

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

Направление подготовки: 09.04.01 «Информатика и вычислительная техника»

Магистерская программа: «Прикладной искусственный интеллект»

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему

**ИССЛЕДОВАНИЕ И РАЗРАБОТКА МЕТОДИКИ
ОЦЕНКИ ПОЛЬЗОВАТЕЛЬСКОГО
ИНТЕРФЕЙСА**

Студент Тумеркин Дамир Рашитович
(подпись, дата) (ФИО полностью)

Руководитель Эпп В.В.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр Бычков А.С.
(подпись, дата) (фамилия, инициалы)

Рецензент Печерский С.В.
(должность, место работы) (подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол зас. кафедры от 27.05.24 № 14)

Заведующий кафедрой Митрохин М.А.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой Хорошо (протокол заседания ГЭК от
25.06.2024г. № 42)

Секретарь ГЭК Е.И. Гурин
(подпись) (фамилия, инициалы)

Пенза 2024

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет
вычислительной техники

Кафедра
«Вычислительная техника»

«Утверждаю»
Заведующий кафедрой
М.А. Митрохин
« 12 » 07 20 17 г.

**ЗАДАНИЕ
НА МАГИСТЕРСКУЮ ДИССЕРТАЦИЮ**

1. Студент Тумеркин Дамир Рашитович гр. 22ВВим2 факультета ВТ

2. Тема работы Исследование и разработка методики оценки пользовательского интерфейса

Тема утверждена приказом ПГУ № 3476 от " 15 " 06 20 17 г.

3. Руководитель магистерской диссертации к.т.н., доцент Эми В.В

4. Задание на работу (назначение разработки, исходные данные и т.п.) Назначение разработки: осуществление анализа входящего интерфейса и отталкиваясь от определенных паттернов представить оценку пользовательского интерфейса. Разработать методику оценки пользовательского интерфейса. В качестве демонстрации методики использовать настольное приложение на ПК

5. Перечень подлежащих разработке вопросов

1. Анализ предметной области методов оценки пользовательского интерфейса приложений

2. Разработка методики для оценки пользовательского интерфейса

3. Отладка методики оценки пользовательского интерфейса для прогнозирования риска наличия или развития определённого списка болезней.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области	15	13.09.2022	<i>Ю.Т.</i>
2	Выбор платформ разработки	10	30.10.2022	<i>Ю.Т.</i>
3	Моделирования системы оценки интерфейса	20	01.04.2023	<i>Ю.Т.</i>
4	Разработка методики оценки интерфейса	40	10.09.2023	<i>Ю.Т.</i>
5	Разработка интерфейса приложения для демонстрации методики	10	10.02.2024	<i>Ю.Т.</i>
6	Оформление пояснительной записки	5	31.05.2024	<i>Ю.Т.</i>

Дата выдачи задания " 13 " 09 " 20 22 г.

Руководитель магистерской диссертации *Ю.Т.* 16.06.2024
(подпись, дата)

Задание к исполнению принял студент *Ю.Т.*
(подпись, дата)

Работу к защите допустить
Декан факультета *А.Р. Фисово* 24.06.2024
(подпись, дата)

РЕФЕРАТ

Магистерская работа содержит 74 с., 29 рис., 1 табл., 12 источн., 2 прил.

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, ИНТЕРФЕЙС, ОЦЕНКА ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА, НЕЙРОННАЯ СЕТЬ, МАШИННОЕ ОБУЧЕНИЕ

Объектом исследования является методика оценки пользовательских интерфейсов.

Целью работы – разработка методики, которая оценивает интерфейс приложений на предмет удобства использования и дизайна.

В ходе выполнения работы была изучена предметная область, спроектированы основные функции.

В результате проделанной работы разработана методика оценки пользовательского интерфейса.

					ПГУ1.090401.60.001 ПЗ			
Изм.	Лист	№ докум.	Подпись	Дата	Исследование и разработка методики оценки пользовательского интерфейса Повышающая записка	Лист	Лист	Листов
Разраб.		Тумерман Д.Р.		24.06.24				
Провер.		Эпп В.В.		24.06.24			4	74
И. Констр.		Бычков А.С.		24.06.24		ФВТ, каф. ВТ		
Утверд.				24.06.24		гр. 22ВВм2		

СОДЕРЖАНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ	7
ВВЕДЕНИЕ.....	8
1 Анализ предметной области	10
1.1 Понятие пользовательского интерфейса	10
1.1.1 Роль пользовательских интерфейсов.....	12
1.1.2 Виды пользовательских интерфейсов.....	12
1.2 Подходы проектирования пользовательского интерфейса	14
1.2.1 Основные принципы создания интерфейса	16
1.2.2 Виды анализа интерфейса	17
1.3 Этапы разработки интерфейса.....	18
1.4 Обзор средств проектирования пользовательских интерфейсов.....	19
1.5 Обзор аналогов	22
2 Проектирование системы оценки пользовательского интерфейса.....	27
2.1 Моделирование системы с использованием UML	27
2.2 Подготовка данных для нейронной сети.....	31
2.3 Обнаружение объектов пользовательского интерфейса.....	32
2.4 Выбор алгоритма обнаружения объектов	34
2.5 Сравнение методов обнаружения объектов	38
2.6 Выбор фреймворка.....	39
2.6.1 Платформа машинного обучения.....	39
3 Разработка методики оценки пользовательского интерфейса	42
3.1 Описание набора данных для обучения	42
3.1.1 Снимки экрана интерфейса и иерархии представлений	42
3.2 Подготовка данных для обучения	43
3.3 Нейронная сеть для оценки пользовательского интерфейса.....	45
3.4 Результат работы приложения.....	52
ЗАКЛЮЧЕНИЕ	56

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	58
ПРИЛОЖЕНИЕ А Листинг программы	60
ПРИЛОЖЕНИЕ В Презентация	65

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящей работе приняты следующие обозначения

ПИ – Профессиональные интерфейсы

ACM (Association for Computing Machinery) – Ассоциация вычислительной техники

CNN (Convolutional Neural Network) – Сверточная нейронная сеть

GOMS (Goals, Operators, Methods, and Selection rules) – Цели, Операторы, Методы и Правила выбора

GUI (Graphical User Interface) – Графический пользовательский интерфейс

KLM (Keystroke-Level Model) – Модель уровня нажатий клавиш

KNN (K-Nearest Neighbors) – Метод ближайших соседей

NLP (Natural Language Processing) – Обработка естественного языка

UI (User Interface) – Пользовательский интерфейс

UML (Unified Modeling Language) – Унифицированный язык моделирования

UX (User Experience) – Пользовательский опыт

WIMP (Windows, Icons, Menus, Pointer) – Окна, Иконки, Меню, Указатель

ВВЕДЕНИЕ

Вне зависимости от типа программного продукта, будь то мобильное приложение, настольное приложение или веб-ресурс, пользовательский интерфейс является неотъемлемой его частью. Современная тенденция коммерческих программных продуктов идет к тому, что разработчики стремятся упростить взаимодействие с их программными продуктами посредством интуитивно понятного интерфейса. В начале развития компьютерных технологий, для использования программ пользователю необходимо было умение пользоваться командной строкой. Для своего времени консольные и псевдографические интерфейсы были прорывом во взаимодействии людей и компьютеров, но в нынешних реалиях у потребителей требования выросли. В нынешних реалиях тяжело представить пользователя, который использовал бы программное средство с непроработанным интерфейсом только ради её опций. Современный рынок программного обеспечения, позволяет выбрать программный продукт с интуитивно понятным интерфейсом, что позволяет в максимально короткие сроки разобраться с функционалом программы, что позволит потратить сэкономленное время на решение поставленной задачи.

Поэтому вопроса в необходимости разработки качественного пользовательского интерфейса не возникает.

Объектом исследования является оценка качества пользовательского интерфейса

Предметом исследования является разработка методики оценки пользовательского интерфейса

Цель работы заключается в анализе имеющихся подходов оценки пользовательских интерфейсов, их сравнении и на основании полученной информации разработке методики оценки пользовательского интерфейса.

В рамках данной работы были выделены следующие задачи:

- выявление проблем и определение основных задач оценки пользовательских интерфейсов,
- исследование основных принципов оценки пользовательского интерфейса,
- разработка нотаций для описания структуры и логики работы пользовательского интерфейса,
- разработка методики оценки пользовательского интерфейса.

Научная новизна. Разрабатываемый метод оценки пользовательского интерфейса позволяет ускорить процесс разработки приложений посредством сокращения времени на проектирование интерфейсов.

1 Анализ предметной области

После появления персональных компьютеров, понятие интерфейса стало широко использоваться в повседневной жизни. Интерфейс — это способ взаимодействия между двумя системами или приложениями. Когда речь идет о взаимодействии человека и машины, используется термин "интерфейс пользователя".

Интерфейс пользователя включает в себя все элементы, с которыми взаимодействует пользователь, такие как окна, меню, кнопки, иконки, текстовые поля и другие визуальные компоненты. Важнейшими аспектами интерфейса являются его удобство и интуитивность, так как от этого зависит, насколько эффективно и комфортно пользователи смогут выполнять свои задачи.

Хорошо спроектированный интерфейс может значительно повысить продуктивность и удовлетворенность пользователей, в то время как плохой дизайн может привести к разочарованию и снижению эффективности работы.

Разработка UI и UX стала неотъемлемой частью создания любого программного продукта или веб-сервиса. В современных условиях конкуренции важно не только предоставить функциональность, но и сделать её доступной и понятной для широкого круга пользователей. Поэтому специалисты по UX и UI занимаются не только визуальным оформлением интерфейса, но и исследованием потребностей пользователей, их поведения и оптимизацией процессов взаимодействия с продуктом. Такой подход помогает создавать продукты, которые не только функциональны, но и приятны в использовании, что способствует повышению конкурентоспособности и удовлетворённости клиентов.

1.1 Понятие пользовательского интерфейса

Интерактивный интерфейс представляет собой такой интерфейс, в котором пользователь вводит определенные данные в машину и получает необходимую информацию в определенной форме, например, графическую

или звуковую. Для ввода данных пользователь использует периферийные устройства.

Периферийные устройства — это дополнительное оборудование, которое позволяет взаимодействовать с компьютером, передавая и получая информацию. К ним относятся устройства ввода информации (клавиатуры, мыши, датчики движения) и устройства вывода информации (дисплеи, проекторы, динамики, вибромоторы и т. д.).

В зависимости от типа устройств ввода, интерфейсы могут быть тактильными, жестовыми, голосовыми, материальными (осязательными) или нейрокомпьютерными.

Методы интерфейса пользователя представляют собой набор правил, разработанных создателем устройства, которые должны привести к нужной реакции устройства на определенные действия пользователя и выполнение определенной задачи. Эти правила должны быть четко определены, интуитивно понятны и легко запоминаемы для пользователя.

Интерфейсы пользователя играют ключевую роль в создании комфортного и эффективного взаимодействия с техническими устройствами. Современные тенденции в развитии интерфейсов направлены на улучшение удобства использования и повышение доступности технологий для различных групп пользователей, включая людей с ограниченными возможностями.

Технологии распознавания жестов, голоса и мышления активно интегрируются для создания более естественного способа взаимодействия между человеком и машиной. Продвинутое системы также учитывают контекст и предпочтения пользователя для персонализированного опыта.

Будущее интерфейсов направлено на дальнейшее развитие искусственного интеллекта и машинного обучения, что позволит создать ещё более интуитивные и адаптивные системы, способные предсказывать потребности пользователя и предлагать решения до того, как это будет сделано им самим.

1.1.1 Роль пользовательских интерфейсов

Интерфейс является неотъемлемой частью современных приложений, сокращая время и увеличивая качество выполнения поставленной задачи. Таким образом, качество пользовательского интерфейса является важным аспектом в проектировании. Если интерфейс не будет соответствовать определенным требованиям, то это усложнит использование приложения. Для этого существует стандартизация интерфейсов. Соблюдая нормы проектирования и требования стандартов, можно исключить вариант что пользователь не разберется.

1.1.2 Виды пользовательских интерфейсов

Текстовый пользовательский интерфейс (от англ. Text user interface, TUI; также Character User Interface, CUI) – вид интерфейса, при котором для визуализации используются только символы.

На рисунке 1 представлен пример текстового пользовательского интерфейса.

```
PING text.pntpa.wikimedia.org (208.80.152.2) 56(84) bytes of data.
^C
--- text.pntpa.wikimedia.org ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 540.528/540.528/540.528/0.000 ms
[root@localhost ~]# pwd
/root
[root@localhost ~]# cd /var
[root@localhost var]# ls -la
total 72
drwxr-xr-x. 18 root root 4096 Jul 30 22:43 .
drwxr-xr-x. 23 root root 4096 Sep 14 20:42 ..
drwxr-xr-x.  2 root root 4096 May 14 00:15 account
drwxr-xr-x. 11 root root 4096 Jul 31 22:26 cache
drwxr-xr-x.  3 root root 4096 May 18 16:03 db
drwxr-xr-x.  3 root root 4096 May 18 16:03 empty
drwxr-xr-x.  2 root root 4096 May 18 16:03 games
drwxrwx--T.  2 root gdm 4096 Jun  2 18:39 gdm
drwxr-xr-x. 38 root root 4096 May 18 16:03 lib
drwxr-xr-x.  2 root root 4096 May 18 16:03 local
lrwxrwxrwx.  1 root root 11 May 14 00:12 lock -> ../run/lock
drwxr-xr-x. 14 root root 4096 Sep 14 20:42 log
lrwxrwxrwx.  1 root root 10 Jul 30 22:43 mail -> spool/mail
drwxr-xr-x.  2 root root 4096 May 18 16:03 nis
drwxr-xr-x.  2 root root 4096 May 18 16:03 opt
drwxr-xr-x.  2 root root 4096 May 18 16:03 preserve
drwxr-xr-x.  2 root root 4096 Jul  1 22:11 report
lrwxrwxrwx.  1 root root  6 May 14 00:12 run -> ../run
drwxr-xr-x. 14 root root 4096 May 18 16:03 spool
drwxrwxrwt.  4 root root 4096 Sep 12 23:50 tmp
drwxr-xr-x.  2 root root 4096 May 18 16:03 yp
[root@localhost var]# yum search wiki
Loaded plugins: langpacks, presto, refresh-packagekit, remove-with-leaves
rpmfusion-free-updates                               | 2.7 kB      00:00
rpmfusion-free-updates/primary_db                     | 206 kB      00:04
rpmfusion-nonfree-updates                             | 2.7 kB      00:00
updates/metalink                                       | 5.9 kB      00:00
updates                                                | 4.7 kB      00:00
updates/primary_db                                     73% [=====] 62 kB/s | 2.6 MB      00:15 ETA
```

Рисунок 1 – Текстовый пользовательский интерфейс

Являлся начальным этапом в проектировании интерфейсов.

К плюсам можно отнести:

- скорость отображения информации,
- низкая требовательность к аппаратной части.

К минусам можно отнести:

- недостаток графических средств из-за ограниченности словаря символов.
- ограниченные возможности для интерактивного взаимодействия с пользователем.

Обычно текстовый интерфейс визуализируется в командной строке, поэтому приложения, использующие данный тип интерфейса, называются консольными программами.

Текстовый интерфейс имеет все основные элементы интерфейса схожие с графическим интерфейсом (кнопки, чек боксы, списки, меню) а также возможность взаимодействовать с периферийными устройствами

Примером текстового интерфейса считается алфавитно-цифровое печатающее устройство, которое повсеместно использовалось в первой половине XX века. Далее с течением времени они были заменены мониторами с отображением символов и знаков, что дало возможность более комфортно предоставлять информацию пользователю.

Графический интерфейс пользователя (от англ. Graphical User Interface (GUI)) – вид интерфейса, при котором для визуализации используются графические изображения.

В 1973 году энтузиастами из компании Xerox был разработан прототип первого графического интерфейса – WIMP (Windows, Icons, Menus, Pointer) интерфейс, что значительно изменил подход к использованию приложений. В прототипе использовались понятия окон, значков, указателей, что кардинально отличалось от того, что было в 1967-68 гг. в Unix. WIMP давала

возможность взаимодействовать с объектами интерфейса, перемещать их и изменять свойства.

На рисунке 2 представлен пример графического пользовательского интерфейса.



Рисунок 2 – Графический пользовательский интерфейс

По итогу можно сказать, что графические интерфейсы на данный момент являются более привычными, интуитивно понятными как для обычных пользователей, так и для более опытных.

1.2 Подходы проектирования пользовательского интерфейса

Удобство и практичность пользовательского интерфейса проверяется с помощью определенных методов изучаемых в юзабилити (от англ. usability).

Юзабилити — это мера того, насколько хорошо конкретный пользователь в конкретном контексте может использовать продукт/дизайн для эффективного, действенного и удовлетворительного достижения поставленной цели [1]. Разработчики обычно измеряют удобство использования дизайна на протяжении всего процесса разработки — от

макетов до конечного результата — чтобы обеспечить максимальное удобство использования.

Эвристическая оценка (от англ. heuristic evaluation) – это метод, в котором с помощью эмпирических правил эксперты производят измерения удобства использования пользовательских интерфейсов и в процессе независимых проверок выделяют их проблемы. Эксперты используют сформированные правила эвристики такие как эвристика Нильсена-Молича и выделяют идеи, которые на раннем этапе разработки могут сформировать у команды разработчиков правильные действия для увеличения удобства использования продукта.

Эксперты, следуя контрольному списку критериев, исследуют недостатки, которые незаметны для команды разработчиков. Эвристика Нильсена-Молича полагает что система должна:

- информировать пользователей своевременно и должным образом,
- отталкиваясь от реального мира, представлять информацию на языке пользователей и доступным образом,
- давать возможность контроля над ситуацией и исправления ошибок,
- последовательно предоставлять информацию, чтобы не перегружать систему,
- для опытных пользователей, давать возможность быть гибкими, сокращая время для получения необходимой информации.

Достоинства эвристической оценки:

- эвристика позволяет выделить недостатки интерфейса на раннем этапе разработки,
- по сравнению с другими методами является недорогим и быстрым.

Недостатки эвристической оценки:

- прямо пропорционально зависит от уровня опыта и знаний экспертов,
- обучение экспертов требует время и финансы,

– возможны ложные результаты, где проблемы могут на самом деле таковыми не являться.

Важнейший момент заключается в том, что эвристическая оценка, какой бы полезной она ни была, не может заменить тестирование юзабилити.

Метод, основанный на проверке по контрольному списку (от англ. Checklist-based testing) – метод, который требует создания контрольных списков по мере выполнения тестирования интерфейса и основывается на опыте инженера-тестировщика. Благодаря данному методу разработчики могут проектировать тесты для проверки каждого аспекта приложения.

Проанализировав продукт, тестировщики могут расширить контрольный список, который включал бы более детальные условия проверки интерфейса или создать новый. Контрольные списки создаются, отталкиваясь от личного опыта проверяющих и отрасли приложения.

Преимущества использования метода по контрольному списку:

- позволяет обнаружить критические ошибки в проектировании интерфейса и минимизирует вероятность их пропуска,
- позволяет начинающим тестировщикам уверенно и эффективно производить тестирование, так как существует набор инструментов для этого.

Недостатки использования метода по контрольному списку:

- в зависимости от объема контрольного списка, возможны большие затраты на время проверки и исправления ошибок.
- может ограничивать креативность тестировщика, приводя к проверке только тех аспектов, которые указаны в списке.

1.2.1 Основные принципы создания интерфейса

Создание четкой отправной точки – позволяет пользователю сократить усилия по выполнению задачи. Для реализации данного принципа необходимо правильно использовать элементы интерфейса. Необходима понятное расположение текста и иконок с точки зрения пользователя.

Поддержание синергии в дизайне – позволяет пользователям быстрое определение общих действий. Если элементы интерфейса или функций не унифицированы, это может вызвать проблемы с навигацией. Должна быть реализована согласованность между макетом, стилями кнопок, цветовой палитры, иконок.

Интеграция знакомых элементов дизайна – позволяет легко ориентироваться пользователю в интерфейсе и интуитивно понимать возможности отдельных элементов. Внедрение редких и экспериментальных элементов может затруднить выполнение задач и такой интерфейс нельзя считать ориентированным на пользователя.

Предоставление четких сообщений об ошибках – улучшает пользовательский опыт. Верное определение ошибки в процессе работы позволяет сократить время её решения. Для этого необходимо ясно и кратко формулировать ошибку, не использовать сложную терминологию.

Таким образом, создание интерфейса является структурированным процессом. Следуя сформированным принципам, можно сделать интерактивный и удобный интерфейс для пользователя

1.2.2 Виды анализа интерфейса

Для анализа интерфейсов должны быть выработаны критерии оценки пользовательских интерфейсов и соответствующие им количественные шкалы, опирающиеся, в первую очередь, на общепризнанные стандарты в области проектирования.

Существует множество методов количественного анализа пользовательских интерфейсов.

Аналитические методы, позволяющие смоделировать и провести анализ выполнения той или иной задачи оператором и на основании полученных данных произвести оценку качества пользовательского интерфейса.

Экспериментальные, направленные на получение качественных и количественных оценок при выполнении профессиональных задач

оператором с помощью ПИ. Экспериментальные методы могут проводиться на тренажерах (лабораторные) или в реальных условиях на реальных рабочих местах (натурные). Экспериментальные методы дают хорошие результаты на этапах технического проекта и опытной эксплуатации, за счет выявления и устранения недостатков создаваемых специализированных систем.

Анкетирования, позволяющие дать оценку отношения операторов к ПИ и (или) его отдельным составляющим, а также выявить основные проблемы, возникающие у операторов, при работе с системой.

Экспертные, позволяющие при исследовании ПИ экспертами выявлять достоинства и недостатки, которые, в дальнейшем, необходимо устранить. В данном случае рассматриваются аналитические методы: GOMS, KLM, а также методика оценки сложности системы Тима Комбера и Джона Мэлтби. GOMS это семейство методов, позволяющих оценить время выполнения задачи как основной критерий качества. KLM является одной из разновидностей метода GOMS, в которой используются только операторы. Методика оценки сложности системы Тима Комбера и Джона Мэлтби базируется на типах элементов интерфейса и их характеристиках.

1.3 Этапы разработки интерфейса

Процесс создания пользовательского интерфейса часто основывается на методологии дизайн-мышления. В основе методологии дизайн-мышления лежит понимание потребности пользователя и его креативное удовлетворение.

Разработка пользовательского интерфейса обычно включает несколько этапов. На первом этапе, называемом проектированием, определяются цели разработки, требования к интерфейсу, анализируются потребности пользователей, структура диалога и сценарии использования. В этом этапе также происходит моделирование процесса, для которого разрабатывается приложение, выбор платформы и средств разработки.

После этапа проектирования следует этап разработки, в ходе которого создаются макеты интерфейсов с возможностью их дальнейшего

редактирования, производится прототипирование. Под прототипом понимается динамическая модель приложения, которая симулирует взаимодействие пользователя с приложением.

Завершающим этапом является тестирование разработанного интерфейса и оценка его качества. Часто в процессе тестирования привлекаются потенциальные пользователи продукта, поскольку для полного понимания ситуации требуется объективное мнение стороннего пользователя, а не только точка зрения разработчика.

В целом, процесс разработки пользовательского интерфейса включает проектирование, разработку и тестирование, при этом активное взаимодействие с пользователями является важным аспектом для достижения оптимальных результатов.

1.4 Обзор средств проектирования пользовательских интерфейсов

Нельзя не упомянуть про инструменты и средства для проектирования пользовательских интерфейсов.

Adobe XD — это программное обеспечение от Adobe Systems, которое предназначено для создания пользовательского интерфейса и пользовательского опыта. С помощью Adobe XD дизайнеры и разработчики могут создавать прототипы, макеты и другие элементы интерфейса с высокой степенью интерактивности [2].

При использовании Adobe XD можно определить переходы между экранами, добавлять анимацию и эффекты для создания более реалистичного пользовательского опыта. Это позволяет взаимодействовать с элементами интерфейса, испытывая и проверяя их функциональность.

Одной из важных особенностей Adobe XD является возможность совместной работы над проектами. Можно пригласить других участников команды для совместной работы над дизайном, комментирования и обмена идеями в режиме реального времени. Это упрощает командную работу и

способствует эффективному взаимодействию между дизайнерами и разработчиками.

На рисунке 3 представлен интерфейс Adobe XD.

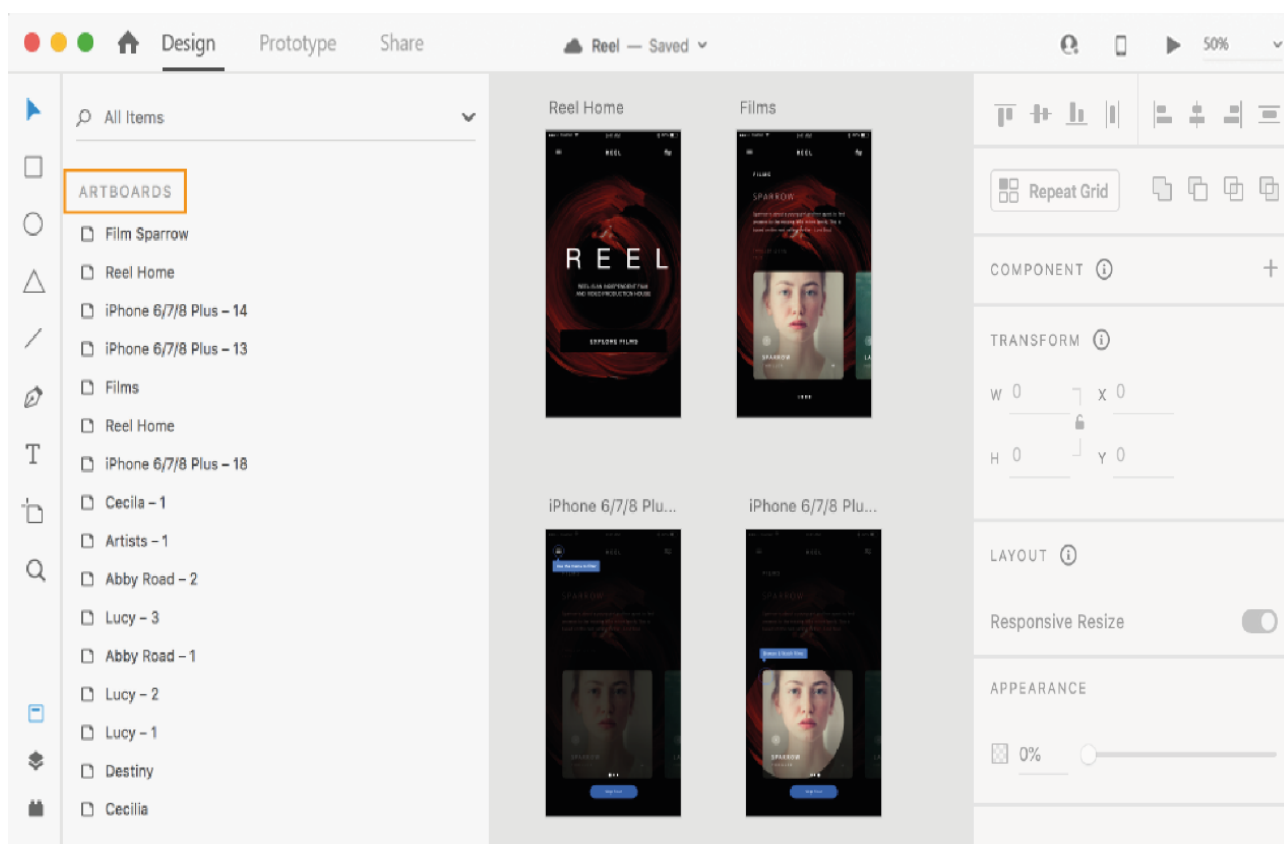


Рисунок 3 – Интерфейс Adobe XD

Figma — это онлайн-платформа, предназначенная для разработки пользовательского интерфейса (UI) и пользовательского опыта (UX). Она предоставляет возможность создавать, прототипировать и совместно работать над дизайном интерфейса [3].

Figma предлагает разнообразные инструменты для создания макетов интерфейса. В рамках платформы можно добавлять графические элементы, текст, импортировать изображения и создавать векторные формы. Также доступны функции выравнивания и распределения элементов, которые облегчают создание сетки и организацию макета.

Платформа Figma широко используется в дизайнерских сообществах и командах разработки для создания прототипов, макетов и совместной работы над дизайном интерфейса. Ее онлайн-формат обеспечивает гибкость и

удобство использования, что делает ее популярным выбором для работы в команде и сотрудничества на расстоянии.

На рисунке 4 представлен интерфейс Figma.

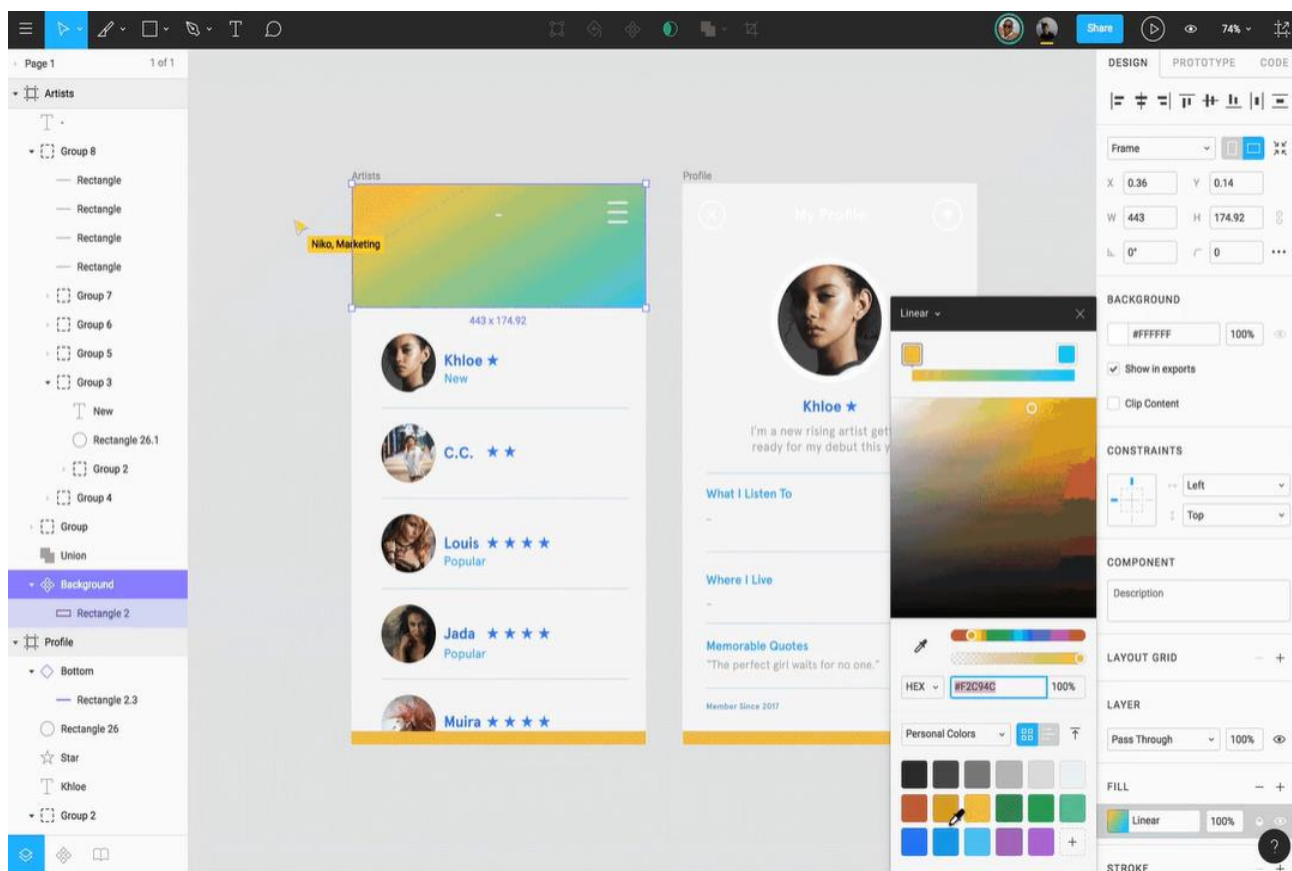


Рисунок 4 – Интерфейс Figma

Scketch – векторный графических редактор, который используется для разработки настольных и мобильный пользовательских интерфейсов, макетов и прототипов. На данный момент считается отраслевым стандартом. Также редактор позволяет редактировать и обрабатывать фотографии [4].

Предназначен для дизайнеров, которые работают в большей мере в команде и которые работают с клиентами

Преимущества Scketch:

- простота и легкость в освоении,
- интуитивно понятный интерфейс,
- присутствует функционал для совместной работы.

Недостатки Scketch:

– отсутствует кроссплатформенность, так как приложение доступно только на MacOS.

– ограниченные возможности интеграции с другими инструментами и сервисами.

На рисунке 5 представлен интерфейс Scketch.

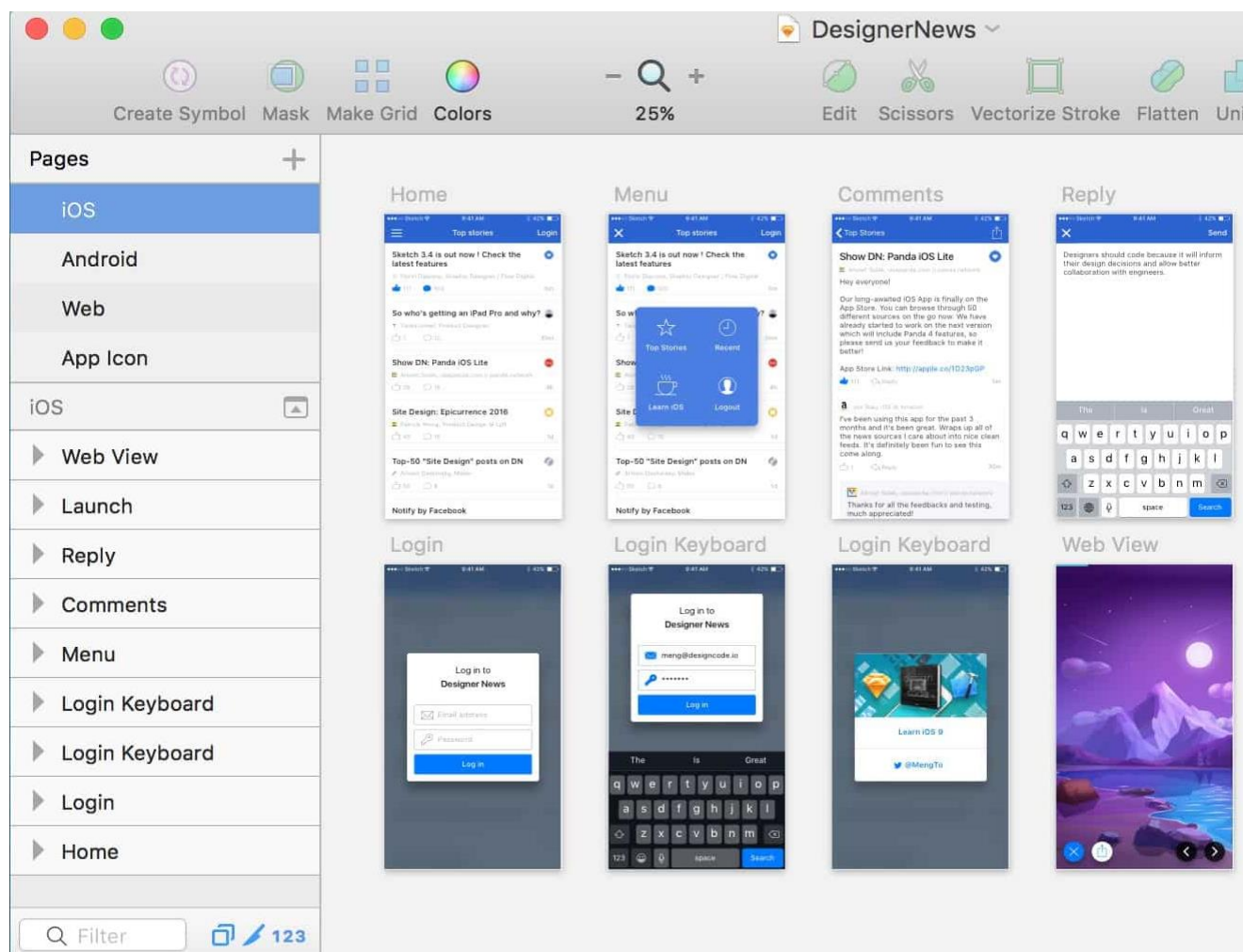


Рисунок 5 – Приложение Scketch

Таким образом были представлены средства для проектирования пользовательских интерфейсов, проанализированы их преимущества и недостатки.

1.5 Обзор аналогов

Существуют различные сервисы для оценки пользовательского интерфейса. Среди всех имеющихся особенно можно выделить:

Usabilla – инструмент, который позволяет оставлять отзывы на веб-сайтах, приложениях или по электронной почте. Обратная связь в приложениях является полностью встроенной, которая не требует перезапуска окна браузера или перенаправления [5].

Позволяет выбрать конкретный элемент приложения, про который пользователь хочет оставить отзыв. Данная возможность избавляет разработчиков от неопределенности и позволяет определить конкретные аспекты сайта, которые нуждаются в улучшении

На рисунке 6 представлен ресурс Usabilla.

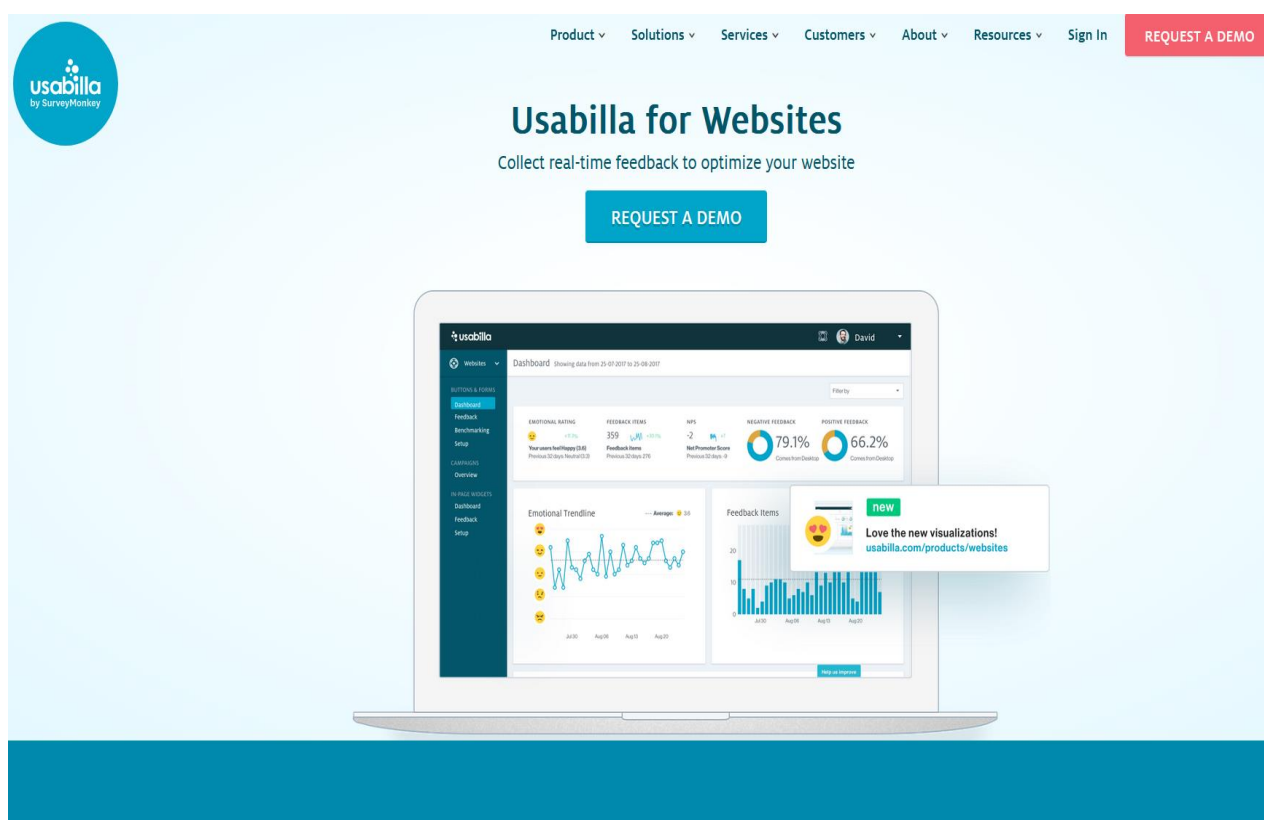


Рисунок 6 – Ресурс Usabilla

UsabilityHub – это платформа для исследования пользователей, которая используется для тестирования клиентоориентированных приложений и сбора данных о аудитории. С помощью инструментов и функций ресурс позволяет совершенствовать дизайн и создавать более интересные пользовательские впечатления [6].

Универсальность позволяет тестировать идеи для различных платформ: настольных приложений, мобильных устройств, веб-приложений. Также UsabilityHub решает распространённую проблему, когда приложение перед выпуском проходит недостаточно проверок в реальных условиях, гарантируя что потенциальные проблемы обнаруживаются до запуска

Платформа UsabilityHub позволяет быстро и эффективно проводить тестирования, собирая мнения и впечатления от широкой аудитории пользователей. Она также предоставляет инструменты для создания и проведения опросов с помощью визуального дизайна, что помогает лучше понять предпочтения и потребности пользователей.

На рисунке 7 представлен ресурс UsabilityHub.

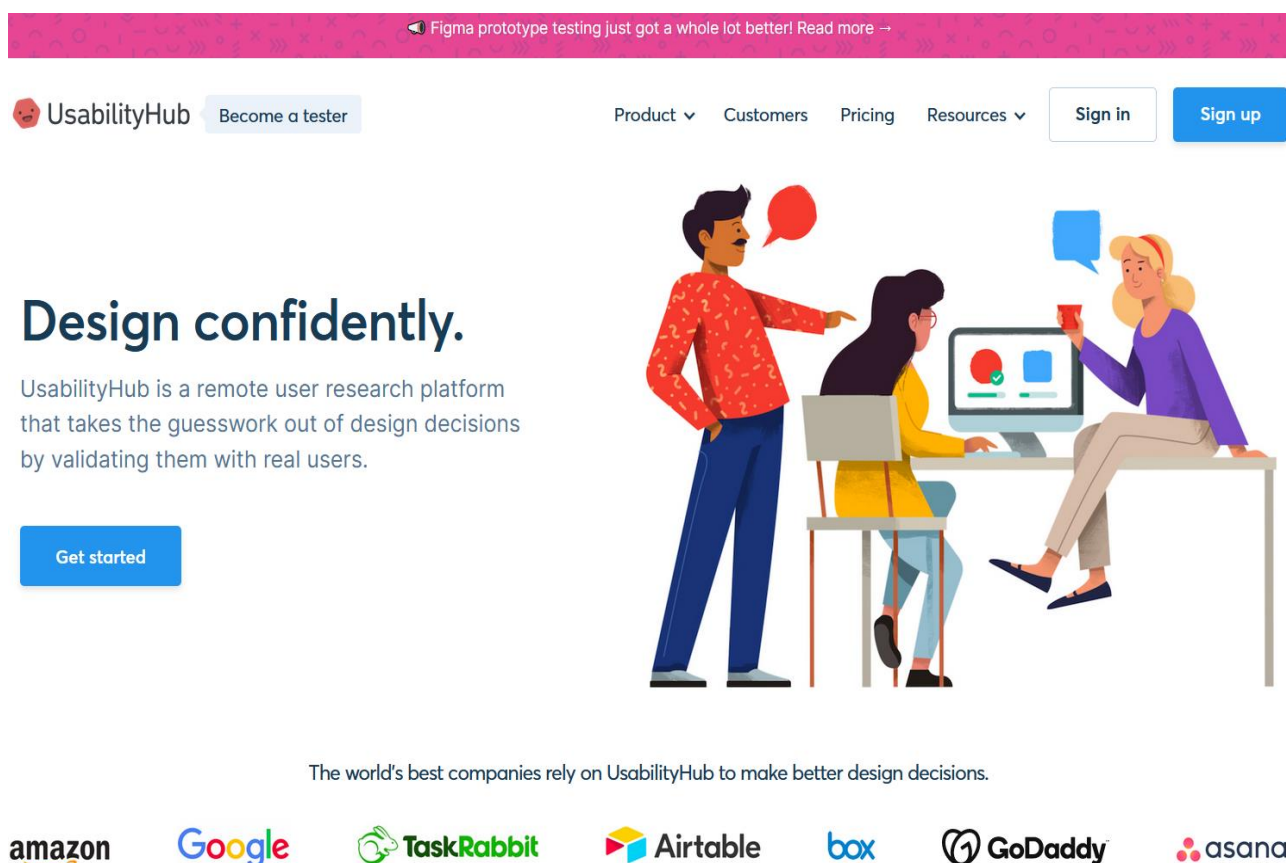


Рисунок 7 – Ресурс UsabilityHub

В итоге, использование UsabilityHub помогает компаниям получить ценные знания о своих продуктах и услугах, чтобы оптимизировать их в соответствии с потребностями пользователей и достичь более успешных результатов.

Userbrain — это инструмент удаленного тестирования юзабилити, который предлагает простоту и удобство в использовании. Этот инструмент позволяет вам получать ценные видеозаписи пользователей, которые взаимодействуют с вашим веб-сайтом. Эти видеозаписи помогают вам выявлять ошибки, проблемные моменты и сомнительные аспекты в пользовательском опыте [7].

На рисунке 8 представлен ресурс Userbrain.

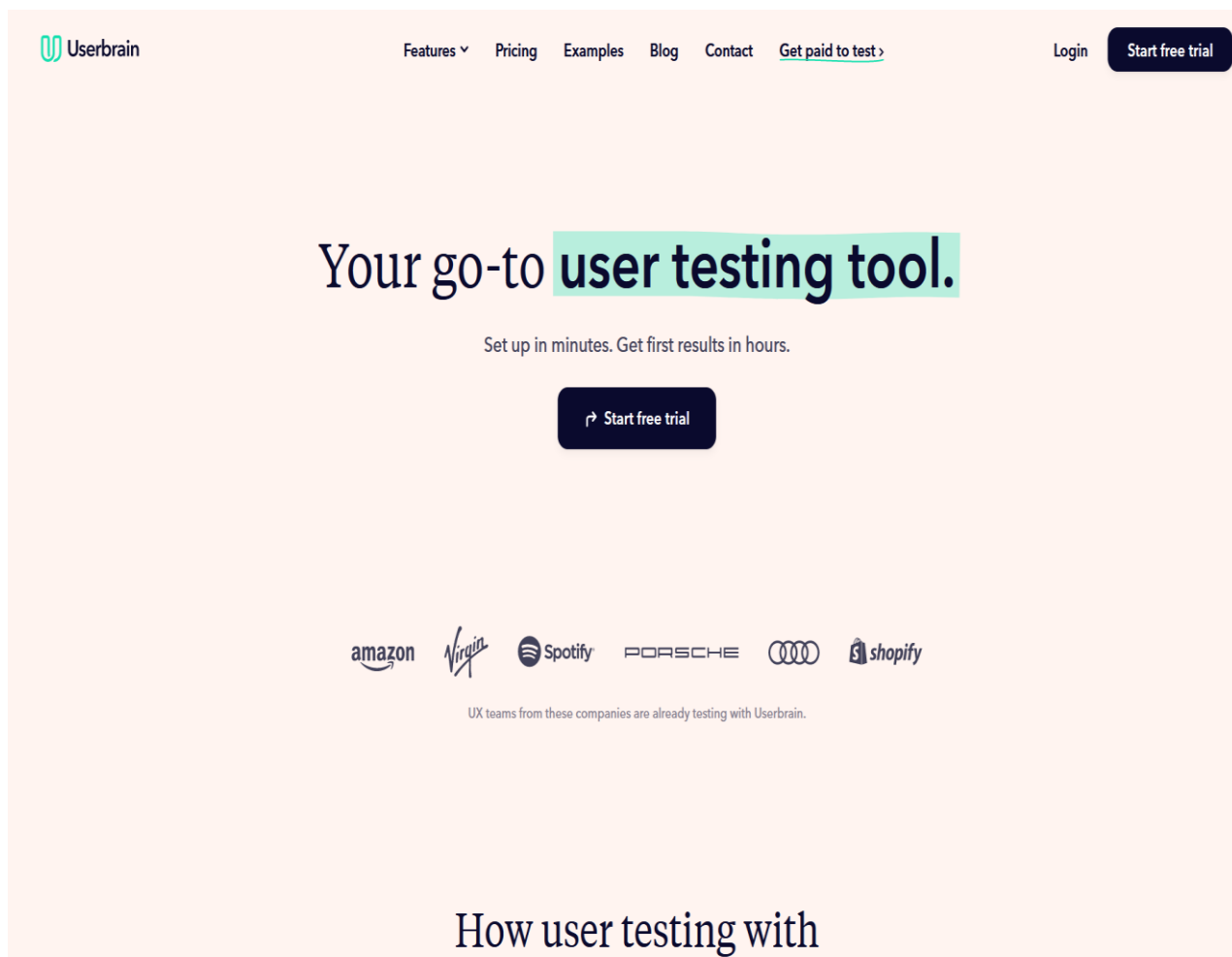


Рисунок 8 – Ресурс Userbrain

Userbrain предлагает возможность проводить еженедельные автоматические пользовательские тесты, которые помогают вам непрерывно улучшать работу вашего сайта и создавать более качественные продукты. Эти тесты обеспечивают постоянный поток обратной связи от реальных пользователей, позволяя вам легко и быстро идентифицировать проблемные области и предпринимать меры для их улучшения.

Каждый представленный ресурс имеет свои достоинства и недостатки, которые перечислены в таблице 1.

Таблица 1 – Аналоги разрабатываемой системы

Название ресурса	Специализация	Методы тестирования	Надежность методов тестирования	Доступность ресурса
Usabilla	Ресурс для тестирования UI	Кнопки обратной связи и визуальные опросы.	Средняя	Значительная
UsabilityHub	Ресурс для тестирования UI	Видео с использованием приложений	Умеренная	Малая
Userbrain	Ресурс для тестирования UI и UX	Автоматические пользовательские тесты	Средняя	Умеренная
Разрабатываемая система	Ресурс для тестирования UI	База оценки UI	Высокая	Значительная

В ходе сравнения существующих решений для оценки пользовательского интерфейса были выявлены недостатки и проанализировав их, было принято решение разработать систему решающую задачу оценки пользовательского интерфейса с учетом достоинств и недостатков аналогов.

2 Проектирование системы оценки пользовательского интерфейса

Первым шагом в разработке методики оценки пользовательского интерфейса является проектирование взаимодействия пользователя с системой, что позволит более качественно заниматься реализацией основной идеи.

2.1 Моделирование системы с использованием UML

Для проектирования системы оценки пользовательского интерфейса первым шагом является разработка методики взаимодействия пользователя с системой, что способствует более эффективной реализации основной концепции.

Ключевую роль играет моделирование системы с использованием UML (Unified Modeling Language) — стандартного языка моделирования, предназначенного для описания различных аспектов разрабатываемой системы [8].

Разрабатываемая система ориентирована на категорию пользователей, в данном случае — разработчиков графических интерфейсов. Важно отметить, что для успешного проектирования и последующей реализации системы оценки пользовательского интерфейса необходимо уделить внимание разработке детальных моделей взаимодействия, функциональных возможностей и структурных компонентов системы.

Оптимальный объем методики включает в себя описание основных этапов проектирования, включая подробное моделирование пользовательских сценариев, архитектуру системы, спецификацию интерфейсов и протоколов взаимодействия. Это позволяет не только лучше понять требования пользователей и специфику разрабатываемой системы, но и обеспечить ее более эффективное функционирование в процессе эксплуатации.

Таким образом, методика оценки пользовательского интерфейса, основанная на системном проектировании с использованием UML, предполагает комплексный подход к созданию и оптимизации интерфейса,

что способствует повышению качества и удобства использования разрабатываемой системы разработчиками графических интерфейсов.

На рисунке 9 представлена структура разрабатываемой системы.

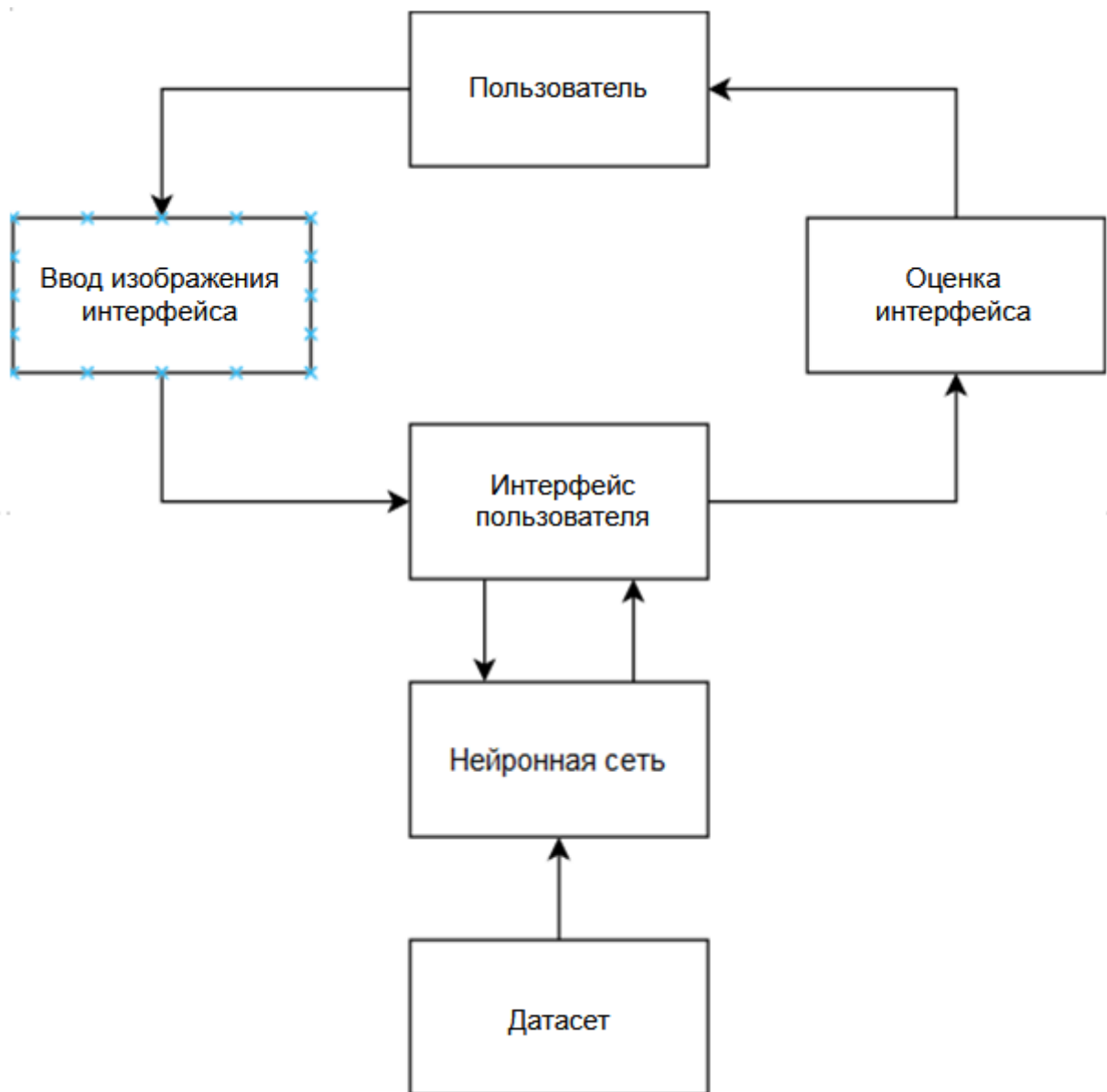


Рисунок 9 – Структура разрабатываемой системы

Данная структура является основой так как в зависимости от идей и желаний разработчика, она может модифицироваться.

На вход системы поступает файл изображения, который обрабатывается нейронной сетью и после сравнения с датасетом подготовленных данных(референсов), пользователю выводится результат оценки загруженного интерфейса.

На рисунке 10 представлена диаграмма классов приложения.

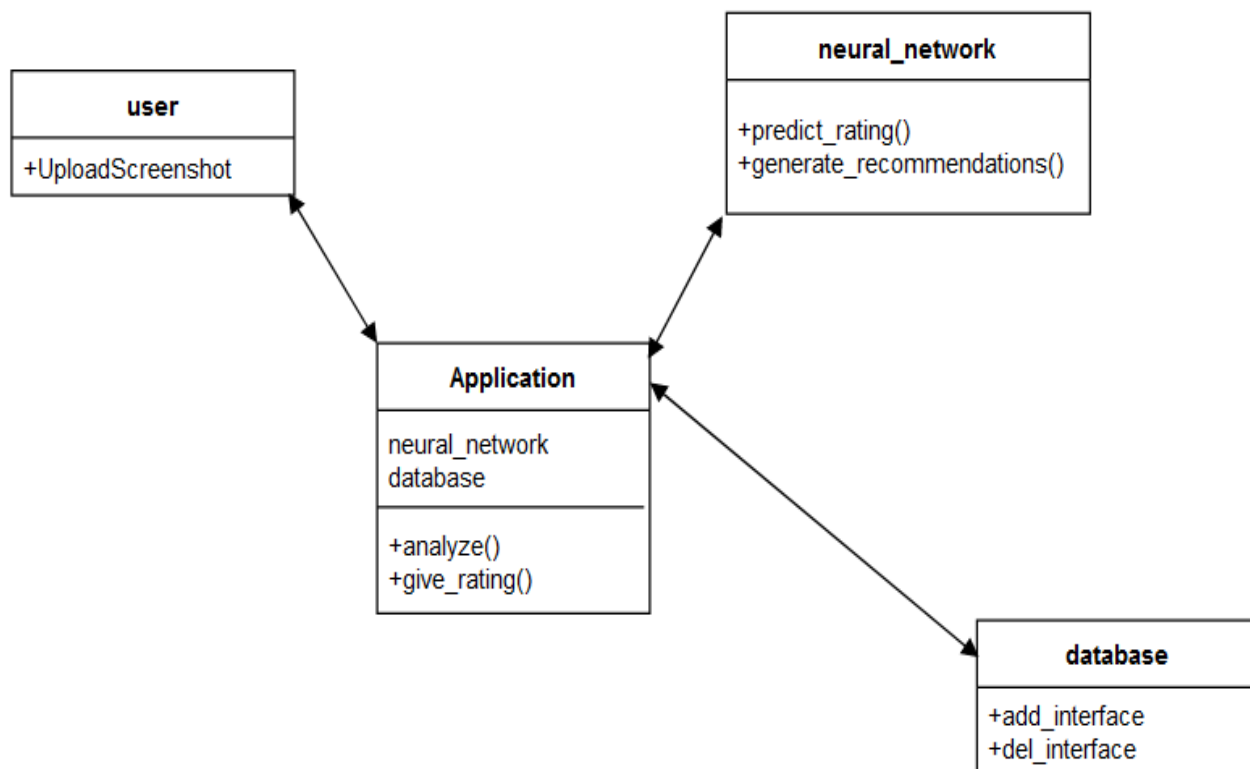


Рисунок 10 – Диаграмма классов системы

Диаграмма классов – статическая структурная диаграмма, которая описывает структуру системы, классы, атрибуты, методы и зависимости между классами.

Также на рисунке 11 представлена диаграмма прецедентов пользователя разрабатываемой системы.

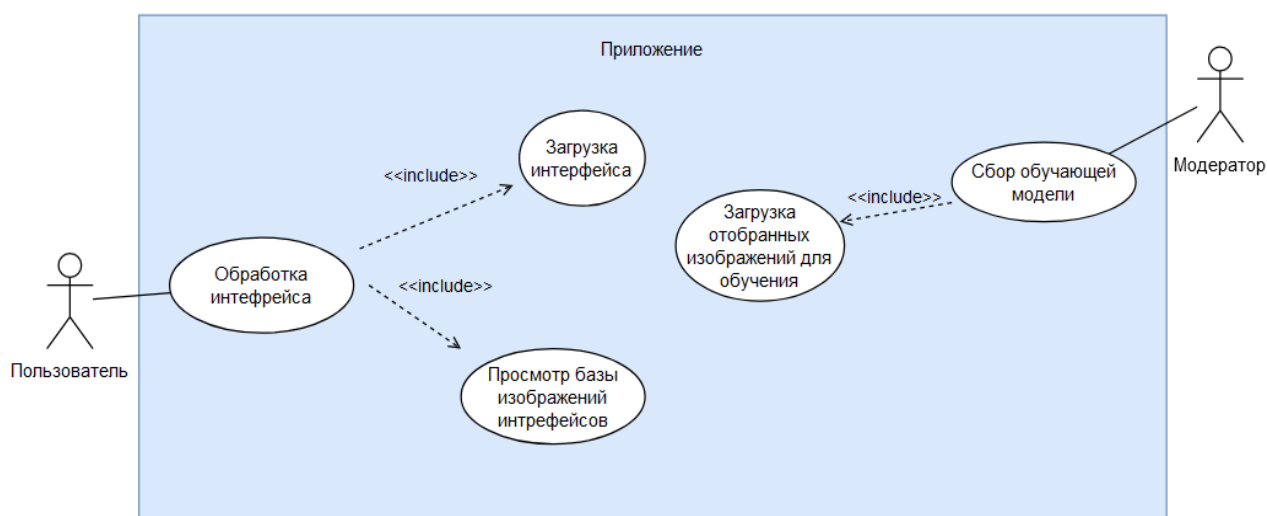


Рисунок 11 – Диаграмма прецедентов для пользователя

Диаграмма прецедентов – это диаграмма вариантов использования в UML, которая позволяет описать основные взаимодействия пользователей с материалом и предоставит описание ее функций.

Нотация диаграммы прецедентов спроектирована таким образом, что даже неподготовленный пользователь может понять их значение и помочь уточнить некоторые моменты.

Пользователю будут доступны следующие возможности:

- просмотр базы отобранных интерфейсов,
- загрузка интерфейса для оценки.

Модератору будут доступны следующие возможности:

- просмотр базы отобранных интерфейсов,
- загрузка интерфейса для оценки,
- загрузка отобранных интерфейсов для обучения.

На рисунке 12 представлена диаграмма деятельности.

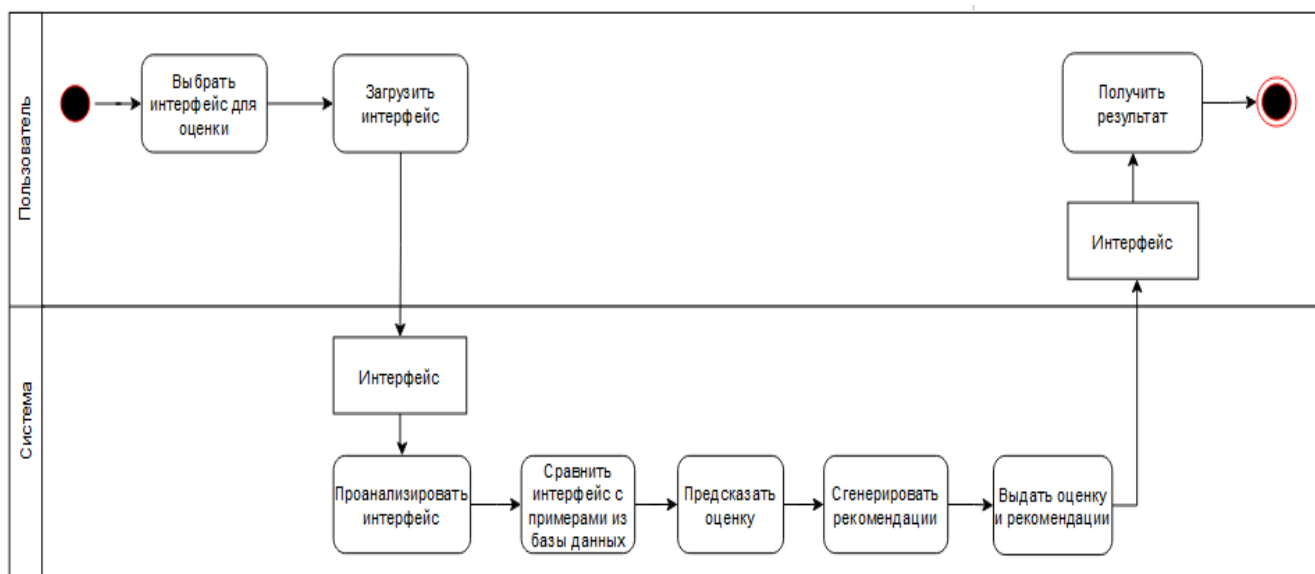


Рисунок 12 – Диаграмма деятельности

Диаграмма деятельности – диаграмма в UML, которая описывает частный случай использования системы. Также диаграмма может быть применена для визуализации событий в бизнес-процессах [9].

На рисунке 13 представлена диаграмма последовательности пользователя системы.

Диаграмма последовательности описывает объекты и их взаимодействия во временном промежутке использования системы.

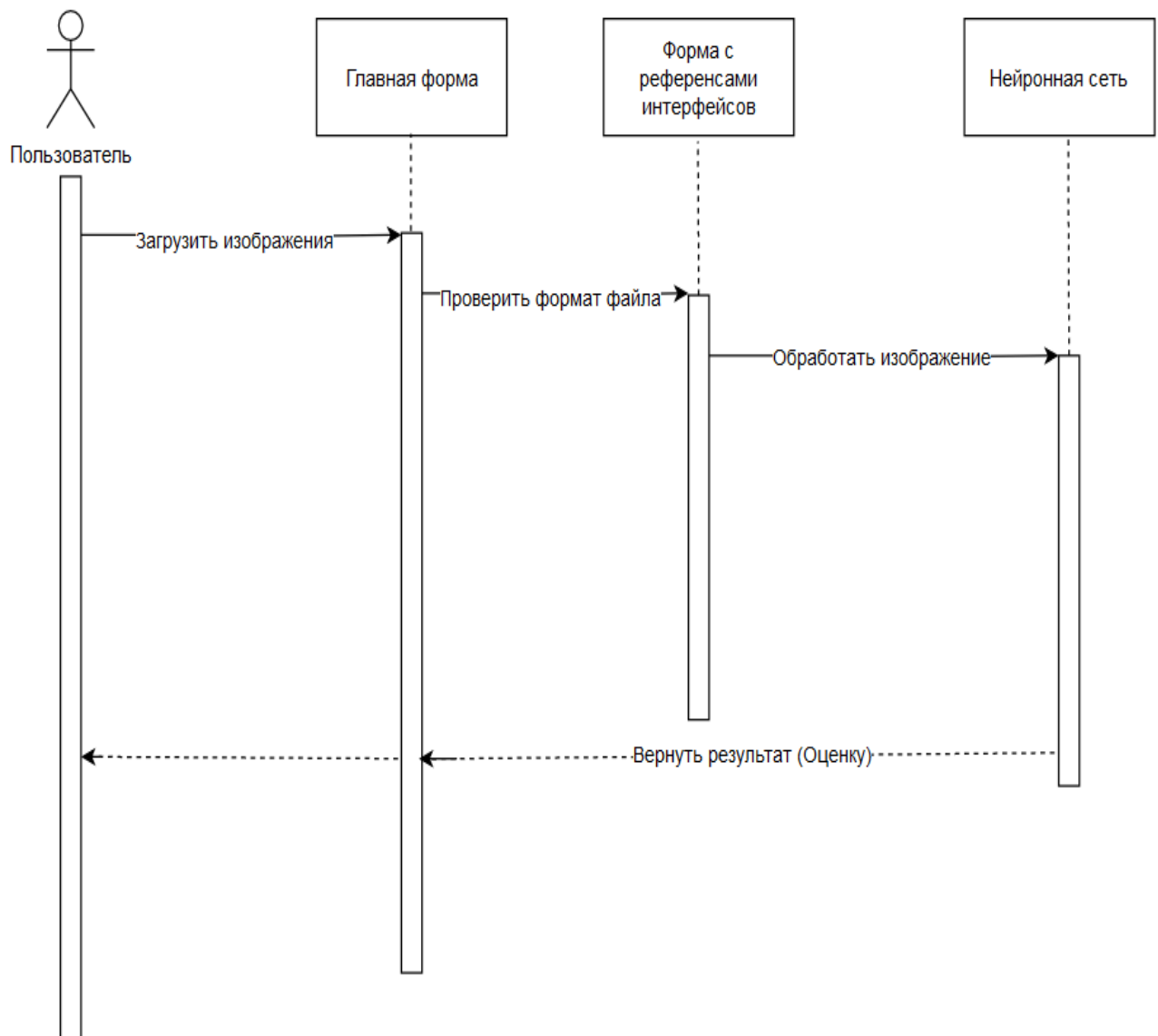


Рисунок 13 – Диаграмма последовательности

2.2 Подготовка данных для нейронной сети

Для реализации системы оценки пользовательского интерфейса необходимо использовать методы глубокого обучения для обнаружения объектов интерфейса.

В качестве набора данных для обучения будут использоваться с ReDraw и Riso, так как они являются общедоступными и широко используются в отрасли

Набор данных ReDraw состоит из совокупности снимков экрана, метаданных пользовательского интерфейса (GUI) и аннотированных изображений компонентов графического интерфейса. Он включает в себя 15 категорий, таких как переключатели, индикаторы выполнения, кнопки, переключатели и чекбоксы, более 14000 изображений пользовательского интерфейса и более 19000 аннотированных компонентов графического интерфейса. Количество каждого компонента достигает 5000 после обработки набора данных.

Этот набор данных используется для обучения и оценки технологий машинного обучения сверточной нейронной сети (CNN) и К-ближайшего соседа (KNN).

Rico на сегодняшний день является крупнейшим набором данных пользовательского интерфейса, созданным для поддержки пяти типов приложений, управляемых данными: поиск дизайна, генерация макета пользовательского интерфейса, генерация кода пользовательского интерфейса, моделирование взаимодействия с пользователем и прогнозирование восприятия пользователя. Набор данных Рико содержит более 25 категорий, более 9 000 приложений и около 75 000 скриншотов.

Этот набор данных был открыт на 30-м ежегодном симпозиуме ACM по программному обеспечению и технологиям пользовательского интерфейса в 2017 (RICO: набор данных приложений для создания приложений для проектирования, управляемых данными).

2.3 Обнаружение объектов пользовательского интерфейса

Процесс машинного обучения можно сравнить с обучением человека.

В зависимости от результата, который получается после ввода выборки x в функцию $f(x)$ можно сказать к какой категории относится решенная проблема, если результатом является значение, то проблема относится к проблеме регрессии, иначе если результатом является категория, то проблема является проблемой классификации.

Концепт машинного обучения схож с обучением человека за исключением того, что для человека требуется гораздо меньше данных для решения задачи. Например, для классификации животных по классам человеку достаточно иметь пару несколько примеров, для машины же нужна большая база данных.

Приложения, в которых задействуется машинное обучение:

- распознавание номерных знаков и распознавание лиц,
- текстовый поиск и поиск изображений,
- идентификация целевой группы и автоматический маркетинг,
- выявление болезней и обнаружение аномалий,
- обработка естественного языка (NLP) для таких приложений, как анализ настроений и маркировка фотографий.

На рисунке 14 представлена схема машинного обучения.

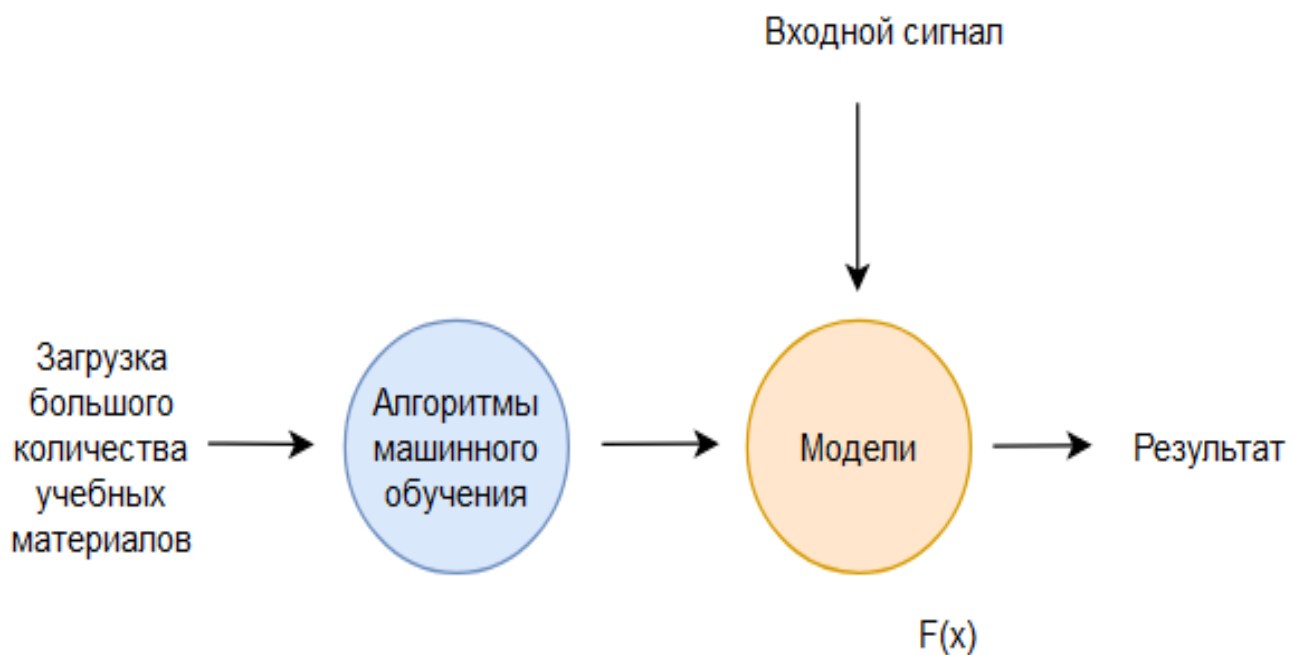


Рисунок 14 – Схема машинного обучения

Обнаружение объектов —технология, связанная с компьютерным зрением и обработкой изображений. Она используется для выявления определенных категорий семантических объектов, таких как люди, животные

и автомобили, в цифровых изображениях и видео. В нашем случае для обнаружения элементов интерфейса.

Обнаружение объектов в контексте элементов интерфейса применяется для автоматического распознавания и анализа компонентов пользовательского интерфейса, таких как кнопки, поля ввода и меню. Это помогает улучшить пользовательский опыт и эффективность работы с приложениями и веб-сайтами.

На рисунке 15 представлен процесс обнаружения объектов.



Рисунок 15 – Обнаружение объектов

Таким образом был рассмотрен процесс обнаружения объектов пользовательского интерфейса с использованием технологий машинного обучения. Процесс машинного обучения схож с обучением человека, где на основе входных данных модель классифицирует или регрессирует результат.

2.4 Выбор алгоритма обнаружения объектов

Методы обнаружения объектов делятся на методы, основанные на машинном обучении (традиционные методы обнаружения объектов) и методы, основанные на глубоком обучении (методы обнаружения объектов на основе глубокого обучения). Методы обнаружения объектов эволюционировали от первых ко вторым.

Традиционные методы обнаружения объектов, основанных на машинном обучении, используют любой из следующих методов для определения функций и используют такие методы, как машины опорных векторов (SVM), для классификации функций:

- система обнаружения объектов Виолы-Джонса, основанная на функциях Хаара,

- преобразование масштабно-инвариантных признаков (SIFT)а,

- гистограмма особенностей ориентированного градиента (HOG).

Методы обнаружения объектов на основе глубокого обучения, основанные на глубоком обучении, сквозное обнаружение объектов (E2E) может выполняться без определения функций, обычно на основе сверточной нейронной сети (CNN). Методы обнаружения объектов на основе глубокого обучения можно разделить на одноэтапные и двухэтапные методы, а также алгоритм RefineDet, который унаследовал преимущества первых алгоритмов.

Одноуровневые алгоритмы обнаружения объектов представляют собой методику, при которой информация о категории и местоположении объектов предоставляется напрямую через магистральную сеть, без использования сети предложения региона (RPN). Одноуровневый алгоритм обнаружения объектов работает быстрее, однако страдает точность, нежели у двухуровневого алгоритма.

Классические алгоритмы одноэтапного обнаружения объектов, следующие:

- YOLO (You Only Look Once),

- SSD (Single Shot MultiBox Detector),

- RetinaNet.

YOLO (You Only Look Once) представляет собой одноэтапный алгоритм обнаружения объектов, который работает очень быстро, так как выполняет обнаружение и классификацию объектов в одной сети, прогоняя изображение через сверточную сеть и выдающую предсказания сразу для всех объектов на изображении.

SSD (Single Shot MultiBox Detector) также является одноэтапным алгоритмом, который интегрирует метод MultiBox для предложения регионов

и применяет несколько слоев свертки для обнаружения объектов различных размеров на изображении.

Двухуровневый алгоритм обнаружения объектов выполняет обнаружение через свёрточную нейронную сеть (ConvNet/CNN) и извлекает сверточные функции. Обучение сети состоит из двух частей: первый шаг — обучение сети региональных интересов (RPN), второй — классификация и уточнение границ регионов (обнаружение области объекта).

RetinaNet также одноэтапный и разработан для решения проблемы несбалансированных данных при обнаружении объектов, используя функцию потерь Focal Loss, что помогает справляться с проблемой низкого качества предсказаний для редких классов.

RefineDet является усовершенствованием алгоритма SSD, вводя двухэтапный подход с использованием дополнительного модуля уточнения, который позволяет улучшить точность обнаружения объектов за счет последующего уточнения их границ.

Каждый из этих алгоритмов имеет свои особенности и применяется в зависимости от конкретной задачи и требований к скорости и точности обнаружения.

Двухуровневый алгоритм обнаружения объектов сначала генерирует серию кадров-кандидатов в качестве выборок, а затем эти выборки классифицируются сверточной нейросетью. двухуровневый алгоритм имеет более высокую точность сети, однако по сравнению с одноуровневым алгоритмом более медленный. Типичные Двухуровневые алгоритмы обнаружения объектов, следующие: R-CNN, Fast R-CNN и Faster R-CNN.

R-CNN был одним из первых алгоритмов, который использовал метод "областей с сверточными нейронными сетями" для обнаружения объектов. Он работает следующим образом:

- сначала из изображения извлекаются кандидаты на объекты с помощью методов, таких как метод выбора избыточных квадратов (Selective

Search), который выделяет потенциальные области, где могут находиться объекты,

- каждый из найденных кандидатов регионов подается на вход сверточной нейронной сети для извлечения признаков,
- извлеченные признаки используются для классификации объекта и уточнения его границ.

Однако R-CNN имел значительные недостатки в скорости из-за необходимости отдельного вычисления признаков для каждого кандидата региона.

Fast R-CNN усовершенствует алгоритм R-CNN посредством использования общей сверточной сети. В отличие от R-CNN, где каждый кандидат региона обрабатывался отдельно, Fast R-CNN использует общую сверточную сеть для извлечения признаков из всего изображения один раз.

После извлечения признаков для всего изображения, Fast R-CNN применяет классификацию и регрессию для каждого кандидата региона, что значительно упрощает процесс и ускоряет работу алгоритма.

Faster R-CNN вводит сеть региональных предложений (англ. Region Proposal Network), которая интегрируется с основной сверточной сетью и позволяет генерировать предложения регионов на лету, значительно ускоряя процесс по сравнению с Fast R-CNN. RPN предсказывает регионы, которые затем используются для извлечения признаков и дальнейшей классификации объектов. Это позволяет значительно ускорить процесс обнаружения объектов по сравнению с Fast R-CNN.

Нейронная сеть однократного уточнения для обнаружения объектов (RefineDet) — это усовершенствование, основанное на алгоритме однократного обнаружения (SSD). Он наследует преимущества одноэтапных и двухэтапных методов обнаружения объектов и преодолевает их недостатки.

RefineDet добавляет дополнительный модуль, который уточняет предсказания границ объектов, что помогает улучшить точность даже для

объектов с малым количеством образцов в обучающем наборе. Этот подход сочетает в себе преимущества как одноэтапных, так и двухэтапных методов обнаружения объектов, минимизируя их недостатки.

Каждый из этих алгоритмов имеет свои уникальные особенности и применяется в зависимости от требований к скорости, точности и ресурсам, доступным для обработки изображений.

2.5 Сравнение методов обнаружения объектов

Традиционный метод обнаружения объектов требует, чтобы вы вручную проектировали функции и получали кадры-кандидаты через скользящее окно. Затем для определения области объекта используется традиционный классификатор. Весь процесс обучения разбит на несколько этапов. Метод обнаружения объектов на основе глубокого обучения использует машины для изучения функций и получения объектов-кандидатов с помощью более эффективного предложения или прямой регрессии. Он имеет более высокую точность и производительность в реальном времени.

На рисунке 16 представлена структура метода обнаружения объектов.

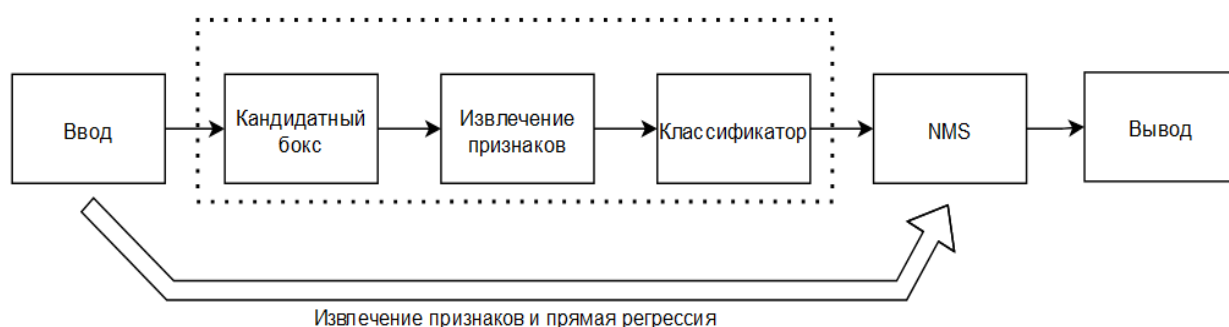


Рисунок 16 – Структура метода обнаружения объектов

Ввод: исходные данные, которые обрабатываются системой.

Кандидатный бокс: область изображения, которая потенциально может содержать объект.

Извлечение признаков: процесс извлечения характеристик из кандидатного бокса, которые используются для классификации объекта.

Классификатор: модель машинного обучения, которая определяет, является ли объект в кандидатном боксе объектом или фоном.

NMS: немаксимальное подавление, метод, который используется для удаления дублирующихся обнаружений объектов.

Выход: окончательное определение объекта или фона для каждого кандидатного бокса.

2.6 Выбор фреймворка

Фреймворк – программная платформа, которая определяет структуру и возможности разработки программной системы и упрощает процесс разработки программных решений.

Для выбора подходящего фреймворка важно учитывать не только функциональные возможности, но и особенности проекта, требования к производительности, удобство использования и экосистему поддержки.

Каждый фреймворк имеет свои специфические алгоритмы и подходы к реализации машинного обучения. В зависимости от требований проекта, необходимо учитывать производительность фреймворка при обучении и развертывании моделей.

Важно выбирать фреймворк с активным сообществом и поддержкой, так как это обеспечивает доступ к обучающим материалам, библиотекам, апдейтам и решению проблем. TensorFlow и PyTorch обладают большими и активными сообществами разработчиков и исследователей.

При выборе фреймворка следует учитывать его возможности интеграции с другими системами и инструментами, а также удобство развертывания обученных моделей.

2.6.1 Платформа машинного обучения

Самыми востребованными вариантами фреймворков и библиотек машинного обучения являются Scikit Learn, TensorFlow, Pytorch и Keras.

Scikit Learn — это универсальная платформа для машинного обучения, предлагающая внутреннюю реализацию различных алгоритмов для классификации, регрессии и кластеризации. Она также предоставляет библиотеки инструментов для задач, таких как уменьшение размерности

данных, выбор модели и предварительная обработка данных. Scikit Learn отличается легкостью установки и использования. Помимо упомянутых возможностей, Scikit Learn особенно популярен среди начинающих благодаря простоте в использовании и хорошей интеграции с другими библиотеками Python, такими как NumPy и pandas. Это делает Scikit Learn отличным выбором для классических задач машинного обучения, таких как обучение моделей, оценка и работа с небольшими и средними наборами данных.

На рисунке 17 представлен пример алгоритма с использованием TensorFlow для распознавания рукописных цифр.

```
Ввод [3]: import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import matplotlib.pyplot as plt

(train_images, train_labels), (test_images, test_labels) = datasets.mnist.load_data()
train_images, test_images = train_images / 255.0, test_images / 255.0

model = models.Sequential()
model.add(layers.Flatten(input_shape=(28, 28)))
model.add(layers.Dense(128, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

history = model.fit(train_images, train_labels, epochs=5)

test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=2)
print('\nТочность на тестовых данных:', test_acc)

predictions = model.predict(test_images)

plt.figure(figsize=(6, 3))
plt.subplot(1, 2, 1)
plt.imshow(test_images[0], cmap=plt.cm.binary)
plt.title("Исходное изображение")
plt.subplot(1, 2, 2)
plt.bar(range(10), predictions[0])
plt.title(f"Предсказание: {predictions[0].argmax()}")
plt.xlabel("Класс")
plt.ylabel("Вероятность")
plt.show()
```

Рисунок 17 – Алгоритм для распознавания рукописных цифр

TensorFlow, Keras и Pytorch на данный момент являются основными инструментами для глубокого обучения. Их можно использовать для вызова различных алгоритмов глубокого обучения. Эти платформы обладают обширной документацией, активным сообществом пользователей и множеством примеров.

TensorFlow является одной из самых мощных библиотек для глубокого обучения. Он поддерживает как низкоуровневое, так и высокоуровневое программирование, что позволяет пользователям создавать модели различной

сложности. TensorFlow также предоставляет TensorFlow Extended (TFX) для построения производственных рабочих процессов и TensorFlow Lite для развертывания моделей на мобильных устройствах.

Keras – это API (от англ. application programming interface) платформы Tensorflow. Предоставляет комфортный интерфейс для решения задач машинного обучения.

Keras изначально был создан как высокоуровневый интерфейс для TensorFlow, что делает его более доступным для новичков и позволяет быстро прототипировать модели глубокого обучения. В Keras доступен широкий спектр предобученных моделей и инструментов для их кастомизации.

PyTorch выделяется поддержкой графических процессоров и использованием автоматического дифференцирования в обратном режиме, что позволяет изменять графики вычислений в процессе. Это делает его популярным выбором для быстрого экспериментирования и прототипирования.

Исходя из выше сказанного в ходе анализа существующих вариантов платформ были выбраны решения для реализации методики оценки пользовательского интерфейса.

PyTorch отличается динамическим вычислительным графом, который позволяет изменять граф на лету, что делает его более гибким и интуитивно понятным. Это также делает его отличным выбором для исследований и разработки новых алгоритмов. PyTorch активно используется в научных исследованиях и быстро набирает популярность среди инженеров благодаря своей простоте и производительности.

3 Разработка методики оценки пользовательского интерфейса

В данной главе будет описание набора данных для анализа и обучения нейронной сети и представление работы метода оценки пользовательского интерфейса.

3.1 Описание набора данных для обучения

Для эффективной оценки интерфейсов требуется надежные набор данных. Особое внимание стоит уделить снимкам экранов интерфейса и иерархии представлений, которые являются важными аспектами для для анализа.

3.1.1 Снимки экрана интерфейса и иерархии представлений

Для тренировки нейронной сети был использован датасет Riso содержащий данные о дизайне более чем 9 тысяч приложений, охватывающих 27 категорий.

На рисунке 18 представлен датасет Riso.

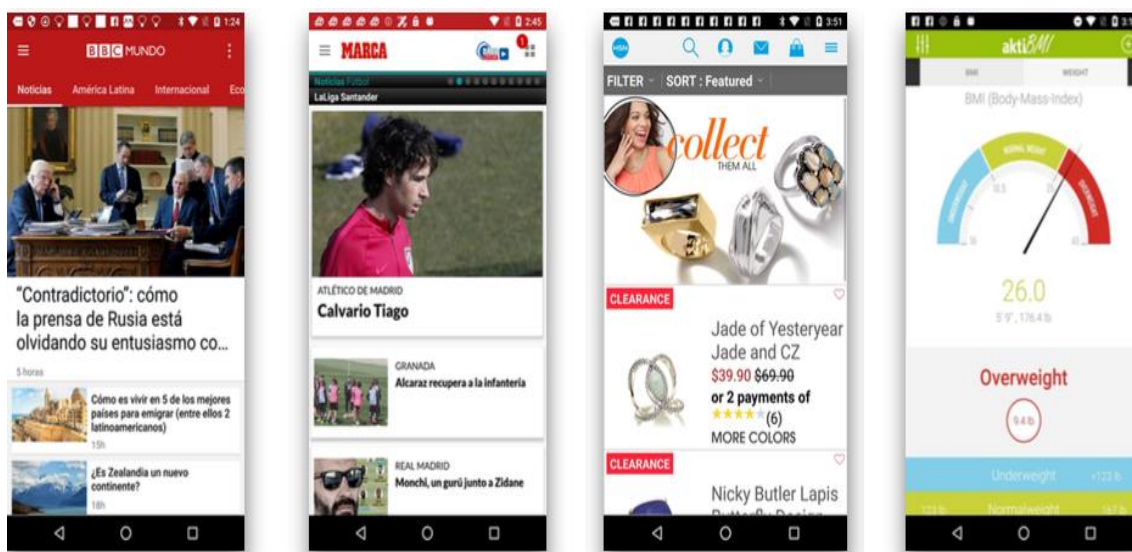


Рисунок 18 – Датасет Riso

Для того чтобы создать обучающие входные данные, было создано новое изображение для каждого пользовательского интерфейса, фиксируя области ограничивающей рамки всех конечных элементов в его иерархии представлений, различая текстовые и нетекстовые элементы. Иерархии представлений Riso устраняют необходимость в обработке зашумленных

изображений или методах оптического распознавания символов для создания этих входных данных.

3.2 Подготовка данных для обучения

Для обучения нейронной сети для оценки пользовательского интерфейса данные были классифицированы, где изображения разделены по классам в разные папки. Это стандартный подход для обучения моделей на основе глубокого обучения, таких как сверточные нейронные сети (CNN). Каждая папка представляет собой класс (или категорию), к которой относится изображение. Изображения каждого класса располагаются в отдельной папке внутри папки обучающего набора.

На рисунке 19 изображена структура данных, которая будет использована для обучения.

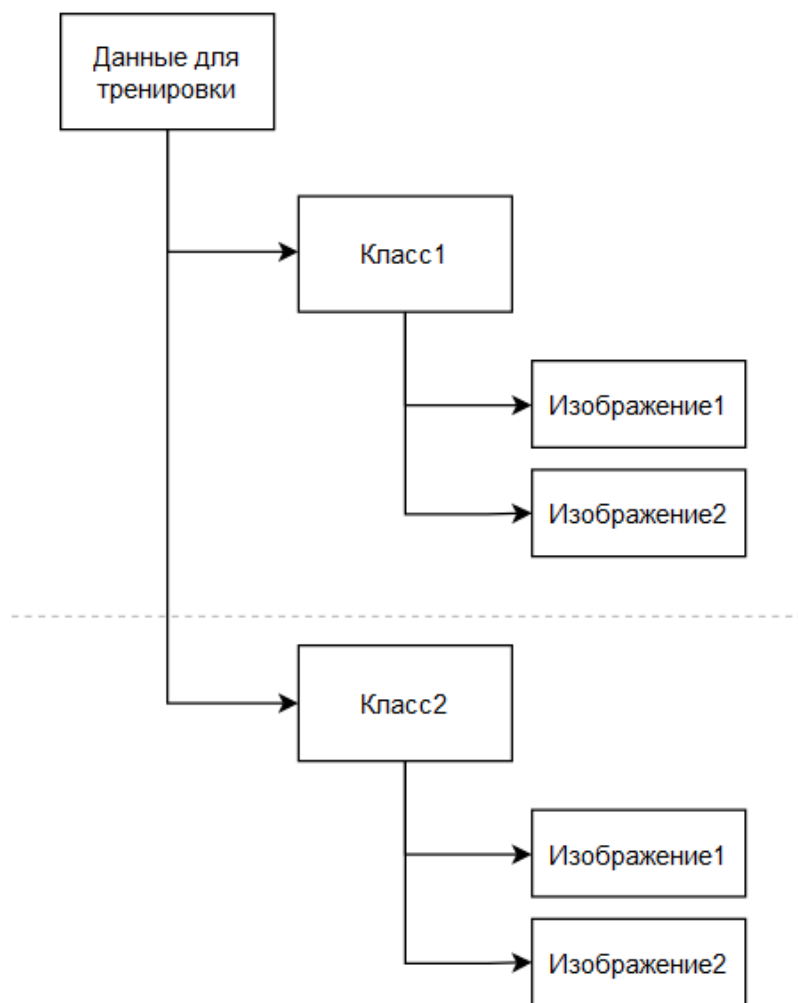


Рисунок 19 – Структура данных для обучения

Таким же образом устроена структура тестового (валидационного) набора данных, который используется для оценивания модели на этапе её обучения.

Также для обучения нейронной сети используется CSV файл, который представляет текстовый файл, где данные между собой разделены запятыми.

На рисунке 20 изображена содержание CSV файла, которая будет использована для обучения.

1	image,score,valid	
2	0001_app-inventor,0.17,False	
3	0002_app-inventor,0.00,False	
4	0003_app-inventor,0.00,False	
5	0005_app-inventor,0.00,False	
6	0006_app-inventor,0.00,False	
7	0007_app-inventor,0.00,False	
8	0008_app-inventor,0.00,False	
9	0009_app-inventor,0.00,False	
10	0013_app-inventor,0.00,False	
11	0014_app-inventor,0.00,False	
12	0015_app-inventor_our-apps,0.67,False	
13	0016_app-inventor,0.00,False	
14	0017_app-inventor,0.17,False	
15	0019_app-inventor,0.67,False	
16	0021_app-inventor,0.17,False	
17	0022_app-inventor,0.00,False	
18	0024_app-inventor,0.33,False	
19	0025_app-inventor,0.00,False	
20	0026_app-inventor,0.00,False	
21	0027_app-inventor,0.00,False	
22	0028_app-inventor_our-apps,0.83,False	
23	0030_app-inventor,0.00,False	

Рисунок 20 – CSV файл

Имя изображения в файле - image, которое используется для нахождения соответствующего файла изображения в исходной директории. Столбец score является оценкой пользовательского интерфейса, которая представлена в диапазоне от 0.0 до 1.0. Столбец valid отвечает за булевское значение (True/False), указывающее, является ли изображение частью валидационного набора данных.

На основе информации из CSV файла нужно распределять изображения по классам и директориям (тренировочные и валидационные данные).

3.3 Нейронная сеть для оценки пользовательского интерфейса

Каждая нейронная сеть состоит из слоев узлов, или искусственных нейронов: входного слоя, одного или нескольких скрытых слоев и выходного слоя. Каждый узел соединяется с другими и имеет свой собственный вес и пороговое значение.

Если выходной сигнал любого отдельного узла превышает заданное пороговое значение, этот узел активируется, отправляя данные на следующий слой сети. В противном случае данные не передаются на следующий уровень сети.

В современных нейронных сетях широко используются активационные функции, такие как ReLU (Rectified Linear Unit) и их модификации, которые обычно обеспечивают стабильное и быстрое обучение глубоких моделей. Но для данной работы будет использована сигмовидная функция.

Преимущества использования сигмовидной функции включают их гладкость и дифференцируемость, что важно для обратного распространения ошибки в процессе обучения нейронных сетей. Однако она может подвержена проблеме затухания градиента при больших или маленьких значениях входа, что снижает её эффективность в глубоких нейронных сетях.

Для реализации алгоритма была использована сигмовидная функция, которая изображена на рисунке 21.

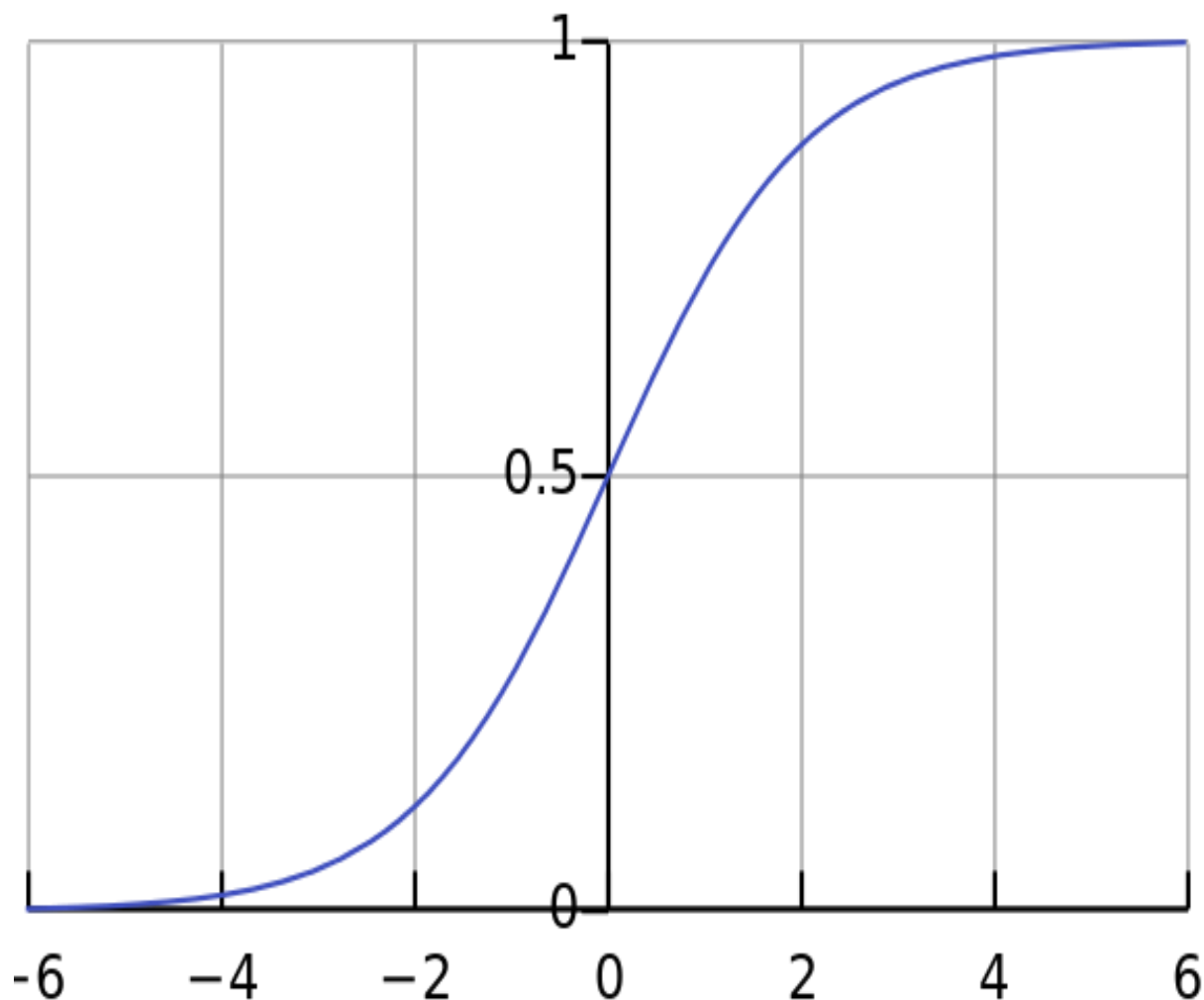


Рисунок 21 – Сигмовидная функция

Каждый слой связан с несколькими входами и для начала вычисляет взвешенную сумму полученных на вход значений, умножая значение на входе на вес синапса и далее суммирует с аналогичными произведениями. К этому значению суммируется смещение, которое сжимается в диапазон от -1 до 1 используя нелинейную функцию активации. Нейронные сеть обучаются на больших объемах данных, что позволяет автоматически настраивать веса для минимизации ошибки. Это позволяет эффективно распознавать сложные паттерны и принимать решения на основе новых данных.

На рисунке 22 представлен нейронный слой нейронной сети.

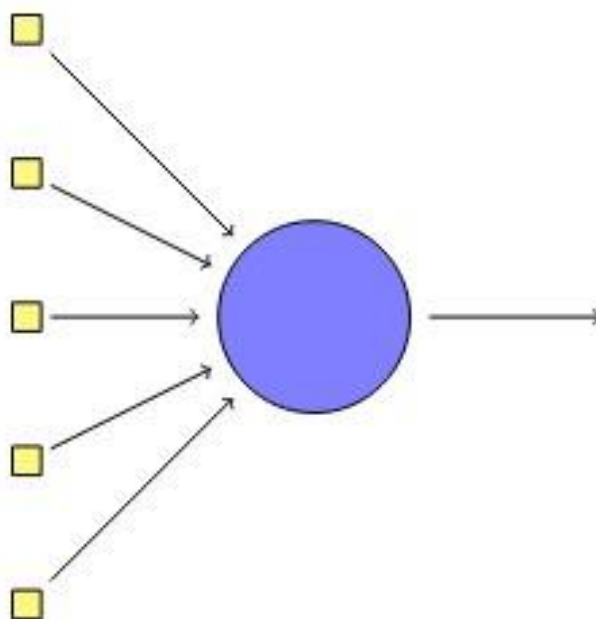


Рисунок 22 – Нейронный слой

При соединении нескольких слоев получается нейронная сеть, визуализация которой представлена на рисунке 23.

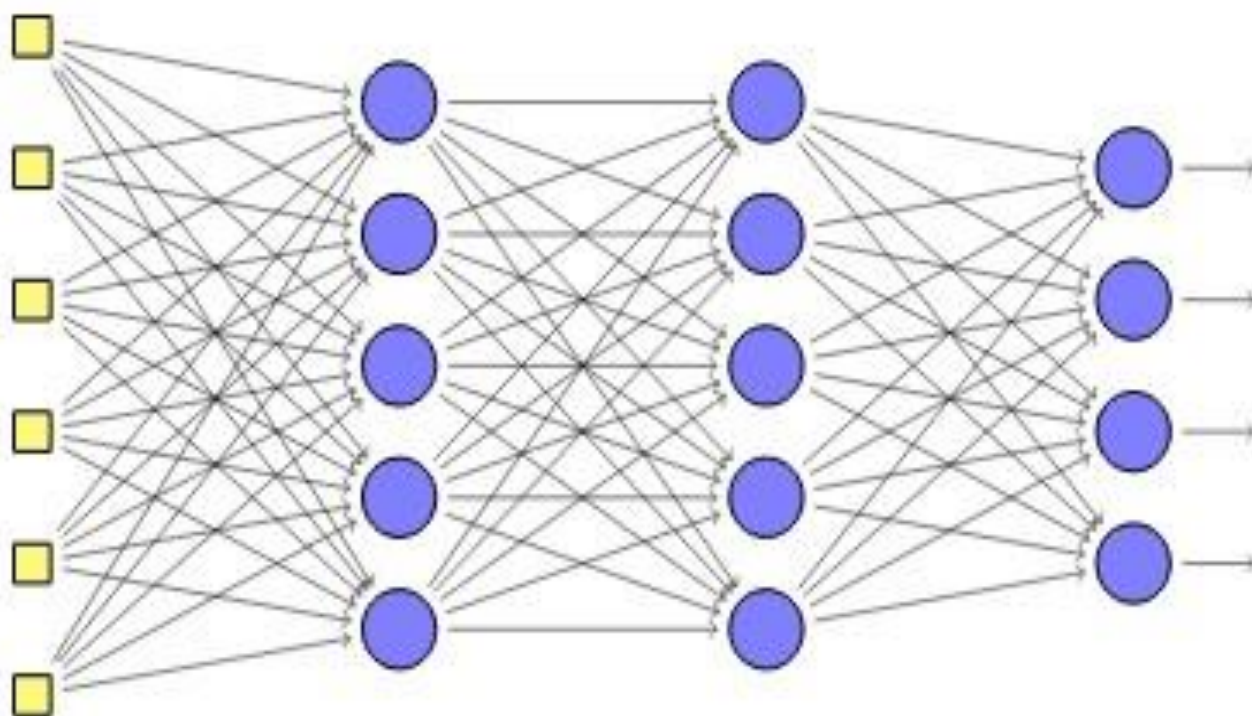


Рисунок 23 – Нейронная сеть

Визуализация каждого слоя по отдельности как обособленные объекты представлена на рисунке 24.

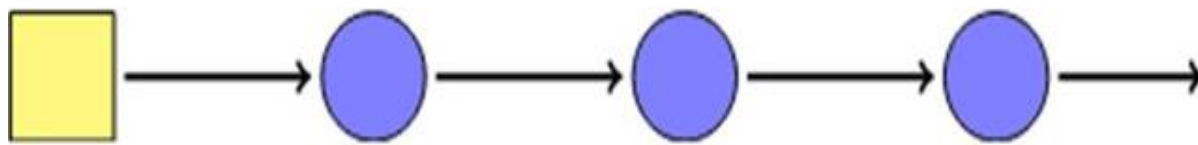


Рисунок 24 – Слой

Дальше будет рассматриваться визуализация нейронной сети, где один круг обозначает целый слой сети, а стрелки представляют векторы значений.

Работа нейронной сети зависит от набора весов и смещений каждого узла, поэтому нам нужно настроить эти параметры для корректного функционирования сети.

Для начала необходимо вычислить насколько точно данное выходное значение учитывает входное.

Определив это, появляется возможность использования обратного распространения. Этот метод сводится к обновлению каждого из весов в сети так, чтобы фактический результат был ближе к целевому выходу, тем самым минимизируя ошибку для каждого выходного нейрона и сети в целом.

Проанализировав, были выбраны свойства, которыми должна обладать построенная нейронная сеть:

- объективная оценка, независимая от изменений в дизайне,
- отсутствие зависимости от расположения отдельных компонентов интерфейса,
- учёт взаимосвязи между компонентами интерфейса.

Для чтения и обработки CSV файла используется библиотека pandas.

Ниже представлена часть кода, где создаются генераторы данных для аугментации тренировочных данных и нормализации валидационных данных.

```
import os
import shutil
import pandas as pd
import numpy as np
```

```

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten,
Dense, Dropout

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image


target_width, target_height = 239, 408


train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest')


validation_datagen = ImageDataGenerator(rescale=1./255)


train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(target_width, target_height),
    batch_size=32,
    class_mode='categorical')


validation_generator =
validation_datagen.flow_from_directory(
    validation_dir,
    target_size=(target_width, target_height),
    batch_size=32,
    class_mode='categorical')

```

Для создания используется `flow_from_directory` для загрузки данных из директорий. Данные масштабируются до размера 239x408 и нормализуются (значения пикселей делятся на 255).

Далее определяется архитектура модели, состоящей из сверточных и полносвязных слоев. Модель заканчивается 10-выходным `softmax`-слоем для классификации.

Компилируется модель с использованием функции потерь `categorical_crossentropy`, оптимизатора `Adam` и метрики `accuracy`.

```
model = Sequential([
    Conv2D(32, (3, 3), activation='relu',
input_shape=(target_width, target_height, 3)),
    MaxPooling2D(2, 2),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),
    Dense(10, activation='softmax') # 10 категорий для
оценок от 1 до 10
])

model.compile(loss='categorical_crossentropy',

optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
metrics=['accuracy'])
```

После создания модели вычисляются количество шагов на эпоху для обучения и проверки на основе количества образцов и размера батча.

Модель обучается на тренировочных данных и проверяется на валидационных данных.

```

        steps_per_epoch      =      train_generator.samples      //
train_generator.batch_size
        validation_steps      =      validation_generator.samples  //
validation_generator.batch_size

```

```

history = model.fit(
    train_generator,
    steps_per_epoch=steps_per_epoch,
    epochs=30,
    validation_data=validation_generator,
    validation_steps=validation_steps)

```

После обучения модели на представленных данных функция загружает изображение, подготавливает его для модели, предсказывает оценку и возвращает её.

```

def predict_score(image_path):
    img = image.load_img(image_path,
target_size=(target_width, target_height))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array /= 255.

    prediction = model.predict(img_array)
    score = np.argmax(prediction) + 1 # Оценки от 1 до 10
    return score

```

Этот код инициализирует, обучает и использует модель глубокого обучения для классификации изображений, поддерживает аугментацию данных для улучшения обобщающей способности модели и предоставляет простой интерфейс для предсказания на новых изображениях.

3.4 Результат работы приложения

При запуске приложение встречает интерфейс который представлен на рисунке 25.

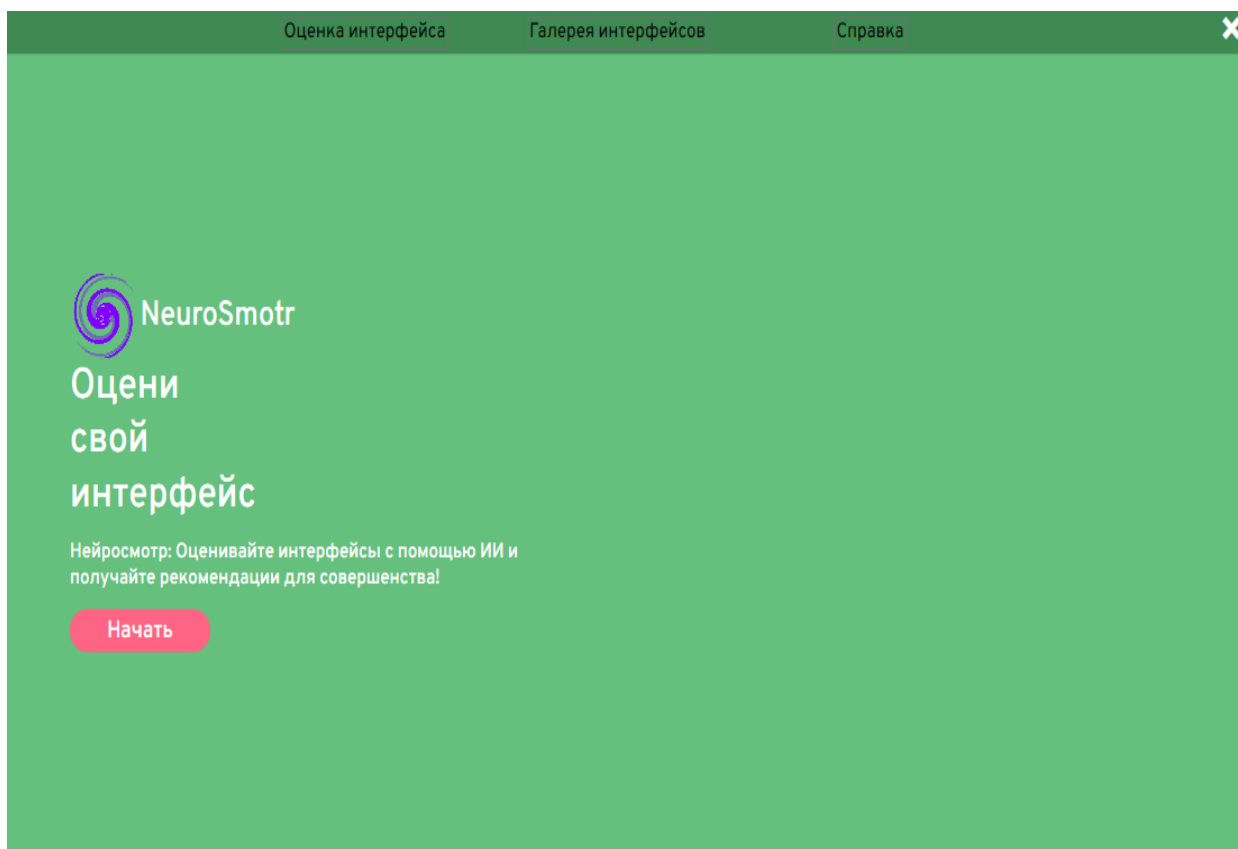


Рисунок 25 – Стартовое окно приложения

После нажатия на соответствующую кнопку “Начать” необходимо выбрать изображение интерфейса, после чего приложение выдаст результат

На данный момент приложение может выставлять оценку от 1 до 10, но при дальнейшей разработке возможно добавление функции для предоставления рекомендаций.

При загрузке неправильного формата файла, приложение выдаст ошибку, что формат файла неверный в ином же случае при загрузке изображения проработанного интерфейса, который соответствует современным стандартам приложение выдает высокую оценку загруженному пользовательскому интерфейсу.

На рисунке 26 представлен вариант работы приложения при проработанном интерфейсе.

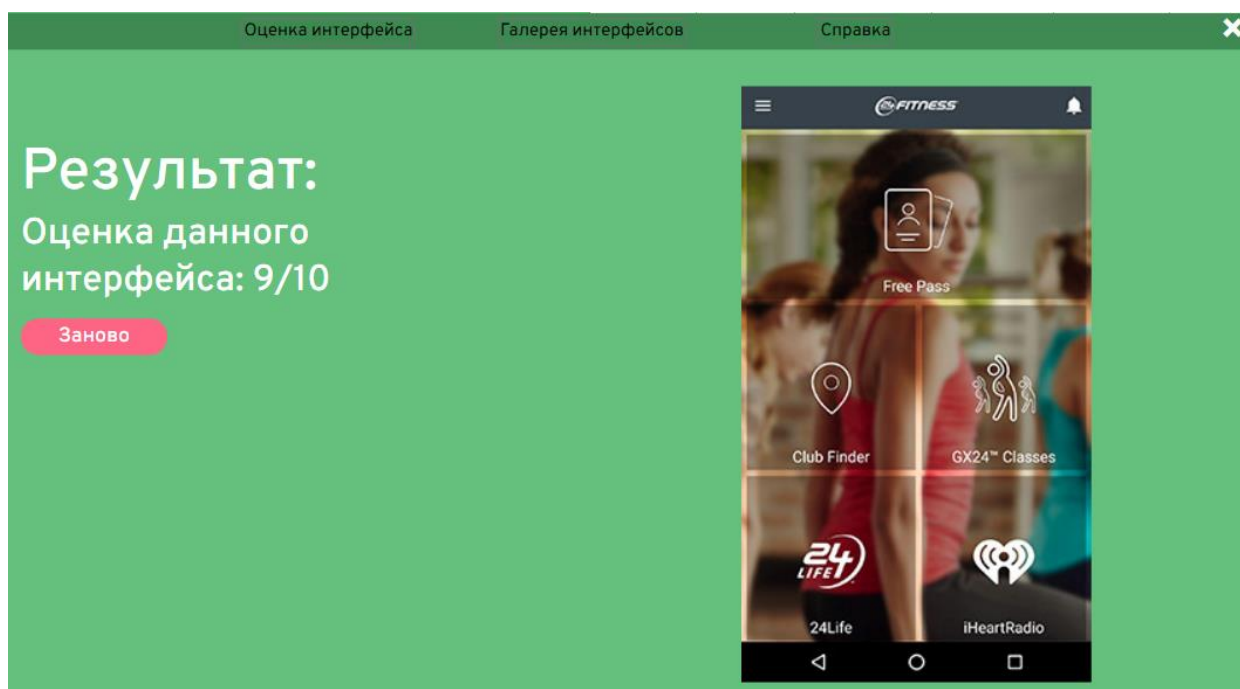


Рисунок 26 – Результат работы приложения на проработанном интерфейсе

Если загруженный интерфейс хуже, то выдает более низкую оценку, которая представлена на рисунке 27.

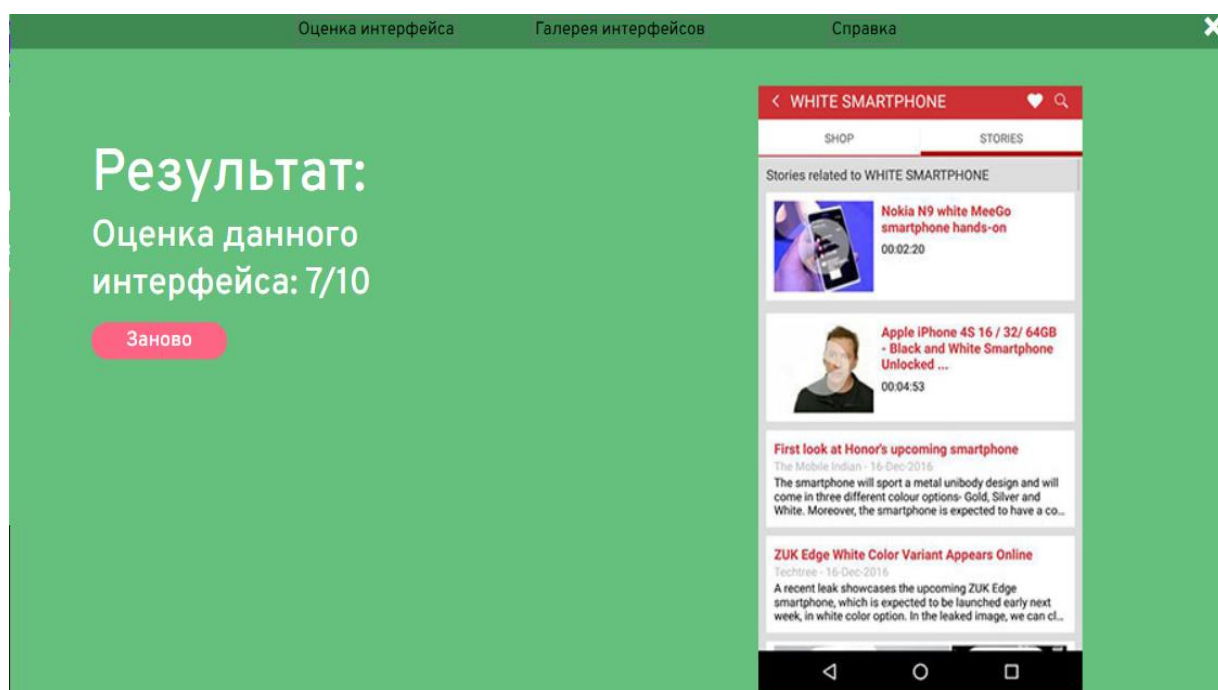


Рисунок 27 – Результат работы на менее проработанном интерфейсе

Если интерфейс не соответствует нынешним стандартам, то приложение поставит низкую оценку, и предложит доработать интерфейс.

Пример работы на плохом интерфейсе представлен на рисунке 28.

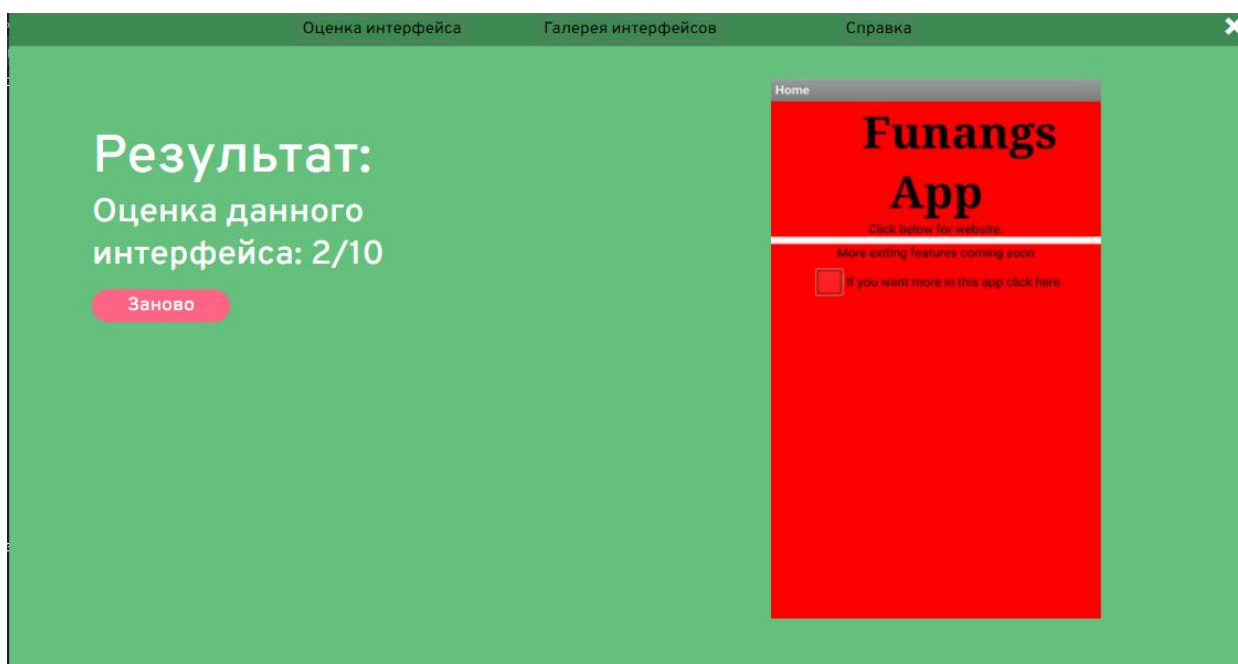


Рисунок 28 – Результат работы приложения на непроработанном интерфейсе

Также приложение позволяет просмотреть отобранные нейронной сетью интерфейсы.

Пример работы галереи интерфейсов представлен на рисунке 29.

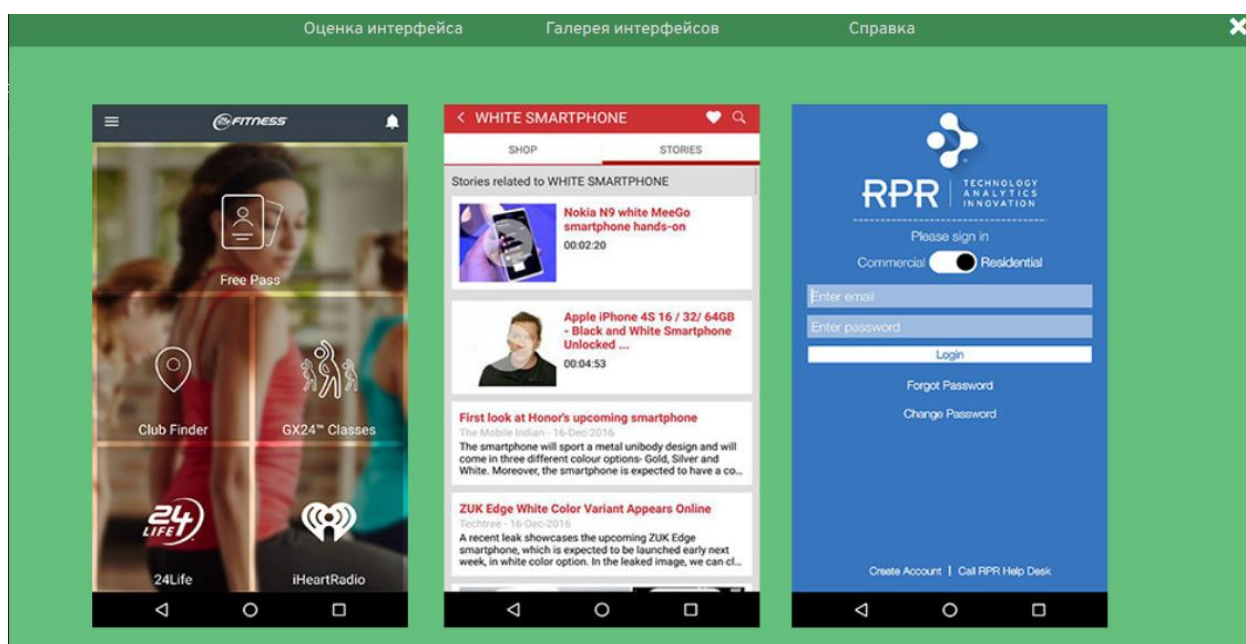


Рисунок 29 – Галерея интерфейсов

Таким образом разделе была описана работа итоговой методики оценки пользовательского интерфейса, продемонстрирована её работа, а также составлено руководство пользователя, где подробно с иллюстрациями был описан алгоритм взаимодействия пользователя с приложением.

ЗАКЛЮЧЕНИЕ

В процессе выпускной квалификационной работы были решены следующие задачи:

- выявление проблем и определение основных задач оценки пользовательских интерфейсов,
- исследование основных принципов оценки пользовательского интерфейса,
- разработка нотаций для описания структуры и логики работы пользовательского интерфейса,
- разработка методики оценки пользовательского интерфейса.

В данной работе были рассмотрены методики оценки качества пользовательских интерфейсов. В ходе их анализа были выявлено, что используемые методы не позволяют целостно оценить качество интерфейса и имеют определенные ограничения. Таким образом для решения задачи оценки пользовательского интерфейса был предложен метод оценки с использованием машинного обучения.

В процессе исследования было проведено выявление ключевых проблем, связанных с оценкой пользовательских интерфейсов. Были рассмотрены существующие подходы и методики, используемые в данной области, и определены их слабые стороны. На основе этого анализа были сформулированы основные задачи, которые необходимо решить для разработки эффективной методики оценки пользовательских интерфейсов.

Одним из важнейших этапов работы стало исследование основных принципов оценки пользовательского интерфейса. Внимание уделялось таким аспектам, как удобство использования, интуитивность, визуальная привлекательность и функциональность. Были изучены различные подходы к оценке этих параметров и определены наиболее подходящие для использования в разработанной методике.

Разработка нотаций для описания структуры и логики работы пользовательского интерфейса стала следующим этапом. Эти нотации позволили структурировать процесс оценки и сделать его более прозрачным и понятным. Были разработаны схемы и модели, описывающие взаимодействие пользователя с интерфейсом, а также логика работы различных элементов интерфейса.

На основе проведенных исследований и разработанных нотаций была создана методика оценки пользовательского интерфейса. Основой этой методики стало использование машинного обучения, что позволило значительно повысить точность и объективность оценки. Были разработаны алгоритмы, которые анализируют данные о взаимодействии пользователей с интерфейсом и выдают оценку его качества.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Нильсен, Дж. Юзабилити: дизайнерский подход. [Электронный ресурс]. URL: <https://www.nngroup.com/books/usability> (дата обращения: 26.04.2024).
2. AdobeXD [Электронный ресурс]. URL: <https://helpx.adobe.com/ru/support/xd.html> (дата обращения 26.04.2024).
3. Figma [Электронный ресурс]. URL: <https://www.figma.com/> (дата обращения 26.04.2024).
4. Scketch [Электронный ресурс]. URL: <https://scketch.com/> (дата обращения 26.04.2024).
5. Usabilla [Электронный ресурс]. URL: <https://demo.usabilla.com/> (дата обращения 26.04.2024).
6. UsabilityHub [Электронный ресурс]. URL: <https://usabilityhub.com/> (дата обращения 26.04.2024).
7. Userbrain [Электронный ресурс]. URL: <https://www.userbrain.com/en/> (дата обращения 26.04.2024).
8. Арлоу, Д. UML 2 и Унифицированный процесс. Второе издание / Д. Арлоу, А. Нейштадт. – СПб.: Питер, 2006. – 624 с. (дата обращения 26.04.2024).
9. Галиаскаров, Э. Г. Анализ и проектирование систем с использованием UML: учебное пособие для вузов / Э. Г. Галиаскаров, А. С. Воробьев. — Москва: Издательство Юрайт, 2024. — 125 с. (дата обращения 26.04.2024).
10. Пономарев, И.А. Методы оценки качества пользовательского интерфейса. [Электронный ресурс]. URL: <http://it-claim.ru/Library/Books/ITS/wwwbook/ist6/ponomarev2/ponomarev2.htm> – (дата обращения: 26.04.2024)

11. Тул, Дж. Основы проектирования интерфейсов пользователя: практическое руководство. [Электронный ресурс]. URL: <https://uxdesign.com/guide> (дата обращения: 26.04.2024).

12. Головач, В.В. Дизайн пользовательского интерфейса: искусство мыть слона [Электронный ресурс]. URL: <http://uibook2.usetheics.ru> –(дата обращения: 26.04.2024)

ПРИЛОЖЕНИЕ А

(Обязательное)

Исследование и разработка методики оценки пользовательского интерфейса

Листинг программы

Файл main.py

```
import os
import shutil
import pandas as pd
import numpy as np
import tensorflow as tf

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten,
Dense, Dropout
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image

# Пути к директориям с изображениями и метками
image_source_dir = 'path/to/rico_screens'
train_dir = 'path/to/train_data'
validation_dir = 'path/to/validation_data'
csv_file = 'path/to/labels.csv'

# Создание директорий для классов (1-10)
for i in range(1, 11):
    os.makedirs(os.path.join(train_dir, str(i)), exist_ok=True)
    os.makedirs(os.path.join(validation_dir, str(i)), exist_ok=True)

# Загрузка данных из CSV
df = pd.read_csv(csv_file)

# Проверка наличия всех изображений
missing_images = [img for img in df['image'] if not
os.path.exists(os.path.join(image_source_dir, img))]
if missing_images:
    raise ValueError(f"Missing images in source directory:
{missing_images}")

# Копирование изображений в соответствующие папки
```

```

for index, row in df.iterrows():
    image_name = row['image']
    score = row['score']
    valid = row['valid']

    # Округление оценки до ближайшего целого числа от 1 до 10
    label = round(score * 10)
    label = min(max(label, 1), 10)

    src_path = os.path.join(image_source_dir, image_name)

    # Определение пути назначения (train или validation в зависимости
    от значения valid)
    if valid:
        dst_path = os.path.join(validation_dir, str(label), image_name)
    else:
        dst_path = os.path.join(train_dir, str(label), image_name)

    shutil.copyfile(src_path, dst_path)

# Проверка наличия данных в директориях
for i in range(1, 11):
    train_class_dir = os.path.join(train_dir, str(i))
    validation_class_dir = os.path.join(validation_dir, str(i))
    if len(os.listdir(train_class_dir)) == 0:
        print(f"Warning: No training images found for class {i}")
    if len(os.listdir(validation_class_dir)) == 0:
        print(f"Warning: No validation images found for class {i}")

# Подготовка данных
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,

```

```

height_shift_range=0.2,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode='nearest')

validation_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='categorical')

validation_generator = validation_datagen.flow_from_directory(
    validation_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='categorical')

# Создание модели
model = Sequential([
    Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)),
    MaxPooling2D(2, 2),
    Conv2D(64, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),

```

```

        Dense(10, activation='softmax') # 10 категорий для оценок от 1 до
10
    ])

model.compile(loss='categorical_crossentropy',
              optimizer=tf.keras.optimizers.Adam(learning_rate=0.001),
              metrics=['accuracy'])

# Расчет количества шагов для обучения и проверки
steps_per_epoch = train_generator.samples // train_generator.batch_size
validation_steps = validation_generator.samples //
validation_generator.batch_size

# Обучение модели
history = model.fit(
    train_generator,
    steps_per_epoch=steps_per_epoch,
    epochs=30,
    validation_data=validation_generator,
    validation_steps=validation_steps)

# Функция для предсказания оценки
def predict_score(image_path):
    img = image.load_img(image_path, target_size=(150, 150))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array /= 255.

    prediction = model.predict(img_array)
    score = np.argmax(prediction) + 1 # Оценки от 1 до 10
    return score

```

ПРИЛОЖЕНИЕ В

(Обязательное)

Исследование и разработка методики оценки пользовательского интерфейса

Презентация

ИССЛЕДОВАНИЕ И РАЗРАБОТКА МЕТОДИКИ ОЦЕНКИ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

выполнил: Тумеркин Д.Р.
Руководитель: Эпп В.В.

Задачи и цели работы

Цель работы – разработка методики, которая оценивает интерфейс приложений на предмет удобства использования и дизайна.

Для достижения цели необходимо выполнение следующих задач:

- выявление проблем и определение основных задач оценки пользовательских интерфейсов
- исследование основных принципов оценки пользовательского интерфейса
- разработка нотаций для описания структуры и логики работы пользовательского интерфейса
- разработка методики оценки пользовательского интерфейса

Актуальность работы

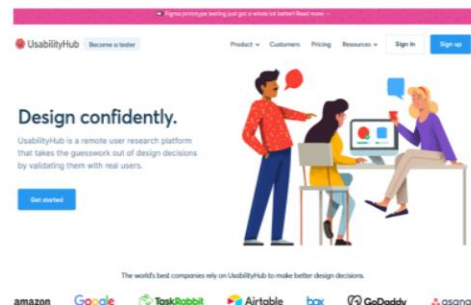
- Вне зависимости от типа программного продукта, будь то мобильное приложение, настольное приложение или веб-ресурс, пользовательский интерфейс является неотъемлемой его частью.
- Процесс оценки пользовательского интерфейса может оказаться долгим, что увеличит в целом длительность разработки программного обеспечения
- Поэтому разработка методики оценки пользовательского интерфейса с использованием нейронных сетей является актуальной

3

Обзор аналогов



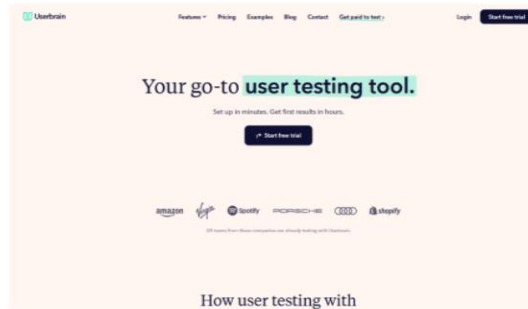
Инструмент Usabilla



Платформа UsabilityHub

4

Обзор аналогов



Интерфейс Userbrain

5

Достоинства и недостатки аналогов

Основные достоинства рассмотренных аналогов:

- предоставляют инструменты для анализа собранной информации

Недостатки аналогов:

- наличие человеческого фактора при оценке
- необходимость присутствия эксперта
- низкая скорость оценок
- малая автоматизация процесса оценки

6

Выбор средств разработки



PyCharm

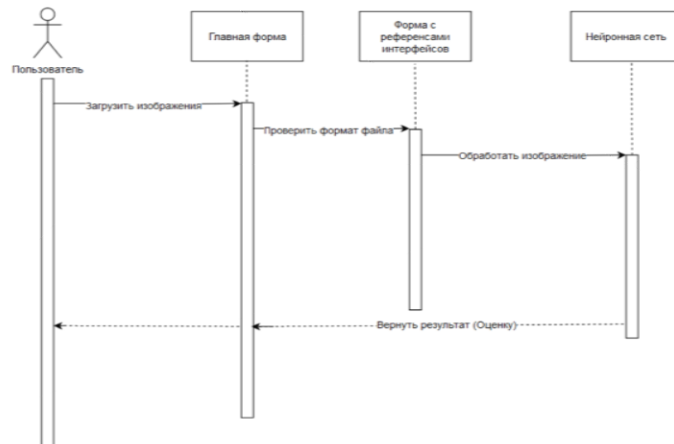
7

Диаграмма прецедентов



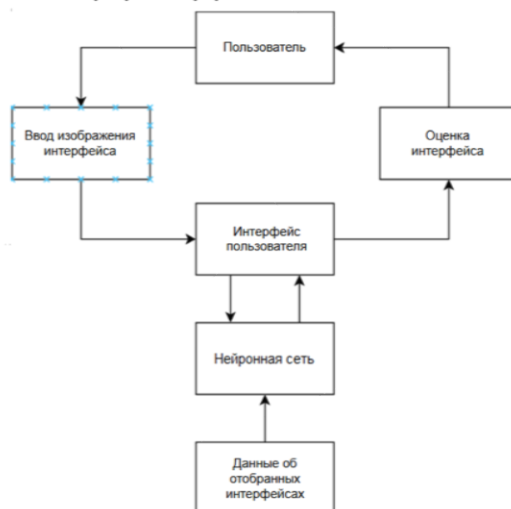
8

Диаграмма последовательностей работы с приложением



9

Структура системы

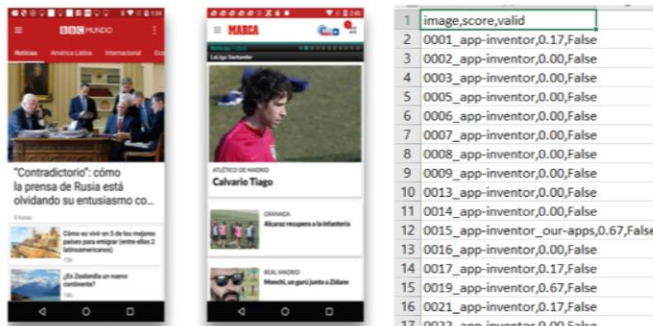


• Основные блоки структуры:

- Интерфейс пользователя – предназначен для взаимодействия пользователя с системой, обеспечивающий управление системой с помощью соответствующих кнопок:
- Нейронная сеть – модуль обучения нейронной сети и вывода оценки загруженного пользовательского интерфейса:
- Данные об отобранных интерфейсах – отобранный датасет с изображениями интерфейсов и метаданными.

10

Датасет RICO

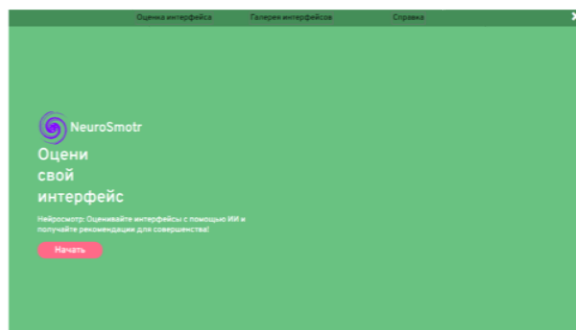


Основное окно

1. Функция оценки интерфейса
2. Просмотр референсов
3. Справка и техническая поддержка

11

Интерфейс системы

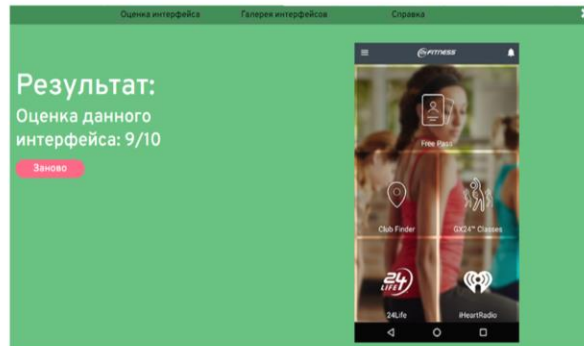


Основное окно

1. Функция оценки интерфейса
2. Просмотр референсов
3. Справка и техническая поддержка

12

Интерфейс системы



Результат работы при проработанном интерфейсе.

Результат работы приложения

13

Интерфейс подсистемы

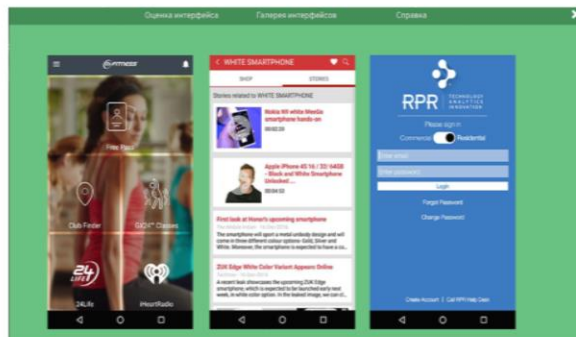


Результат работы при непроработанном интерфейсе.

Результат работы приложения

14

Интерфейс системы



Галерея проработанных изображений интерфейсов

Результат работы приложения

15

Заключение

- Был произведён анализ предметной области, а также осуществлен обзор аналогов проектируемой системы.
- Были построены UML диаграммы вариантов использования, последовательности, классов и компонентов.
- Разработаны структура системы и ее компоненты, обеспечивающие получение и анализ данных о критических событиях на распределенных объектах городской инфраструктуры.
- В дальнейшем предполагается доработка и внедрение решения в процессы проектирования приложений

16

Спасибо за внимание!