САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Направление: 02.03.02 «Фундаментальная информатика и информационные технологии

ООП: Программирование и информационные технологии

**ОТЧЕТ О НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ**

**Тема задания:**

Описание реализации серверной части приложения тренировки ходьбы людей, страдающих заболеваниями нижних конечностей.

**Выполнил:**

Панюшин Даниил Васильевич – студент группы 19Б12-пу бакалавриата.

**Руководитель научно-исследовательской работы:**

Раевская Анастасия Павловна, доцент кафедры математической теории экономических решений Санкт-Петербургского Государственного Университета, кандидат физико-математических наук.

Санкт-Петербург

2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 3

Постановка задачи 4

Глава 1. 5

1.1. Первый параграф 5

1.2. Второй параграф 5

ЗАКЛЮЧЕНИЕ 6

Список использованных источников 7

# ВВЕДЕНИЕ

Практически каждое приложение в современном мире использует сервера для различных целей. Бэкенд определяет логику хранения и обработки данных, важных для конкретного бизнеса, осуществляет рендер веб страниц и многое другое. Наш проект по созданию приложения для тренировки ходьбы людей, страдающих заболеваниями нижних конечностей – не исключение. Фронтенд и мобильное приложение не могут обмениваться информацией сами по себе – удалённый сервер служит необходимой прослойкой для работы с данными, полученными от пользователей и визуализируемыми на страницах фронтенда.

Проведение исследовательской работы необходимо с целью определения и описания методов и технологий, которые могут быть использованы для реализации вышеупомянутых задач. Данная работа позволит избежать архитектурных ошибок и их исправлений на последующих этапах разработки продукта.

На текущий момент бекенд сервисы практически любого назначения могут быть реализованы на многих языках программирования, таких как *Java, Python, Golang, PHP, .NET* и т.д. Популярность различных языков проиллюстрирована на Рисунке 1.

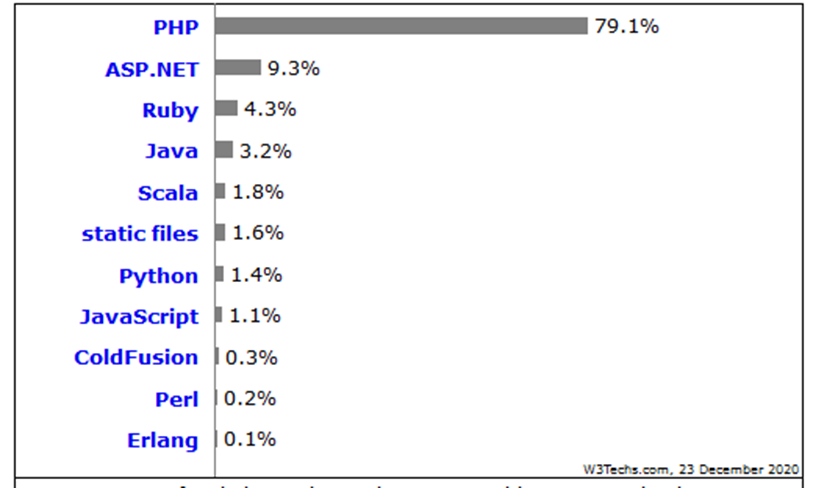


Рисунок 1 - Популярность различных языков программирования в сфере веб-разработки

Для каждого из вышеперечисленных языков программирования написаны фреймворки[[1]](#footnote-1), обладающие как достоинствами, так и недостатками, исходящими как от конкретной реализации, так и от особенностей языка.

# Глава 1. Язык программирования

## Golang

*Golang* или просто *Go* - компилируемый многопоточный язык программирования, разработанный внутри компании Google. Go позволяет создавать высокоэффективные программы, исполняющиеся на современных распределённых системах и многоядерных процессорах. Этот язык обладает строгой статической типизацией, имеет достаточно простой синтаксис, основанный на *C*, но существенно доработанный, с большим количеством синтаксического сахара[[2]](#footnote-2).

Ключевые достоинства языка:

* Go компилируется в один небольшой двоичный файл с нулевыми зависимостями, что делает его очень быстрым и обеспечивает кроссплатформенное[[3]](#footnote-3) развертывание.
* Скорость работы – модель параллельной работы и масштабируемости использования ЦПУ[[4]](#footnote-4) позволяет Go получить хорошую скорость работы, особенно при параллельных вычислениях.
* Для веб-разработки в большинстве случаев не требуются сторонние библиотеки – язык включает собственную библиотеку веб шаблонизации.
* Строгая статическая типизация[[5]](#footnote-5) данного языка важна в крупных приложениях.
* Go позволяет быстро разрабатывать масштабируемые и безопасные веб-приложения.
* Поддержка - язык разработан компанией Google, имя которой говорит само за себя, так что поддержка и обновление языка будет происходить стабильно в ближайшие годы. К тому же, сообщество пользователей языка постоянно растёт, одной из причин чего является простота освоения.

## Python

Python – высокоуровневый интерпретируемый мультипарадигменный[[6]](#footnote-6) язык программирования общего назначения с открытым исходным кодом, обладающий динамической[[7]](#footnote-7)строгой типизацией и автоматическим управлением памятью.

Наиболее активно данный язык применяется в сферах машинного обучения, DevOps[[8]](#footnote-8) и веб-разработки.

Основные преимущества:

* Язык обладает простым минималистическим синтаксисом, что делает его простым в освоении.
* Огромное количество всевозможных библиотек, начиная с библиотек для машинного обучения и заканчивая библиотеками для написания веб или мобильных приложений.
* Python позволяет быстро писать рабочий код с необходимым функционалом, что особенно полезно при работе со строгими временными ограничениями.

Главным же недостатком языка Python является зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках.

## Java

*Java* – строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle).

Особенность языка Java – использование Java-машины (или, сокращённо, JVM[[9]](#footnote-9)), которая транслирует java-код в специальный байт-код, что позволяет запускать приложение на любой компьютерной архитектуре, поддерживающая JVM.

Java используется для создания: мобильных приложений, веб приложений, видеоигр, приложений для работы с базами данных и прочее.

Основные достоинства:

* Кроссплатформенность – как было сказано выше, JVM позволяет писать приложения на Java для широкого спектра платформ.
* Простота синтаксиса по сравнению с C/C++.
* Безопасность. При создании платформы Java создатели проделали большую работу по её защите. Обойти или взломать механизмы крайне сложно. Наглядным примером механизма безопасности Java является использование классов, использующих цифровую подпись, а, следовательно, полные права предоставляются только при полном доверии автору класса.
* Производительность. Изначально она вызывала вопросы. Новые версии динамических компиляторов Java не уступают традиционным из других платформ. Мощный прирост скорости обработки даёт оптимизация тех фрагментов кода, которая исполняется чаще. При необходимости те или иные приёмы оптимизации включаются или отменяются JIT-компилятором[[10]](#footnote-10).
* Надёжность. Программы на Java стабильно работают в любых условиях. Компилятор способен выявить ошибки ещё до выполнения кода, то есть на ранних стадиях. Контроль выполнения позволяет предотвратить сбои в памяти. Сами указатели можно применять не везде, а только там, где это необходимо.
* Динамичность и адаптируемость. При необходимости можно добавить в библиотеки новые объекты и методы. При этом нет необходимости изменять приложения, использующие данные библиотеки.
* Эффективные и удобные сетевые возможности. Имеется библиотека для передачи данных по самым распространённым протоколам. Существует механизм удалённого вызова методов.
* Поддержка. Язык пользуется популярностью на протяжении более двадцати лет и за это время вокруг него сформировалось значительное сообщество, способное дать ответ практически на любой вопрос касательно данного языка.

Недостатки Java:

* Платное коммерческое использование.
* Отсутствует нативный дизайн. Для создания интерфейса пользователя на компьютере для java-приложения нет ни одного java-инструмента, поэтому разработчики используют сторонние инструменты и библиотеки.
* Синтаксис Java всё ещё сложен. Это приводит к необходимости создания сложных и объёмных структур кода.
* Отсутствие поддержки корутин[[11]](#footnote-11).

## Выбор

Рассмотрев приведённые выше языки программирования автор данной научной работы сделал выбор в пользу языка Java для написания логики связи мобильного приложения или фронтенда с бекендом.

Причинами такого решения являются надёжность и безопасность, предоставляемая языком Java. Это необходимо для нашего проекта, так как имеет место обработка приватных данных клиентов.

Другой причиной такого решения стоит отметить поддержку данного языка, которая сильно облегчает работу с ним.

# Глава 2. Система управления базами данных

## **2.1 Классификация**

Существует два основных вида баз данных: SQL[[12]](#footnote-23688) (так же известные как реляционные) базы и NoSQL. Вторые, в свою очередь делятся на следующие подтипы по принципу хранения данных: колоночные, ключ-значение, документные.

## **2.2 Реляционные СУБД**

Классические реляционные базы данных чаще всего применяются для создания продуктов, использующих Online Transaction Processing - принцип организации базы данных, при котором система работает с большим потоком малообъёмных транзакций[[13]](#footnote-25850). Клиенты таких систем должны получать ответы от системы с минимальным временем отклика. Так же, ввиду использования транзакций, являющихся атомарными[[14]](#footnote-425) операциями, должна существовать возможность откатить любые операции в рамках одной транзакции.

Реляционные СУБД чаще всего применяются при работе с большим количеством сущностей (представленных в базе таблицами), соединенных различными видами связей: один-к-одному, один-ко-многим, многие-ко-многим.

Наиболее популярными реляционными СУБД являются Oracle Database, Microsoft SQL Server, PostgreSQL, MySQL. Далее приведено их описание:

*Oracle Database* - СУБД, написанная компанией Oracle, преднзначенная для облачнростых сред и может быть размещена на одном или нескольких серверах.

Достоинства:

* Обрабатывает огромные данные.
* Очень высокая степень надёжности. Каждая транзакция изолирована от других.

Недостатки:

* Является платным продуктом, что может быть слишком дорогим решением для небольших компаний.
* Высокая требовательность к аппаратным ресурсам.

*Microsoft SQL Server* - СУБД, созданная компанией Microsoft, и изначально привязанная к операционной системе Windows. Впоследствии была адаптирована и для использования на Linux.

Достоинства:

* Простота использования.
* Скорость.
* Стабильность.
* Наличие графического интерфейса.
* Синергия с другими продуктами Microsoft.

Недостатки:

* Является платным продуктом, что может быть слишком дорогим решением для небольших компаний.
* Способна задействовать все возможные ресурсы платформы, на которой запущена.

*PotgreSQL* - масштабируемая объектно-реляционная кроссплатформенная СУБД.

Достоинства:

* Позволяет работать с огромными данными.
* Полная SQL совместимость.
* Поддержка json формата.
* Множество встроенных функций.
* Доступен ряд интерфейсов, в том числе и графических.
* Бесплатная. В том числе и для коммерческого использования.

Недостатки:

* Туманная документации.
* Сложная конфигурация.
* Низкая производительность при выполнении запросов чтения или пакетных запросов.
* Проблемы с поиском провайдера для хостинга из-за не высокой популярности и трудности настройки.

*MySQL* – СУБД с открытым исходным кодом.

Достоинства:

* Простота использования.
* Широкий функционал.
* Отличная документация.
* Надёжность.
* Оптимальное использование аппаратных средств.
* Позволяет работать с огромными данными.
* Скорость работы. MySQL пренебрегает некоторыми стандартами СУБД, что позволяет получить прирост в производительности.
* Возможность взаимодействия с другими СУБД.
* Доступен ряд интерфейсов, в том числе и графических.

Недостатки:

* Некоторые операции реализованы менее недежно чем в других СУБД.
* Застой в разработке данной СУБД.
* Отсутствие встроенной поддержки XML[[15]](#footnote-2215) и OLAP.
* Для бесплатной версии доступна только платная поддержка.

## **2.3 СУБД типа ключ-значение**

# ЗАКЛЮЧЕНИЕ

Заключение должно содержать:

* краткие выводы по результатам выполненной НИР или отдельных ее этапов;
* оценку полноты решений поставленных задач;
* разработку рекомендаций и исходных данных по конкретному использованию результатов НИР.

# Список использованных источников

Список должен содержать сведения об источниках, использованных при составлении отчета. Сведения об источниках приводятся в соответствии с требованиями ГОСТ 7.1, ГОСТ 7.80, ГОСТ 7.82.

1. программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта. [↑](#footnote-ref-1)
2. синтаксические возможности, применение которых не влияет на поведение программы, но делает использование языка более удобным для человека. [↑](#footnote-ref-2)
3. способность программного обеспечения работать с несколькими аппаратными платформами или операционными системами. [↑](#footnote-ref-3)
4. центральное процессорное устройство. [↑](#footnote-ref-4)
5. типизация, при которой конечные типы переменных и функций устанавливаются на этапе компиляции. [↑](#footnote-ref-5)
6. язык программирования, поддерживающий несколько парадигм программирования, например, объектно-ориентированное программирование или процедурное. [↑](#footnote-ref-6)
7. типизация, при которой все типы выясняются уже во время выполнения программы. [↑](#footnote-ref-7)
8. методология автоматизации технологических процессов сборки, настройки и развёртывания программного обеспечения. [↑](#footnote-ref-8)
9. Java Virtual Machine [↑](#footnote-ref-9)
10. JIT-компиляция - технология увеличения производительности программных систем, использующих байт-код, путём компиляции байт-кода в машинный код или в другой формат непосредственно во время работы программы. [↑](#footnote-ref-10)
11. облегченный поток исполнения кода, который может работать поверх основных потоков программы. [↑](#footnote-ref-11)
12. Structured Query Language - язык запросов, применяющийся для управления реляционными базами данных. [↑](#footnote-ref-23688)
13. группа последовательных операций с базой данных, которая представляет собой логическую единицу работы с данными. [↑](#footnote-ref-25850)
14. единый, неделимый [↑](#footnote-ref-425)
15. eXtensible Markup Language - расширяемый язык разметки. [↑](#footnote-ref-2215)