ЭСТОНСКИЙ УНИВЕРСИТЕТ ПРЕДПРИНИМАТЕЛЬСТВА МАЙНОР

Учебная программа: Веб-дизайн и цифровая графика

Специализация: Веб-технологии

Ivan Ireev

**Определение наиболее востребованных технологий и инструментов для iOS и Android разработчиков на основе анализа вакансий из стран Европы и Северной Америки**

Заключительная работа

Руководитель: Boris Shvartsman, PhD

Таллинн 2025

ДЕКЛАРАЦИЯ ОБ АВТОРСКИХ ПРАВАХ

Декларирую, что составил/составила настоящую работу самостоятельно и что ранее она не представлялась к защите кем-либо другим. На все использованные при составлении работы исследования других авторов, принципиальные точки зрения, данные, полученные из литературных источников, имеются ссылки.

Автор: [Имя Фамилия]

[дд. мм. гггг]

РЕЗЮМЕ

Текст.

RESÜMEE

Текст.

ABSTRACT

Текст.

ОГЛАВЛЕНИЕ

[ДЕКЛАРАЦИЯ ОБ АВТОРСКИХ ПРАВАХ 1](#_Toc193715130)

[РЕЗЮМЕ 2](#_Toc193715131)

[RESÜMEE 3](#_Toc193715132)

[ABSTRACT 4](#_Toc193715133)

[ВВЕДЕНИЕ 7](#_Toc193715134)

[1. АНАЛИЗ СУЩЕСТВУЮЩИХ ИССЛЕДОВАНИЙ 10](#_Toc193715135)

[1.1. Обзор исследований по рынку труда в *IT*-сфере 10](#_Toc193715136)

[1.2. Исследования, фокусирующиеся на *iOS* и *Android* разработчиках 12](#_Toc193715137)

[1.3. Выводы и пробелы в исследованиях 15](#_Toc193715138)

[2. ВЫБОР ИНСТРУМЕНТОВ И ПОДХОДОВ 17](#_Toc193715139)

[2.1. *SerpApi* 17](#_Toc193715140)

[2.2. *Langdetect* 17](#_Toc193715141)

[2.3. *ChatGPT-4o* 18](#_Toc193715142)

[2.4. *Fuzzy matching* 19](#_Toc193715143)

[3. МЕТОДИКА ИССЛЕДОВАНИЯ 21](#_Toc193715144)

[3.1. Подготовка к сбору данных 21](#_Toc193715145)

[3.2. Сбор данных 21](#_Toc193715146)

[3.3. Создание набора данных 22](#_Toc193715147)

[3.4. Первичная очистка данных 22](#_Toc193715148)

[3.5. Определение языка вакансий 22](#_Toc193715149)

[3.6. Перевод не англоязычных вакансий на английский 23](#_Toc193715150)

[3.7. Разделение вакансий на логические части 24](#_Toc193715151)

[ОГРАНИЧЕНИЯ 26](#_Toc193715152)

[РЕЗУЛЬТАТЫ 27](#_Toc193715153)

[ЗАКЛЮЧЕНИЕ 28](#_Toc193715154)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 29](#_Toc193715155)

[ПРИЛОЖЕНИЯ 31](#_Toc193715156)

[Приложение 1. Полный список стран 31](#_Toc193715157)

[Приложение 2. Директории с вакансиями 32](#_Toc193715158)

[Приложение 3. Единый набор данных 33](#_Toc193715159)

[Приложение 4. Промпт для разбиения вакансий на логические части 34](#_Toc193715160)

ВВЕДЕНИЕ

Мобильная разработка представляет собой процесс создания программного обеспечения, предназначенного для работы на мобильных устройствах, таких как смартфоны и планшеты. Эта область активно развивается и растет ([Statista, 2023](#Statista_2023_App_Worldwide)) в ответ на растущее число пользователей мобильных устройств ([IDC, 2024](#IDC_2024_Smartphone_Market_Insights); [Statista, 2021](#Statista_2021_Forecast_num_of_mibile_use)), увеличение количества скачиваний приложений ([Statista, 2024](#Statista_2024_Number_of_mobile_app)) и всё большее проникновение цифровых технологий во все сферы жизни человека. Мобильные приложения сегодня являются неотъемлемой частью повседневной жизни, используясь для общения, обучения, работы, развлечений и множества других задач.

Одной из ключевых сложностей является стремительная эволюция технологий: платформы *iOS* и *Android* регулярно выпускают обновления, внедряют новые языки программирования, фреймворки и подходы к разработке. Это создаёт ситуацию, в которой разработчикам становится трудно своевременно ориентироваться в меняющихся требованиях рынка и выбирать наиболее актуальные и востребованные инструменты, а университетские программы теряют актуальность и выпускают менее конкурентоспособных выпускников.

**Актуальность темы**

Актуальность данного исследования обусловлена потребностью в комплексном анализе ключевых тенденций мобильной разработки. Выявление наиболее востребованных технологий и инструментов для мобильных разработчиков на рынках Европы и Северной Америки позволит специалистам более эффективно ориентироваться в текущих тенденциях и выбирать правильное направление профессионального развития. Для образовательных учреждений подобные исследования служат основой для корректировки и обновления учебных программ, обеспечивая их актуальность и соответствие современным реалиям рынка.

Сосредоточение внимания на регионах Европы и Северной Америки не случайно: по данным *SlashData* ([2024](#SlashData_Mobile_Developer_Population_F)), именно Северная Америка и Западная Европа выступают двумя ключевыми региональными центрами развития мобильных приложений. Понимание тенденций в этих регионах может служить ориентиром для глобального сообщества разработчиков при выборе соответствующих технологических стеков и инструментов.

**Проблема и цель исследования**

Проблема исследования заключается в сложности определения наиболее востребованных навыков и инструментов для мобильных разработчиков в условиях стремительного развития технологий.

Цель исследования — выявить наиболее востребованные технологии и составить рекомендации, которые помогут специалистам адаптироваться к текущим тенденциям рынка труда, а образовательным учреждениям — формировать программы, соответствующие современным требованиям индустрии.

**Задачи исследования**

Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести обзор существующих исследований, посвященных анализу рынка труда мобильных разработчиков;
2. Определить список стран для анализа;
3. Разработать и реализовать методику сбора вакансий с использованием *Python* и *SerpApi*;
4. Выполнить предобработку собранных данных, включая:
   * удаление дубликатов;
   * фильтрацию нерелевантных вакансий.
5. Определить язык вакансий с помощью *Langdetect*;
6. Перевести неанглоязычные вакансии на английский с использованием *ChatGPT*-*4o*;
7. Извлечь упомянутые в вакансиях технологии и инструменты с помощью *ChatGPT*-*4o*;
8. Провести статистический анализ собранных данных для выявления наиболее востребованных языков программирования, фреймворков и других технологий в области мобильной разработки;
9. Сравнить требования работодателей к мобильным разработчикам в Европе и Северной Америке, выявив сходства и различия для этих регионов;
10. Сформулировать выводы и рекомендации, которые помогут:
    * мобильным разработчикам выбрать оптимальный технологический стек;
    * образовательным учреждениям адаптировать учебные программы в соответствии с актуальными требованиями рынка труда.

1. АНАЛИЗ СУЩЕСТВУЮЩИХ ИССЛЕДОВАНИЙ

В современных условиях рынок труда в *IT*-сфере привлекает значительное внимание исследователей, однако анализ рынка труда для *iOS*- и *Android*-разработчиков остается недостаточно изученным. Существующие работы, затрагивающие смежные вопросы, чаще всего ограничиваются либо общими тенденциями *IT*-рынка, либо рассматривают более широкую категорию «разработчиков программного обеспечения» в целом, либо фокусируются на анализе рынка труда в рамках отдельных стран.

## 1.1. Обзор исследований по рынку труда в *IT*-сфере

Популярные отраслевые опросы и аналитические отчеты регулярно фиксируют глобальные тенденции. К примеру, *Stack Overflow Developer Survey* ([2024](#Stack_Overflow_2024_Annual_Dev_Survey)) представляет широкий срез *IT*-специалистов по всему миру (более 50 тысяч респондентов) и содержит данные о наиболее распространённых языках программирования, технологиях и профессиональных ролях. Подобные исследования дают ценную макрокартину, но не фокусируются на отдельных платформах: мобильные разработчики выступают в них лишь малой частью выборки (лишь около 3,4% от всех респондентов составляют *mobile*-разработчики ([Stack Overflow, 2024, Developer Profile](#Stack_Overflow_2024_Annual_Dev_Survey))).​

Исследование ([Cummings, Janicki, & Matthews, 2023](#Cummings_Janicki_Matthews_2023)) анализирует востребованные технологии и навыки на основе опроса более 500 специалистов в области информационных систем и технологий, включая работодателей, разработчиков и рекрутеров. В работе рассматриваются популярные инструменты, тенденции их развития и прогнозы по изменению спроса на навыки в *IT*-сфере. Однако исследование фокусируется на индустрии в целом и не предоставляет детальной информации о технологиях, специфичных для мобильной разработки.

В исследовании ([Apatsidis, Georgiou, Mittas, & Angelis, 2021](#Apatsidis_Georgiou_Mittas_Angelis_2021)) проведён анализ 8514 объявлений о вакансиях с платформы *Stack Overflow* для выявления различий между удалёнными и офисными позициями в сфере информационно-коммуникационных технологий (*ICT*). Авторы применили методы текстового анализа и теорию графов, чтобы определить востребованные технологии и типы должностей. Хотя исследование даёт представление о ключевых технологиях в разработке программного обеспечения, оно не фокусируется на мобильной разработке.

Исследование ([Kang, Park, & Shin, 2020](#Kang_Park_Shin_2020)) посвящено анализу современных тенденций и ключевых технических навыков, востребованных в сфере разработки программного обеспечения. Основное внимание авторы уделяют выявлению разрыва между академической подготовкой и требованиями индустрии, который особенно актуален в *IT*-сфере из-за её динамичного и быстро меняющегося характера. В рамках исследования был проведён анализ более 120 000 вакансий, собранных с различных платформ для поиска работы по всей территории США. Это позволило определить наиболее востребованные навыки, включая языки программирования, интегрированные среды разработки (*IDE*), инструменты управления проектами, а также "мягкие навыки" (*soft skills*), которые активно запрашиваются работодателями. Однако отсутствие фокуса на мобильной разработке в исследовании ограничивает его применимость для специалистов в этой области.

Исследование ([Ternikov, & Aleksandrova, 2020](#Ternikov_Aleksandrova_2020)) посвящено анализу востребованности навыков на рынке труда в *IT*-секторе. Авторы предложили алгоритм, который позволяет извлекать и унифицировать наборы ключевых навыков, востребованных различными профессиональными группами в этой сфере. Для выполнения задачи использовалась база данных с описанием вакансий из онлайн-ресурсов российского рынка труда. Анализ данных проводился с применением методов обработки текста, таких как *TF-IDF* и *n-grams*. В результате исследования были выявлены основные навыки, актуальные для различных *IT*-профессий, а также популярные комбинации навыков, пользующиеся наибольшим спросом у работодателей. Важно отметить, что мобильные разработчики не выделялись как отдельная профессиональная группа. Их навыки рассматривались в контексте более широких категорий, таких как высококвалифицированные *IT*-специалисты или разработчики программного обеспечения.

Проект ([Data Nerds, n.d.](#Data_Nerds_n_d_)), разработанный *Luke Barousse* ([2023](#Barousse_Luke_2023)), представляет собой аналитическую платформу, исследующую востребованные навыки и уровень заработных плат в сфере информационных технологий (*IT*). Для сбора данных о вакансиях сервис использует *SerpApi*, получая информацию из результатов поиска *Google*. В среднем ежедневно обрабатывается около 6500 вакансий, преимущественно относящихся к *data science* и смежным направлениям. Полученные данные проходят предобработку и анализ с использованием *NLP*-пайплайна, после чего агрегируются для выявления наиболее востребованных компетенций и расчёта медианных значений заработных плат. Основной целью сервиса является предоставление обоснованной информации специалистам в области данных (*Data Analysts*, *Data Engineers*, *Data Scientists* и т. д.), однако платформа может быть полезна и для разработчиков программного обеспечения. На веб-сайте представлена статистика по наиболее часто упоминаемым в вакансиях языкам программирования, инструментам, базам данных, облачным сервисам, библиотекам и фреймворкам, а также анализ динамики популярности различных технологий с течением времени. Несмотря на то, что проект даёт общее представление о ключевых технологиях в сферах разработки программного обеспечения и *data science*, он не фокусируется на мобильной разработке.

## 1.2. Исследования, фокусирующиеся на *iOS* и *Android* разработчиках

**1.2.1. Исследования команды Яндекс Практикума**

Отдельного упоминания заслуживают исследования, опубликованные на *Habr*, проведенные командой Яндекс Практикума ([2024, 3 июня](#Яндекс_Практикум_2024_03_06_Android); [2024, 10 июня](#Яндекс_Практикум_2024_10_06_iOS)), специально посвящённые рынку труда *iOS*- и *Android*-разработчиков. В рамках этих исследований анализировалась ситуация с наймом младших специалистов (*junior* и *junior plus*) в сфере мобильной разработки на российском рынке.

Методология включала комбинированный подход: анализ вакансий на *job*-портале *HH.ru* и экспертные интервью. В частности, эксперты опросили технических руководителей команд и ведущих разработчиков ряда компаний (в основном российских, а также некоторых зарубежных) для выяснения требований к кандидатам. Параллельно были собраны и проанализированы актуальные вакансии уровня *junior* и *junior plus* по *iOS*- и *Android*-разработке, опубликованные в мае 2024 г. на *HH.ru*​. Такой подход позволил выявить перечень ключевых навыков и знаний, которые работодатели ожидают от начинающих мобильных разработчиков, а также понять особенности процесса найма.

***iOS*** — В исследовании ([Яндекс Практикум, 2024, 10 июня](#Яндекс_Практикум_2024_10_06_iOS)) было установлено, что работодатели чаще всего требуют от начинающих *iOS*-разработчиков следующие компетенции:

* Владение языком *Swift*.
* Опыт работы с системой контроля версий *Git*.
* Знание принципов объектно-ориентированного программирования (ООП).
* Навыки работы в *Xcode*.
* Знакомство с фреймворком *UIKit*.
* Знание архитектурного паттерна *MVVM*.

Помимо этого, от кандидатов ожидается понимание концепции разделения кода на слои и общее представление о шаблонах проектирования, таких как *MVVM*, *VIPER*, *MVI*, *Singleton*, *Delegate* и *Factory*. Знание *SwiftUI* и *Combine* является желательным, однако их необходимость зависит от специфики компании и проекта: одни активно внедряют *SwiftUI* в новых разработках, в то время как другие продолжают использовать *UIKit*.

Также для начинающих специалистов *soft skills* имеют значение, сопоставимое с техническими знаниями. Отсутствие способности к логическому рассуждению или проявление агрессивного поведения во время интервью существенно снижает шансы на успешное трудоустройство.

***Android*** — В исследовании ([Яндекс Практикум, 2024, 3 июня](#Яндекс_Практикум_2024_03_06_Android)) было установлено, что работодатели чаще всего требуют от начинающих *Android*-разработчиков следующие компетенции:

* Владение языками *Kotlin* и *Java*.
* Знание *Android* *SDK*.
* Умение работать с системой контроля версий *Git*.
* Понимание принципов ООП.
* Знание архитектурных паттернов *MVVM* и *MVI*.

Кроме того, ценятся навыки работы со *Spring* *Framework*, понимание клиент-серверного взаимодействия, опыт работы с базами данных (например, *SQLite* и *Room*), а также базовые знания алгоритмов, кэширования и управления памятью.

Особое внимание уделяется умению находить и исправлять ошибки, работе с версткой (*XML*, *Jetpack* *Compose*, *HTML*) и использованию *Compound* *View*. При этом в современных проектах всё чаще применяется *Jetpack* *Compose*, хотя знание *XML* остаётся актуальным. Среди архитектурных требований выделяются умения работы с *ViewModel* и внедрения зависимостей посредством *Koin* или *Dagger*. Работодатели также ожидают базовых знаний многопоточности и функционального программирования, что подчёркивает важность владения *Coroutines* или *RxJava*.

Также для начинающих специалистов *soft skills* имеют значение, сопоставимое с техническими знаниями. Работодатели отдают предпочтение кандидатам, которые проявляют инициативу, проявляют искренний интерес к работе, умеют анализировать информацию и подходят к задачам внимательно. В то же время наиболее нежелательными качествами считаются агрессивность и раздражительность.

**1.2.2. Исследование *Hanna Samer Odeh***

В исследовании ([Hanna Samer Odeh, 2024](#Hanna_Samer_Odeh_2024_Android_Jobs)) был проведен анализ 300 вакансий *Android*-разработчиков из шести стран: США, Канады, Великобритании, Австралии, ОАЭ и Индии (по 50 вакансий из каждой страны). Вакансии были собраны с 44 сайтов по поиску работы, включая такие платформы, как *LinkedIn*, *Indeed* и *Seek*. Методология исследования основывалась на текстовом анализе объявлений о вакансиях, что позволило выделить наиболее часто упоминаемые требования к кандидатам. Исследование фокусировалось на выявлении как технических, так и мягких навыков (*soft skills*), востребованных на рынке труда *Android*-разработчиков.

Среди наиболее востребованных технических навыков, упоминаемых в более чем 30% проанализированных объявлений, были выявлены:

* Владение языками программирования *Kotlin* и *Java*;
* Навыки обеспечения производительности *Android*-приложений;
* Опыт работы с *RESTful* *API* и взаимодействием с сетевыми сервисами;
* Владение системой контроля версий *Git*;
* Знание *Android* *SDK*;
* Навыки модульного тестирования (*Unit* *Testing*);
* Проектирование интерфейсов *Android*-приложений, в том числе использование *XML* для разработки активностей;
* Владение архитектурным шаблоном *MVVM*;
* Знание *Agile*-методологий разработки программного обеспечения;
* Опыт публикации приложений в *Google* *Play* *Store*.

Помимо технических компетенций, значительное внимание уделяется и мягким навыкам. В частности, наибольшее значение придается:

* Умению работать в команде (*Collaboration* — 47%, *Teamwork* — 28% вакансий);
* Коммуникативным навыкам (*Communication* — 29%);
* Способности к решению проблем (*Problem*-*Solving* — 21%).

## 1.3. Выводы и пробелы в исследованиях

Анализ существующих исследований показал, что, несмотря на высокий интерес к рынку труда в *IT*-сфере, детальное изучение наиболее востребованных технологий и инструментов для *iOS*- и *Android*-разработчиков до сих пор остаётся ограниченным.

Так, в целом ряде работ ([Apatsidis et al., 2021](#Apatsidis_Georgiou_Mittas_Angelis_2021); [Cummings et al., 2023](#Cummings_Janicki_Matthews_2023); [Data Nerds, n.d.](#Data_Nerds_n_d_); [Kang et al., 2020](#Kang_Park_Shin_2020); [Stack Overflow, 2024](#Stack_Overflow_2024_Annual_Dev_Survey); [Ternikov et al., 2020](#Ternikov_Aleksandrova_2020)) представлена ценная информация о глобальных потребностях рынка разработчиков программного обеспечения, однако в них мобильные специалисты не выделяются как отдельная категория, что затрудняет выявление специфических требований именно к *iOS*- и *Android*-разработчикам.

Исследование, проведённое *Hanna* *Samer* *Odeh* ([2024](#Hanna_Samer_Odeh_2024_Android_Jobs)), предоставляет детализированное представление о востребованных навыках и технологиях для *Android*-разработчиков на глобальном уровне, однако аналогичного комплексного исследования, охватывающего *iOS*-разработчиков, не существует на момент написания работы.

Исследования Яндекс Практикума ([2024, 3 июня](#Яндекс_Практикум_2024_03_06_Android); [2024, 10 июня](#Яндекс_Практикум_2024_10_06_iOS)) обладают высокой практической ценностью, однако они ограничены анализом российского рынка труда. С одной стороны, можно предположить, что требования к *iOS*- и *Android*-разработчикам между различными странами могут быть схожи — подтверждением этому служит обнаруженная в исследовании ([Hanna Samer Odeh, 2024](#Hanna_Samer_Odeh_2024_Android_Jobs)) сильная положительная корреляция (≥ 0,7) между некоторыми навыками *Android*-разработчиков в разных странах. С другой стороны, не исключено, что локальные особенности и различия в корпоративных культурах могут приводить к вариациям в наборе необходимых навыков.

Таким образом, данное исследование направлено на устранение пробелов в существующих знаниях, позволяя более детально изучить и сопоставить востребованные навыки и инструменты мобильной разработки в регионах Европы и Северной Америки.

2. ВЫБОР ИНСТРУМЕНТОВ И ПОДХОДОВ

## 2.1. *SerpApi*

*SerpApi* — это сервис, предоставляющий *API* для получения результатов поисковых систем в реальном времени, включая *Google*, *Bing*, *Yahoo* и другие. Он обеспечивает доступ к структурированным данным поисковой выдачи, облегчая извлечение и обработку информации для различных целей. ([SerpApi, n.d. -a](#SerpApi_n_d_Google_Search_API))

В данном исследовании *SerpApi* используется для сбора вакансий *iOS*- и *Android*-разработчиков в Европе и Северной Америке. *SerpApi* позволяет автоматизировать процесс сбора данных о вакансиях, обеспечивая доступ к актуальной информации с различных поисковых систем и сайтов.

Преимущества использования *SerpApi* в исследовании:

* Автоматизированное решение для обхода *CAPTCHA*. В отличие от стандартных методов веб-скрейпинга, *SerpApi* автоматически обходит *CAPTCHA*, что значительно упрощает процесс сбора вакансий ([SerpApi, n.d. -a](#SerpApi_n_d_Google_Search_API));
* Доступ к структурированным данным. *SerpApi* предоставляет структурированные данные о вакансиях, включая описание должностей, требования, зарплаты (если указаны) и местоположение ([SerpApi, n.d. -b](#SerpApi_n_d_Google_Jobs_API));
* Гибкость в настройке поисковых параметров. *SerpApi* поддерживает использование различных локальных доменов *Google* (например, *google*.*de*, *google*.*fr*), что позволяет получать результаты, специфичные для конкретных стран или регионов ([SerpApi, n.d. -b](#SerpApi_n_d_Google_Jobs_API));
* Точный поиск по ключевым словам. *SerpApi* позволяет формулировать точные поисковые запросы, такие как “*iOS developer*” или “*Android developer*”, что помогает находить наиболее релевантные вакансии ([SerpApi, n.d. -b](#SerpApi_n_d_Google_Jobs_API)).

Благодаря перечисленным преимуществам *SerpApi* был выбран в качестве оптимального инструмента для сбора данных о вакансиях в данном исследовании.

## 2.2. *Langdetect*

Для определения языка вакансий была выбрана библиотека *Python Langdetect*. Хотя существуют более современные решения, такие как *fastText*, *Langdetect* показывает достаточно высокую точность, что подтверждено в сравнительном исследовании ([*ModelPredict*, 2021](#ModelPredict_2021_Comparison_of_language)). Несмотря на более низкую скорость работы, в условиях офлайн-анализа и умеренного объёма данных её производительности достаточно. Простота интеграции делают её удобным инструментом для применения в исследованиях. Дополнительным преимуществом является наличие у автора опыта работы с *Langdetect*, что позволило быстрее приступить к анализу.

## 2.3. *ChatGPT-4o*

​*ChatGPT-4o* — это мультимодальная большая языковая модель, разработанная *OpenAI* и выпущенная 13 мая 2024 года. Она способна обрабатывать и генерировать текст, изображения и аудио в реальном времени, обеспечивая более быстрые и эффективные результаты по сравнению с предыдущими моделями. ([GPT-4o, n.d.](#Wikipedia_n_d_GPT_4o))

В данном исследовании используется *API ChatGPT-4o* версии *gpt-4o-2024-11-20*.

**2.3.1. Формирование промпта для перевода вакансий на английский**

Структура промпта была выбрана на основе исследования ([Gao, Wang & Hou, 2023](#Gao_Wang_Hou_2023)), где эмпирически тестировались различные формулировки для машинного перевода с использованием *ChatGPT* (*GPT-3.5*). В рамках исследования авторы сравнивали *ChatGPT* с *Google* *Translate* и *DeepL* — двумя наиболее мощными коммерческими системами машинного перевода. Они использовали их в качестве эталонных систем, чтобы понять, насколько *ChatGPT* приближается (или уступает) им в качестве перевода.

Результаты экспериментов показали, что оптимальная формулировка промпта может существенно повысить точность перевода. *ChatGPT* (*GPT-3.5*) превзошёл *Google* *Translate* и *DeepL* в переводе на английский, согласно метрикам *BLEU* и *ChrF++*.

В данной работе была адаптирована наиболее эффективная версия промпта из исследования, но с использованием *GPT-4o* — более совершенной модели.

## 2.4. *Fuzzy matching*

*Fuzzy matching* (нечёткое сопоставление) — это метод, используемый для сравнения текстовых данных с учётом возможных орфографических вариаций, перестановок слов и незначительных расхождений в формулировках. В отличие от строгого (точного) сопоставления строк, *fuzzy matching* позволяет выявлять схожесть текстов, даже если они содержат частичные совпадения или небольшие различия. ([DataCamp, 2025](#DataCamp_2025_Fuzzy_String_Matching))

В данном исследовании метод *fuzzy matching* был выбран для оценки точности автоматического разбиения вакансий на логические части с использованием *API ChatGPT-4o*. Основная причина выбора данного подхода заключается в том, что модель *ChatGPT-4o* не всегда возвращает текст в точности в том же виде, что и исходный, но при этом сохраняет смысл и структуру данных. Поэтому применение нечёткого сопоставления позволяет объективно измерить степень соответствия между автоматической и ручной разметкой, не наказывая за незначительные различия в формулировках.

**2.4.1. Token Set Ratio**

Одной из наиболее эффективных метрик для сравнения текстов в рамках *fuzzy matching* является *Token Set Ratio*. Она основана на классическом методе — расстоянии Левенштейна (*Levenshtein Distance*), которое вычисляет минимальное число операций (вставки, удаления или замены) для преобразования одной строки в другую ([DataCamp, 2025](#DataCamp_2025_Fuzzy_String_Matching)).

Однако *Token Set Ratio* отличается от расстояния Левенштейна следующим ([DataCamp, 2025](#DataCamp_2025_Fuzzy_String_Matching)):

1. Токенизация строк — разбиение текста на отдельные слова (токены);
2. Удаление дубликатов — одинаковые слова внутри каждой строки не учитываются повторно;
3. Приведение к одному регистру — сравнение проводится без учёта регистра букв;
4. Расчёт сходства — после нормализации строк вычисляется расстояние Левенштейна между отсортированными множествами токенов.

Эта метрика особенно эффективна при анализе текстов вакансий, где одна и та же информация может быть представлена разными формулировками, но при этом иметь одинаковый смысл.

**2.4.2. Rapidfuzz**

Для реализации *Token Set Ratio* в *Python* была выбрана библиотека *rapidfuzz*. Она представляет собой высокопроизводительную альтернативу *fuzzywuzzy*, обеспечивая большую точность и скорость вычислений за счёт оптимизированных алгоритмов на языке *C++* ([RapidFuzz, n.d.](#RapidFuzz_n_d_GitHub)).

3. МЕТОДИКА ИССЛЕДОВАНИЯ

## 3.1. Подготовка к сбору данных

Перед началом сбора данных был определён список стран (см. [Приложение 1](#Приложение_1_Полный_список_стран)) для регионов: “Европа” и “Северная Америка”. В регион “Европа” вошли все страны Европейского Союза и Шенгенской зоны, а также Кипр, Ирландия и Великобритания. В регион “Северная Америка” вошли страны: Канада, Мексика и США.

Чтобы упростить последующий сбор и обработку, данные о странах были оформлены в виде *CSV*-файла “*location\_domain\_table.csv*”. В этот файл были включены следующие столбцы:

* ***Location*** — название страны;
* ***Region*** — регион, к которому относится страна (*Europe* или *Northern* *America*);
* ***EU Member*** — флаг, указывающий на членство страны в Европейском Союзе (*True/False*);
* ***Schengen Agreement*** — флаг, указывающий на участие страны в Шенгенском соглашении (*True/False*);
* ***Google Domain*** — локальный домен *Google*, связанный с данной страной (например, *google.ee* для Эстонии, *google.co.uk* для Великобритании).

## 3.2. Сбор данных

Сбор данных осуществлялся с использованием *Python*-функции, которая выполняла автоматизированные *API*-запросы к *SerpApi*. Все вакансии были собраны в период с “2025-01-13 12:08:48 *UTC*” по “2025-01-13 12:45:14 *UTC*”.

Для максимального охвата уникальных вакансий сбор данных выполнялся дважды для каждой страны с использованием двух различных доменов *Google*:

1. Домен по умолчанию (*google.com*) — для получения результатов глобального поиска;
2. Локальный домен (например, *google.ee* для Эстонии) — для учета локальных особенностей поисковой выдачи.

Вакансии были собраны с помощью следующих поисковых запросов:

* *“iOS developer”* — для поиска вакансий *iOS*-разработчиков.
* *“Android developer”* — для поиска вакансий *Android*-разработчиков.

Все собранные вакансии сохранялись в соответствующие директории в зависимости от домена и типа вакансии (см. [Приложение 2](#_Приложение_2._Директории)). Подобное разбиение позволило упростить дальнейшую обработку данных.

## 3.3. Создание набора данных

После сбора вакансий данные были объединены в единый набор (см. [Приложение 3](#_Приложение_3._Итоговый)) с помощью *Python*-функции. Перед этим была выполнена предобработка: удалены нестандартные символы-разделители строк (\*u*2028\*u*2029), которые были заменены на пробел. Функция принимала на вход директории с вакансиями, а также данные из файла *“location\_domain\_table.csv”*, содержащего информацию о странах.

## 3.4. Первичная очистка данных

Поскольку вакансии собирались через два разных домена, ожидалось, что в данных будет значительное количество дубликатов, что и подтвердил анализ: при 5 834 строках в наборе данных содержалось всего 2 863 уникальных *Job ID*, что свидетельствует о наличии дубликатов.

Для удаления дубликатов было установлено следующее правило: если одна и та же вакансия (с одинаковым *Job ID*) встречалась несколько раз в рамках одной страны, приоритет отдавался варианту, найденному через *google.com*.

В результате очистки было удалено 2 971 строк, и итоговый набор данных уменьшился до 2 863 вакансий. Также были удалены вакансии, содержащие 14 слов или меньше.

## 3.5. Определение языка вакансий

Для определения языка вакансий была написана *Python*-функция с использованием библиотеки *Langdetect*. Для воспроизводимости результатов был задан параметр “*DetectorFactory.seed* = 0”. Эта функция выполняла следующие задачи:

* Определяла язык текста для каждой вакансии;
* Возвращала код языка в формате *ISO* (например, *en* — *English*, *fr* — *French* и т. д.);
* Возвращала уровень уверенности в определённом языке (например, 0.999).

Если доля уверенности модели *Langdetect* в определённом языке была меньше 0,99 (что произошло только для 16 вакансий), то язык вакансии определялся повторно с помощью *Python*-функции, которая отправляла *API*-запрос к *ChatGPT-4o*.

Для определения языка использовался следующий промпт:

“*Detect the language of the text and return ONLY the ISO language code* (*e.g., en, fr, de, etc*.). *Text*: {*Job Description*}”.

Где {*Job Description*} — описание вакансии.

Окончательное решение о языке принималось на основе ручной проверки по следующему алгоритму:

* Если выводы *Langdetect* и *ChatGPT-4o* совпадали (например, оба определили текст как английский), то данный язык фиксировался как окончательный.
* Если текст содержал примерно равные доли двух языков, то вакансия не относилась к англоязычным.

## 3.6. Перевод не англоязычных вакансий на английский

В ходе исследования было выявлено, что 86% всех вакансий были опубликованы на английском языке.

Перед переводом была составлена языковая разметка (*mapping*), в которой каждому *ISO*-коду языка было сопоставлено его название. Например:

* “*en*” → “*English*”;
* “*fr*” → “*French*”;
* “*de*” → “*German*” и т. д.

Для перевода вакансий на английский была написана *Python*-функция, которая формировала динамический промпт в зависимости от языка оригинального текста и отправляла *API*-запрос к *ChatGPT-4o*.

Для перевода вакансий на английский язык использовался следующий промпт (подробнее об обосновании выбора промпта см. [раздел 2.3.1.](#Раздел_2_3_1_Формирование_промпта)):

“*This is a* {*Mapped Language*} *to English translation, please provide the English translation for this job description*: {*Job Description*}”

Где {*Mapped Language*} — язык оригинального текста, а {*Job Description*} — описание вакансии.

## 3.7. Разделение вакансий на логические части

Для дальнейшей обработки данных было решено разделить вакансии на следующие логические части:

1. *Platform*;
2. *Salary*;
3. *Requirements*;
4. *Nice to have*;
5. *Responsibilities*;
6. *Benefits*.

Подобное разделение упростит будущий процесс извлечения навыков и технологий за счёт сокращения объёма обрабатываемого текста, что также снижает затраты на использование *API ChatGPT-4o*.

**3.7.1. Валидация**

Для оценки качества автоматического разбиения вакансий на логические части была проведена процедура валидации. Основной целью данного этапа была проверка точности работы метода, использующего *API ChatGPT-4o*, по сравнению с ручной разметкой.

Для проведения валидации была сформирована случайная выборка из 30 вакансий. Для обеспечения воспроизводимости эксперимента в процессе выборки использовался параметр “*random\_state = 42*”. Далее вакансии были разделены на логические части двумя способами:

1. Ручная разметка, выполненная автором, который классифицировал информацию в соответствии с установленными категориями: *Platform*, *Salary*, *Requirements*, *Nice to have*, *Responsibilities* и *Benefits*.
2. Автоматическая разметка, выполненная при помощи *Python*-функции и *API* *ChatGPT-4o* на основе заранее сформулированного промпта (см. [Приложение 4](#_Приложение_2._Промпт)).

Для оценки степени соответствия двух методов разметки был применён метод *fuzzy matching* с использованием метрики *Token Set Ratio* из библиотеки *rapidfuzz* (подробнее см. [раздел 2.4.](#_2.4._Fuzzy_matching)).

В качестве порога значимости была выбрана граница 85%, выше которой автоматическая разметка считается удовлетворительно точной. В результате тестирования были получены следующие средние значения *Token Set Ratio* по секциям:

* *Platform*: 98,33%
* *Salary*: 97,35%
* *Requirements*: 98,61%
* *Nice* *to* *have*: 93,38%
* *Responsibilities*: 90,88%
* *Benefits*: 91,85%

Итоговое среднее значение *Token Set Ratio* составило 95,07%, что существенно превышает установленный порог. Полученные результаты подтверждают высокую степень схожести автоматической разметки с ручной аннотацией и свидетельствуют о надёжности применяемого метода.

После успешной валидации все вакансии были автоматически разделены на логические части с помощью *Python*-функции и *API* *ChatGPT*-*4o*.

**3.7.2. Преобразование секций в столбцы**

После автоматического разбиения вакансий на логические части каждая запись представляла собой единый блок текста, в котором секции следовали друг за другом по шаблону “*Ключ*: *Значение*” (см. пример в [Приложении 5](#_Приложение_5._Пример)). Для последующего анализа требовалось перевести эти выжимки в табличный вид, чтобы каждая логическая часть располагалась в отдельном столбце и могла обрабатываться как самостоятельная переменная.

При помощи регулярных выражений блок текста разбивался на пары “ключ–значение”. Ключами выступали фиксированные заголовки (*Platform, Salary, Requirements, Nice to have, Responsibilities, Benefits*), а значениям сохранялся исходный порядок строк с учётом возможных маркеров списков.

## 3.8. Отбор только *iOS*- и *Android*-вакансий

Для достижения целей исследования требовалось оставить в выборке только вакансии, относящиеся к платформам *iOS* и *Android*. Для этого был проведён предварительный анализ столбца *Platform*, содержащего результаты автоматического определения платформы на основе описаний вакансий.

Из 2846 строк набора данных было выявлено 71 уникальное значение в поле *Platform*. Чистое распределение по платформам составило: 1094 вакансии для *iOS* и 1080 вакансий для *Android*, что в сумме дало 2174 вакансии, строго соответствующие требованиям.

Дополнительно были обнаружены вакансии, содержащие упоминания сразу нескольких платформ. Эти вакансии разделялись на две группы:

* Кроссплатформенные позиции, ориентированные на разработку сразу под несколько платформ;
* Вакансии, совмещающие требования к разработчикам *iOS* и *Android* в рамках одного объявления, вероятно, для оптимизации процесса найма.

Такие позиции не соответствовали целям сравнения и были удалены как методически “шумовые” либо чрезмерно ресурсоёмкие для раздельной обработки.

Отдельно были рассмотрены редкие случаи потенциально пригодных вакансий из группы смешанных платформ. В результате ручной проверки были выявлены две группы вакансий (“*iOS, tvOS*” и “*iOS/macOS*”), каждая из которых включала по 2 позиции. Поскольку они явно относились к экосистеме *iOS*, соответствующие записи были отнесены к категории *iOS*.

У 77 объявлений поле *Platform* содержало значение “*Not mentioned*”. Визуальный осмотр показал, что это либо крайне краткие публикации, либо общие описания без технических деталей; в обоих случаях модель *ChatGPT*-*4o* обоснованно не смогла определить платформу. Такие строки были исключены как информационно неполные.

Таким образом, после всех этапов фильтрации в итоговый набор данных вошли 2178 вакансий, относящихся строго к разработке под платформы *iOS* или *Android*.

## 3.9. Извлечение технологий и инструментов

Для последующего анализа были использованы три информационные секции вакансии — *Requirements*, *Nice* *to* *Have* и *Responsibilities*. Содержимое этих секций было конкатенировано в единый текстовый блок, добавленный в новый столбец *Full* *Requirements*.

**3.9.1. Валидация автоматического извлечения**

Для проверки корректности работы метода была сформирована случайная выборка из 40 вакансий (*random\_state* = 42). Каждая запись размечалась двумя способами:

* Ручная аннотация — выделение всех упомянутых технологий, библиотек и инструментов, выполненное автором.
* Автоматическая аннотация — *Python*-функция, отправлявшая текст блока *Full* *Requirements* в *API* *ChatGPT*-*4o* с промптом, приведённым в [Приложении 6](#_Приложение_6._Промпт).

Ответ модели представлял собой *JSON*-объект формата “ключ–значение”, при этом сравнивались только значения, а точность определения ключей не оценивалась. Это связано с тем, что на будущем этапе данные будут перераспределены по ключам, а на данном этапе ключи использовались исключительно для упрощения понимания требований к модели.

Сходимость результатов оценивалась посредством нечёткого сопоставления (метрика *Token* *Set* *Ratio* библиотеки *rapidfuzz*; см. раздел 2.4). Порог удовлетворительной точности был установлен на уровне 85 %. Полученный средний *Token* *Set* *Ratio* составил 96,73 %, что существенно превышает заданное значение и подтверждает надёжность автоматического метода.

После успешной валидации процедура автоматического извлечения технологий была применена ко всем 2178 отобранным вакансиям.

**3.9.2. Удаление не существующих технологий**

Несмотря на высокую точность извлечения, крупные языковые модели могут “галлюцинировать” — то есть добавлять в ответ названия технологий, фактически отсутствующие в тексте вакансии. Чтобы исключить подобные артефакты и не завысить частоту отдельных инструментов, был введён дополнительный этап очистки.

Был сформирован чёрный список потенциально ошибочных терминов, состоящих из одного слова (см. полный список в [Приложении 7](#_Приложение_7._Чёрный)). Для каждой вакансии алгоритм сравнивал извлечённые моделью технологии с исходным текстом секции *Full* *Requirements*, приведённым к нижнему регистру. Термин исключался, если одновременно выполнялись два условия:

* он входит в чёрный список;
* его строковое представление отсутствует в требованиях данной вакансии.

Такой критерий минимизировал риск случайного удаления действительно упомянутых инструментов. По итогам очистки было удалено 391 ложное упоминание.

**3.9.3. Очистка и нормализация**

Сначала был сформирован словарь переименований, в котором к одному каноническому виду сводились близкие или тождественные обозначения (например, “*compose* *ui*” → “*jetpack* *compose*”, “*objective* *c*” → “*objective*-*c*”).

Параллельно применялся “стоп-список” — перечень слишком общих или методологических понятий (“*unit* *testing*”, “*dependency* *injection*” и др.), присутствие которых не добавляло ценности для исследования технологического стека.

Каждая ячейка столбца проходила нормализацию:

1. разбиение по запятым на отдельные слова;
2. приведение к нижнему регистру и удаление лишних пробелов;
3. применение словаря переименований;
4. исключение терминов из стоп-списка;
5. объединение уникальных названий в строку.

Результат сохранялся в обновлённом столбце той же таблицы.

Для уменьшения статистического шума были сохранены лишь те технологии, которые встречались не менее трёх раз в корпусе данных. После применения порога осталось 189 уникальных названий.

**3.9.4. Категоризация**

Для повышения информативности анализа было принято решение разбить все 189 значений на более общие категории. Для этого был создан *JSON*-файл, содержащий 28 ключей, между которыми были распределены 189 значений. Далее с помощью *Python*-функции строка с перечисленными технологиями автоматически преобразовывалась в словарь вида “категория → список технологий”: значения разделялись по запятым, сопоставлялись с соответствующей категорией и сохранялись в новую колонку.

Финальным этапом стало приведение всех значений к эталонному виду: *git* → *Git*, *github* → *GitHub*, *aws* → *AWS* и так далее.

Финальный результат выглядел следующим образом:

*{'languages\_and\_runtimes': ['Swift'], 'build\_and\_dependency\_management': ['Gradle'], 'ci\_cd\_and\_release\_automation': ['Jenkins']}*

После этого все ключи были преобразованы в отдельные колонки со своими значениями.

## 3.10. Анализ

ОГРАНИЧЕНИЯ

Автор исследования не проводил собственную оценку качества перевода для всего массива текстов, что может приводить к определённым рискам недостоверности результатов перевода.

Тем не менее, учитывая ранее упомянутое исследование ([Gao, Wang & Hou, 2023](#Gao_Wang_Hou_2023)), демонстрирующее высокую степень адекватности машинного перевода, автор считает, что возможные погрешности незначительно влияют на понимание ключевых слов, касающихся технологий, и не искажают общую суть переводимых текстов.

РЕЗУЛЬТАТЫ

Дополнительный поиск через локальные домены позволил обнаружить 149 вакансий, уникальных для локального домена, что составило 5% от общего числа уникальных вакансий.

## 1. Базовые языки и инструменты

**1.1. languages\_and\_runtimes**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **languages\_and\_runtimes** | **EU count** | **EU %** | **languages\_and\_runtimes** | **NA count** | **NA %** |
| Swift | 683 | 84.95 | Swift | 234 | 79.59 |
| Objective-C | 279 | 34.7 | Objective-C | 148 | 50.34 |
| Kotlin | 34 | 4.23 | JavaScript | 26 | 8.84 |
| Java | 28 | 3.48 | Java | 18 | 6.12 |
| C++ | 24 | 2.99 | C++ | 13 | 4.42 |
| JavaScript | 18 | 2.24 | Kotlin | 11 | 3.74 |
| Kotlin Multiplatform | 14 | 1.74 | C | 9 | 3.06 |
| Python | 13 | 1.62 | Python | 9 | 3.06 |
| TypeScript | 10 | 1.24 | SQL | 8 | 2.72 |
| C | 8 | 1 | C# | 5 | 1.7 |
| C# | 6 | 0.75 | Go | 3 | 1.02 |
| SQL | 5 | 0.62 | PHP | 2 | 0.68 |
| PHP | 5 | 0.62 | Dart | 2 | 0.68 |
| Go | 4 | 0.5 | TypeScript | 2 | 0.68 |
| Ruby | 3 | 0.37 | Rust | 1 | 0.34 |
| Dart | 2 | 0.25 | Kotlin Multiplatform | 1 | 0.34 |
| Node.js | 2 | 0.25 | CSS | 1 | 0.34 |
| HTML | 1 | 0.12 | HTML | 1 | 0.34 |
| CSS | 1 | 0.12 | Ruby | 1 | 0.34 |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **languages\_and\_runtimes** | **EU count** | **EU %** | **languages\_and\_runtimes** | **NA count** | **NA %** |
| Kotlin | 600 | 78.23 | Kotlin | 208 | 66.45 |
| Java | 368 | 47.98 | Java | 186 | 59.42 |
| C++ | 79 | 10.3 | C++ | 29 | 9.27 |
| Kotlin Multiplatform | 29 | 3.78 | C | 23 | 7.35 |
| C | 24 | 3.13 | JavaScript | 20 | 6.39 |
| Python | 22 | 2.87 | Kotlin Multiplatform | 12 | 3.83 |
| JavaScript | 12 | 1.56 | Python | 12 | 3.83 |
| Swift | 10 | 1.3 | Swift | 10 | 3.19 |
| SQL | 8 | 1.04 | Objective-C | 7 | 2.24 |
| C# | 8 | 1.04 | C# | 6 | 1.92 |
| TypeScript | 6 | 0.78 | SQL | 3 | 0.96 |
| Go | 4 | 0.52 | Node.js | 3 | 0.96 |
| Dart | 4 | 0.52 | PHP | 3 | 0.96 |
| Node.js | 3 | 0.39 | CSS | 1 | 0.32 |
| Rust | 3 | 0.39 | Ruby | 1 | 0.32 |
| PHP | 2 | 0.26 | Go | 1 | 0.32 |
| Ruby | 1 | 0.13 | Dart | 1 | 0.32 |
| Objective-C | 1 | 0.13 | TypeScript | 1 | 0.32 |
|  |  |  | HTML | 1 | 0.32 |

**ide\_and\_code\_editors**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Название** | **EU count** | **EU %** | **Название** | **NA count** | **NA %** |
| Xcode | 143 | 17.79 | Xcode | 70 | 23.81 |
| Android Studio | 2 | 0.25 | Android Studio | 1 | 0.34 |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Название** | **EU count** | **EU %** | **Название** | **NA count** | **NA %** |
| Android Studio | 124 | 16.17 | Android Studio | 64 | 20.45 |
| Eclipse | 2 | 0.26 | Eclipse | 2 | 0.64 |

**version\_control\_and\_branching**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Название** | **EU count** | **EU %** | **Название** | **NA count** | **NA %** |
| Git | 235 | 29.23 | Git | 101 | 34.35 |
| GitFlow | 22 | 2.74 | SVN | 16 | 5.44 |
| GitLab | 18 | 2.24 | Mercurial | 12 | 4.08 |
| Bitbucket | 16 | 1.99 | GitHub | 10 | 3.4 |
| GitHub | 12 | 1.49 | Bitbucket | 3 | 1.02 |
| SVN | 10 | 1.24 | GitFlow | 2 | 0.68 |
| Mercurial | 3 | 0.37 | Apache Subversion | 2 | 0.68 |
| Gerrit | 2 | 0.25 | GitLab | 1 | 0.34 |
| Apache Subversion | 1 | 0.12 |  |  |  |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Название** | **EU count** | **EU %** | **Название** | **NA count** | **NA %** |
| Git | 213 | 27.77 | Git | 109 | 34.82 |
| GitHub | 26 | 3.39 | GitLab | 11 | 3.51 |
| GitLab | 11 | 1.43 | GitHub | 10 | 3.19 |
| Gerrit | 11 | 1.43 | SVN | 9 | 2.88 |
| GitFlow | 9 | 1.17 | Gerrit | 7 | 2.24 |
| SVN | 7 | 0.91 | Bitbucket | 4 | 1.28 |
| Bitbucket | 6 | 0.78 | GitFlow | 2 | 0.64 |

## 2. Сборка, зависимости, архитектура

**build\_and\_dependency\_management**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Название** | **EU count** | **EU %** | **Название** | **NA count** | **NA %** |
| CocoaPods | 72 | 8.96 | CocoaPods | 15 | 5.1 |
| Swift Package Manager | 43 | 5.35 | Swift Package Manager | 7 | 2.38 |
| Carthage | 6 | 0.75 | Bazel | 4 | 1.36 |
| Gradle | 4 | 0.5 | Apache Maven | 2 | 0.68 |
| Bazel | 1 | 0.12 |  |  |  |
| Apache Maven | 1 | 0.12 |  |  |  |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Название** | **EU count** | **EU %** | **Название** | **NA count** | **NA %** |
| Gradle | 66 | 8.6 | Gradle | 32 | 10.22 |
| Apache Maven | 5 | 0.65 | Apache Maven | 4 | 1.28 |

**dependency\_injection\_frameworks**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **dependency\_injection\_frameworks** | **EU count** | **EU %** | **dependency\_injection\_frameworks** | **NA count** | **NA %** |
| Swinject | 2 | 0.25 | Swinject | 3 | 1.02 |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **dependency\_injection\_frameworks** | **EU count** | **EU %** | **dependency\_injection\_frameworks** | **NA count** | **NA %** |
| Dagger | 44 | 5.74 | Dagger | 52 | 16.61 |
| Hilt | 34 | 4.43 | Hilt | 19 | 6.07 |
| Koin | 10 | 1.3 | Koin | 4 | 1.28 |

**architectural\_patterns**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **architectural\_patterns** | **EU count** | **EU %** | **architectural\_patterns** | **NA count** | **NA %** |
| MVVM | 165 | 20.52 | MVVM | 64 | 21.77 |
| MVC | 70 | 8.71 | MVC | 43 | 14.63 |
| VIPER | 47 | 5.85 | VIPER | 19 | 6.46 |
| MVP | 38 | 4.73 | MVP | 16 | 5.44 |
| Clean Architecture | 29 | 3.61 | Clean Architecture | 10 | 3.4 |
| VIP | 5 | 0.62 | Redux | 2 | 0.68 |
| Redux | 3 | 0.37 | MVI | 1 | 0.34 |
| MVI | 2 | 0.25 |  |  |  |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **architectural\_patterns** | **EU count** | **EU %** | **architectural\_patterns** | **NA count** | **NA %** |
| MVVM | 159 | 20.73 | MVVM | 74 | 23.64 |
| MVP | 68 | 8.87 | MVP | 29 | 9.27 |
| Clean Architecture | 54 | 7.04 | Clean Architecture | 24 | 7.67 |
| MVI | 26 | 3.39 | MVC | 20 | 6.39 |
| MVC | 25 | 3.26 | MVI | 15 | 4.79 |
| VIPER | 2 | 0.26 | VIPER | 5 | 1.6 |
| Redux | 2 | 0.26 | Redux | 1 | 0.32 |

**asynchronous\_parallel\_and\_reactive\_programming**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **y** | **EU count** | **EU %** | **y** | **NA count** | **NA %** |
| RxSwift | 63 | 7.84 | Combine | 38 | 12.93 |
| Combine | 63 | 7.84 | RxSwift | 18 | 6.12 |
| GCD | 41 | 5.1 | GCD | 4 | 1.36 |
| RxJava | 1 | 0.12 | Kotlin Coroutines | 1 | 0.34 |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **y** | **EU count** | **EU %** | **y** | **NA count** | **NA %** |
| Kotlin Coroutines | 98 | 12.78 | Kotlin Coroutines | 40 | 12.78 |
| RxJava | 58 | 7.56 | RxJava | 35 | 11.18 |
| Flows | 36 | 4.69 | Flows | 8 | 2.56 |
| RxKotlin | 3 | 0.39 |  |  |  |

## 3. Бизнес-логика и взаимодействие с бэкендом

**backend\_or\_baas\_integration**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **y** | **EU count** | **EU %** | **y** | **NA count** | **NA %** |
| Firebase | 38 | 4.73 | AWS | 5 | 1.7 |
| AWS | 14 | 1.74 | Firebase | 5 | 1.7 |
| Azure | 2 | 0.25 | Azure | 2 | 0.68 |
| Firestore | 2 | 0.25 |  |  |  |
| Amazon S3 | 1 | 0.12 |  |  |  |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **y** | **EU count** | **EU %** | **y** | **NA count** | **NA %** |
| Firebase | 45 | 5.87 | Firebase | 19 | 6.07 |
| AWS | 7 | 0.91 | AWS | 4 | 1.28 |
| Firestore | 2 | 0.26 | Firestore | 1 | 0.32 |
| Azure | 2 | 0.26 | Amazon S3 | 1 | 0.32 |
| Amazon S3 | 1 | 0.13 |  |  |  |

**networking\_and\_api**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **networking\_and\_api** | **EU count** | **EU %** | **networking\_and\_api** | **NA count** | **NA %** |
| REST API | 184 | 22.89 | REST API | 73 | 24.83 |
| GraphQL | 26 | 3.23 | GraphQL | 18 | 6.12 |
| Alamofire | 15 | 1.87 | Alamofire | 11 | 3.74 |
| WebSockets | 14 | 1.74 | WebSockets | 2 | 0.68 |
| URLSession | 6 | 0.75 | Moya | 1 | 0.34 |
| Protocol Buffers | 2 | 0.25 | URLSession | 1 | 0.34 |
| Moya | 2 | 0.25 | Retrofit | 1 | 0.34 |
| gRPC | 2 | 0.25 |  |  |  |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **networking\_and\_api** | **EU count** | **EU %** | **networking\_and\_api** | **NA count** | **NA %** |
| REST API | 160 | 20.86 | REST API | 98 | 31.31 |
| Retrofit | 49 | 6.39 | Retrofit | 36 | 11.5 |
| GraphQL | 18 | 2.35 | GraphQL | 21 | 6.71 |
| OkHttp | 10 | 1.3 | OkHttp | 12 | 3.83 |
| WebSockets | 3 | 0.39 | WebSockets | 7 | 2.24 |
| gRPC | 3 | 0.39 | Protocol Buffers | 3 | 0.96 |
| Protocol Buffers | 2 | 0.26 | gRPC | 1 | 0.32 |

**data\_and\_caching**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **data\_and\_caching** | **EU count** | **EU %** | **data\_and\_caching** | **NA count** | **NA %** |
| Core Data | 102 | 12.69 | Core Data | 53 | 18.03 |
| Realm | 29 | 3.61 | SQLite | 9 | 3.06 |
| SQLite | 17 | 2.11 | MySQL | 9 | 3.06 |
| MySQL | 4 | 0.5 | Realm | 8 | 2.72 |
| NoSQL | 2 | 0.25 | NoSQL | 6 | 2.04 |
| Room | 1 | 0.12 | MongoDB | 5 | 1.7 |
|  |  |  | Amazon DynamoDB | 2 | 0.68 |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **data\_and\_caching** | **EU count** | **EU %** | **data\_and\_caching** | **NA count** | **NA %** |
| Room | 54 | 7.04 | Room | 25 | 7.99 |
| SQLite | 31 | 4.04 | Realm | 5 | 1.6 |
| Realm | 9 | 1.17 | SQLite | 5 | 1.6 |
| NoSQL | 5 | 0.65 | MySQL | 4 | 1.28 |
| MySQL | 3 | 0.39 | MongoDB | 1 | 0.32 |
| MongoDB | 1 | 0.13 | Amazon DynamoDB | 1 | 0.32 |

**native\_interop**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **native\_interop** | **EU count** | **EU %** | **native\_interop** | **NA count** | **NA %** |
|  |  |  | AIDL | 1 | 0.34 |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **native\_interop** | **EU count** | **EU %** | **native\_interop** | **NA count** | **NA %** |
| Android NDK | 30 | 3.91 | JNI | 7 | 2.24 |
| JNI | 18 | 2.35 | Binder | 6 | 1.92 |
| Binder | 16 | 2.09 | Android NDK | 5 | 1.6 |
| AIDL | 13 | 1.69 | AIDL | 4 | 1.28 |

## 4. Безопасность и доступ к сервисам устройства

**security\_and\_cryptography**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **security\_and\_cryptography** | **EU count** | **EU %** | **security\_and\_cryptography** | **NA count** | **NA %** |
| OAuth 2.0 | 8 | 1 | OAuth | 6 | 2.04 |
| Keychain | 7 | 0.87 | Keychain | 6 | 2.04 |
| OAuth | 3 | 0.37 | TLS Pinning | 1 | 0.34 |
| OWASP | 2 | 0.25 |  |  |  |
| TLS Pinning | 1 | 0.12 |  |  |  |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **security\_and\_cryptography** | **EU count** | **EU %** | **security\_and\_cryptography** | **NA count** | **NA %** |
| OAuth 2.0 | 3 | 0.39 | OAuth 2.0 | 3 | 0.96 |
| TLS Pinning | 1 | 0.13 | OAuth | 1 | 0.32 |
| OWASP | 1 | 0.13 |  |  |  |
| OAuth | 1 | 0.13 |  |  |  |

**device\_and\_platform\_services\_and\_third\_party\_sdks**

**iOS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **y** | **EU count** | **EU %** | **y** | **NA count** | **NA %** |
| iOS SDK | 35 | 4.35 | Push Notifications | 26 | 8.84 |
| Push Notifications | 34 | 4.23 | iOS SDK | 21 | 7.14 |
| Core ML | 34 | 4.23 | Location | 10 | 3.4 |
| HealthKit | 30 | 3.73 | Bluetooth | 7 | 2.38 |
| Bluetooth | 10 | 1.24 | Facebook | 6 | 2.04 |
| BLE | 9 | 1.12 | Google Maps | 5 | 1.7 |
| ARKit | 9 | 1.12 | Facebook SDK | 5 | 1.7 |
| Core Location | 9 | 1.12 | AVPlayer | 4 | 1.36 |
| MapKit | 7 | 0.87 | WebKit | 4 | 1.36 |
| AVFoundation | 4 | 0.5 | Core Location | 4 | 1.36 |
| Location | 4 | 0.5 | AVFoundation | 3 | 1.02 |
| watchOS | 3 | 0.37 | HealthKit | 2 | 0.68 |
| Camera | 3 | 0.37 | Stripe SDK | 2 | 0.68 |
| Stripe SDK | 2 | 0.25 | Camera | 2 | 0.68 |
| Sensors | 1 | 0.12 | ARKit | 2 | 0.68 |
| WebViews | 1 | 0.12 | NFC | 1 | 0.34 |
| CarPlay | 1 | 0.12 | Sensors | 1 | 0.34 |
| Core Bluetooth | 1 | 0.12 | Core ML | 1 | 0.34 |
| NFC | 1 | 0.12 | CarPlay | 1 | 0.34 |
| Facebook SDK | 1 | 0.12 | WebViews | 1 | 0.34 |
| Google Maps | 1 | 0.12 |  |  |  |

**Android**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **y** | **EU count** | **EU %** | **y** | **NA count** | **NA %** |
| Android SDK | 173 | 22.56 | Android SDK | 107 | 34.19 |
| Push Notifications | 18 | 2.35 | Push Notifications | 16 | 5.11 |
| Android HAL | 12 | 1.56 | Bluetooth | 11 | 3.51 |
| BLE | 9 | 1.17 | Location | 7 | 2.24 |
| Android Auto | 7 | 0.91 | Android Auto | 7 | 2.24 |
| Camera | 7 | 0.91 | BLE | 5 | 1.6 |
| Location | 5 | 0.65 | Google Maps | 5 | 1.6 |
| Bluetooth | 4 | 0.52 | Google Play Services | 4 | 1.28 |
| ExoPlayer | 4 | 0.52 | CarPlay | 4 | 1.28 |
| Android TV | 4 | 0.52 | Android HAL | 3 | 0.96 |
| AOSP | 4 | 0.52 | Camera | 3 | 0.96 |
| ARCore | 3 | 0.39 | Facebook SDK | 3 | 0.96 |
| Fire TV | 3 | 0.39 | NFC | 2 | 0.64 |
| WebViews | 3 | 0.39 | ARKit | 2 | 0.64 |
| Sensors | 3 | 0.39 | Core Bluetooth | 2 | 0.64 |
| Google Play Services | 2 | 0.26 | Sensors | 2 | 0.64 |
| Facebook SDK | 1 | 0.13 | WebViews | 1 | 0.32 |
| Google Maps | 1 | 0.13 | ExoPlayer | 1 | 0.32 |
| NFC | 1 | 0.13 | Android TV | 1 | 0.32 |
| CarPlay | 1 | 0.13 | Stripe SDK | 1 | 0.32 |
|  |  |  | AOSP | 1 | 0.32 |

|  |
| --- |
|  |

## 5. Пользовательский интерфейс

**ui\_native**

**iOS**

**Android**

**architectural\_patterns**

**iOS**

**fAndroid**

Текст.

ЗАКЛЮЧЕНИЕ

Текст.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Apatsidis, I., Georgiou, K., Mittas, N., & Angelis, L. (2021). A Study of Remote and On-site ICT Labor Market Demand using Job Offers from Stack Overflow. *ResearchGate*. <https://www.researchgate.net/publication/355685512_A_Study_of_Remote_and_On-site_ICT_Labor_Market_Demand_using_Job_Offers_from_Stack_Overflow>

Barousse, L. (2023, 17 февраля). *I analyzed 3,275,597 jobs to solve THIS* [Видео]. <https://www.youtube.com/watch?v=7G_Kz5MOqps&ab_channel=LukeBarousse>

Cummings, J., Janicki, T., & Matthews, K. (2023). Invited Paper: Survey of Technology and Skills in Demand: The 2022 Update. *Journal of Information Systems Education*, 34(2), 243-255. <https://aisel.aisnet.org/jise/vol34/iss2/10>

Data Nerds. (n.d.). *Top Skills & Pay of Data Nerds*. <https://datanerd.tech/>

DataCamp. (2025). *Fuzzy String Matching in Python Tutorial*. <https://www.datacamp.com/tutorial/fuzzy-string-python>

Gao, Y., Wang, R., & Hou, F. (2023). How to Design Translation Prompts for ChatGPT: An Empirical Study. *arXiv*. <https://arxiv.org/pdf/2304.02182>

Georgakopoulos, S. (2024, 11 апреля). Mobile Developer Population Forecast 2024. *SlashData*. <https://www.slashdata.co/post/mobile-developer-population-forecast-2024>

GPT-4o. (n.d.). *Wikipedia*. <https://en.wikipedia.org/wiki/GPT-4o>

Hanna, Samer Odeh. (2024). Analysis of the Required Skills by Industry for Android Jobs. *SSRN*. [http://dx.doi.org/10.2139/ssrn.5008188](https://dx.doi.org/10.2139/ssrn.5008188)

IDC. (2024). *Smartphone Market Insights*. <https://www.idc.com/promo/smartphone-market-share>

Kang, T., Park, H., & Shin, S. (2020). Searching, Examining, and Exploiting In-demand Technical (SEE IT) Skills using Web Data Mining. *Software Engineering and Knowledge Engineering*. <https://ksiresearch.org/seke/seke20paper/paper058.pdf>

ModelPredict. (2021). *Comparison of language identification models*. <https://modelpredict.com/language-identification-survey>

RapidFuzz. (n.d.). *RapidFuzz* [GitHub-репозиторий]. <https://github.com/rapidfuzz/RapidFuzz>

SerpApi. (n.d. -a). *Google Search API*. <https://serpapi.com>

SerpApi. (n.d. -b). *Google Jobs API*. <https://serpapi.com/google-jobs-api>

Stack Overflow. (2024). *Annual Developer Survey 2024*. <https://survey.stackoverflow.co/2024>

Statista. (2021). *Forecast number of mobile users worldwide from 2020 to 2025*. <https://www.statista.com/statistics/218984/number-of-global-mobile-users-since-2010>

Statista. (2023). *App — Worldwide*. <https://www.statista.com/outlook/amo/app/worldwide>

Statista. (2024). *Number of mobile app downloads worldwide from 2016 to 2023*. <https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads>

Ternikov, A. A., & Aleksandrova, E. A. (2020). Demand for skills on the labor market in the IT sector. *Business Informatics*, 14(2), 64—83. <https://doi.org/10.17323/2587-814X.2020.2.64.83>

Яндекс Практикум. (2024, 3 июня). Android-разработчики в 2024 году: какие нужны скилы и как проходит процесс найма. *Habr*. <https://habr.com/ru/companies/yandex_praktikum/articles/824672/>

Яндекс Практикум. (2024, 10 июня). iOS-разработчики в 2024 году: какие нужны скилы и как проходит процесс найма. *Habr*. <https://habr.com/ru/companies/yandex_praktikum/articles/819715/>

ПРИЛОЖЕНИЯ

## Приложение 1. Полный список стран

* *Austria*
* *Belgium*
* *Bulgaria*
* *Canada*
* *Croatia*
* *Cyprus*
* *Czechia*
* *Denmark*
* *Estonia*
* *Finland*
* *France*
* *Germany*
* *Greece*
* *Hungary*
* *Iceland*
* *Ireland*
* *Italy*
* *Latvia*
* *Liechtenstein*
* *Lithuania*
* *Luxembourg*
* *Malta*
* *Mexico*
* *Netherlands*
* *Norway*
* *Poland*
* *Portugal*
* *Romania*
* *Slovakia*
* *Slovenia*
* *Spain*
* *Sweden*
* *Switzerland*
* *United Kingdom*
* *United States*

## Приложение 2. Директории с вакансиями

Директории, в которые собирались вакансии:

* *“./data/dotcom\_domain/Android”* — вакансии для *Android*-разработчиков, найденные через *google.com*;
* *“./data/dotcom\_domain/iOS”* — вакансии для *iOS*-разработчиков, найденные через google.com;
* *“./data/local\_domain/Android”* — вакансии для *Android*-разработчиков, найденные через локальные домены;
* *“./data/local\_domain/iOS”* — вакансии для *iOS*-разработчиков, найденные через локальные домены.

## Приложение 3. Единый набор данных

Итоговый набор данных включал следующие столбцы:

* ***Location*** — страна, в которой проводился поиск вакансий;
* ***Region*** — регион страны поиска (*“Europe”* или*” Northern* *America”*);
* ***EU* *Member*** — флаг, указывающий на членство страны в Европейском Союзе (*True/False*);
* ***Schengen* *Agreement*** — флаг, указывающий на участие страны в Шенгенском соглашении (*True/False*);
* ***Google* *Domain* *Type*** — тип домена, используемого для поиска (“*default*”/ “*local*”);
* ***Google Domain Used*** — конкретный домен *Google*, использованный при поиске (например, “*google.com*”);
* ***Job* *Title*** — название вакансии (например, *“Senior Android Developer”*);
* ***Company Name*** — название компании, разместившей вакансию;
* ***Job Location*** — местоположение вакансии (например, *“Vienna, Austria”* или *“Anywhere”* для удалённой работы);
* ***Apply Options*** — список сайтов, где можно подать заявку на вакансию (например, *“Indeed, LinkedIn, Startup Jobs”*);
* ***Job Description*** — описание вакансии, включая требования и обязанности;
* ***Work from home*** — возможность удалённой работы (*True/False* или *NaN*, если не указано);
* ***Salary*** — указанная в вакансии заработная плата (если предоставлена);
* ***Schedule type*** — тип занятости (*Full-time*, *Part-time* и т. д.);
* ***Qualifications*** — требования к кандидату (например, *“No degree mentioned”*, если степень не указана, *NaN*, если указано);
* ***Job ID*** — уникальный идентификатор вакансии;
* ***Search Date*** — дата и время выполнения поиска (например, “2025-01-13 12:08:48 *UTC*”);
* ***Search Query*** — поисковый запрос, использованный для поиска вакансий (*“Android developer”* или *“iOS developer”*).

## Приложение 4. Промпт для разбиения вакансий на логические части

***System Prompt:*** *"You are an AI assistant. Your role is to extract specific information from job descriptions and format them in a strict structure. "*

***User Prompt:*** *""" I will provide a job description. Please extract and present the information in \*\*this exact order\*\*:*

1. *Platform: (Android/iOS/Cross-platform)*
2. *Salary: (If stated; otherwise "Not mentioned")*
3. *Requirements: (verbatim from the job description or "Not mentioned")*
4. *Nice to have: (verbatim or "Not mentioned")*
5. *Responsibilities: (verbatim or "Not mentioned")*
6. *Benefits: (verbatim or "Not mentioned")*

*\*\*Guidelines\*\*:*

*- \*\*DO NOT reword, paraphrase, or summarize\*\* any part of the job description. Copy the sentences exactly as they appear.*

*- Combine all mandatory or required skill sections (e.g., "Requirements," "Skills," "Key Technologies," "About You") under \*\*Requirements\*\*.*

*- If the job description specifically says something is "a plus," "beneficial," or otherwise indicates it’s optional, place it under "Nice to have" even if it appears under a "Requirements" heading in the job description.*

*- If there is an "About you" or "About Role" section (or similar) that describs duties or tasks, include those under "Responsibilities".*

*- If the information is not in the job description, write "Not mentioned" for that section.*

*- For multiple platforms (e.g., Android, iOS), list them all in \*\*Platform\*\* and use headings under Requirements (and other sections, if needed) like "General Requirements:", "For Android Developers:", "For iOS Developers:".*

*- Present your answer \*\*only\*\* in the format above.*

*---*

*Here is the job description:*

*"""*

## Приложение 5. Пример вакансии после преобразования

*1. Platform: Android*

*2. Salary: Not mentioned*

*3. Requirements:*

*• Passionate about mobile platforms and translating requirements into a user friendly app*

*• Excellent knowledge of Kotlin*

*• Strong knowledge of architecture patterns such as MVVM*

*• Proven experience of developing and publishing native Android apps to the Google Play store in a professional environment*

*• Good knowledge of Kotlin Coroutines, Hilt and Retrofit.*

*4. Nice to have: Not mentioned*

*5. Responsibilities:*

*• Ensure that the app meets our quality standards by creating unit tests and doing code reviews*

*• Design and maintain the architecture of new features and existing ones*

*• Collaborate with cross-functional teams, designers and technical product owners*

*• Stay up to date with programming best practices and new technology trends*

*• Working autonomously you will bring strong communication skills and the ability to mentor colleagues*

*6. Benefits: Not mentioned*

## Приложение 6. Промпт для извлечения технологий из вакансий

*system\_prompt\_1 = """*

*You are a senior extraction assistant.*

*GOAL*

*Given a natural-language job-vacancy text, return one and only one valid, minified JSON object that maps every technology term found in the input to exactly ONE key from ALLOWED\_KEYS.*

*STRICT RULES*

*1. Output = raw JSON only (no Markdown, no comments, no extra text).*

*2. Use keys from ALLOWED\_KEYS exactly as written. Omit any key whose array would be empty.*

*3. Each value is an \*\*array of unique strings\*\*, sorted alphabetically and preserving the term’s original spelling/case from the input.*

*4. Ignore soft skills and vague nouns (e.g. communication, documentation, performance, detail, English, collaboration, problem-solving, university).*

*5. \*\*IGNORE\*\* overly generic technology words/phrases (e.g. "design patterns", "android ui", "json", "xml", "clean code", "API integration", "unit testing", "Continuous integration", "CI", "CD", "CI/CD", "cryptography", "application testing" ect.).*

*6. DO NOT hallucinate. Include a term \*\*only\*\* if it appears verbatim in the input text.*

*7. If a technology term matches \*\*exactly\*\* (case-insensitive) one of the values in ALLOWED\_KEYS, then it MUST be assigned to that exact key — even if the name could hypothetically fit other categories.*

*8. The order of keys in the output JSON MUST strictly follow the order of keys as they appear in the ALLOWED\_KEYS section below. Only include keys that have at least one matched term. If a key has no matched terms, it MUST be completely omitted from the output, even if this breaks the visual continuity of the ALLOWED\_KEYS order.*

*ALLOWED\_KEYS (with examples)*

*{*

*"languages\_and\_runtimes": ["Swift", "Kotlin", "Java", "Dart", "Objective-C", "Coroutines", "RxSwift", "Combine", "GCD", "RxJava", "Flows", "JavaScript", "TypeScript", "C", "C++", "Python", "Golang", "PHP"],*

*"ui\_and\_cross\_platform\_frameworks": ["SwiftUI", "UIKit", "Jetpack Compose", "Flutter", "React Native", "Xamarin", "Ionic", "WatchKit", "Cocoa Touch", "PhoneGap", "Cordova", "Kotlin Multiplatform"],*

*"architectural\_patterns": ["MVVM", "VIPER", "Clean Architecture", "MVI", "MVC", "MVP", "Redux"],*

*"dependency\_injection\_frameworks": ["Dagger", "Hilt", "Koin", "Swinject"],*

*"build\_and\_dependency\_management": ["Gradle", "CocoaPods", "SPM", "Bazel", "Buck", "Xcode", "Android Studio", "CircleCI", "Bamboo", "CocoaPods"],*

*"data\_and\_caching": ["Core Data", "Room", "Realm", "SQLite", "Firestore", "MongoDB", "SAP UltraLite", "MySQL", "NoSQL"],*

*"networking\_and\_api": ["Retrofit", "OkHttp", "URLSession", "Alamofire", "GraphQL", "REST API", "WebSockets", "gRPC", "Protocol Buffers"],*

*"backend\_or\_baas\_integration": ["Firebase", "AWS Amplify", "Azure Mobile", "Parse", "AWS Mobile Hub", "AWS Cognito", "AWS S3"],*

*"device\_and\_platform\_services\_and\_third\_party\_sdks": ["ARKit", "HealthKit", "CoreML", "Core Animation", "Android SDK", "Android NDK", "Push Notifications", "BLE", "NFC", "Camera", "Location", "Sensors", "Stripe SDK", "Facebook SDK", "AdMob", "Google Maps", "ExoPlayer", "Glide", "Stripe", "PayPal SDK", "Binder", "AIDL", "JNI", "CTS", "HAL", "LoRa", "CarPlay", "Android Auto", "CydiaSubstrate", "Frida", "WebViews"],*

*"security\_and\_cryptography": ["Keychain", "TLS pinning", "OAuth2", "Veracode", "Checkmarx", "OWASP"],*

*"testing\_frameworks": ["XCTest", "JUnit", "Espresso", "Mockito", "Robolectric"],*

*"debugging\_or\_profiling": ["Instruments", "Android Profiler"],*

*"version\_control": ["Git", "GIT", "git", "SVN", "Mercurial", "Gerrit", "GitFlow", "SourceTree", "Fork", "Bitbucket"],*

*"ci\_cd\_and\_release\_automation": ["GitHub Actions", "Jenkins", "Bitrise", "fastlane", "CircleCI", "Bamboo", "GitLab CI", "Docker"],*

*"monitoring\_analytics\_and\_crash\_reporting": ["Crashlytics", "Sentry", "Datadog", "Firebase Analytics", "App Center"],*

*"development\_methodologies": ["Scrum", "Kanban", "Agile", "SAFe", "TDD", "BDD", "DevOps"],*

*"testing\_process\_and\_qa": ["test coverage", "regression testing"],*

*"code\_quality\_and\_static\_analysis": ["SonarQube", "SwiftLint", "Veracode", "Checkmarx"],*

*"documentation\_and\_knowledge\_sharing": ["Swagger", "OpenAPI", "Javadoc", "Confluence", "HIG"],*

*"collaboration\_pm\_and\_design\_handoff": ["Jira", "Trello", "Figma", "Zeplin", "Rally/AgileCentral"],*

*"distribution\_and\_store\_operations": ["TestFlight", "App Store Connect", "Google Play Console"],*

*"compliance\_and\_certifications": ["SOC 2", "GDPR"]*

*}*

*EXAMPLE RESPONSE*

*{*

*"languages\_and\_runtimes":["Swift","Java"],*

*"version\_control":["Git"]*

*}*

*"""*

*user\_prompt\_1 = """*

*Extract every technology, tool, framework, library, service or formal methodology mentioned in the text below and output the JSON exactly as specified in the system prompt.*

*---*

*INPUT TEXT:*

*"""*

## Приложение 7. Список технологий для проверки

*blacklist = {*

*'objective-c', 'git', 'agile', 'android sdk', 'android ndk', 'hig', 'tdd',*

*'android studio', 'c++', 'swift', 'c#', 'dagger', 'java', 'swiftui',*

*'kotlin', 'dart', 'python', 'javascript', 'typescript', 'golang', 'php',*

*'hilt', 'koin', 'swinject',*

*'gradle', 'bazel', 'buck', 'xcode',*

*'sqlite', 'realm', 'room', 'firestore', 'mongodb',*

*'retrofit', 'okhttp', 'graphql',*

*'firebase', 'parse',*

*'camera', 'sensors', 'location', 'nfc', 'ble',*

*'mvvm', 'mvc', 'mvp', 'mvi', 'viper', 'redux',*

*'scrum', 'kanban', 'bdd', 'devops', 'safe',*

*'jenkins', 'docker', 'bitrise', 'fastlane', 'circleci',*

*'xctest', 'junit', 'espresso', 'mockito', 'robolectric',*

*'crashlytics', 'sentry', 'datadog',*

*'jira', 'trello', 'figma', 'zeplin'*

*}*

## Приложение 8. Список технологий для проверки