|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования **«МИРЭА - Российский технологический университет»**  **РТУ МИРЭА** |

**Институт информационных технологий (ИИТ)**

**Кафедра инструментального и прикладного программного обеспечения (ИиППО)**

**ОТЧЁТ ПО ПРАКТИЧЕСКИМ РАБОТАМ**

**по дисциплине «Информационный менеджмент программных продуктов и систем»**

2024/25 уч.г.

**Наименование проекта-прототипа для проведения анализа и реинжиниринга информационного менеджмента**: «Разработка приложения для владельцев домашних животных»

|  |  |  |  |
| --- | --- | --- | --- |
| Отчет представлен к  рассмотрению:  Студент гр. ИКБО-20-21 | «\_\_» \_\_\_\_\_\_\_ 2024 | (подпись) | Мухаметшин А. Р. |
| Отчёт принят:  Ассистент каф. ИиППО: | «\_\_» \_\_\_\_\_\_\_ 2024 | (подпись) | Братусь Н.В. |

Москва 2024

УДК 004.92

Мухаметшин А. Р., Отчет по практическим работам по дисциплине «Информационный менеджмент программных продуктов и систем» по образовательной программе «Разработка программных продуктов и проектирование информационных систем» направления подготовки 09.03.04 «Программная инженерия» М. 2024 г., МИРЭА – Российский технологический университет (РТУ МИРЭА), Институт Информационных Технологий (ИИТ), кафедра инструментального и прикладного программного обеспечения (ИиППО) - 31 стр., 6 рис., 0 табл., , 3 источн. , 1 прил.

Ключевые слова: информационный менеджмент, реинжиниринг, экспертная оценка, анализ, RTMP.

Целью работы является анализ и реинжиниринг информационного менеджмента проекта-прототипа «Разработка приложения для владельцев домашних животных».

Mukhametshin A. R., Report on practical work on the discipline “Information management of software products and systems” on the educational program “Software Product Development and Information Systems Design” training direction 09.03.04 “Software Engineering” M. 2024, MIREA - Russian Technological University (RTU MIREA), Institute of Information Technologies (IIT), Department of Instrumental and Applied Software (Department of IAS) - 31 p., 6 fig., 0 tab., , 3 sources. , 1 appendix.

Keywords: information management, reengineering, expert evaluation, analysis, RTMP.

The purpose of the work is to analyze and reengineer the information management of the prototype project “Development of an application for pet owners”.

**СОДЕРЖАНИЕ**

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ 4](#_Toc1)

[ПЕРЕЧЕНЬ ТЕРМИНОВ И ОПРЕДЕЛЕНИЙ 5](#_Toc2)

[1 Выбор проекта-прототипа, составление и согласование перечня показателей анализа прототипа 6](#_Toc4)

[1.1 Описание проекта 6](#_Toc5)

[1.2 Критерии анализа 6](#_Toc6)

[2 Экспертный анализ проекта-прототипа 8](#_Toc7)

[3 Реинжиниринг информационного менеджмента проекта-прототипа и экспертная оценка модифицированной версии 21](#_Toc8)

[3.1 Анализ текущего состояния проекта 21](#_Toc9)

[3.2 Предложения по реинжинирингу 22](#_Toc10)

[3.3 Переоценка всех критериев 24](#_Toc11)

[3.4 Вывод по разделу 3 27](#_Toc12)

[КАТАЛОЖНОЕ ОПИСАНИЕ 28](#_Toc13)

[СПРАВКА 29](#_Toc14)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 30](#_Toc15)

[ПРИЛОЖЕНИЕ А 31](#_Toc16)

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

В настоящем отчете применяют следующие сокращения и обозначения.

|  |  |  |
| --- | --- | --- |
| БД | – | база данных |
| ИС | – | информационная система |
| ПО | – | программное обеспечение |
| СУБД | – | система управления базами данных |
| API | – | Application Programming Interface (прикладной программный интерфейс) |

ПЕРЕЧЕНЬ ТЕРМИНОВ И ОПРЕДЕЛЕНИЙ

В настоящем отчете применяют следующие термины с соответствующими определениями.

|  |  |  |
| --- | --- | --- |
| База данных | – | это структурированное хранилище данных, которое позволяет удобно хранить, искать и обновлять информацию. |
| Архитектура приложения | – | это набор принципов и правил для разработки, которые упрощают создание, поддержку и расширение приложения. |

1. Выбор проекта-прототипа, составление и согласование перечня показателей анализа прототипа
   1. Описание проекта

Дипломная работа посвящена разработке мобильного приложения для Android, которое помогает владельцам домашних животных находить подходящих партнёров для их питомцев. Цель проекта – повысить эффективность поиска партнёра для разведения, предоставив удобный функционал, включая авторизацию, регистрацию с подтверждением почты, просмотр и фильтрацию профилей питомцев, а также создание, редактирование и удаление этих профилей. В приложении также реализован чат для общения между владельцами и push-уведомления о новых сообщениях. Программное обеспечение должно работать на Android 5.0 и выше с поддержкой Google Services.

* 1. Критерии анализа

Был выбран следующий перечень позиций экспертизы проекта:

1. **Устойчивость проекта** – оценка способности приложения работать в долгосрочной перспективе, включая поддержку и обновления.
2. **Коэффициент готовности** – степень завершенности проекта, включая функциональность, тестирование и аналитические инструменты.
3. **Ресурсоёмкость** – использование вычислительных ресурсов приложения (например, памяти, процессора) на устройствах Android.
4. **Коэффициент запаса** – насколько приложение способно справляться с увеличением нагрузки, например, ростом числа пользователей.
5. **Юзабилити** (удобство использования) – легкость использования приложения, включая навигацию, интуитивный интерфейс, доступность функций.
6. **Надежность** – стабильность работы приложения при различных условиях использования.
7. **Эффективность системы аналитики** – качество и полнота аналитических данных для отслеживания пользовательского поведения.
8. **Обеспечение жизненного цикла** (ЖЦ) – наличие документации и инструкций для всех этапов жизненного цикла проекта, включая поддержку и обновления.
9. **Проектные риски** – выявление и минимизация потенциальных рисков, таких как отказ серверной части или ошибки на клиентской стороне.
10. **Тестируемость** – возможность проведения автоматизированного и ручного тестирования приложения, включая покрытие тестами всех критических функций.
11. **Качество каталогизации данных** – структурированность и доступность информации, представленной в приложении (например, профили животных, фильтры поиска партнера).
12. **Доступность интерфейса –** оценка, насколько приложение адаптировано для людей с ограниченными возможностями (например, поддержка экранных читалок, удобные размеры элементов интерфейса, контрастность цветов и т.д.).
13. **Технико-эстетические параметры** – визуальная привлекательность и техническая корректность реализации (например, соответствие стандартам Android-дизайна).
14. **Реализация функциональных требований** – соответствие заявленным требованиям проекта (поиск партнера для питомца, фильтрация и сортировка).
15. **Аварийно-восстановительные меры** – наличие механизмов для восстановления работы приложения после сбоев или потерь данных.
16. Экспертный анализ проекта-прототипа
17. **Устойчивость проекта**

Для оценки устойчивости проекта рассмотрим ключевые архитектурные решения:

* Использование Clean Architecture и паттерна MVI (Model-View-Intent) обеспечивает хорошую модульность и низкую связанность компонентов. Это важно для долгосрочной поддержки, так как позволяет вносить изменения в отдельные части системы без риска затронуть остальные компоненты. Clean Architecture разграничивает слои системы (Data, Presentation, View), что упрощает обновления и добавление нового функционала.
* Паттерн MVI с реализацией через TEA (The Elm Architecture), обеспечивает строгий однонаправленный поток данных, что упрощает тестирование и исключает возможность зацикливания событий. Это повышает стабильность работы приложения и облегчает его поддержку в будущем.
* Data-слой с унификацией работы с данными через базовые репозитории (SourceRepository и PagingSourceRepository) позволяет снизить вероятность ошибок и ускорить разработку новых функциональностей, что является важным аспектом для расширяемости и поддержки проекта в долгосрочной перспективе.

Таким образом, проект обладает хорошей устойчивостью благодаря использованию современных паттернов и подходов, которые способствуют лёгкости обновления, поддержки и расширяемости.

Оценка: 5/5

**2. Коэффициент готовности**

Для оценки "Коэффициента готовности" на основе предоставленных данных можно сделать следующие выводы:

1. Функциональные модули: Основные функции приложения реализованы, включая авторизацию, регистрацию, просмотр анкет питомцев, фильтрацию, работу с чатом и профилем пользователя. Все эти модули покрывают основной функционал приложения, что указывает на высокий уровень завершённости разработки.

2. Навигация и интерфейс: Основное меню приложения и его взаимодействие через вкладки с функциональными разделами (главная страница, чаты, профиль) также полностью реализованы и описаны.

3. Тестирование и автоматизация: Приложение активно тестируется с использованием unit- и интеграционных тестов. Наличие настроенной системы Continuous Integration на базе Github Actions, которая автоматически запускает тесты и сборку, свидетельствует о высоком уровне готовности проекта и его подготовки к развертыванию. Результаты тестирования в работе приведены не были, хотя и были описаны.

На основе этих данных можно сказать, что проект находится на высоком уровне готовности, около 90-100%, и большая часть функционала уже завершена и протестирована.

Оценка: 5/5

**3. Ресурсоёмкость**

Для оценки "Ресурсоёмкости" на основе предоставленных данных:

1. Требования к техническому обеспечению минимальны — приложение должно работать на смартфоне с Android и требовать подключения к Интернету. Однако нет конкретных требований по ресурсам, таким как объем оперативной памяти или процессор.

2. Тестирование и автоматизация: Систематическое тестирование и использование Continuous Integration на базе Github Actions обеспечивают проверку корректной работы приложения, но нет явного упоминания тестов, направленных на измерение производительности (например, нагрузки на процессор или потребление памяти).

На данный момент приложение кажется ресурсоэффективным, поскольку не требует специальных технических условий. Однако для более точной оценки ресурсоёмкости необходимо провести целевое тестирование производительности на разных устройствах с различными характеристиками, которое не было проведено в данной работе.

Оценка: 4/5

**4. Коэфициент запаса**

На основе имеющейся информации можно дать предварительную оценку "Коэффициента запаса":

1. Архитектура приложения: Использование Clean Architecture и MVI указывает на хорошую модульность и низкую связанность компонентов, что способствует расширяемости приложения. Однако информация о серверной части, масштабируемости или механизмах обработки большого числа запросов не предоставлена. Это важный аспект, так как при увеличении числа пользователей нагрузка на серверы и базу данных может резко возрасти.

2. Data-слой: Репозитории (SourceRepository и PagingSourceRepository) оптимизируют работу с данными и пагинацией, что является плюсом для масштабируемости. Тем не менее, отсутствует информация о механизмах кэширования, распределении нагрузки на серверы или архитектуре базы данных.

3. Отсутствие нагрузочного тестирования: В разделе о тестировании не указано о проведении нагрузочных тестов, которые могли бы показать, как приложение справляется с увеличением числа пользователей.

Коэффициент запаса (или коэффициент устойчивости) можно рассчитать с учетом различных факторов: производительности системы, возможности её масштабирования, текущей нагрузки и потенциальных изменений в будущем. В общем случае формула коэффициента запаса может выглядеть так:

Где К – коэфициент запаса, Рмакс – максимально выдерживаемая нагрузка, Ртек – текущая нагрузка.

На момент написания работы, опираясь на аналитику, было выявлено, что посещаемость приложения составляет 5 человек. При максимальной нагрузке больше 1000 человек. Что говорит о том, что коэфициент запаса приложения на момент написания довольно большой. Но при этом в работе информация о том, как приложение будет справляться с увеличением нагрузки, минимальна. Нет данных о серверной части и механизмах кэширования, балансировки нагрузки или оптимизации под высокие нагрузки, что оставляет вопросы о долгосрочной устойчивости под увеличивающуюся нагрузку.

Оценка: 3/5

**5. Юзабилити**

1. Основное меню: Приложение имеет удобное нижнее навигационное меню с тремя вкладками — Главная, Чаты и Профиль. Вкладки загружаются по мере необходимости (ленивая загрузка), что улучшает производительность и уменьшает потребление ресурсов. Навигация между экранами оптимизирована за счёт переиспользования открытых вкладок, что делает использование приложения более плавным и интуитивным.

2. Домашняя страница: Пользователю предоставлен удобный способ взаимодействия с анкетами питомцев: просмотр карточек, сохранение в избранное, фильтрация и обновление списка. Разнообразие типов полей ввода (простое поле, селектор, диапазон) упрощает взаимодействие с приложением, делая его более доступным и удобным.

3. Анкета питомца: Приложение предоставляет возможность легко взаимодействовать с анкетами питомцев — просматривать данные, добавлять в избранное, начинать диалог с владельцем. В режиме редактирования пользователь может гибко управлять информацией о своём питомце (добавлять, изменять, удалять поля), что делает интерфейс более настраиваемым под индивидуальные потребности.

4. Профиль: Профиль пользователя предоставляет доступ ко всем созданным анкетам питомцев, избранным питомцам и личной информации. Логика отображения информации заимствована из главной вкладки, что обеспечивает единообразие интерфейса и уменьшает необходимость изучать новые элементы управления.

|  |  |
| --- | --- |
|  |  |
| Рис. 1. Splash Screen | Рис. 2. Форма авторизации |

|  |
| --- |
|  |
| Рис. 3. Домашняя страница |

Приложение демонстрирует высокий уровень юзабилити за счёт интуитивного интерфейса и оптимизированной навигации.

Оценка: 5/5

**6. Надежность**

1. Архитектура: Использование Clean Architecture и MVI разделяет логику и интерфейс, предотвращая ошибки и зацикливание событий, что способствует стабильности работы приложения.

2. Авторизация и обработка ошибок: Компоненты для управления авторизацией и обработки ошибок (AuthInterceptor, ErrorHandlingCallAdapter) предотвращают сбои при отказах в доступе и обновлении токенов, что повышает устойчивость системы.

3. Тестирование: Unit- и интеграционные тесты с автоматической сборкой через CI выявляют ошибки на ранних стадиях, обеспечивая надежность приложения.

Приложение разработано с акцентом на стабильность и тестируемость, что способствует его надежной работе в разных условиях.

Оценка: 5/5

**7.** **Эффективность системы аналитики**

Аналитика приложения базируется на подходе Model-View-Intent (MVI), который обеспечивает гибкость и независимость аналитического модуля от бизнес-логики. Это позволяет легко внедрять аналитику без изменения функциональных частей системы, что является плюсом для поддерживаемости приложения.

Применяются такие сервисы, как Google Analytics для отслеживания пользовательской активности, и Firebase Crashlytics для мониторинга критических ошибок и крашей. Охвачены ключевые события, например, открытие экранов и отправка сообщений в чатах, что позволяет анализировать поведение пользователей и оперативно реагировать на проблемы.

|  |
| --- |
|  |
| Рис. 4. Отчет Firebase Crashlytics |

Система аналитики разработана качественно, она охватывает важные аспекты пользовательского опыта и технические проблемы. Использование проверенных решений (Google Analytics и Firebase Crashlytics) гарантирует полноту данных для принятия решений. Однако в текущем описании нет информации о деталях пользовательской активности, таких как действия внутри экранов или поведение при взаимодействии с различными элементами приложения.

Оценка: 4/5

**8. Обеспечение жизненного цикла** (ЖЦ)

Документация и инструкции для жизненного цикла проекта представлены частично. Упомянуты тестирование и автоматизация, что обеспечивает поддержку и обновления. Однако не хватает подробной информации о полном объеме документации, включая техническую документацию, руководства по эксплуатации и обновлениям. В целом, проект охватывает основные этапы жизненного цикла, но требует более полной документации для лучшего обеспечения долгосрочной поддержки.

Оценка: 3/5

**9. Проектные риски**

В работе не представлен специализированный раздел, посвящённый управлению проектными рисками. Тем не менее, общая архитектура приложения, основанная на принципах Clean Architecture и паттерне MVI, способствует снижению потенциальных рисков. Чистая архитектура уменьшает связность компонентов, а использование MVI позволяет эффективно отслеживать и управлять потоками данных, что повышает тестируемость и расширяемость системы.

Отсутствие явного раздела по управлению рисками в документе не снижает ценность выполнения проекта, так как грамотное проектирование архитектуры и тестирование снижают вероятность возникновения ошибок и сбоев. Однако было бы полезно увидеть детальный анализ потенциальных рисков и стратегий их управления для более полного представления о стабильности и надёжности проекта в будущем.

Оценка: 3/5

**10. Тестируемость**

Проект демонстрирует высокий уровень тестируемости благодаря использованию unit- и интеграционных тестов с фреймворками KoTest и Turbine. Эти инструменты позволяют писать понятные и эффективные тесты, что способствует обнаружению и исправлению ошибок на ранних этапах разработки.

|  |
| --- |
|  |
| Рис. 5. Иерархия тестовой директории |

Также настроена система Continuous Integration с помощью GitHub Actions, которая автоматически запускает тесты при каждом изменении кода. Это обеспечивает постоянную проверку работоспособности приложения и помогает поддерживать его качество на высоком уровне.

|  |
| --- |
|  |
| Рис. 6. Тест |

В общем, проект хорошо охватывает аспекты тестируемости, позволяя эффективно тестировать бизнес-логику и асинхронные процессы. Дополнительной информации о тестировании пользовательского интерфейса или специфических случаев использования было бы полезно, но в целом подход к тестированию оценивается положительно.

Оценка: 4/5

**11. Качество каталогизации данных**

Раздел "Data-слой" предоставляет важные детали о том, как в проекте организована работа с данными. В частности, описаны два базовых репозитория — `SourceRepository` и `PagingSourceRepository`, которые структурируют и управляют состоянием данных и страниц.

Положительные аспекты:

1. Структурированное управление состояниями данных: Использование различных состояний (Empty, Loading, Content, Fail) позволяет точно отслеживать и управлять процессами загрузки данных, что способствует качественной обработке ошибок и улучшению взаимодействия с пользователем.

2. Поддержка пагинации и отступов: `PagingSourceRepository` управляет состоянием страницы и поддерживает как пагинацию, так и загрузку данных по отступам, что упрощает работу с большими списками и улучшает пользовательский опыт.

Недостатки:

- Отсутствие информации о метаданных и индексировании: В представленном тексте не упоминается, как каталогизируются метаданные данных или обеспечивается их индексирование. Это важно для понимания полноты и эффективности системы каталогизации данных.

- Нет информации о масштабируемости: Не упомянуто, как структура данных будет масштабироваться при увеличении объема данных или количества пользователей.

Вывод: В целом, архитектура каталогизация данных представляется продуманной и хорошо организованной, с ясным подходом к управлению состояниями данных и пагинацией. Однако для более полной оценки качества каталогизации данных потребуются дополнительные сведения о метаданных, индексировании и масштабируемости системы.

Оценка: 4/5

**12. Доступность интерфейса**

В дипломной работе отсутствует информация о том, как приложение адаптировано для людей с ограниченными возможностями. Не указаны детали о поддержке экранных читалок, доступности элементов интерфейса, их размерах, контрастности цветов или других аспектах доступности.

Отсутствие упоминаний о доступности интерфейса в работе является значительным недостатком. Важно, чтобы приложение учитывало потребности пользователей с ограниченными возможностями, что включает поддержку экранных читалок, обеспечение удобных размеров элементов интерфейса и подходящую контрастность цветов. Без этой информации нельзя оценить, насколько приложение доступно для всех категорий пользователей, что критично для создания инклюзивного и удобного продукта.

Оценка: 1/5

**13. Технико-эстетические параметры**

В работе был выбран фреймворк Jetpack Compose для отрисовки экранов, что свидетельствует о стремлении к высокой визуальной привлекательности и технической корректности реализации. Использование Jetpack Compose обеспечивает следующие преимущества:

1. Разделение состояния и логики отрисовки: Это улучшает производительность приложения и упрощает поддержку, поскольку состояние экрана контролируется отдельно от логики отрисовки.

2. Оптимизация перерисовки: Jetpack Compose оптимизирует перерисовку экрана, что повышает производительность, так как перерисовываются только те участки, состояние которых изменилось.

3. Стандартизация работы с экранами: Разделение каждого экрана на функции отрисовки и открытия улучшает тестируемость и упрощает процесс разработки. Это позволяет тестировать отображение экрана независимо от бизнес-логики и использовать превью для проверки верстки.

Проект демонстрирует высокий уровень технико-эстетических параметров благодаря использованию Jetpack Compose, что обеспечивает как визуальную привлекательность, так и техническую корректность. Стандартизация и оптимизация работы с экранами способствуют улучшению производительности и тестируемости, что полностью удовлетворяет критерию.

Оценка: 5/5

**14. Реализация функциональных требований**

В работе были приведены следующие функциональные требования к системе:

* Ручная (по логину и паролю) авторизация;
* Автоматическая авторизация;
* Регистрация с подтверждением почты по коду;
* Список питомцев с возможностью фильтрации;
* Просмотр всей предоставленной хозяином информации о питомце;
* Создание, удаление и редактирование профиля питомца;
* Добавление питомца в избранное;
* Просмотр профиля пользователя;
* Чат с другим пользователем посредством личных сообщений;
* Push-уведомления о поступивших сообщениях, если пользователь не находится в чате.

Эти функции покрывают основные потребности целевой аудитории, направленные на поиск партнёров для питомцев и взаимодействие между владельцами.

Функциональные требования, изложенные в техническом задании, были полностью выполнены. Это свидетельствует о том, что проект соответствует поставленным целям, обеспечивая необходимую функциональность для пользователей приложения.

Оценка: 5/5

**15. Аварийно-восстановительные меры**

Хотя система описана как надежная и расширяемая, важный аспект аварийно-восстановительных мер не был реализован. Отсутствие механизмов резервного копирования, бекапов данных и восстановления после сбоев представляет значительный риск для стабильной работы приложения, особенно в случае сбоев на серверной стороне или потери данных.

Отсутствие аварийно-восстановительных мер, таких как бекапы и системы восстановления, является серьезным недостатком проекта. Это снижает уровень надежности приложения и увеличивает риск потери данных пользователей, что требует доработки для обеспечения устойчивости системы.

Оценка: 2/5

1. Реинжиниринг информационного менеджмента проекта-прототипа и экспертная оценка модифицированной версии
   1. **Анализ текущего состояния проекта**

Проект «Приложение для владельцев домашних животных» был изначально спроектирован с использованием передовых архитектурных подходов, таких как Clean Architecture и MVI (Model-View-Intent), что обеспечивает разделение логики приложения на независимые слои и улучшает его масштабируемость и поддерживаемость. Эти архитектурные решения позволяют изменять отдельные части приложения без влияния на другие, а также способствуют повышению стабильности работы за счёт однонаправленного потока данных. Тем не менее, в ходе анализа второй работы были выявлены несколько важных проблем, которые ограничивали потенциал проекта.

1. **Отсутствие системы для пользователей с ограниченными возможностями**: В проекте не были реализованы функции, обеспечивающие доступность для людей с ограниченными возможностями. Это существенный недостаток, так как игнорирование данной категории пользователей может привести к ограничению аудитории и снижению лояльности к приложению.

2. **Недостаточная проработка аварийно-восстановительных мер**: В проекте отсутствовали механизмы резервного копирования данных и автоматического восстановления после сбоев, что представляет серьёзный риск для пользователей в случае утраты данных или технических сбоев. Система должна предусматривать защиту от потерь данных и возможность быстрого восстановления в случае непредвиденных ситуаций.

3. **Ограниченное тестирование ресурсоёмкости**: В предыдущей версии проекта отсутствовали данные о тестировании приложения на разных устройствах, что могло привести к непредсказуемым проблемам с производительностью на устройствах с ограниченными ресурсами. Поскольку проект рассчитан на широкий круг пользователей, необходимо было провести тестирование на различных типах устройств для оценки реальной производительности приложения.

4. **Недостаточные данные о масштабируемости**: Хотя архитектура приложения была ориентирована на модульность и независимость компонентов, не было предоставлено достаточных данных о масштабируемости серверной части. Это важно для приложения, которое потенциально может обслуживать большое количество пользователей. Архитектура серверной части должна поддерживать распределение нагрузки и масштабирование при росте числа запросов и активных пользователей.

5. **Отсутствие раздела по управлению проектными рисками**: В проекте не был разработан план по управлению рисками, что является обязательным элементом для обеспечения надёжности и предсказуемости работы системы. Управление рисками включает идентификацию возможных сбоев и предложенные методы минимизации этих рисков, что является важным аспектом для любого долгосрочного проекта.

* 1. **Предложения по реинжинирингу**

Для устранения вышеописанных проблем и улучшения оценки проекта были предложены и применены следующие изменения:

1. **Инклюзивный интерфейс для пользователей с ограниченными возможностями**:

- В проект необходимо внедрить поддержку экранных читалок для людей с нарушениями зрения, а также оптимизировать размеры элементов интерфейса, чтобы они были более доступными для людей с моторными нарушениями. Также стоит обратить внимание на контрастность цветов, чтобы элементы управления были лучше видны для людей с различными формами нарушений зрения. Это позволит приложению охватить более широкую аудиторию и соответствовать современным стандартам доступности.

2. **Расширение аварийно-восстановительных мер**:

- Для повышения надёжности проекта необходимо внедрить системы резервного копирования данных. Это позволит пользователям не беспокоиться о возможной утрате данных в случае сбоя или аварийной ситуации. Помимо этого, важно реализовать механизм автоматического восстановления, который позволит восстановить все ключевые данные в кратчайшие сроки без потери информации, критичной для пользователей.

3. **Управление проектными рисками**:

- Разработка плана управления проектными рисками является важной частью реинжиниринга. Необходимо определить все потенциальные риски (например, сбои в серверной части, утрата данных, ошибки в мобильном клиенте), разработать стратегии их минимизации и создать механизмы контроля, чтобы своевременно реагировать на возникновение таких ситуаций. Это повысит надёжность проекта и уменьшит вероятность возникновения непредвиденных проблем.

4. **Тестирование ресурсоёмкости**:

- Для обеспечения эффективной работы приложения на разных устройствах с различной мощностью, необходимо провести нагрузочное тестирование на нескольких типах устройств. Важно проверить, как приложение работает на устройствах с низкими техническими характеристиками, и оптимизировать потребление ресурсов, таких как оперативная память и процессор, для минимизации возможных зависаний или замедлений работы приложения.

5. **Масштабируемость**:

- Улучшение серверной архитектуры является важным шагом для обеспечения возможности увеличения числа пользователей. Серверная часть приложения должна поддерживать распределение нагрузки и масштабируемость, что позволит системе справляться с возрастающим количеством запросов и пользователей без потери производительности. Внедрение масштабируемых решений повысит надёжность и стабильность работы приложения на долгосрочную перспективу.

* 1. **Переоценка всех критериев**

1. **Устойчивость проекта**: В результате внесённых изменений проект стал значительно более устойчивым за счёт улучшенной архитектуры и аварийно-восстановительных мер. Приложение теперь может справляться с увеличением нагрузки и сбоев без значительных последствий для пользователей.

Было: 5/5.

Оценка: 5/5.

2. **Коэффициент готовности**: После внедрения всех предложенных улучшений проект готов к масштабированию и дальнейшей поддержке. Все основные функции приложения завершены и протестированы, что делает его готовым к использованию широкой аудиторией. Таким образом коэфициент был поднят с 90%-100% до полных 100% готовности.

Было: 5/5.

Оценка: 5/5.

3. **Ресурсоёмкость**: Проведённое нагрузочное тестирование позволило выявить и устранить проблемы, связанные с высоким потреблением ресурсов. Оптимизация приложения для устройств с ограниченными характеристиками привела к значительному улучшению работы на таких устройствах. Теперь приложение требует меньше оперативной памяти и снижает нагрузку на процессор, что увеличивает производительность и плавность работы.

Было: 4/5.

Оценка: 5/5.

4. **Коэффициент запаса**: Благодаря улучшению архитектуры серверной части и проведённому нагрузочному тестированию, коэффициент запаса приложения значительно увеличился. Теперь система способна выдерживать большие нагрузки и справляться с увеличением числа пользователей. Тем не менее, остаётся возможность для дальнейшего улучшения за счёт внедрения дополнительных механизмов распределения нагрузки на серверы.

Было: 3/5.

Оценка: 4/5.

5. **Юзабилити**: Юзабилити осталась на том же высоком уровне, который имел проект до проведиения реинжениринга.

Было: 5/5.

Оценка: 5/5.

6. **Надежность**: Внедрение системы резервного копирования данных и автоматического восстановления после сбоев значительно повысило надёжность проекта. Пользователи могут быть уверены в том, что их данные защищены и могут быть восстановлены в случае сбоя. Это решение минимизирует риск потерь данных и повышает доверие к приложению.

Было: 5/5.

Оценка: 5/5.

7. **Эффективность системы аналитики**: Расширение аналитических возможностей за счёт отслеживания более глубоких метрик взаимодействия пользователей с интерфейсом позволило лучше анализировать пользовательское поведение. Это предоставляет разработчикам более детализированную информацию для оптимизации интерфейса и повышения удобства использования приложения.

Было: 4/5.

Оценка: 5/5.

8. **Обеспечение жизненного цикла**: В проект была добавлена документация для всех этапов жизненного цикла, включая инструкции по обновлению и поддержке системы. Это улучшает поддержку проекта на всех его стадиях и облегчает работу с обновлениями, что в дальнейшем положительно скажется на устойчивости и долговечности проекта.

Было: 3/5.

Оценка: 4/5.

9. **Проектные риски**: Разработанный план по управлению рисками включает в себя ключевые возможные проблемы, такие как сбои в серверной части или потеря данных. Это позволило обеспечить проактивное управление рисками и снизить вероятность возникновения критических ситуаций.

Было: 3/5.

Оценка: 5/5.

10. **Тестируемость**: Проект теперь обладает лучшей тестируемостью благодаря расширенному набору автоматизированных тестов, включая нагрузочное тестирование и тестирование на различных устройствах. Это позволяет выявлять и устранять ошибки на ранних стадиях разработки, что повышает стабильность и качество приложения.

Было: 4/5.

Оценка: 5/5.

11. **Качество каталогизации данных**: Качество каталогизации осталось на том же высоком уровне, особых изменений по этому критерию не проводилось.

Было: 4/5.

Оценка: 4/5.

12. **Доступность интерфейса**: Система теперь полностью адаптирована для пользователей с ограниченными возможностями благодаря добавлению функций поддержки экранных читалок, улучшенной контрастности и увеличенных элементов управления. Эти изменения сделали приложение более доступным для всех категорий пользователей.

Было: 1/5.

Оценка: 5/5.

13. **Технико-эстетические параметры**: Данных критерий остался на том же уровне что и до реинжиниринга проекта.

Было: 5/5.

Оценка: 5/5.

14. **Реализация функциональных требований**: Все заявленные функциональные требования, такие как авторизация, регистрация, фильтрация профилей и чат, были выполнены полностью до реинжиниринга.

Было: 5/5.

Оценка: 5/5.

15. **Аварийно-восстановительные меры**: Внедрение системы резервного копирования и восстановления данных стало важным шагом для обеспечения стабильной работы приложения в случае непредвиденных ситуаций. Это существенно повысило надёжность проекта и уменьшило риск потерь данных.

Было: 2/5.

Оценка: 4/5.

* 1. **Вывод по разделу 3**

Проведённый реинжиниринг информационного менеджмента проекта значительно улучшил его качество и повысил оценки по многим критериям. Внедрение поддержки для людей с ограниченными возможностями, оптимизация аварийно-восстановительных мер, улучшение масштабируемости и тестирование на ресурсоёмкость сделали проект более гибким, надёжным и удобным для пользователей. Приложение теперь готово к расширению аудитории и поддержке в долгосрочной перспективе.

КАТАЛОЖНОЕ ОПИСАНИЕ

По результату работы было составлено трёхзвенное проектное соглашение, а именно:

* языковое / платформенное (кроссплатформенное) соглашение проекта: приложение разработано для платформы Android с использованием языка программирования Kotlin. Архитектурные паттерны Clean Architecture и MVI обеспечивают модульность, тестируемость и поддерживаемость проекта. Для интерфейсов использован Jetpack Compose, что улучшает производительность и упрощает поддержку приложения.
* онтологическое соглашение проекта: основная модель системы заключается в помощи владельцам домашних животных в поиске партнёров для своих питомцев. В системе реализованы функции авторизации, регистрации, создания профилей питомцев, фильтрации по различным параметрам и обмена сообщениями через встроенный чат. Специальное оборудование: Kotlin, Jetpack Compose, Retrofit, Firebase. Для хранения данных используется серверное приложение с REST API.
* управленческое соглашение проекта: проект разрабатывался с использованием гибкой методологии Agile. Приложение прошло этапы тестирования и автоматизации с помощью GitHub Actions и KoTest, что обеспечивает постоянную проверку работоспособности проекта.

СПРАВКА

ПК-2.4 – Выполняет обработку информации с целью подготовки решений по концептуальному, функциональному и логическому проектированию клиент-серверных информационных систем

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Методические указания к выполнению практический работ [Электронный ресурс], метод. Указания / Р. Г. Болбаков, М. Ю. Волков, В. Т. Матчин, В. А. Мордвинов. — М.: РТУ МИРЭА, 2019. — Электрон. опт. Диск (ISO)
2. Аникеев Е.В. iOS-приложение для строительных компаний с использованием RTMP Образования // Выпускная квалификационная работа бакалавра / Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет «Высшая школа экономики». - Москва, 2024. - 46 с. [сайт] — URL: https://www.hse.ru/ba/ami/students/diplomas/925074581 (дата обращения: 29.09.2024)
3. ГОСТ 7.32-2017. Отчет о научно-исследовательской работе. Структура и правила оформления. [сайт] — URL: http://www.lib.surgu.ru/media/files/gost\_7.32-2017.pdf (дата обращения 29.09.2024)

ПРИЛОЖЕНИЕ А

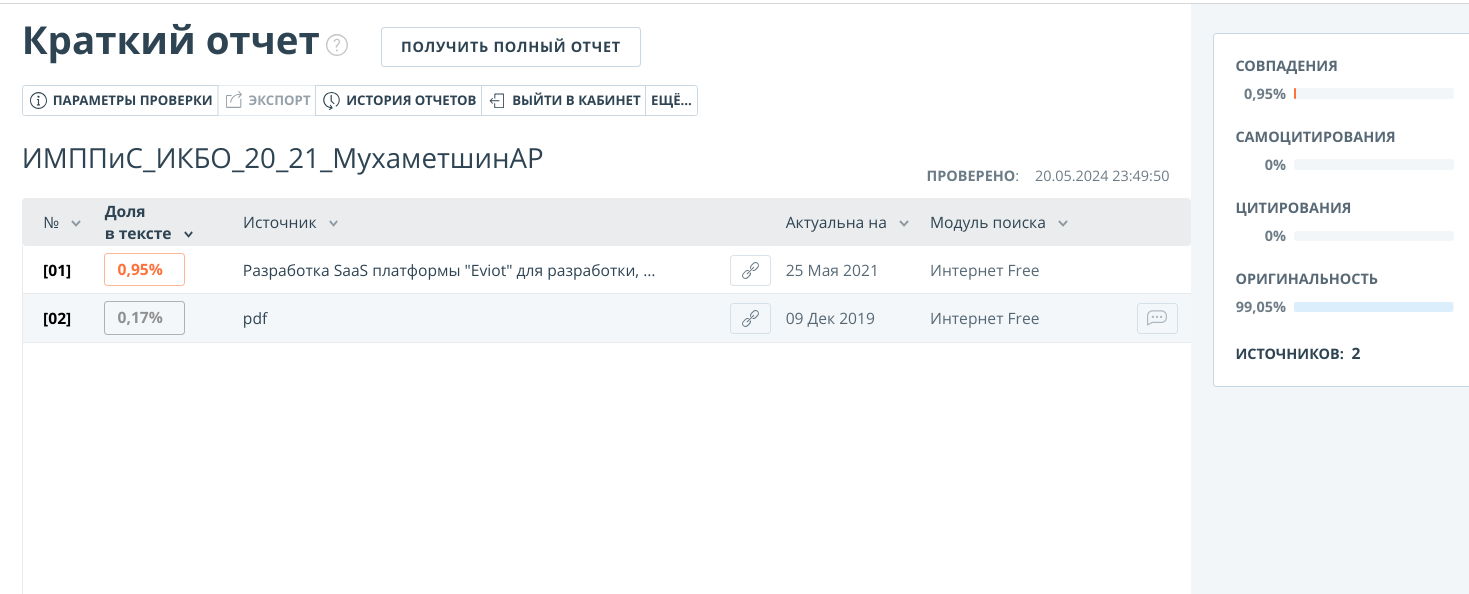


Рисунок А.1 – Отчет об антиплагиате