Федеральное государственное бюджетное образовательное учреждение высшего

образования

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»**

**Департамент анализа данных и машинного обучения**

Пояснительная записка к курсовой работе

по дисциплине “Технологии разработки приложений для мобильных устройств”

на тему:

«Мобильная галерея»

Выполнил(а):

студент группы ПИ20-6 факультета информационных технологий и анализа больших данных

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сухолозов К. С.

Научный руководитель:

доцент Департамента

кандидат технических наук

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Болтачев Э. Ф.

2022 г

Оглавление

[ВВЕДЕНИЕ 3](#_Toc103688806)

[Глава 1. Описание интерфейса приложения 4](#_Toc103688807)

[1.1 Экран «Мои альбомы» 4](#_Toc103688808)

[1.2. Экран галереи 8](#_Toc103688809)

[Глава 2. Изложение состава приложения и алгоритмических решений 10](#_Toc103688810)

[2.1 React Native 10](#_Toc103688811)

[2.2. Expo 11](#_Toc103688812)

[2.3. Компоненты 13](#_Toc103688813)

[2.4. Async storage 19](#_Toc103688814)

[2.5. Используемые пакеты 20](#_Toc103688815)

[ЗАКЛЮЧЕНИЕ 24](#_Toc103688816)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ И ИНТЕРНЕТ-РЕСУРСОВ 25](#_Toc103688817)

# ВВЕДЕНИЕ

Мы ежедневно взаимодействуем со своим смартфоном. Современный человек не может представить себе жизнь без мобильного устройства. Смартфон позволяет нам ежедневно узнавать новости, следить за погодой, а также общаться с друзьями, коллегами и близкими посредством мессенджеров и социальных сетей. В процессе организации профессиональной деятельности или досуга нередко случается ситуация, когда необходимо запечатлеть важный момент с помощью фотографий.

Однако у каждого из нас есть возможность с помощью камеры нашего смартфона сделать фотографию, а для того, чтобы её посмотреть было реализовано приложение мобильной галереи.

Актуальность темы – поскольку в наше время, в силу того что широкое распространение фотографий в цифровом формате набрало обороты, человечество уменьшило количество распечатываемых снимков, банально удобнее хранить их на своём мобильном гаджете. В этом как раз и заключается актуальность приложения «Мобильная галерея»

Цель проекта – демонстрация знаний и навыков создания мобильных приложений с использованием языков Java или JavaScript, умение работать с данными.

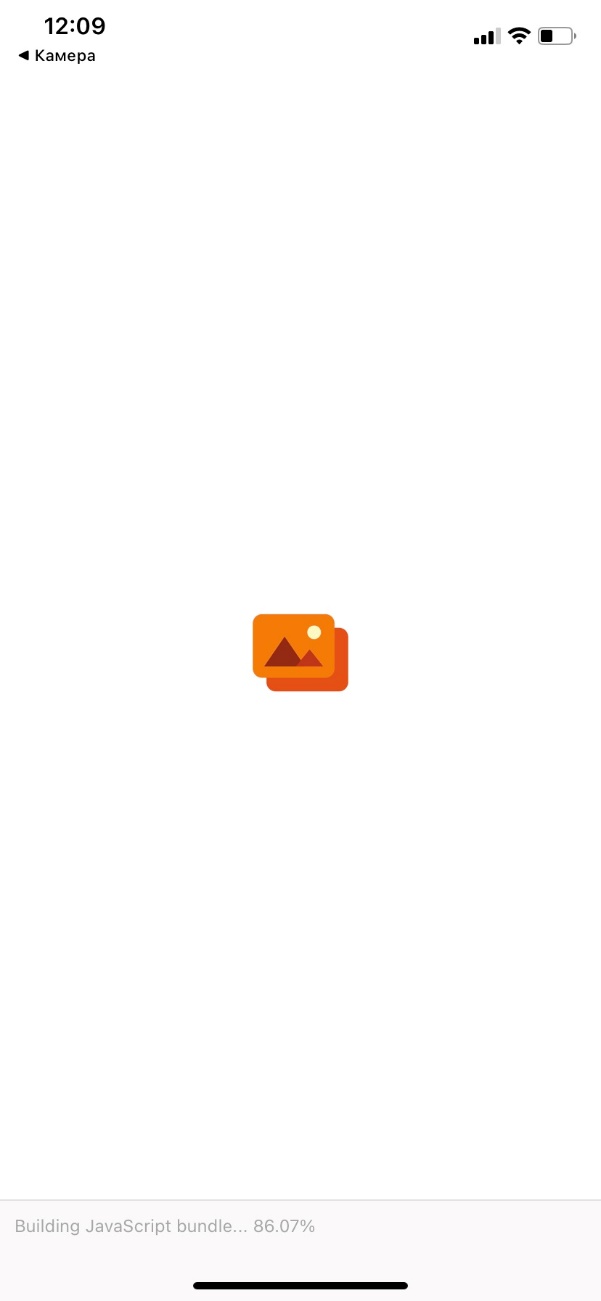
Задача проекта – создание прототипа мобильного приложения для создания, хранения и последующего комфортного использования фотографий в удобном для человека интерфейсе. Для реализации приложения необходимо разработать компоненты для пользовательского интерфейса чтобы в дальнейшем построить на их основании экраны, а также разработать навигацию по ним.

# Глава 1. Описание интерфейса приложения

## Экран «Мои альбомы»

При входе в приложение пользователя встречает загрузочный экран с логотипом по середине (см. рис. 1).

Рисунок 1



После загрузки приложения появляется главный экран, в котором можно создать свой первый альбом предварительно нажав кнопку «+» в правом верхнем углу, ввести название, а также присвоить ему соответствующий логотип (см. рис. 2,3).

Рисунок 2

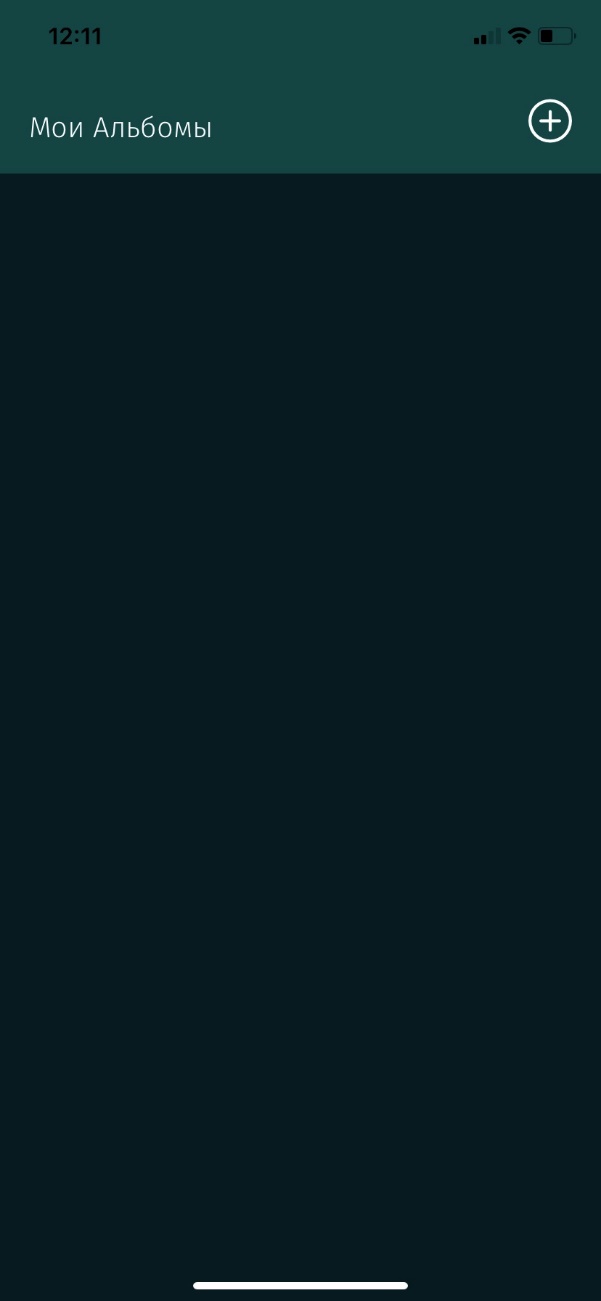
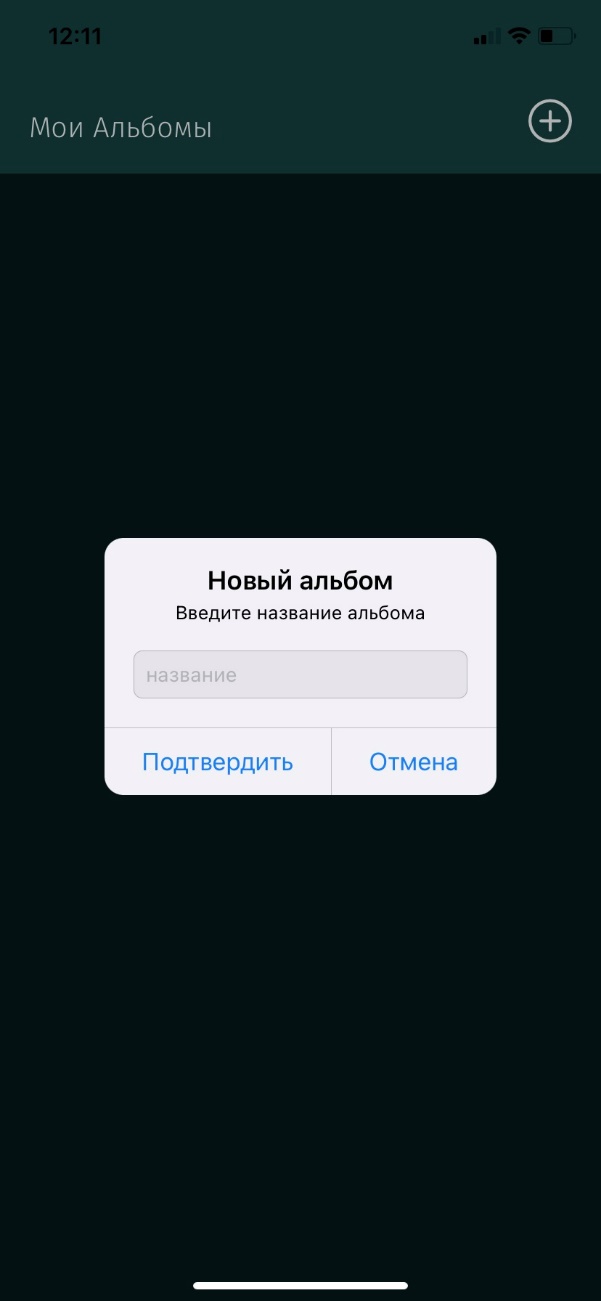
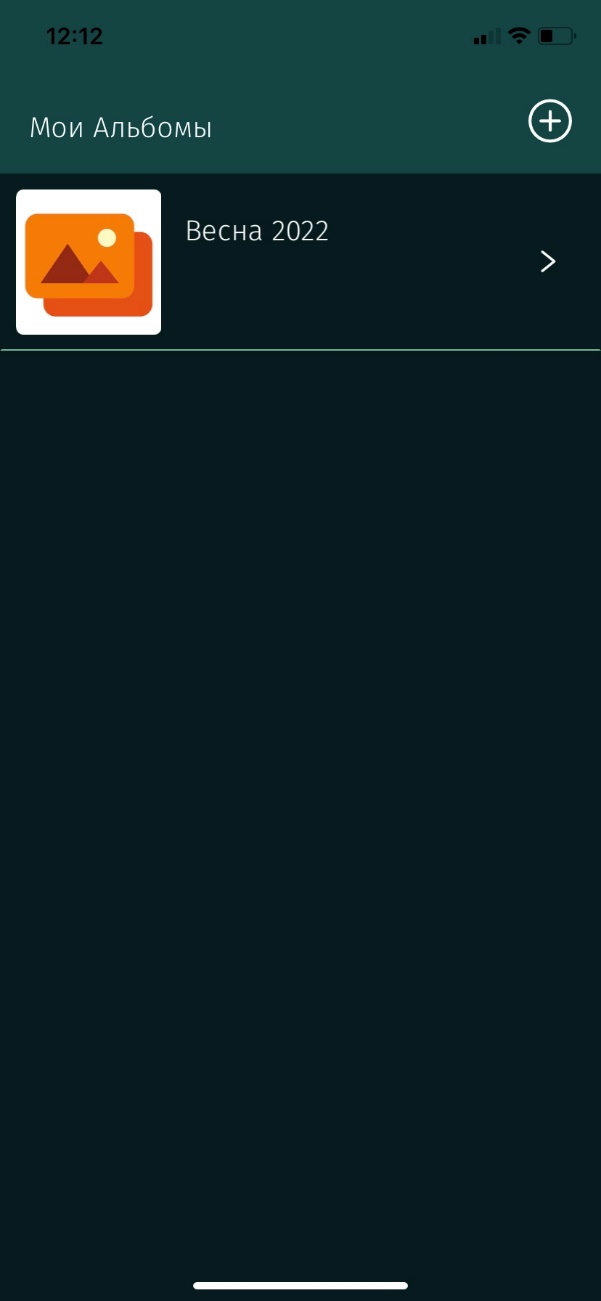


Рисунок 3



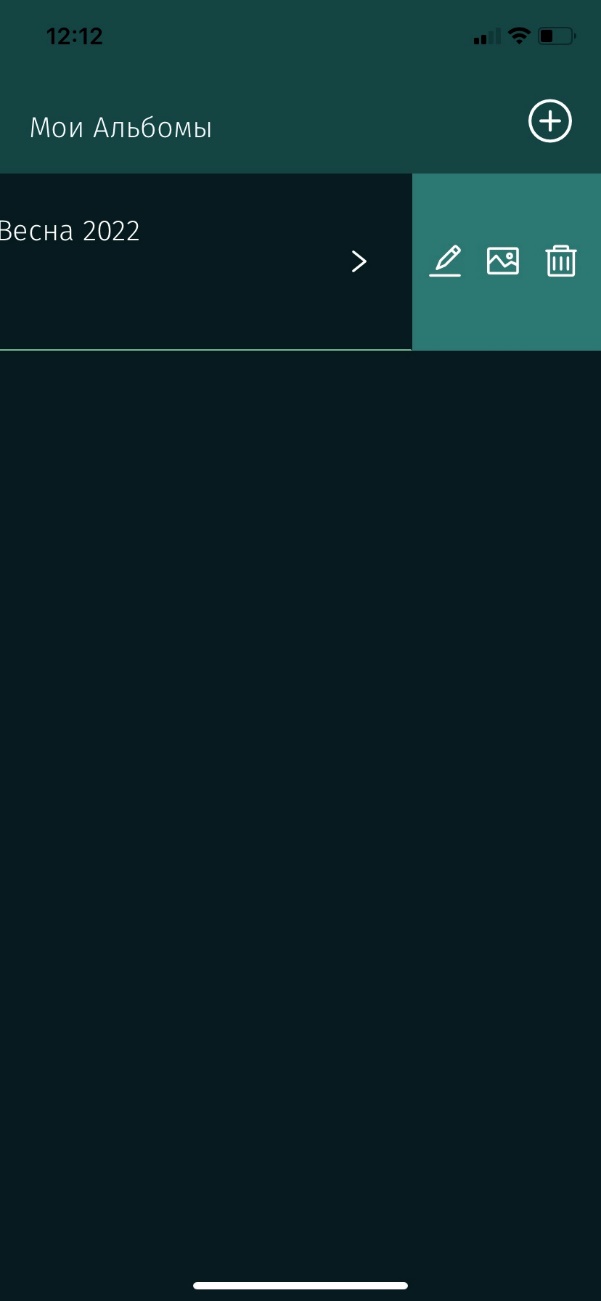
По умолчанию альбом создаётся с логотипом самого приложения (см. рис. 4), но можно изменить его на собственный. Это полезно в тех случаях, когда альбомов становится слишком много, то на поиски нужного сокращается время, так как поиск по логотипу намного удобнее чем по названию. Оригинальная иконка позволяет улучшить визуальное восприятие в целом.

Рисунок 4



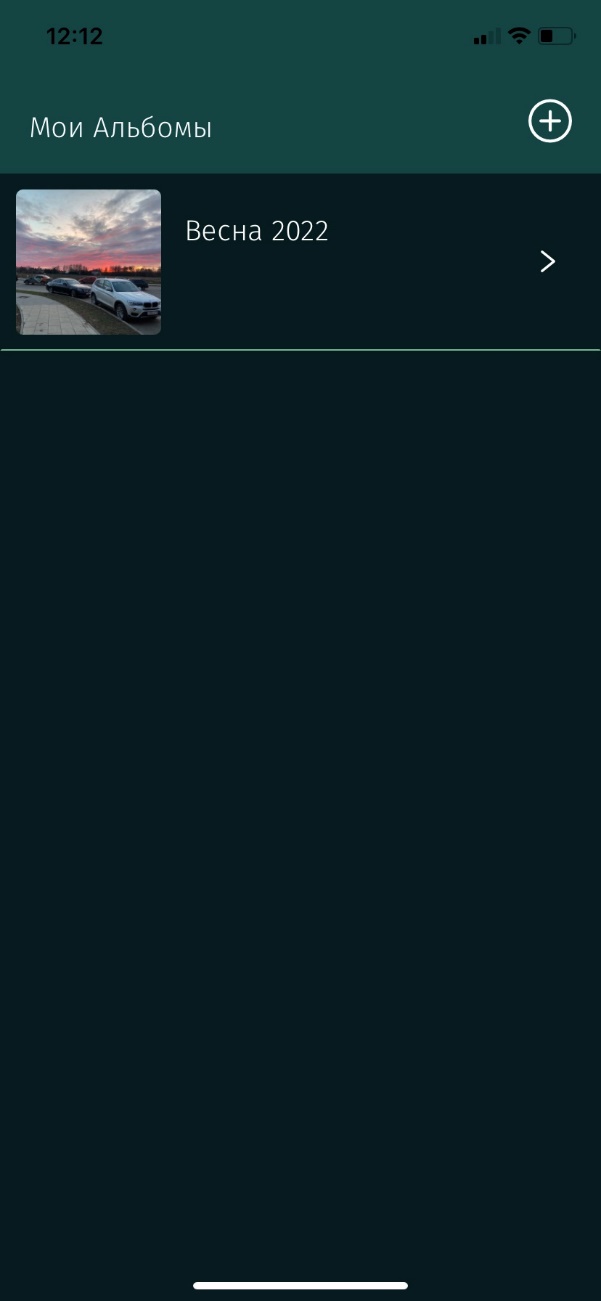
Для того чтобы изменить иконку нужно свайпнуть влево по нашему альбому, появятся три кнопки (см. рис. 5). Первая «карандаш» отвечает за функционал переименования альбома, вторая «изображение» для смены логотипа, а третья «корзина» для его удаления.

Рисунок 5



Нажав на кнопку с «изображением», пользователю даётся выбор из его уже сохранённых фотографий на смартфоне. Выбранная нужная фотография заменяет стандартный логотип альбома (см. рис. 6).

Рисунок 6



Также было реализовано перемещение созданных альбомов между собой местами (см. рис. 7 и 8).

Для того чтобы изменить порядок альбомов в списке созданных, пользователю необходимо удерживать альбом нажатием по нему, а затем перетащить его на нужно место, вверх или вниз.

Рисунок 7

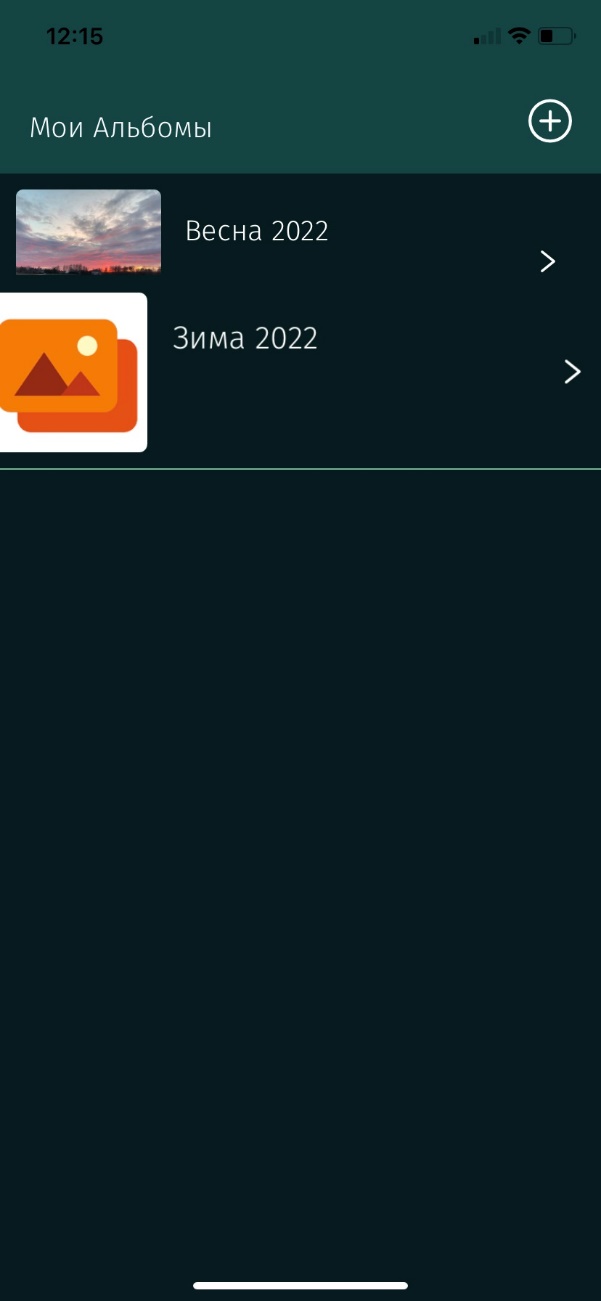
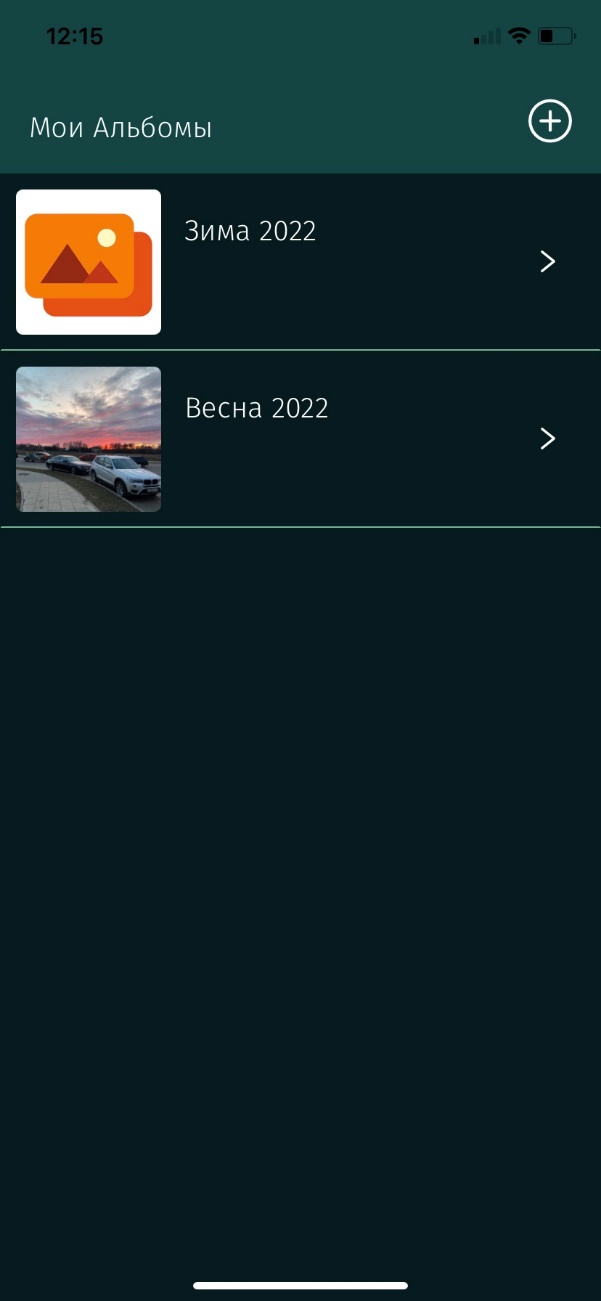


Рисунок 7 показывает реализацию перемещения альбома «Зима 2022» на место альбома «Весна 2022»

Рисунок 8



Таким образом на экране «Мои Альбомы» пользователю доступны создание альбомов, их дальнейшее редактирование, просмотр содержимого, а также удаление.

## 1.2. Экран галереи

Интерфейс данного экрана встречает пользователя надписью выбранного альбома и доступным реализованным функционалом с помощью кнопок в правом верхнем углу (см. рис. 9). Изображения на экране галереи хранятся в уменьшенном формате «квадратиками».

Кнопка «камера» позволяет создавать фотографии с помощью камеры мобильного устройства с последующей загрузкой в выбранный альбом.

Кнопка «файл с плюсом» предназначена для добавления в галерею существующих у пользователя изображений на его смартфоне. При её нажатии попадаем в хранилище существующих фотографий на устройстве. Выбираем нужное и загружаем в галерею (см рис. 10).

Рисунок 9

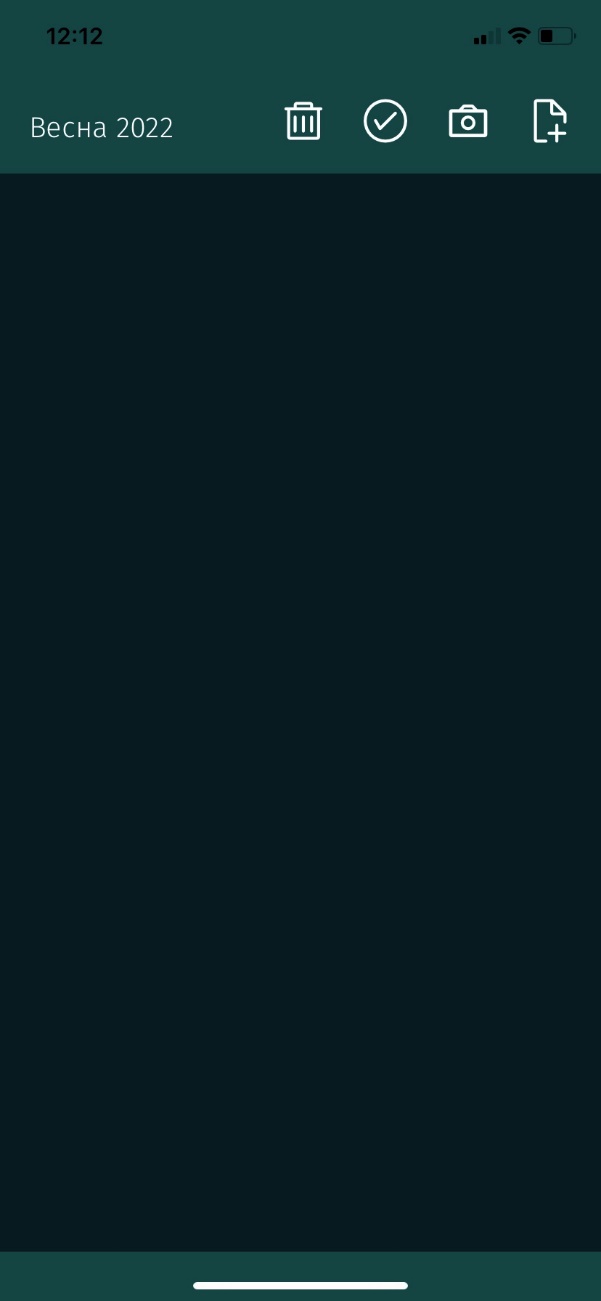
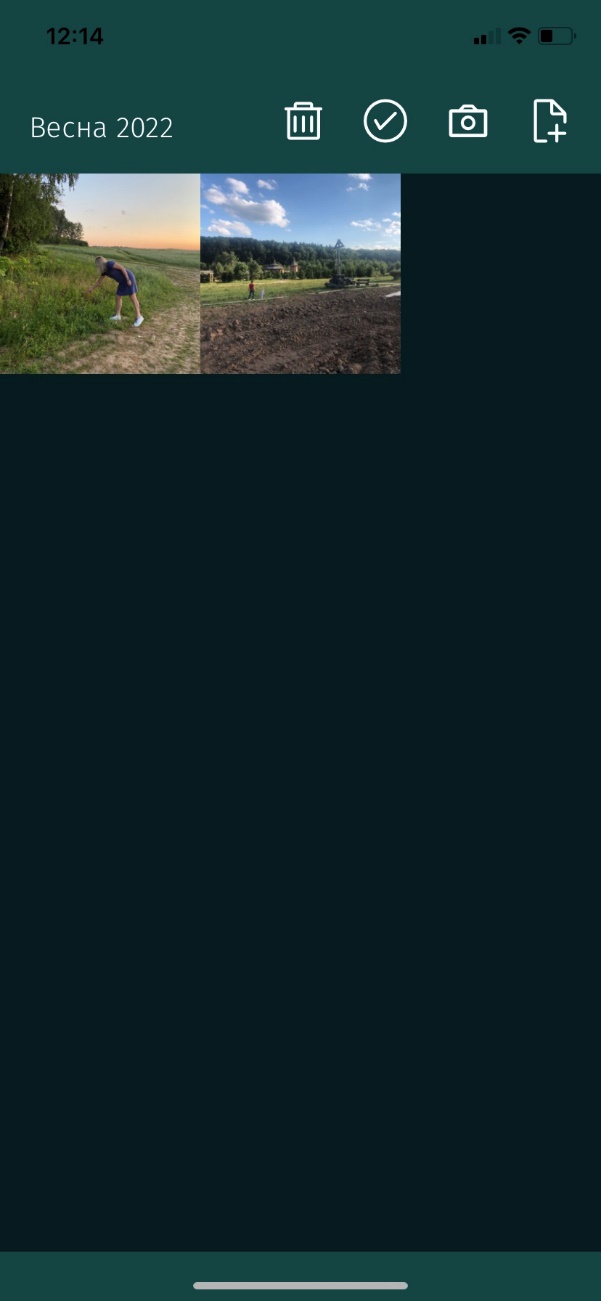


Рисунок 10



Кроме этого, если пользователь захочет удалить ненужные, по его мнению, фотографии, то это реализуется с помощью кнопок «галочка» и «корзина». Чтобы убрать изображение из галереи потребуется нажать кнопку «галочка», выбрав фотографию она потемнеет (см. рис. 11), а после нажать кнопку «корзина» для её удаления. Множественная выборка изображений поможет пользователю выбирать ненужные фотографии и удалять их намного быстрее единичной выборки.

Рисунок 11



Если пользователь нажмёт на фотографию, то изображение развернётся на весь экран (см. рис. 12). Кнопка «Закрыть» возвращает обратно в галерею.

Рисунок 12



# Глава 2. Изложение состава приложения и алгоритмических решений

## 2.1 React Native

В процессе проектирования приложения передо мной стояла задача в определении языка программирования. Первый вариант – написать приложение на Java, но, в таком случае, приложение являлось бы нативным (только для пользователей Android), а это негативно скажется на охвате аудитории и распространении приложения. Второй вариант – использовать JS и React Native, созданный компанией Facebook. Данный фреймворк позволяет быстро разрабатывать нативные приложения как для Android, так и IOS. Так как кроссплатформенность имеет большое значение для моего проекта, второй вариант является более выигрышным решением.

Также стоит отметить, что разработка с использованием React Native дает возможность создавать интерфейс приложения с продуманным дизайном и без вреда для удобства пользователя. Например, такие компоненты, как View, Text, и Image, независимы и сопоставляются с собственными строительными блоками пользовательского интерфейса платформы.

Дизайн — важный аспект, на который стоит обратить внимание. Несмотря на всю функциональность приложения, непривлекательный дизайн скорее всего оттолкнет пользователя. У него есть и другая сторона — удобство взаимодействия пользователя с продуктом. При разработке дизайна есть два подхода. Первый, наиболее сложный подход заключается в создании дизайн-системы с нуля: разрабатывать компоненты и правила их использования. Второй и более простой подход — использовать существующую дизайн-систему, например, Ant Design, специально для React Native. Для React Native существует библиотека react-native-vector-icons/AntDesign, которая имплементирует компоненты из Ant Design. Он предоставляет выбрать множество иконок для приложения, чем я и воспользовался (см. рис. 13)

Рисунок 13



Тем не менее, в процессе написания программы недостаточно пользоваться готовыми решениями, поэтому возникает необходимость продумывать свои собственные компоненты.

## 2.2. Expo

Expo — это платформа с открытым исходным кодом для создания универсальных нативных приложений, которые работают на Android, iOS и в интернете. Он включает в себя универсальную среду выполнения и библиотеки, которые позволяют создавать нативные приложения с помощью написания на React и JavaScript.

Это клиентское программное обеспечение Expo, которое можно найти в AppStore и Google Play Маркет, включая приложения, модули приложения и многое другое. Expo CLI содержит инструменты разработки Expo. С помощью него пользователь будет запускать нашу программу.

Expo дает вам варианты использования. Пользователь может предварительно просмотреть наше приложение через клиентское приложение Expo на своем смартфоне или с помощью симулятора iOS и Android на своем компьютере.

Устанавливая пакет Expo CLI с его помощью, можно запустить локальный сервер разработки, он откроет новое окно для просмотра сервера и предоставит нам QR-код, с помощью которого можно запустить приложение на платформе Android и iOS.

Кроме этого, приложение через Expo можно опубликовать в Google Play Маркет и App Store. Для этого предназначена библиотека EAS Build для Expo CLI, при помощи команд (см. рис. 14) можем выполнить сборку приложения как на Android, так и для iOS. При создании приложения я использовал Expo на операционной системе iOS разработанной компанией Apple.

Рисунок 14

Изображение выглядит как текст

Автоматически созданное описание

Для запуска мобильного приложения «Мобильная галерея» потребуется установленный Node.js и пакет к нему под названием Expo CLI. Установить его модно командой (см. рис. 14) и использовать для старта проекта. Сначала нужно загрузить необходимые для проекта библиотекой командой (см. рис. 15), а уже после ввести команду «npm start». Она запустит процесс expo start с развёртыванием локального сервера, с помощью которого пользователь и сможет зайти в приложение.

Рисунок 15

Изображение выглядит как текст

Автоматически созданное описание

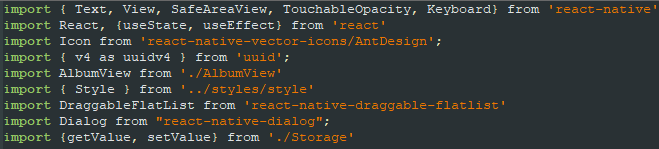
Процесс установки всех компонентов и библиотек довольно прост, поэтому сложностей возникнуть не должно. Благодаря Expo разработчику легче разрабатывать приложение, так как платформа позволяет просматривать проекты в стадии разработки.

## 2.3. Компоненты

В основе любых приложений, написанных на React, лежит компонентный подход. Чтобы построить полноценный React Native приложение, чаще всего недостаточно использовать уже готовые сторонние компоненты. В связи с этим на их основании приходится создавать свою библиотеку компонентов.

Экран «Мои Альбомы» содержится в компоненте ListOfAlbums.js. Он содержит в себе импорт нужных для работы приложения библиотек (см. рис. 16), а также исходный код главной страницы. Здесь выполнена реализация добавления альбомов в список, выбора его названия, логотипа и функциональных кнопок.

Рисунок 16



Когда пользователь нажимает на кнопку «плюс», с помощью которой было реализовано добавления нового альбома в список, используется «handleConfirmNew» который заносит новый альбом в список (см. рис 17) и «ListOfAlbums» который отображает существующие альмбомы (см. рис. 18).

Рисунок 17

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 18

Изображение выглядит как текст

Автоматически созданное описание

«showDialog» предназначен для вывода диалогового окна, «onChange» при нажатии кнопки «карандаш» для изменения названия альбома.

Также был использован компонент react-native-dialog, с его помощью был реализован пользовательский интерфейс всплывающего окна при создании нового альбома (см. рис. 19).

* Dialog.Container: этот компонент является корневым компонентом диалога, и все остальные компоненты должны быть вложены в него.
* Dialog.Title: Text компонент, оформленный в виде родного заголовка диалогового окна.
* Dialog.Description: Text компонент, стилизованный под собственное диалоговое описание.
* Dialog.Button: компонент, стилизованный под нативную диалоговую кнопку.
* Dialog.Input: TextInput компонент, стилизованный под собственный диалоговый ввод.

Рисунок 19

Изображение выглядит как текст

Автоматически созданное описание

За вид одного альбома на экране «Мои альбомы» отвечает компонент «AlbumView.js» (см. рис. 20). Здесь реализован функционал навигации с помощью компонента «@react-navigation» благодаря которому мы можем переместиться в компонент Album.js с помощью «openAlbum» при нажатии на выбранный пользователем альбом.

Также благодаря «handleConfirmEdit», «handleDelete» и «handlePicture» были реализованы функциональные кнопки у каждого альбома для изменения его имени, изменении иконки и удалении соответственно. Функция «handleCancelEdit» отменяет переименование альбома, а «onChange» наоборот, меняет название.

Рисунок 20

Изображение выглядит как текст

Автоматически созданное описание

За внешний вид также отвечает компоненты «react-native-vector-icons/AntDesign» с помощью которого реализованы иконки кнопок и «react-native-dialog» для основы стилей (см. рис. 21).

Рисунок 21

Изображение выглядит как текст

Автоматически созданное описание

Переходя в выбранный пользователем альбом, мы перемещаемся в компонент «Album.js» который и реализует экран галереи изображений (см. рис. 9). Нас встречает название выбранного альбома в левом верхнем блоке, в правом блоке функциональные кнопки и по середине сама сетка с изображениями.

Необходимо рассказать про функциональные кнопки на данном экране. За кнопку добавления уже имеющихся на мобильном гаджете пользователя фотографий отвечает «handleAddFile» (см. рис. 22), он открывает новое окно с возможностью выбора фотографии для последующей загрузки в галерею. Следующая кнопка «Камера» отвечает за возможность с помощью камеры сделать новый снимок и добавить его в галерею это реализовано в «handleCamera». Кнопки «галочка» и «корзина» соотвественно в «handleChoice» благодаря которому можно выбрать нужные нам изображения для того чтобы в дальнейшем использовать «handleDelete» для их удаления.

Рисунок 22

Изображение выглядит как текст

Автоматически созданное описание

Дизайн и стили на экране галереи были также созданы с помощью компонентов «react-native-vector-icons/AntDesign» и «react-native-dialog».

## 2.4. Async storage

Для хранения изображений пользователя я выбрал компонент «Async Storage». Этот пакет нужен для сохранения фотографий и манипулирования с данными в хранилище через асинхронное хранилище. Оно может хранить в себе только string данные, поэтому для хранения объектных данных было необходимо сначала их преобразовать. Для преобразования в JSON были использованы JSON.stringify() для сохранения данных и JSON.parse() для их загрузки. Использование компонента было реализовано в файле «Storage.js» (см. рис. 23).

Рисунок 23

Изображение выглядит как текст

Автоматически созданное описание

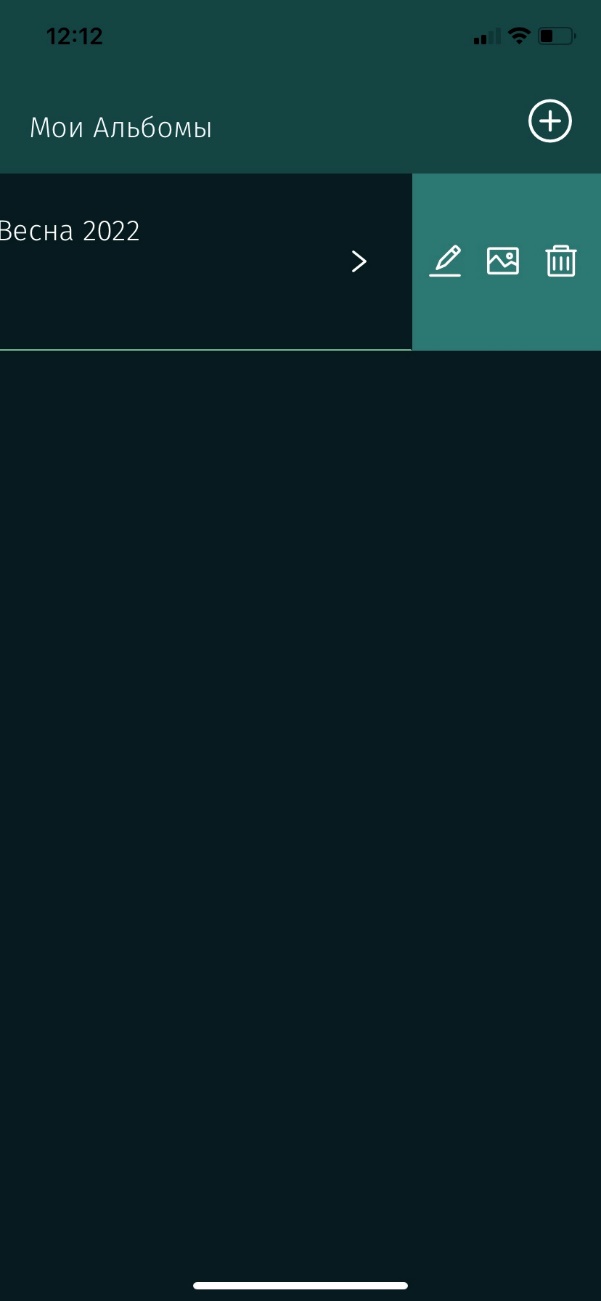
* setItem() был использован для добавления нового элемента данных и для изменения существующего элемента.
* getItem() возвращает нам элемент данных, либо возвращает null в обратном случае.
* removeItem() удаляет элемент.

AsyncStorage использует SQLite в качестве внутреннего хранилища. Новая база данных, которая используется нашим приложением, будет создана на основе текущего файла базы данных, если новой не существует, то есть с пустыми альбомами и данными в них. При повторной загрузке приложения если пакет находит существующую базу данных, то новую он не создаёт.

## 2.5. Используемые пакеты

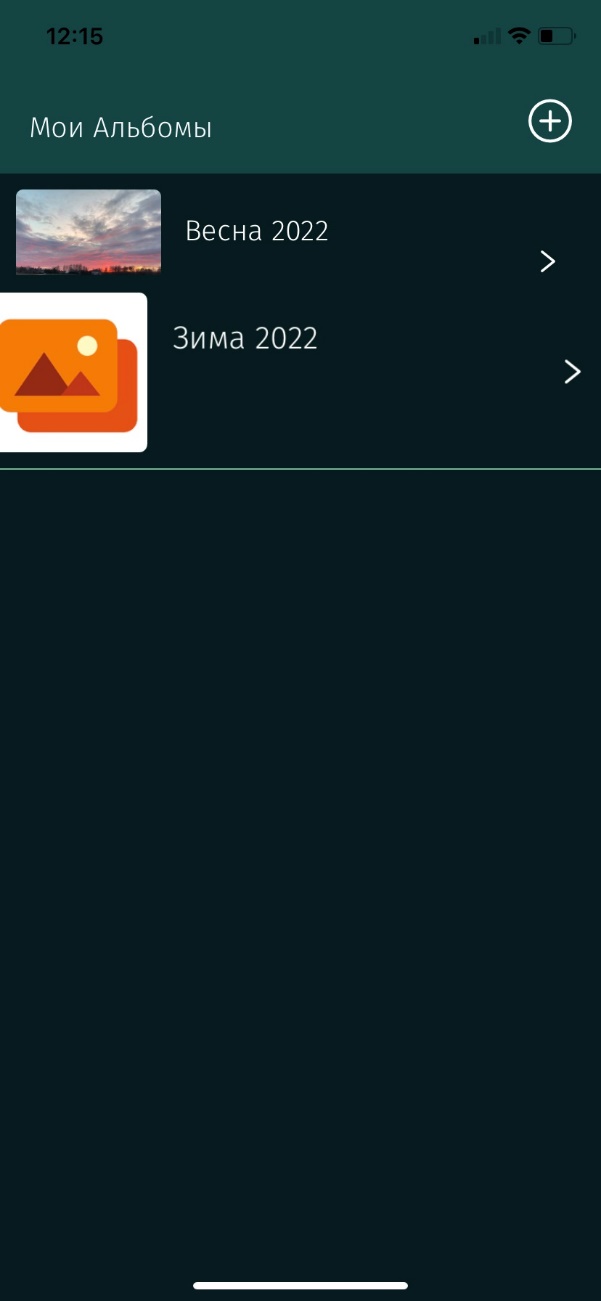
* Для создания универсального уникального идентификатора UUID был использован компонент uuid4.
* Компонент React Native Vector Icons был взять для реализации кнопок, логотипа и панелей навигации. С помощью данного пакета было намного проще стилизировать проект.
* Для свайпов был взять за основу пакет React Native Swiper.
* Благодаря компоненту React Native Swipe List View реализован функционал свайпов на экране со списком альбомов для вызова кнопок (см. рис. 24).

Рисунок 24



* React Native Draggable FlatList предоставил возможность реализации перемещения альбомов между собой путём свайпа в нужно сторону, вверх или вниз (см. рис. 25).

Рисунок 25



* Компонент ImagePicker (см. рис. 26) предоставляет возможность пользователю доступ к пользовательскому интерфейсу систему для выбора изображений из библиотеки смартфона или создания новых снимков с помощью камеры устройства.

Рисунок 26

Изображение выглядит как текст

Автоматически созданное описание

* pickAlbumProfile открывает пользовательский интерфейс системы пользователя для выбора изображения.
* useCamera используется для создания снимка.
* getImage предоставляет возможность выбранной фотографии в системном пользовательском интерфейсе операционной системы загружать файл в альбом.

# ЗАКЛЮЧЕНИЕ

Главным образом была достигнута цель проекта: были продемонстрированы имеющиеся навыки разработки мобильных приложений. Создано кроссплатформенное мобильное приложение на Java Script и React Native, которое было протестировано на стабильную работу в операционных системах iOS и Android. Приложение имеет удобный и понятный пользовательский интерфейс, с помощью которого пользователь может взаимодействовать с приложением и просматривать свою галерею изображений сортируя их по альбомам. Предусмотрена работа с изменением альбомов путём перемещения их между собой, что-то важное можно поднять повыше. Данное приложение имеет возможность дальнейшего расширения и добавления новых компонентов и дополнительного функционала. В процессе реализации работы над проектом были получены дополнительные знания в области создания мобильных приложений на React Native. Поставленная цель была достигнута: были продемонстрированы имеющиеся навыки разработки мобильных приложений.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ И ИНТЕРНЕТ-РЕСУРСОВ

1. React Native: официальный сайт. URL: https://reactnative.dev/ (дата обращения: 01.05.2022).
2. React Navigation: официальный сайт. URL: https://reactnavigation.org/ (дата обращения: 01.05.2022).
3. Expo: официальный сайт. URL: https://docs.expo.dev/ (дата обращения: 01.05.2022).
4. UUID4: официальный сайт. URL: <https://www.npmjs.com/package/uuidv4/> (дата обращения: 01.05.2022).
5. React Native Vectors Icons: документация на GitHub. URL: <https://github.com/oblador/react-native-vector-icons>/ (дата обращения: 01.05.2022).
6. React Native Swiper: документация на GitHub. URL: <https://github.com/leecade/react-native-swiper>/ (дата обращения: 01.05.2022).
7. React Native Swipe List View: документация на GitHub. URL: <https://github.com/jemise111/react-native-swipe-list-view/> (дата обращения: 01.05.2022).
8. React Native Draggable FlatList: документация на GitHub. URL: <https://github.com/computerjazz/react-native-draggable-flatlist/> (дата обращения: 01.05.2022).
9. ImagePicker: официальный сайт с документацией. URL: <https://docs.expo.dev/versions/latest/sdk/imagepicker/> (дата обращения: 01.05.2022).