**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический университет им. А.Н. Туполева-КАИ»

(КНИТУ-КАИ)

Институт компьютерных технологий и защиты информации Кафедра компьютерных систем

09.03.01 «Информатика и вычислительная техника»

**К защите допустить**

**Зав. каф. КС**

**/Вершинин И.C/**

**« » 20 г.**

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему «Мобильный сервис для выбора профессионального обслуживания дома»

ОБУЧАЮЩИЙСЯ Данилкин А.Д.

*(фамилия, имя, отчество) (личная подпись)*

РУКОВОДИТЕЛЬ ассистент каф КС, Яковлев И.В.

*(ученая степень, звание, фамилия, инициалы) (личная подпись)*

Mobile service for professional home maintenance

*By*

*Danilkin Alexandr Dmitrievich*

Submitted to the Department of Computer Systems

in partial fulfillment of the Requirements for the degree of BACHELOR OF SCIENCE

at the

Federal State Budgetary Educational Institution of Higher Education

«Kazan National Research Technical University named after A.N.Tupolev-KAI» (KNRTU-KAI)

|  |  |  |
| --- | --- | --- |
| Author |  | Danilkin Alexandr Dmitrievich |
| *(signature)* |  |  |
| Supervisor |  | Yakovlev Igor Vladimirovich |
| *(signature)* |  | PhD, Department of Computer Science |
| Certified by |  | Vershinin Igor Sergeevich |
| *(signature)* |  | PhD, Head of the Department of Computer Science |
| date |  |  |

**федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический**

**университет им. А.Н. Туполева-КАИ» (КНИТУ-КАИ)**

Институт (факультет), филиал компьютерных технологий и защиты информации Кафедра компьютерных систем Направление подготовки/специальность 09.03.01 информатика и вычислительная техника

**ЗАДАНИЕ**

на выпускную квалификационную работу

обучающегося Данилкина Александра Дмитриевича

(фамилия, имя, отчество)

1. Тема выпускной квалификационной работы

Мобильный сервис для выбора профессионального обслуживания дома

утверждена приказом № от « » 20 г.

1. Срок сдачи обучающимся законченной ВКР « » 20 г.
2. Исходные данные к выпускной квалификационной работе
3. Перечень подлежащих разработке вопросов и исходные данные к ним:
   1. Анализ предметной области 4.2 Выбор инструментов разработки 4.3 Разработка мобильного приложения
4. Перечень графического материала (при наличии):

Презентация на слайдах

1. Консультанты по ВКР (при их наличии, с указанием относящихся к ним разделов):

*(наименование раздела, ФИО консультанта, подпись)*

*(наименование раздела, ФИО консультанта, подпись)*

*(наименование раздела, ФИО консультанта, подпись)*

*(наименование раздела, ФИО консультанта, подпись)*

Дата выдачи задания « 15 » февраля 2022г.

Руководитель ВКР Яковлев И.В.

(подпись) (ФИО)

Задание к исполнению принял Данилкин А.Д.

(подпись) (ФИО)

**Календарный план выполнения ВКР**

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование этапов (разделов) выпускной квалификационной работы | Срок выполнения  этапов (разделов) ВКР | Примечание |
| 1 | Анализ предметной области | 1.03.22 |  |
| 2 | Обзор существующих приложений | 15.03.22 |  |
| 3 | Разбор особенности фреймворка Flutter | 24.03.22 |  |
| 4 | Разработка дизайна приложения | 27.03.22 |  |
| 5 | Выбор архитектуры приложения | 11.04.22 |  |
| 6 | Разработка мобильного приложения на платформе iOS и Android | 18.04.22 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Обучающийся | |  |  |
| *(подпись)* | | *(Фамилия, инициалы)* | *( дата)* |
| Руководитель | *(подпись)* | *(Фамилия, инициалы)* | *(дата* |

# АННОТАЦИЯ

Выпускная квалификационная работа посвящена разработке сервиса для выбора профессионального обслуживания дома с помощью фреймворка Flutter . Рассматривается разработка мобильного приложения на языке Dart, а именно:

* среда разработки Android Studio, Xcode;
* архитектура операционной системы iOS и Android;
* жизненный цикл приложения;
* работа с пользовательским UI интерфейсом;

В заключении представлена экспериментальная часть с результатом в виде программного обеспечения на платформе iOS для выбора профессионального обслуживания дома.

**Ключевые слова:** Язык программирования Dart, мобильное приложение, Android и IOS разработка, UI интерфейс.

Количество страниц: - Количество иллюстраций: - Количество таблиц: - Количество приложений: -

Количество использованных источников: -

# ABSTRACT

-

**Keywords:** Dart programming language, mobile application, Android and iOS development, UI interface.

Page count: - Picture count: - Table count: - Appendix count: - Reference count: -

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Мобильные телефоны перестали быть чем-то поразительным и замечательно справляются со своей функцией – являются средством коммуникации между людьми. Смартфоны стали настолько функциональны, что трудно сказать, чего они не умеют. По сути, они стали маленькой копией компьютера, который постоянно можно иметь при себе. В наше время все больше и больше смартфонов и других видов устройств выпускаются на базе ОС Android, а так же популярные iPhone на iOS.

Причины распространения этих операционных систем разнообразны:

1. Android поддерживает большое количество устройств разных производителей.

2. ОС характеризуются высокой доступностью средств разработки. Средства разработки для платформы Android и iOS бесплатны.

3. Так же преимуществом является наличие бесплатных библиотек для работы со сторонними ресурсами, в то время как для Windows Phone Mobile такие библиотеки не распространены.

Уход за домом отнимает много времени – это большая проблема. Каждый из нас сталкивался с этим в повседневной жизни. Приложение для обслуживания дома поможет это исправить, так как оно позволяет решить ряд проблем.

Раньше, находя разнорабочего по вопросам связанных с обслуживанием дома нужно было искать кого-то кто специализируется на конкретной работе, нужно позвонить кому-то что отнимает наше время.

Сегодня, благодаря развитию технологий с помощью мобильного приложения можно получить доступ к множеству услуг. Спрос на приложения для обслуживания дома быстро растет, так как нас беспокоит повседневный образ жизни, где каждый второй человек обращается за помощью, чтобы управлять своей хозяйственной деятельностью.

В рамках данной работы будет описана разработка мобильного приложения для обслуживания дома ОС Android и IOS.

# ГЛАВА 1.АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Домашние услуги охватывают широкий спектр промышленных рынков, таких как ремонт и обслуживание, модернизация, уборка дома и другие. Если количество услуг, предлагаемых сектором услуг на дому, растет, масштабы также растут. Домашние услуги включают в себя различные нематериальные активы, которые предприятия предоставляют для домов. С ростом технологического прогресса растут как онлайн, так и мобильные платформы бронирования. Домашние услуги в основном являются более удобным и оптимизированным способом покупки и организации квалифицированных услуг. Сегодня наше поколение является крупнейшим потребителем интернет-услуг, и это расширило сферу бизнеса по оказанию домашних услуг. Занятый образ жизни клиентов заставляет их выбирать поставщиков прямо по требованию, и это расширило пропускную способность роста отрасли в ближайшие годы.

В то время как быстрый рост предоставит игрокам рынка огромные возможности для роста, эффективность поставщиков будет угрожать росту участников рынка. Чтобы оптимизировать возможности, рыночные поставщики будут больше сосредоточиваться на потенциале роста в быстрорастущих сегментах, сохраняя при этом свое место в медленнорастущих сегментах.

Рост этой отрасли также вызван ростом и удобством использования мобильного сервиса, что позволяет потребителям заказывать информацию и бронировать услуги одним касанием. Домашние услуги по запросу пересекают разрыв между реальной инфраструктурой и непосредственными интернет-платформами для предоставления производственных услуг.

Приложение для домашних услуг ориентировано на помощь в выборе профессиональных мастеров. Приложение предоставляет следующие возможности:

* Услуги по содержанию дома легко доступны;
* Предоставление услуг в удобное время;
* Простой-онлайн заказ;
* Простая авторизация?;
* Вывод лучших специалистов.

Для подтверждения конкурентности и необходимости мобильного приложения для профессионального обслуживания дома необходимо провести сравнения с похожими сервисами.

<https://oyelabs.com/on-demand-services-startups/>

## Обзор существующих сервисов для выбора профессионального обслуживания дома

Для сравнения мы рассмотрим три разных приложения, которые помогают c обслуживанием дома. Сравнение будет производиться по следующим критериям:

* возможности;
* доступность;
* простота и удобство в использовании.

**1.2.1 Приложение Братья Чистовы**

«Братья Чистовы» — это современный клининговый сервис. Его создали люди, которые стремятся к идеальной чистоте и порядку, и любят помогать другим людям.   
  
Основная адача — сделать уборку таким же простым и легким делом, как доставка продуктов и вызов такси. Заказывайте уборку на сайте или через мобильное приложение, отметьте фотографиями важные места в доме и доверьте свой дом профессиональным клинерам.   
  
Мы тщательно отбираем и обучаем каждого клинера, а также мотивируем исполнителей убираться по-настоящему чисто. В «Братьях Чистовых» используют специальные, экологически-чистые моющие средства и профессиональное оборудование.   
  
Ны рынке чистоты мы работаем с 2014 года, и представлены в 35 городах: Москве, Санкт-Петербурге, Воронеже, Калининграде, Калуге, Краснодаре, Липецке, Омске, Пушкино, Чите, Ярославле и еще в 24 регионах. Мы растем благодаря вниманию к клиентам и любви к своему труду. «Братьев Чистовых» рекомендуют друзьям и «забирают с собой» при переезде. Гордимся тем, что дружим с клиентами и оправдываем их доверие.

**1.2.2 Приложение Qlean**

На сайте сервиса Qlean вы можете заказать генеральную уборку квартиры в Москве. Она называется так, потому что помогает убрать пыль и загрязнения в самых труднодоступных местах дома.

Качественная генеральная уборка позволяет сохранить первоначальный вид квартиры надолго. С Qlean это легко и недорого.

Вы можете вызвать профессиональную команду клинеров по телефону и рассказать подробнее о вашей квартире и о том, какие услуги вам понадобятся. Для клиентов, которые заказывали генеральную уборку, действует скидка до 20% на регулярные уборки.

Регулярная уборка состоит из основ клининга: наведения порядка в вещах и влажной уборки поверхностей, а также дополнительных опций, которые сделают уборку индивидуальной

**1.2.1 Приложение Qweep.ru**

Мы предоставляем нашим клиентам услуги по уборке квартир (от поддерживающей уборки по подписке до уборки после ремонта). Об истории возникновения самой компании я писал раньше в материалах «[Как мы запустили клининговый бизнес перед началом пандемии, и что из этого вышло](https://vc.ru/life/126200-kak-my-zapustili-kliningovyy-biznes-pered-nachalom-pandemii-i-chto-iz-etogo-vyshlo)» и «[Грязные танцы, или Как клининговый сервис искал маркетологов](https://vc.ru/marketing/135978-gryaznye-tancy-ili-kak-kliningovyy-servis-iskal-marketologov)», поэтому в деталях повторяться не буду, а коротко напишу нашу главную цель: мы хотим изменить рынок клининговых услуг в России за счет digital-подхода и высокого уровня сервиса. Начали делать это, как принято, с самого «маленького» рынка — московского.

## О платформе: сайт и приложения

Задача, которую мы поставили в основу функционала платформы, проста — экономить время клиента для получения необходимой ему услуги. Такой себе Uber для уборки — сделал несколько кликов — клинеры уже едут на удобное для тебя время. Как это выглядит?

## Как сделать заказ

Сделать заказ очень легко, никаких блужданий по интерфейсу нет. По сути, все сводится к трём простым шагам на первом этапе:

## Выводы по главе

Проанализировав и сравнив приложения можно сделать вывод, что приложения делают упор на однонаправленный функционал – работа с графиками, ценными бумагами или сервис для учета личных финансов. Отсюда следует, что разработка многофункционального приложения для учета финансовых активов позволит приложению быть конкурентоспособным на рынке мобильных приложений. Отличительными особенностями являются:

* образовательный блок для повышения финансовой грамотности и квалификации инвестора;
* учет акций, криптовалюты, доходов и расходов в одном приложении;
* статистика по всем видам финансовых активов.

# ГЛАВА 2.ВЫБОР ИНСТРУМЕНТОВ РАЗРАБОТКИ

## Обзор операционной системы iOS

iOS – мобильная операционная система, разрабатываемая и выпускаемая американской корпорацией Apple. В отличии от других операционных систем, таких как Windows Phone и Android, iOS выпускается только для устройств, производимых фирмой Apple.

Как и в любой операционной системе в iOS существуют интерфейсы (уровни, слои, API), то есть предоставление интерфейса для работы с более низким уровнем. iOS предоставляет собой многоуровневую архитектуру, состоящую из 4-х уровней: Cocoa Touch, Media, Core Services, Core OS (рисунок [2.1](#_bookmark10)).

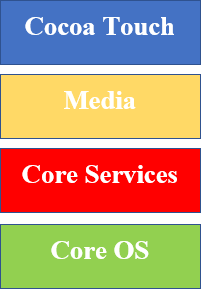


Рис 2.1. Уровни архитектуры операционной системы iOS

## Cocoa Touch

Слой Cocoa Touch представляет с собой верхний слой в iOS архитектуре. Он содержит в себе ключевой фреймворк UIKit, который позволяет программировать пользовательский интерфейс. Именно на этом слое находятся кнопки (UIButton), текстовые поля (UITextField) и все другие элементы пользовательского интерфейса. Чаще всего разработчикам приходится взаимодействовать с этим слоем.

## Media

Этот уровень предоставляет мультимедийные сервисы, которые используется на iPhone и других устройства iOS. Слой медиа состоит из аудио, видео и графики. На этом уровне используется такие фреймворк как: Core

Audio, Open AL, Core Graphics, Core Animation и многие другие, которые отвечают за отрисовку изображения, рендеринг данных и буферизации видео.

## Core Services

Уровень Core Services предоставляет объектно-ориентированное API над Core OS для работы со строками, коллекциями, сетевым взаимодействием, базами данных и многопоточностью. Также данный уровень предоставляет возможность пользоваться аппаратными особенностями, такие как GPS, компас, акселерометр и гироскоп. Включает в себя такие фреймворки, как Foundation, Address Book, Core Data и другие.

## Core OS

Слой Core OS является самым низкоуровневым iOS стека и находится непосредственно поверх аппаратного обеспечения устройства. На данном уровне осуществляется работа с базовыми операциями операционной системы, управление памятью и потоками.

Операционная система iOS управляет состоянием приложения, но у самого приложения есть свой жизненный цикл (рисунок [2.2](#_bookmark15)).

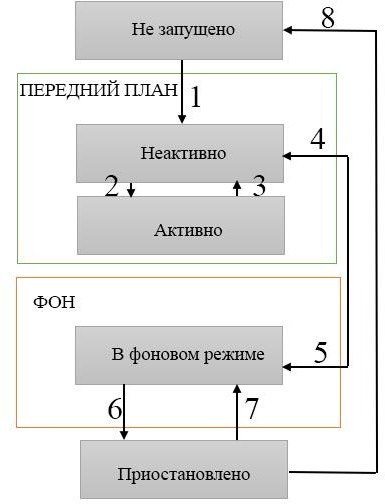


Рис 2.2. Жизненный цикл iOS приложения

Рассмотрим жизненный цикл поэтапно:

1. На начальном этапе приложение имеет состояние **не запущено**.
2. После запуска приложения оно переходит в передний план и имеет состояние **неактивно –** на данном этапе выполняется код программы, но не обрабатывается пользовательский интерфейс.
3. Далее переход в состояние **активно**, в котором выполняется код и обрабатываются все события пользовательского интерфейса.
4. Если пользователь запустит другое приложение, то текущее приложение перейдет в состояние **неактивно** и затем уйдет **в фоновый режим.** В этом состоянии код выполняется ограниченное время, и не обрабатывается события пользовательского интерфейса. В этом состоянии можно выполнять действия, которые покажут актуальную информации при возврате пользователя в приложение.
5. После короткого промежутка времени в фоновом режиме приложение переходит в состояние **приостановлено**. В этом состоянии код не выполняется вовсе, и система может деинициализировать приложение для освобождения памяти.

Каждое из этих состояний сопровождается методами жизненного цикла, которые находятся в файле AppDelegate, который является основной точкой входа для приложения (рисунок [2.3](#_bookmark16)).



Рис 2.3. Класс AppDelegate

Каждое из этих состояний сопровождаются методами класса AppDelegate, который является основной точкой входа для приложения:

* didFinishLaunchingWithOptions: метод, который выполняется непосредственно перед запуском приложения. Предназначен для конфигурации приложения.
* applicationWillResignActive: метод, который выполняется непосредственно перед сворачиванием приложения. Предназначен для остановки активных задач.
* applicationDidEnterBackground: дает знать, что приложение неактивно и может быть выключено в любое время.
* applicationWillEnterForeground: метод, которые выполняет перед выходом приложения в передний план. Предназначен для возврата задач, которые были остановлены.
* applicationDidBecomeActive: приложение стало активным.
* applicationWillTerminate: метод, который уведомляет, что приложение выключено. Метод не вызовется, если приложение было в приостановленном состоянии.

## Технический обзор Flutter

Flutter использует язык программирования Dart (спецификация v2.5). Он создан компанией Google и вдохновлен такими языками, как JavaScript. Dart использует систему статической типизации с проверкой во время выполнения, но, как и многие другие языки, активно использует вывод типов. Dart можно использовать как для написания простых скриптов, так и для полнофункциональных приложений. В Dart реализована гибкая технология компиляции, в которой компилятор может по-разному определять выполнение кода в зависимости от целевой платформы.

• Dart Native — для программ, предназначенных для устройств (мобильных, настольных, серверных и т. д.), Dart Native включает в себя виртуальную машину Dart с компиляцией Just-in-Time (JIT) и компилятор Ahead-of-Time (AOT) для создания машинного кода.

• Dart Web — для программ, ориентированных на Интернет, Dart Web включает в себя как компилятор времени разработки (dartdevc), так и компилятор времени производства (dart2js).

Flutter выполняет использование обоих способов. Если целевой платформой является Интернет, используется Dart Web. Для других платформ выбран Dart Native. JIT-компиляция Dart Native широко используется для поддержки быстрого процесса разработки с функцией “hot-reload”. Затем компиляция AOT используется для получения наилучшего оптимизированного готового к производству результата на нативной платформе.

Фреймворк Flutter состоит из нескольких слоев (рис. 1.3), где каждый слой использует предыдущий. Верхние уровни чаще используются разработчиками ежедневно, а нижние уровни используются только в том случае, если разработчикам необходимо создать определенные настройки.

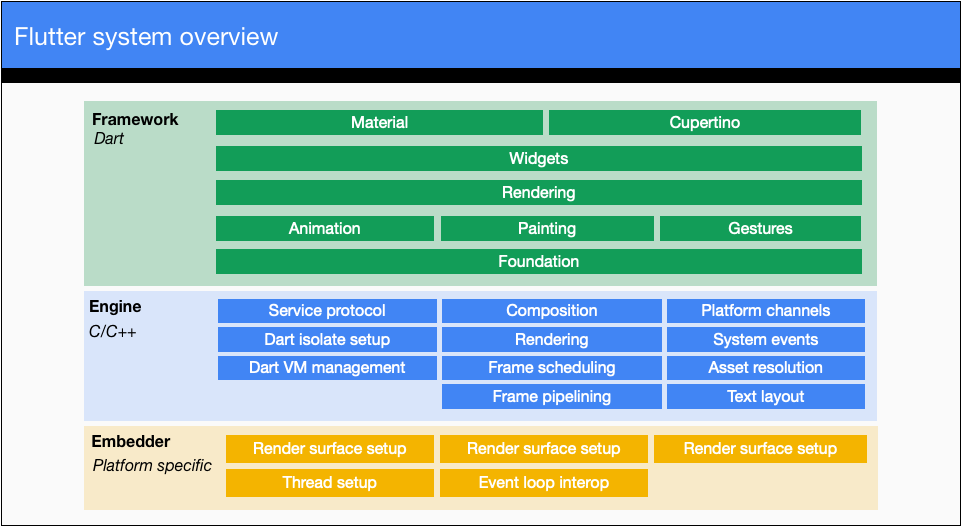


Рис 1.3. Обзор системы Flutter

В отличие от других фреймворков, Flutter использует высокопроизводительный движок 2D-рендеринга и рисует все прямо на экране. Это отражает точный до пикселя контроль над тем, что и как отображается. Самые верхние слои, Mate-rial и Cupertino, представляют собой набор виджетов, определяющих компоненты Material Design (системы Android) и Apple Design соответственно. Чтобы подчеркнуть это, Flutter не использует нативные компоненты, а все рисует сам по себе. Эти два уровня помогают разработчикам привнести стандартизированный внешний вид на целевую платформу.

**2.2.1 Все о виджетах**

Каждый пользовательский интерфейс состоит из макета и отдельных компонентов. Макет определяет базовую структуру экрана, например, меню сверху и последующие действия внизу. Затем макет состоит из отдельных компонентов, таких как меню, кнопки или значки. Вместе они создают окончательный интерфейс.

Эти строительные блоки во Flutter называются «виджетами». Будь то простой текст, кнопка или сложные части макета, такие как сетка с несколькими столбцами и строками — все это виджет. Виджеты описывают, как должно выглядеть их представление с учетом их текущей конфигурации и состояния. Когда состояние виджета изменяется, виджет перестраивает свое описание, которое фреймворк сравнивает с предыдущим описанием, чтобы определить минимальные изменения, необходимые в базовом дереве рендеринга для перехода из одного состояния в другое . Как следует из композиции дерева, каждый виджет имеет не более одного родителя и ноль или более дочерних виджетов. Это дерево, называемое «деревом виджетов», на самом деле является одним из трех задействованных деревьев. Фреймворк Flutter использует только один язык для определения как пользовательского интерфейса, так и бизнес-логики.

Widget — это класс Dart, который наследуется от некоторых базовых классов виджета (обычно это StatelessWidget или StatefulWidget). У каждого виджета есть метод build(), который определяет, как виджет должен быть построен (и отрисован на экране).

## Фреймворк Flutter

Для проекта был выбран фреймворк Flutter - это пользовательский фреймворк с открытым исходным кодом, созданный Google для создания высококачественных приложений для iOS и Android.

Именно архитектура и дизайн Flutter делают его такой мощной платформой для разработки приложения.

В настоящее время большинство компаний в восторге от Flutter. Это связано с тем что можно разрабатывать приложение для Android, IOS, Windows, Mac, Linux и интернета из единой кодовой базы.

Flutter имеет многоуровневую архитектуру, которая позволяет управлять каждым пикселем на экране. Благодаря мощным возможностям можно накладывать и анимировать графику, текст, видео и другие элементы управления без каких-либо ограничений.

Ядро механизма компоновки Flitter – это виджеты.

Visible Widgets – виджеты которые мы видим на экране.

Single Child Layout Widgets – виджеты которые меняют (расширяют) конфигурацию (поведение) дочернего виджета.

Multi Child Layout Widgets – виджеты которые размещают внутри себя несколько виджетов.