**Итоговый отчёт**

**Wallet App  
Роман Харченко 13 группа**

Разработка велась при помощи следующего стека технологий:

Back-end: Swift, Firebase, Foundation

Database: PostgreSQL

Front-end: Swift, UIKit  
Design: Figma  
Testing: Manual Testing, Mobile AQA, TestRail, Jira Atlassian

**Стратегия дизайна**

Стратегия дизайна включает следующие подразделы:

•                **заинтересованные стороны** — лица, которые хотят быстро и свободно пользоваться своими дисконтными картами, оставить свою большую пачку карт дома и ходить по своим любимым магазинам, не переживая о том, взяли ли вы карту;

•                **видение продукта заинтересованными лицами** (задачи продукта) — удобное и компактное хранение данных большого количества дисконтных карт

•                **конфликты и противоречия** — необходимо получить разрешение магазина на хранение дисконтных карт этого магазина

•                **задачи бизнеса (верифицируемые), задачи маркетинга и брендинга** — удобное хранения всех дисконтных карт пользователя с быстрым доступом к ним

•                **измеримые критерии успешности** — популярность приложения на рынке, оценки пользователей, динамика и количество скачиваний приложения

•                **технические возможности и ограничения** — Языки программирования — JS (NativeScript), Python (Django, Django Rest Framework), Поддерживаемые платформы -  Android;

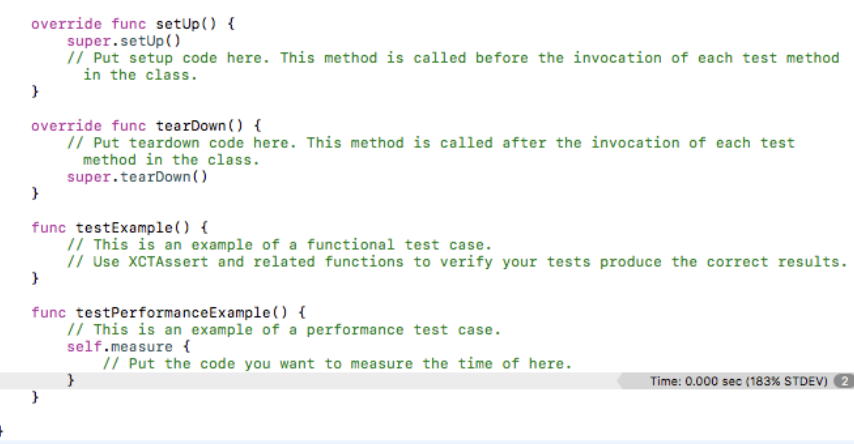
•                **представления заинтересованных лиц о пользователях (целевая аудитория)** — люди, пользующиеся дисконтными карточками, и желающие сделать пользование ими более удобным и безопасным (чтобы не потерять карточку).

•                **бюджет и график проекта** — 100K$. 4 месяца на реализацию.

# Тестирование

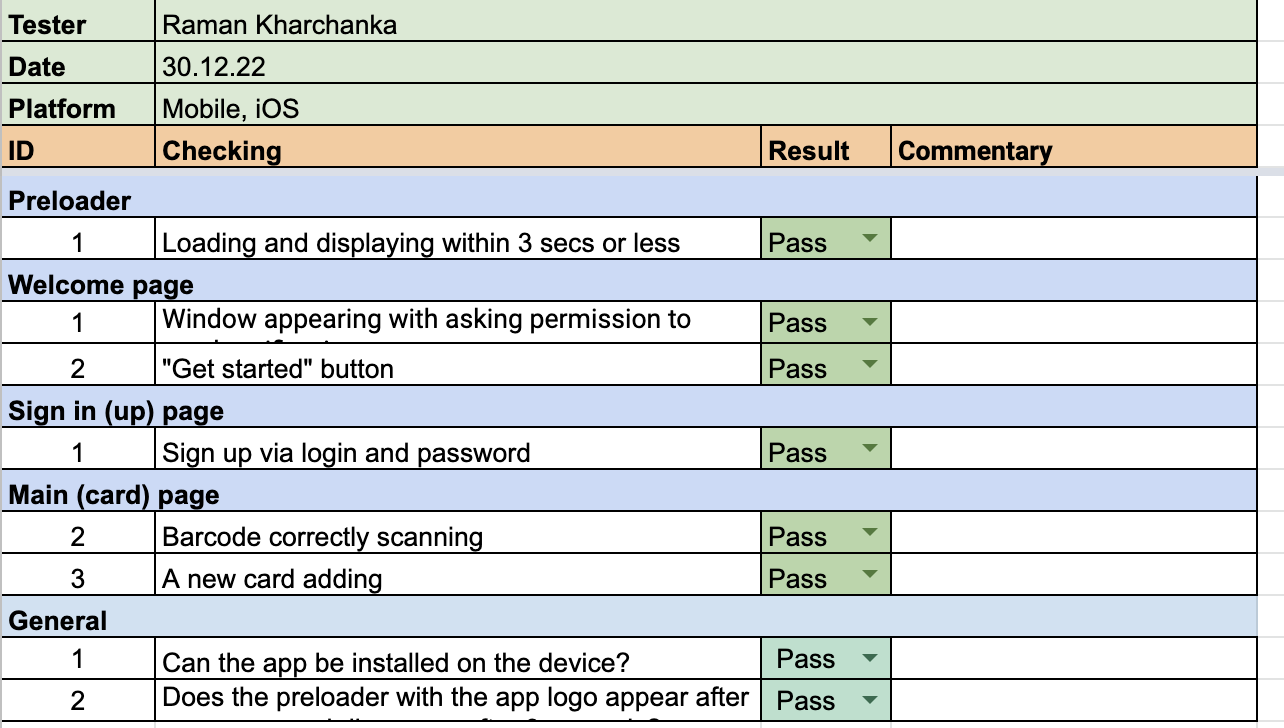
## **Тестирование кода. Юнит-тесты**

Пример написанных текстов в исходном тексте программы:



## **Тестирование приложения. Юзабилити-тестирование**

Были написаны check-list’ы.

****

# **Разработка**

ВНИМАНИЕ ГОВНОКОД !!!

//

**import** UIKit

**class** WelcomeController: UIViewController {

**let** titleLabel = UILabel()

**let** descriptionLabel = UILabel()

**let** getStartedButton = UIButton()

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

**self**.view.backgroundColor = hexStringToUIColor(hex: "#12151B")

        getStartedButton.backgroundColor = hexStringToUIColor(hex: "#405EFC")

        getStartedButton.titleLabel?.font = UIFont(name: "Arial", size: 16)

        getStartedButton.titleLabel?.textAlignment = .center

        getStartedButton.setTitleColor(hexStringToUIColor(hex: "FFFFFF"), for: .normal)

        getStartedButton.setTitle("Get Started", for: .normal)

        getStartedButton.layer.cornerRadius = 8

        getStartedButton.addTarget(**self**, action: **#selector**(presentLoginController), for: .touchUpInside)

        descriptionLabel.textColor = UIColor(red: 255, green: 255, blue: 255, alpha: 0.8)

        descriptionLabel.font = UIFont(name: "Arial", size: 16)

        descriptionLabel.textAlignment = .left

        descriptionLabel.numberOfLines = 2

        descriptionLabel.text = "This is your own wallet. Save your existing\nand new discount cards for free."

        titleLabel.textColor = UIColor(red: 255, green: 255, blue: 255, alpha: 1.0)

        titleLabel.font = UIFont(name: "Arial", size: 28)

        titleLabel.textAlignment = .left

        titleLabel.numberOfLines = 2

        titleLabel.text = "Your personal\nwallet"

**self**.view.addSubview(getStartedButton)

        getStartedButton.translatesAutoresizingMaskIntoConstraints = **false**

        getStartedButton.leftAnchor.constraint(equalTo: **self**.view.leftAnchor, constant: 24).isActive = **true**

        getStartedButton.rightAnchor.constraint(equalTo: **self**.view.rightAnchor, constant: -24).isActive = **true**

        getStartedButton.bottomAnchor.constraint(equalTo: **self**.view.bottomAnchor, constant: -35).isActive = **true**

        getStartedButton.heightAnchor.constraint(equalToConstant: 46).isActive = **true**

**self**.view.addSubview(descriptionLabel)

        descriptionLabel.translatesAutoresizingMaskIntoConstraints = **false**

        descriptionLabel.leftAnchor.constraint(equalTo: **self**.view.leftAnchor, constant: 24).isActive = **true**

        descriptionLabel.rightAnchor.constraint(equalTo: **self**.view.rightAnchor, constant: -24).isActive = **true**

        descriptionLabel.bottomAnchor.constraint(equalTo: getStartedButton.topAnchor, constant: -95).isActive = **true**

**self**.view.addSubview(titleLabel)

        titleLabel.translatesAutoresizingMaskIntoConstraints = **false**

        titleLabel.leftAnchor.constraint(equalTo: **self**.view.leftAnchor, constant: 24).isActive = **true**

        titleLabel.rightAnchor.constraint(equalTo: **self**.view.rightAnchor, constant: -24).isActive = **true**

        titleLabel.bottomAnchor.constraint(equalTo: descriptionLabel.topAnchor, constant: -16).isActive = **true**

    }

**@objc** **func** presentLoginController() {

**let** loginController = LoginController()

        loginController.modalPresentationStyle = .fullScreen

**self**.present(loginController, animated: **true**, completion: **nil**)

    }

}

**extension** UIViewController {

**func** hexStringToUIColor (hex:String) -> UIColor {

**var** cString:String = hex.trimmingCharacters(in: .whitespacesAndNewlines).uppercased()

**if** (cString.hasPrefix("#")) {

            cString.remove(at: cString.startIndex)

        }

**if** ((cString.count) != 6) {

**return** UIColor.gray

        }

**var** rgbValue:UInt64 = 0

        Scanner(string: cString).scanHexInt64(&rgbValue)

**return** UIColor(

            red: CGFloat((rgbValue & 0xFF0000) >> 16) / 255.0,

            green: CGFloat((rgbValue & 0x00FF00) >> 8) / 255.0,

            blue: CGFloat(rgbValue & 0x0000FF) / 255.0,

            alpha: CGFloat(1.0)

        )

    }

**import** UIKit

**class** LoginController: UIViewController {

**let** titleLabel = UILabel()

**let** emailField = UITextField()

**let** passwordField = UITextField()

**let** signInButton = UIButton()

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

**self**.view.backgroundColor = hexStringToUIColor(hex: "#12151B")

        titleLabel.textColor = UIColor(red: 255, green: 255, blue: 255, alpha: 1.0)

        titleLabel.font = UIFont(name: "Arial-BoldMT", size: 26)

        titleLabel.textAlignment = .left

        titleLabel.numberOfLines = 1

        titleLabel.text = "Sign in"

        //titleLabel.bold()

**self**.view.addSubview(titleLabel)

        titleLabel.translatesAutoresizingMaskIntoConstraints = **false**

        titleLabel.leftAnchor.constraint(equalTo: **self**.view.leftAnchor, constant: 24).isActive = **true**

        titleLabel.rightAnchor.constraint(equalTo: **self**.view.rightAnchor, constant: -24).isActive = **true**

        titleLabel.topAnchor.constraint(equalTo: **self**.view.topAnchor, constant: 108).isActive = **true**

        emailField.attributedPlaceholder = NSAttributedString(

            string: "Username",

            attributes: [NSAttributedString.Key.foregroundColor: hexStringToUIColor(hex: "#838390")]

        )

        emailField.layer.cornerRadius = 8

        emailField.font = UIFont(name: "Arial", size: 16)

        emailField.layer.borderWidth = 1

        emailField.textColor = .white

        emailField.layer.borderColor = hexStringToUIColor(hex: "#3D485C").cgColor

        emailField.indent(size: 16)

        emailField.autocapitalizationType = .none

**self**.view.addSubview(emailField)

        emailField.translatesAutoresizingMaskIntoConstraints = **false**

        emailField.leftAnchor.constraint(equalTo: **self**.view.leftAnchor, constant: 24).isActive = **true**

        emailField.rightAnchor.constraint(equalTo: **self**.view.rightAnchor, constant: -24).isActive = **true**

        emailField.topAnchor.constraint(equalTo: **self**.titleLabel.bottomAnchor, constant: 27).isActive = **true**

        emailField.heightAnchor.constraint(equalToConstant: 46).isActive = **true**

        passwordField.attributedPlaceholder = NSAttributedString(

            string: "Password",

            attributes: [NSAttributedString.Key.foregroundColor: hexStringToUIColor(hex: "#838390")]

        )

        passwordField.layer.cornerRadius = 8

        passwordField.font = UIFont(name: "Arial", size: 16)

        passwordField.textColor = .white

        passwordField.layer.borderWidth = 1

        passwordField.layer.borderColor = hexStringToUIColor(hex: "#3D485C").cgColor

        passwordField.indent(size: 16)

        passwordField.autocapitalizationType = .none

        passwordField.isSecureTextEntry = **true**

**self**.view.addSubview(passwordField)

        passwordField.translatesAutoresizingMaskIntoConstraints = **false**

        passwordField.leftAnchor.constraint(equalTo: **self**.view.leftAnchor, constant: 24).isActive = **true**

        passwordField.rightAnchor.constraint(equalTo: **self**.view.rightAnchor, constant: -24).isActive = **true**

        passwordField.topAnchor.constraint(equalTo: **self**.emailField.bottomAnchor, constant: 24).isActive = **true**

        passwordField.heightAnchor.constraint(equalToConstant: 46).isActive = **true**

        signInButton.backgroundColor = hexStringToUIColor(hex: "#405EFC")

        signInButton.titleLabel?.font = UIFont(name: "Arial", size: 16)

        signInButton.titleLabel?.textAlignment = .center

        signInButton.setTitleColor(hexStringToUIColor(hex: "FFFFFF"), for: .normal)

        signInButton.setTitle("Sign in", for: .normal)

        signInButton.layer.cornerRadius = 8

        signInButton.addTarget(**self**, action: **#selector**(signInButtonTapped), for: .touchUpInside)

**self**.view.addSubview(signInButton)

        signInButton.translatesAutoresizingMaskIntoConstraints = **false**

        signInButton.leftAnchor.constraint(equalTo: **self**.view.leftAnchor, constant: 24).isActive = **true**

        signInButton.rightAnchor.constraint(equalTo: **self**.view.rightAnchor, constant: -24).isActive = **true**

        signInButton.topAnchor.constraint(equalTo: **self**.passwordField.bottomAnchor, constant: 56).isActive = **true**

        signInButton.heightAnchor.constraint(equalToConstant: 46).isActive = **true**

    }

      // **MARK: - Actions**

**@objc** **private** **func** signInButtonTapped() {

**guard** **let** username = emailField.text, !username.isEmpty **else** {

//          errorLabel.text = "Please enter a username."

//          errorLabel.isHidden = false

            print("Please enter a username.")

**return**

        }

**guard** **let** password = passwordField.text, !password.isEmpty **else** {

//          errorLabel.text = "Please enter a password."

//          errorLabel.isHidden = false

            print("Please enter a password.")

**return**

        }

        // Validate the login credentials

**let** isValid = validate(username: username, password: password)

**if** isValid {

          // Save the login credentials

          save(username: username, password: password)

          // Dismiss the login screen

            // example ViewController

**let** myVC = CardsController()

            myVC.modalPresentationStyle = .fullScreen

            // create the NavigationController with my VC as root

**let** navCon = UINavigationController(rootViewController: myVC)

            navCon.modalPresentationStyle = .fullScreen

**self**.present(navCon, animated: **true**, completion: **nil**)

        } **else** {

//          errorLabel.text = "Invalid username or password."

//          errorLabel.isHidden = false

            print("Invalid username or password.")

        }

      }

      // **MARK: - Helper Methods**

**private** **func** validate(username: String, password: String) -> Bool {

        // Validate the login credentials here

        // For example, you can check if the username and password match a pre-defined set of credentials

**let** isValid = (username == "test" && password == "test")  // Replace this with your validation logic

**return** isValid

      }

**private** **func** save(username: String, password: String) {

      // Save the login credentials to the user defaults

      UserDefaults.standard.set(username, forKey: "username")

      UserDefaults.standard.set(password, forKey: "password")

      UserDefaults.standard.synchronize()

    }

}

**extension** UILabel{

**func** bold() -> UILabel {

**if** **let** descriptor = **self**.font.fontDescriptor.withSymbolicTraits(UIFontDescriptor.SymbolicTraits.traitBold){

**self**.font = UIFont(descriptor: descriptor, size: 0)

        }

**return** **self**

    }

}

**extension** UITextField {

**func** indent(size:CGFloat) {

**self**.leftView = UIView(frame: CGRect(x: **self**.frame.minX, y: **self**.frame.minY, width: size, height: **self**.frame.height))

**self**.leftViewMode = .always

    }

}

**import** UIKit

**struct** Card {

**var** storeName: String

**var** barCode: String

**var** cardNumber: String

**var** image: UIImage

**var** barImage: UIImage?

**var** favorites: Bool

}

**class** CardsController: UIViewController, UICollectionViewDataSource, UICollectionViewDelegate, UICollectionViewDelegateFlowLayout {

**let** reuseIdentifier = "cell"

**var** favoritesFlag = **false**

**var** cardsArray = [Card]()

**var** cardsArrayFavorites = [Card]()

**let** allCardsButton = UIButton()

**let** favoritesButton = UIButton()

    // create a collection view and set its constraints

**let** collectionView: UICollectionView = {

**let** layout = UICollectionViewFlowLayout()

        layout.scrollDirection = .vertical

**let** cv = UICollectionView(frame: .zero, collectionViewLayout: layout)

        cv.translatesAutoresizingMaskIntoConstraints = **false**

**return** cv

    }()

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

        cardsArray = getProducts()

**for** i **in** cardsArray {

**if** (i.favorites == **true**) {

                cardsArrayFavorites.append(i)

            }

        }

        allCardsButton.backgroundColor = hexStringToUIColor(hex: "#405EFC")

        allCardsButton.setTitle("All cards", for: .normal)

        allCardsButton.titleLabel?.textColor = .white

        allCardsButton.titleLabel?.font = UIFont(name: "Arial", size: 16)

        allCardsButton.layer.cornerRadius = 20

        allCardsButton.addTarget(**self**, action: **#selector**(allCardsButtonClick), for: .touchUpInside)

        favoritesButton.backgroundColor = hexStringToUIColor(hex: "#242B37")

        favoritesButton.setTitle("Favorites", for: .normal)

        favoritesButton.titleLabel?.textColor = .white

        favoritesButton.setTitleColor(.white, for: .normal)

        favoritesButton.titleLabel?.font = UIFont(name: "Arial", size: 16)

        favoritesButton.layer.cornerRadius = 20

        favoritesButton.addTarget(**self**, action: **#selector**(favoritesButtonClick), for: .touchUpInside)

        allCardsButton.translatesAutoresizingMaskIntoConstraints = **false**

        favoritesButton.translatesAutoresizingMaskIntoConstraints = **false**

**self**.view.backgroundColor = hexStringToUIColor(hex: "#12151B")

        // Настраиваем навигационное бар с надписью "My cards" слева

        setLeftAlignTitleView(font: UIFont(name: "Arial-BoldMT", size: 26)!, text: "My cards", textColor: .white)

**self**.navigationController?.view.backgroundColor = hexStringToUIColor(hex: "#12151B")

        // Создаем кнопку с иконкой плюса

**let** addButton = UIBarButtonItem(barButtonSystemItem: .add, target: **self**, action: **#selector**(addButtonTapped))

        addButton.tintColor = .white

        // Добавляем кнопку справа в навигационное бар

**self**.navigationItem.rightBarButtonItem = addButton

**self**.view.addSubview(allCardsButton)

**self**.view.addSubview(favoritesButton)

        NSLayoutConstraint.activate([

            allCardsButton.topAnchor.constraint(equalTo: view.safeAreaLayoutGuide.topAnchor, constant: 5),

            allCardsButton.heightAnchor.constraint(equalToConstant: 35),

            allCardsButton.widthAnchor.constraint(equalToConstant: 94),

            allCardsButton.leftAnchor.constraint(equalTo: view.leftAnchor, constant: 24),

            favoritesButton.topAnchor.constraint(equalTo: allCardsButton.topAnchor),

            favoritesButton.heightAnchor.constraint(equalToConstant: 35),

            favoritesButton.widthAnchor.constraint(equalToConstant: 94),

            favoritesButton.leftAnchor.constraint(equalTo: allCardsButton.rightAnchor, constant: 8),

        ])

        collectionView.backgroundColor = hexStringToUIColor(hex: "#12151B")

        // register the cell class and its reuse identifier

        collectionView.register(UICollectionViewCell.**self**, forCellWithReuseIdentifier: reuseIdentifier)

        // set the collection view's data source and delegate

        collectionView.dataSource = **self**

        collectionView.delegate = **self**

        // add the collection view to the view hierarchy and set its constraints

        view.addSubview(collectionView)

        NSLayoutConstraint.activate([

            collectionView.topAnchor.constraint(equalTo: allCardsButton.bottomAnchor, constant: 24),

            collectionView.bottomAnchor.constraint(equalTo: view.bottomAnchor),

            collectionView.leftAnchor.constraint(equalTo: allCardsButton.leftAnchor),

            collectionView.rightAnchor.constraint(equalTo: view.rightAnchor)

        ])

    }

**@objc** **func** allCardsButtonClick() {

        favoritesFlag = **false**

        allCardsButton.backgroundColor = hexStringToUIColor(hex: "#405EFC")

        favoritesButton.backgroundColor = hexStringToUIColor(hex: "#242B37")

**self**.collectionView.reloadData()

    }

**@objc** **func** favoritesButtonClick() {

        favoritesFlag = **true**

        allCardsButton.backgroundColor = hexStringToUIColor(hex: "#242B37")

        favoritesButton.backgroundColor = hexStringToUIColor(hex: "#405EFC")

**self**.collectionView.reloadData()

    }

**func** saveProducts(\_ products: [Card]) {

        // Encode the products array as Data

**guard** **let** encodedData = **try**? NSKeyedArchiver.archivedData(withRootObject: products, requiringSecureCoding: **false**) **else** { **return** }

        // Save the encoded data to UserDefaults

        UserDefaults.standard.set(encodedData, forKey: "cards")

    }

**func** getProducts() -> [Card] {

        // Get the encoded data from UserDefaults

**guard** **let** encodedData = UserDefaults.standard.data(forKey: "cards") **else** {

**return** []

        }

        // Decode the data into an array of products

**guard** **let** products = **try**? NSKeyedUnarchiver.unarchiveTopLevelObjectWithData(encodedData) **as**? [Card] **else** {

**return** []

        }

**return** products

    }

**@objc** **func** addButtonTapped() {

        // Код, который будет выполняться при нажатии на кнопку с иконкой плюса

        // Например, открытие другого контроллера

**let** otherViewController = AddCardController()

        otherViewController.delegate = **self**

        otherViewController.modalPresentationStyle = .fullScreen

**self**.navigationController?.pushViewController(otherViewController, animated: **true**)

    }

    // **MARK: - UICollectionViewDataSource**

**func** numberOfSections(in collectionView: UICollectionView) -> Int {

**return** 1

    }

**func** collectionView(\_ collectionView: UICollectionView, numberOfItemsInSection section: Int) -> Int {

**if** (!favoritesFlag) {

**return** cardsArray.count

        } **else** {

**return** cardsArrayFavorites.count

        }

    }

**func** collectionView(\_ collectionView: UICollectionView, cellForItemAt indexPath: IndexPath) -> UICollectionViewCell {

**var** card: Card

**if** (!favoritesFlag) {

            card = cardsArray[indexPath.row]

        } **else** {

            card = cardsArrayFavorites[indexPath.row]

        }

**let** cell = collectionView.dequeueReusableCell(withReuseIdentifier: reuseIdentifier, for: indexPath)

        cell.backgroundColor = hexStringToUIColor(hex: "#12151B")

**let** imageViewTmp = UIImageView(image: card.image)

        imageViewTmp.layer.masksToBounds = **true**

        imageViewTmp.layer.cornerRadius = 8

        cell.contentView.addSubview(imageViewTmp)

        imageViewTmp.translatesAutoresizingMaskIntoConstraints = **false**

        imageViewTmp.leftAnchor.constraint(equalTo: cell.contentView.safeAreaLayoutGuide.leftAnchor).isActive = **true**

        imageViewTmp.rightAnchor.constraint(equalTo: cell.contentView.safeAreaLayoutGuide.rightAnchor, constant: -24).isActive = **true**

        imageViewTmp.topAnchor.constraint(equalTo: cell.contentView.topAnchor).isActive = **true**

        imageViewTmp.bottomAnchor.constraint(equalTo: cell.contentView.bottomAnchor, constant: -24).isActive = **true**

**return** cell

    }

**func** collectionView(\_ collectionView: UICollectionView, layout collectionViewLayout: UICollectionViewLayout, sizeForItemAt indexPath: IndexPath) -> CGSize

        {

                // In this function is the code you must implement to your code project if you want to change size of Collection view

**return** CGSize(width: (**self**.collectionView.frame.width - 10) / 2, height: (**self**.view.frame.height - 10) / 7)

        }

    // **MARK: - UICollectionViewDelegate**

**func** collectionView(\_ collectionView: UICollectionView, didSelectItemAt indexPath: IndexPath) {

        print("Start")

**var** card: Card

**if** (!favoritesFlag) {

            card = cardsArray[indexPath.row]

        } **else** {

            card = cardsArrayFavorites[indexPath.row]

        }

        print(cardsArray[indexPath.row])

**let** vc = ShowItemController(card: card)

        print("Next")

        vc.modalPresentationStyle = .fullScreen

        print("Push Controller")

**self**.navigationController?.pushViewController(vc, animated: **true**)

        print("Pushed")

    }

}

**extension** UIViewController {

**func** setLeftAlignTitleView(font: UIFont, text: String, textColor: UIColor) {

**guard** **let** navFrame = navigationController?.navigationBar.frame **else**{

**return**

        }

**let** parentView = UIView(frame: CGRect(x: 0, y: 0, width: navFrame.width\*3, height: navFrame.height))

**self**.navigationItem.titleView = parentView

**let** label = UILabel(frame: .init(x: parentView.frame.minX + 20, y: parentView.frame.minY, width: parentView.frame.width, height: parentView.frame.height))

        label.backgroundColor = .clear

        label.numberOfLines = 2

        label.font = font

        label.textAlignment = .left

        //titleLabel.textAlignment = .left

        label.textColor = textColor

        label.text = text

        parentView.addSubview(label)

    }

}

**extension** CardsController: AddCardDelegate {

**func** addedCard(card: Card) {

**if** (card.favorites != **false**) {

            cardsArrayFavorites.append(card)

        }

        cardsArray.append(card)

        collectionView.reloadData()

        saveProducts(cardsArray)

    }

}

**import** UIKit

**class** AddCardController: UIViewController, UIImagePickerControllerDelegate, UINavigationControllerDelegate {

**let** storeNameTextField = UITextField()

**let** barCodeTextField = UITextField()

**let** cardNumberTextField = UITextField()

**let** imageView = UIImageView()

**let** barImageView = UIImageView()

**let** favoritesCheckbox = UISwitch()

**let** saveButton = UIButton()

**weak** **var** delegate: AddCardDelegate?

**var** product = Card(storeName: "", barCode: "", cardNumber: "", image: UIImage(), favorites: **false**)

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

        imageView.layer.masksToBounds = **true**

        barImageView.layer.masksToBounds = **true**

        imageView.layer.cornerRadius = 8

        barImageView.layer.cornerRadius = 8

        setLeftAlignTitleView(font: UIFont(name: "Arial-BoldMT", size: 26)!, text: "My cards", textColor: .white)

**self**.navigationController?.view.backgroundColor = hexStringToUIColor(hex: "#12151B")

**self**.navigationController?.view.tintColor = .white

        // Set up the text fields, image picker, and checkbox

        imageView.image = UIImage(named: "customImagePlaceholder")

        barImageView.image = UIImage(named: "customImagePlaceholder")

        // Set up the save button

        saveButton.backgroundColor = hexStringToUIColor(hex: "#405EFC")

        saveButton.titleLabel?.font = UIFont(name: "Arial", size: 16)

        saveButton.titleLabel?.textAlignment = .center

        saveButton.setTitleColor(hexStringToUIColor(hex: "FFFFFF"), for: .normal)

        saveButton.layer.cornerRadius = 8

        saveButton.setTitle("Save", for: .normal)

        storeNameTextField.attributedPlaceholder = NSAttributedString(

            string: "Store name",

            attributes: [NSAttributedString.Key.foregroundColor: hexStringToUIColor(hex: "#838390")]

        )

        storeNameTextField.font = UIFont(name: "Arial", size: 16)

        storeNameTextField.layer.borderWidth = 1

        storeNameTextField.textColor = .white

        storeNameTextField.layer.borderColor = hexStringToUIColor(hex: "#3D485C").cgColor

        storeNameTextField.indent(size: 16)

        storeNameTextField.autocapitalizationType = .none

        storeNameTextField.layer.cornerRadius = 8

        barCodeTextField.attributedPlaceholder = NSAttributedString(

            string: "Bar code",

            attributes: [NSAttributedString.Key.foregroundColor: hexStringToUIColor(hex: "#838390")]

        )

        barCodeTextField.font = UIFont(name: "Arial", size: 16)

        barCodeTextField.layer.borderWidth = 1

        barCodeTextField.textColor = .white

        barCodeTextField.layer.borderColor = hexStringToUIColor(hex: "#3D485C").cgColor

        barCodeTextField.indent(size: 16)

        barCodeTextField.autocapitalizationType = .none

        barCodeTextField.layer.cornerRadius = 8

        cardNumberTextField.attributedPlaceholder = NSAttributedString(

            string: "Card number",

            attributes: [NSAttributedString.Key.foregroundColor: hexStringToUIColor(hex: "#838390")]

        )

        cardNumberTextField.font = UIFont(name: "Arial", size: 16)

        cardNumberTextField.layer.borderWidth = 1

        cardNumberTextField.textColor = .white

        cardNumberTextField.layer.borderColor = hexStringToUIColor(hex: "#3D485C").cgColor

        cardNumberTextField.indent(size: 16)

        cardNumberTextField.autocapitalizationType = .none

        cardNumberTextField.layer.cornerRadius = 8

        // Add the text fields, image picker, checkbox, and save button to the view

        view.addSubview(storeNameTextField)

        view.addSubview(barCodeTextField)

        view.addSubview(cardNumberTextField)

        view.addSubview(imageView)

        view.addSubview(barImageView)

        view.addSubview(favoritesCheckbox)

        view.addSubview(saveButton)

        // Add constraints

        storeNameTextField.translatesAutoresizingMaskIntoConstraints = **false**

        barCodeTextField.translatesAutoresizingMaskIntoConstraints = **false**

        cardNumberTextField.translatesAutoresizingMaskIntoConstraints = **false**

        imageView.translatesAutoresizingMaskIntoConstraints = **false**

        barImageView.translatesAutoresizingMaskIntoConstraints = **false**

        favoritesCheckbox.translatesAutoresizingMaskIntoConstraints = **false**

        saveButton.translatesAutoresizingMaskIntoConstraints = **false**

**let** margins = view.layoutMarginsGuide

        NSLayoutConstraint.activate([

            storeNameTextField.topAnchor.constraint(equalTo: margins.topAnchor, constant: 24),

            storeNameTextField.leftAnchor.constraint(equalTo: margins.leftAnchor, constant: 8),

            storeNameTextField.rightAnchor.constraint(equalTo: margins.rightAnchor, constant: -8),

            storeNameTextField.heightAnchor.constraint(equalToConstant: 46),

            barCodeTextField.topAnchor.constraint(equalTo: storeNameTextField.bottomAnchor, constant: 24),

            barCodeTextField.leftAnchor.constraint(equalTo: margins.leftAnchor, constant: 8),

            barCodeTextField.rightAnchor.constraint(equalTo: margins.rightAnchor, constant: -8),

            barCodeTextField.heightAnchor.constraint(equalToConstant: 46),

            cardNumberTextField.topAnchor.constraint(equalTo: barCodeTextField.bottomAnchor, constant: 24),

            cardNumberTextField.leftAnchor.constraint(equalTo: margins.leftAnchor, constant: 8),

            cardNumberTextField.rightAnchor.constraint(equalTo: margins.rightAnchor, constant: -8),

            cardNumberTextField.heightAnchor.constraint(equalToConstant: 46),

            imageView.topAnchor.constraint(equalTo: cardNumberTextField.bottomAnchor, constant: 24),

            imageView.leftAnchor.constraint(equalTo: cardNumberTextField.leftAnchor),

            imageView.heightAnchor.constraint(equalToConstant: **self**.view.frame.height / 8.5),

            imageView.widthAnchor.constraint(equalToConstant: (**self**.view.frame.width - 72) / 2),

            barImageView.topAnchor.constraint(equalTo: imageView.topAnchor),

            barImageView.rightAnchor.constraint(equalTo: cardNumberTextField.rightAnchor),

            barImageView.heightAnchor.constraint(equalToConstant: **self**.view.frame.height / 8.5),

            barImageView.widthAnchor.constraint(equalToConstant: (**self**.view.frame.width - 72) / 2),

            favoritesCheckbox.topAnchor.constraint(equalTo: imageView.bottomAnchor, constant: 24),

            favoritesCheckbox.leftAnchor.constraint(equalTo: imageView.leftAnchor),

            saveButton.centerYAnchor.constraint(equalTo: favoritesCheckbox.centerYAnchor),

            saveButton.leftAnchor.constraint(equalTo: favoritesCheckbox.rightAnchor, constant: 24),

            saveButton.rightAnchor.constraint(equalTo: view.rightAnchor, constant: -24),

            saveButton.heightAnchor.constraint(equalToConstant: 46),

            ])

**let** tapGestureRecognizer = UITapGestureRecognizer(target: **self**, action: **#selector**(didTapImageView))

        imageView.isUserInteractionEnabled = **true**

        imageView.addGestureRecognizer(tapGestureRecognizer)

**let** tapGestureRecognizerBarImage = UITapGestureRecognizer(target: **self**, action: **#selector**(didTapBarImageView))

        barImageView.isUserInteractionEnabled = **true**

        barImageView.addGestureRecognizer(tapGestureRecognizerBarImage)

        saveButton.addTarget(**self**, action: **#selector**(saveButtonTapped), for: .touchUpInside)

    }

**@objc** **func** didTapImageView() {

        // Open the image picker

**let** imagePicker = UIImagePickerController()

        imagePicker.view.tag = 1

        imagePicker.delegate = **self**

        //imagePicker.allowsEditing = true

        present(imagePicker, animated: **true**)

    }

**@objc** **func** didTapBarImageView() {

        // Open the image picker

**let** imagePicker = UIImagePickerController()

        imagePicker.view.tag = 2

        imagePicker.delegate = **self**

        //imagePicker.allowsEditing = true

        present(imagePicker, animated: **true**)

    }

**func** imagePickerController(\_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [UIImagePickerController.InfoKey : **Any**]) {

        // Dismiss the image picker

        dismiss(animated: **true**)

        // Get the selected image

**guard** **let** selectedImage = info[.originalImage] **as**? UIImage **else** {

**return**

        }

**if** (picker.view.tag == 1) {

            imageView.image = selectedImage

        } **else** {

            barImageView.image = selectedImage

        }

        // Update the image view with the selected image

    }

**@objc** **func** saveButtonTapped() {

        print("hi@")

            product.storeName = storeNameTextField.text ?? ""

            product.barCode = barCodeTextField.text ?? ""

            product.cardNumber = cardNumberTextField.text ?? ""

            product.image = imageView.image!

        product.barImage = barImageView.image!

**if** favoritesCheckbox.isOn {

            product.favorites = **true**

        }

        delegate?.addedCard(card: product)

**self**.navigationController?.popViewController(animated: **true**)

//        let vc = CardsController()

//        dismiss(animated: true, completion: {

//            vc.cardsArray.append(self.product)

//                    })

    }

}

**protocol** AddCardDelegate : **class** {

**func** addedCard(card: Card)

}

**import** UIKit

**class** ShowItemController: UIViewController {

**var** card:Card? = **nil**

**let** mainImageView = UIImageView()

**let** barImageView = UIImageView()

**let** cardNumberLabel = UILabel()

**init**(card: Card?)

    {

**self**.card = card

**super**.init(nibName: **nil**, bundle: **nil**)

    }

**required** **init**?(coder: NSCoder) {

        fatalError("init(coder:) has not been implemented")

    }

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

**self**.view.backgroundColor = hexStringToUIColor(hex: "#12151B")

        setLeftAlignTitleView(font: UIFont(name: "Arial-BoldMT", size: 26)!, text: card!.storeName, textColor: .white)

        mainImageView.image = card!.image

        barImageView.image = card!.barImage

        mainImageView.layer.masksToBounds = **true**

        barImageView.layer.masksToBounds = **true**

        mainImageView.layer.cornerRadius = 8

        barImageView.layer.cornerRadius = 8

        cardNumberLabel.text = "\(card!.barCode) \(card!.cardNumber)"

        cardNumberLabel.font = UIFont(name: "Arial-BoldMT", size: 16)

        cardNumberLabel.textColor = .white

        cardNumberLabel.textAlignment = .center

**self**.view.addSubview(mainImageView)

**self**.view.addSubview(barImageView)

**self**.view.addSubview(cardNumberLabel)

        mainImageView.translatesAutoresizingMaskIntoConstraints = **false**

        barImageView.translatesAutoresizingMaskIntoConstraints = **false**

        cardNumberLabel.translatesAutoresizingMaskIntoConstraints = **false**

        mainImageView.leftAnchor.constraint(equalTo: **self**.view.safeAreaLayoutGuide.leftAnchor, constant: 24).isActive = **true**

        mainImageView.rightAnchor.constraint(equalTo: **self**.view.safeAreaLayoutGuide.rightAnchor, constant: -24).isActive = **true**

        mainImageView.topAnchor.constraint(equalTo: **self**.view.safeAreaLayoutGuide.topAnchor, constant: 24).isActive = **true**

        mainImageView.heightAnchor.constraint(equalToConstant: **self**.view.frame.height / 4).isActive = **true**

        barImageView.leftAnchor.constraint(equalTo: mainImageView.leftAnchor).isActive = **true**

        barImageView.rightAnchor.constraint(equalTo: mainImageView.rightAnchor).isActive = **true**

        barImageView.topAnchor.constraint(equalTo: mainImageView.bottomAnchor, constant: 24).isActive = **true**

        barImageView.heightAnchor.constraint(equalToConstant: **self**.view.frame.height / 8).isActive = **true**

        cardNumberLabel.leftAnchor.constraint(equalTo: mainImageView.leftAnchor).isActive = **true**

        cardNumberLabel.rightAnchor.constraint(equalTo: mainImageView.rightAnchor).isActive = **true**

        cardNumberLabel.topAnchor.constraint(equalTo: barImageView.bottomAnchor, constant: 24).isActive = **true**

    }

    /\*

    // **MARK: - Navigation**

    // In a storyboard-based application, you will often want to do a little preparation before navigation

    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {

        // Get the new view controller using segue.destination.

        // Pass the selected object to the new view controller.

    }

    \*/

}

Схема БД

*Схема базы данных*

## **Диаграммы вариантов использования**https://lh6.googleusercontent.com/qJr-Mhmwm3nueFW3MvgA9SBgvR2xsQJnzZuzAFA4xG-Ve0rUzy18o2Ri7M7zmBDPwbyULWf1OPWOBY9zf2fQIZ5r3iJuLPPT7oFOS3owwyYQQpTY8BlovnBgOToieH1hEVhM0xHWMsXta7khOx5bNKpVIQuKlssece7B_Ba_vRyxMEaE5uLicqs__KfO

Основной функционал приложения

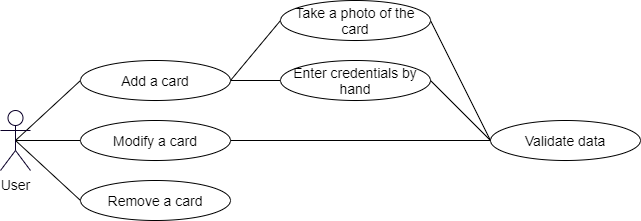


Диаграмма работы пользователя со своими картами

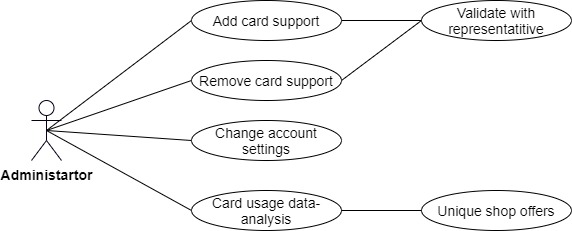


Диаграмма работы администратора с магазинами и их дисконтными картами

## **Бизнес-диаграмма**

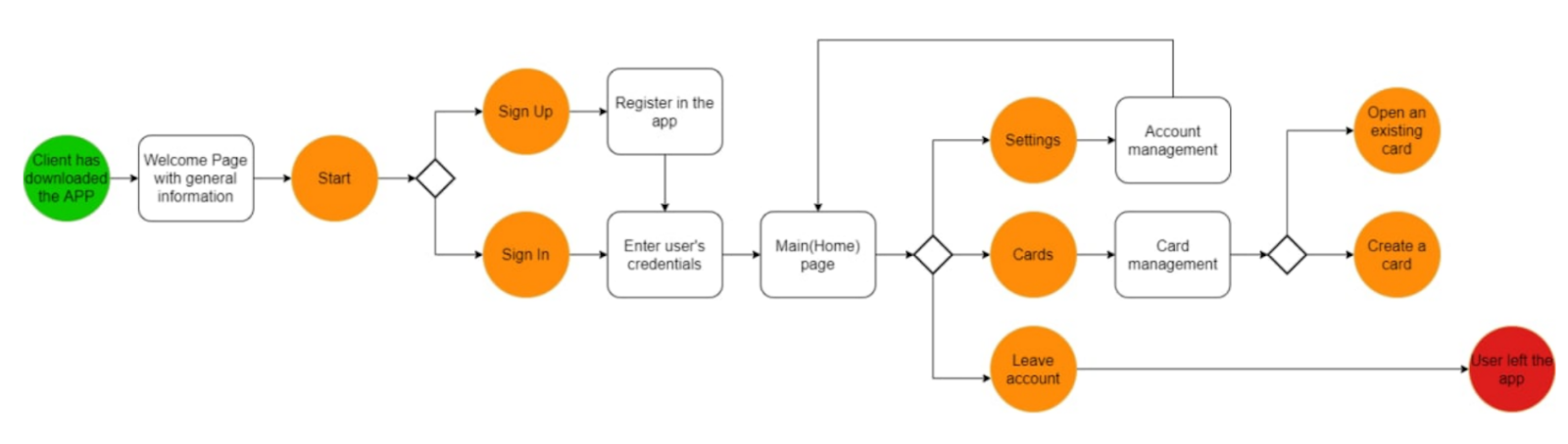


Диаграмма бизнес использования приложения

