Курсовая работа по прикладному программированию.

Твой вариант соответствует последней цифре зачетной книжки.

0. Разработка клиент-серверного приложения для управления задачами
- Цель работы: Обосновать необходимость приложения для управления задачами и его возможности.
- Техническое задание: Описание функционала (добавление, редактирование, удаление задач, управление пользователями).
- Архитектура приложения: Схема клиент-серверной архитектуры.
- Используемые технологии: Go, gRPC или REST, база данных (например, MongoDB).
- Реализация: Описание кода, примеры использования, а также настройка серверной и клиентской части.
- Тестирование: Методы тестирования (юнит-тесты, интеграционные тесты).
- Заключение: Итоги работы, достижения и возможные улучшения.

1. Создание мессенджера на основе gRPC
- Цель работы: Обосновать актуальность мессенджеров и их особенности.
- Техническое задание: Функциональные требования (отправка текстовых сообщений, поддержка пользователей, аутентификация).
- Архитектура приложения: Описание клиент-серверной архитектуры.
- Используемые технологии: gRPC, Protocol Buffers, база данных.
- Реализация: Код серверной и клиентской частей, примеры запросов и ответов.
- Безопасность: Механизмы аутентификации и шифрования.
- Заключение: Итоги, достижения и возможные направления для улучшения.

2. API для обработки изображений
- Цель работы: Пояснить значимость обработки изображений в приложениях.
- Техническое задание: Перечень функций (загрузка, обрезка, изменение размера, форматирование).
- Архитектура приложения: Структура API, описание эндпоинтов.
- Используемые технологии: Go, gRPC или REST, библиотеки для обработки изображений.
- Реализация: Код сервера и примеры запросов на обработку изображений.
- Тестирование: Методики тестирования API.
- Заключение: Достижения и области для будущего развития.

3. Система мониторинга IoT устройств
- Цель работы: Обосновать важность мониторинга IoT устройств.
- Техническое задание: Описание функциональности (подключение устройств, сбор данных).
- Архитектура приложения: Схема архитектуры, взаимодействие между устройствами и сервером.
- Используемые технологии: Go, MQTT, базы данных для хранения данных.
- Реализация: Примеры кода и описание функционала системы.
- Тестирование: Способы тестирования системы и ее компонентов.
- Заключение: Итоги работы и возможности расширения системы.

4. Чат-приложение с использованием WebSocket
- Цель работы: Обосновать необходимость использования WebSocket для обмена сообщениями.
- Техническое задание: Описание функционала (отправка и получение сообщений, поддержка нескольких пользователей).
- Архитектура приложения: Схема работы клиент-серверного взаимодействия.
- Используемые технологии: Go, WebSocket, HTML/CSS/JavaScript для клиентской части.
- Реализация: Примеры кода серверной и клиентской частей, использование WebSocket.
- Тестирование: Методики тестирования, включая нагрузочное тестирование.
- Заключение: Итоги, достижения и возможности для улучшения.

5. Система управления пользователями с аутентификацией
- Цель работы: Обосновать важность управления пользователями в приложениях.
- Техническое задание: Перечень функций (регистрация, вход, управление профилями).
- Архитектура приложения: Схема взаимодействия компонентов.
- Используемые технологии: Go, JWT для аутентификации, базы данных.
- Реализация: Код реализации аутентификации и управления пользователями.
- Безопасность: Обсуждение безопасности и защиты данных пользователей.
- Заключение: Итоги работы, достижения и возможные улучшения.

6. Веб-приложение для совместного редактирования документов
- Цель работы: Обосновать значимость совместного редактирования документов.
- Техническое задание: Описание функционала (редактирование, версия, управление правами доступа).
- Архитектура приложения: Структура приложения, взаимодействие клиентов и сервера.
- Используемые технологии: Go, WebSocket, базы данных.
- Реализация: Примеры кода и использование реальных сценариев.
- Тестирование: Методы тестирования функционала приложения.
- Заключение: Итоги работы и направления для развития.

7. Приложение для трекинга времени
- Цель работы: Пояснить важность трекинга времени для повышения продуктивности.
- Техническое задание: Описание функций (начало/окончание задач, генерация отчетов).
- Архитектура приложения: Схема клиент-серверного взаимодействия.
- Используемые технологии: Go, базы данных, библиотеки для отчетности.
- Реализация: Примеры кода и описание реализации функционала.
- Тестирование: Способы тестирования и валидации данных.
- Заключение: Достижения и возможности для будущего развития.

8. Система рекомендаций на основе машинного обучения
- Цель работы: Обосновать необходимость систем рекомендаций.
- Техническое задание: Перечень функций и источников данных для обучения модели.

- Архитектура приложения: Схема архитектуры приложения и взаимодействия компонентов.
- Используемые технологии: Go, библиотеки для машинного обучения, базы данных.
- Реализация: Примеры кода и описание процесса создания системы рекомендаций.
- Тестирование: Методы тестирования точности рекомендаций.
- Заключение: Итоги работы и рекомендации по улучшению.

9. Микросервисная архитектура для e-commerce платформы
- Цель работы: Обосновать преимущества микросервисной архитектуры для e-commerce.
- Техническое задание: Описание функционала (управление товарами, заказами и пользователями).
- Архитектура приложения: Схема микросервисной архитектуры.
- Используемые технологии: Go, Docker, Kubernetes для деплоя.
- Реализация: Примеры кода и описание сервисов, их взаимодействия.
- Тестирование: Методики тестирования микросервисов и их взаимодействия.
- Заключение: Итоги работы, достижения и возможные направления для развития.

Общие рекомендации для всех тем:

- Содержание отчета: Каждый отчет должен содержать:
  - Введение
  - Цель и задачи работы
  - Описание методов и технологий
  - Реализация (с примерами кода)
  - Тестирование
  - Заключение
  - Список использованных источников