Общество с ограниченной ответственностью «Ребреин»
ИНН 7727409582, ОГРН 1197746106161
Адрес: 123056, город Москва, Большая Грузинская ул, д. 36а стр. 5а, офис 13

Утверждено
Приказом № ПР-1 от 17.06.2025 г.

Генеральный директор

_____ Фролкина Е.А.
«17» июня 2025 г.

ДОПОЛНИТЕЛЬНАЯ ПРОФЕССИОНАЛЬНАЯ ПРОГРАММА
– ПРОГРАММА ПОВЫШЕНИЯ КВАЛИФИКАЦИИ
«РАЗРАБОТКА ВЫСОКОНАГРУЖЕННЫХ СЕРВИСОВ НА GO: ПРОДВИНУТЫЙ УРОВЕНЬ»

Срок реализации: 2 месяца
Количество часов: 106 акад. ч.
Форма обучения: заочная форма
Формат обучения: с применением исключительно дистанционных технологий
Возраст обучающихся: для лиц старше 17 лет, имеющих или получающих среднее профессиональное и (или) высшее образование

Москва, 2025 г.

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Настоящая дополнительная профессиональная программа – программа повышения квалификации «Разработка высоконагруженных сервисов на Go: продвинутый уровень» (далее – Программа) разработана в соответствии с:

- Федеральным законом от 29 декабря 2012 г. № 273-ФЗ «Об образовании в Российской Федерации»;
- Приказом Министерства образования и науки РФ от 1 июля 2013 г. № 499 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным профессиональным программам»;
- Постановлением Правительства РФ от 11.10.2023 № 1678 «Об утверждении Правил применения организациями, осуществляющими образовательную деятельность, электронного обучения, дистанционных образовательных технологий при реализации образовательных программ»;
- Профессиональным стандартом 06.026 «Системный администратор информационно-коммуникационных систем», утвержден приказом Министерства труда и социальной защиты Российской Федерации от 26.10.2020 года № 60580;
- ФГОС высшего образования – бакалавриат по направлению 09.03.02 Информационные системы и технологии, утв. приказом Минобрнауки России от 19.09.2017 №926;
- Локальными нормативными актами ООО «Ребреин».

В данной программе учтены основные идеи формирования универсальных учебных действий учащихся и соблюдена преемственность с программами высшего и/или среднего профессионального образования.

Направленность программы: Программа имеет техническую направленность.

Адресат:

- Разработчики, желающие углубить знания в Go и построении высоконагруженных сервисов.
- Специалисты по тестированию, работающие с нагрузочным тестированием и мониторингом.
- Системные архитекторы, проектирующие микросервисные решения.
- DevOps-инженеры, участвующие в развертывании и сопровождении сервисов.
- Системные аналитики, работающие с проектированием и оптимизацией процессов обработки данных.

Требования к входным знаниям обучающегося:

Базовые знания Linux: команды, файловая система, работа с процессами.
Базовые знания сетевых протоколов: HTTP, TCP/UDP, WebSocket.
Базовые навыки работы с системами контроля версий (GitHub/GitLab): клонирование репозиториев, коммиты, ветки.

Актуальность реализации:

Современные сервисы требуют высокой производительности, отказоустойчивости и масштабируемости. Язык Go стал стандартом для разработки высоконагруженных систем благодаря простоте параллельного программирования, высокой скорости выполнения и удобной работе с микросервисной архитектурой. Освоение продвинутых техник работы с Go, оптимизации, межсервисного взаимодействия и мониторинга позволяет создавать устойчивые и масштабируемые приложения, востребованные в IT-индустрии.

Отличительные особенности программы:

- Фокус на продвинутой разработке на Go с реальными кейсами высоконагруженных систем.
- Практическая работа с микросервисами, контейнерами, Docker и кластеризацией.
- Интеграция с современными средствами межсервисного взаимодействия: gRPC, Kafka, GraphQL.
- Углубленная работа с оптимизацией производительности: in-memory, Redis, LRU-кеши.
- Подробное изучение логирования, метрик, healthcheck и graceful shutdown для поддержания отказоустойчивости.

Объем и срок освоения программы: 106 академ. ч. в течение 2 месяцев (8 недель)

Доступ к материалам Программы у обучающихся остаётся и после окончания периода обучения. Это позволяет повторять изученный материал в удобное время, восполнять пробелы в знаниях, а также возвращаться к практическим заданиям при решении рабочих задач. Такой формат способствует более глубокому закреплению навыков и поддерживает профессиональное развитие выпускников даже после завершения обучения.

Выдаваемый документ о квалификации: удостоверение о повышении квалификации и/или сертификат об успешном освоении программы.

Цели и задачи программы:

Сформировать у слушателей продвинутые знания и практические навыки по проектированию, разработке и сопровождению высоконагруженных сервисов на языке Go с использованием современных архитектурных подходов, инструментов оптимизации и технологий межсервисного взаимодействия.

Программа направлена на решение следующих основных задач:

Обучающие:

- дать представление о работе с базами данных (реляционными и нереляционными) в Go;
- освоить инструменты для построения серверов, роутинга и middleware;
- изучить методы взаимодействия сервисов: HTTP, gRPC, Kafka, GraphQL;
- научить оптимизации приложений с помощью кеширования, in-memory и Redis;
- показать подходы к мониторингу, логированию и работе сервисов в кластере.

Развивающие:

- развить навыки анализа производительности и поиска узких мест;
- сформировать умение проектировать архитектуру микросервисов;
- укрепить практические навыки DevOps-подходов при работе с Docker и CI/CD;
- повысить способность к самостоятельному решению комплексных инженерных задач.

Воспитательные:

- способствовать развитию ответственного отношения к качеству и надежности сервисов;
- формировать культуру командной разработки и следования best practices;
- прививать ценность документирования, тестирования и прозрачности процессов разработки.

Планируемые результаты:

Знания:
- принципы работы реляционных (Postgres) и нереляционных (MongoDB) БД, подходы к миграциям и ORM;
- архитектуру серверных приложений на Go, особенности роутинга и middleware;
- возможности и отличия HTTP, gRPC, Kafka и GraphQL во взаимодействии сервисов;
- методы оптимизации приложений, алгоритмы кеширования и использование Redis;
- подходы к построению микросервисной архитектуры и управлению конфигурацией сервисов;
- практики логирования, сбора метрик и мониторинга в кластере;
- основы работы с инструментами DevOps (Docker, CI/CD, Graylog).

Умения:
- конфигурировать подключение к БД и реализовывать доступ к данным на Go;
- использовать современные роутеры (Gorilla, Chi, FastHTTP) и настраивать middleware;
- реализовывать межсервисное взаимодействие через HTTP, gRPC, Kafka;
- разрабатывать и отлаживать микросервисы, подготавливать их Docker-образы;
- проектировать системы с учётом масштабируемости и высокой нагрузки;
- интегрировать сервисы с системами логирования и мониторинга;
- применять методы оптимизации кода и сервисов для повышения производительности.

Навыки:
- разработки и сопровождения высоконагруженных сервисов на Go;
- применения инструментов управления миграциями и ORM-библиотек;
- проектирования и реализации архитектуры микросервисов;
- работы с in-memory хранилищами и кешированием;
- настройки отказоустойчивости сервисов (graceful shutdown, healthcheck);
- комплексной диагностики, анализа логов и метрик в продакшн-среде;
- командной работы с использованием Git и CI/CD-инструментов.

Перечень профессиональных компетенций, на получение которых направлено обучение:
На основе профстандарта 06.026 «Системный администратор информационно-коммуникационных систем»:
- B/02.5 Обеспечение работы технических и программных средств информационно-коммуникационных систем;
- C/05.6 Выполнение обновления программного обеспечения сетевых устройств информационно-коммуникационных систем;
- C/08.6 Планирование и проведение работ по распределению нагрузки между имеющимися ресурсами, снятию нагрузки на сетевые устройства информационно-коммуникационных систем перед проведением регламентных работ, восстановлению штатной схемы работы в случае сбоев.

Таким образом, в результате освоения программы у обучающихся формируются следующие профессиональные компетенции:
- ОПК-5. Способен инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем;
- ОПК-6. Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий;

- ОПК-7. Способен осуществлять выбор платформ и инструментальных программно-аппаратных средств для реализации информационных систем.

Организационно-педагогические условия реализации программы дополнительного профессионального образования

Язык реализации образовательной программы: обучение проводится на русском языке.

Форма обучения: заочная форма.

Особенности реализации программы: программа реализуется с использованием электронного обучения и исключительно дистанционных образовательных технологий.

Условия набора: на обучение принимаются все желающие лица, оплатившие обучение и заключившие договор об образовании. Обучение проходит в индивидуальном формате без формирования учебных групп. Обучающийся самостоятельно определяет время освоения Программы.

Формы проведения занятий:
- занятия в текстовом формате;
- практическая работа;
- самостоятельная работа с литературой;
- индивидуальные вопросы.

Материально-техническое оснащение

Материальное обеспечение программы

Занятия проводятся в системе дистанционного обучения «Rebrain». Каждый обучающийся и педагог оснащены доступом к системе дистанционного обучения: https://rebrainme.com/.

У педагога дополнительного профессионального образования имеется необходимое оборудование средства для реализации программы: ноутбук с подключением к интернету, программное обеспечение.

Методическое обеспечение программы

Программа обеспечена:
- учебно-методическими материалами (текстовые занятия, полезными материалами);
- практическими заданиями.

Кадровое обеспечение:

К реализации программы в качестве педагогов дополнительного образования допускаются лица:

1) отвечающее одному из требований:

а) имеющее высшее образование или среднее профессиональное образование в рамках укрупненных групп специальностей и направлений подготовки высшего образования и специальностей среднего профессионального образования «Образование и педагогические науки»;

б) имеющее высшее образование либо среднее профессиональное образование в рамках иных укрупненных групп специальностей и направлений подготовки высшего образования и специальностей среднего профессионального образования при условии его соответствия дополнительной общеобразовательной общеразвивающей программе,

реализуемой ООО «Ребреин», и получение при необходимости дополнительного профессионального образования педагогической направленности;

в) успешно прошедшее промежуточной аттестации не менее чем за два года обучения по образовательным программам высшего образования по специальностям и направлениям подготовки, соответствующей направленности дополнительной общеобразовательной общеразвивающей программе;

2) не имеющее ограничений на занятие педагогической деятельностью, установленных законодательством Российской Федерации;

3) прошедшее обязательный предварительный (при поступлении на работу) и периодические медицинские осмотры (обследования), а также внеочередные медицинские осмотры (обследования) в порядке, установленном законодательством Российской Федерации.

Реализация Программы также возможна лицами, привлекаемыми на условиях гражданско-правового договора в соответствии с действующим законодательством РФ.

2. УЧЕБНЫЙ ПЛАН

№ п/п	Наименование модуля	Количество часов			Формы контроля / аттестация
		Всего	Теория	Практика	
1	Модуль 1. Онбординг	2	1	1	Входное тестирование
2	Модуль 2. Работа с БД	8	3	5	Практическое задание
3	Модуль 3. Сервер на Go (обработка запросов, context, middleware)	18	5	13	Практическое задание
4	Модуль 4. Низкоуровневость + продвинутая сборка	16	5	11	Практическое задание
5	Модуль 5. Микросервисная архитектура	16	4	12	Практическое задание
6	Модуль 6. Межсервисное взаимодействие	17	3	14	Практическое задание
7	Модуль 7. Оптимизация	12	4	8	Практическое задание
8	Модуль 8. Работа сервиса в кластере	9	4	5	Практическое задание
9	Итоговая аттестация	8		8	Итоговое практическое задание

3. КАЛЕНДАРНЫЙ УЧЕБНЫЙ ГРАФИК

№ п/п	Наименование модуля	1 неделя	2 неделя	3 неделя	4 неделя	5 неделя	6 неделя	7 неделя	8 неделя
1	Модуль 1. Онбординг	2							
2	Модуль 2. Работа с БД	8							
3	Модуль 3. Сервер на Go (обработка запросов, context, middleware)	3	15						
4	Модуль 4. Низкоуровневость + продвинутая сборка			14	2				
5	Модуль 5. Микросервисная архитектура				13	3			
6	Модуль 6. Межсервисное взаимодействие					10	7		
7	Модуль 7. Оптимизация						6	6	
8	Модуль 8. Работа сервиса в кластере							7	2
9	Итоговая аттестация								8 | А

4. РАБОЧАЯ ПРОГРАММА

Модуль 1. Онбординг
Теория 1 академ. ч. Практика 1 академ. ч.
Модуль состоит из следующих тем:

Тема 1: Онбординг

В модуле обучающемуся предоставляется вводный конспект, содержащий общую информацию о программе, структуре курса, форматах взаимодействия с материалами и ожидаемых результатах обучения.

Предусмотрено прохождение входного тестирования, включающего 7 вопросов, направленных на закрепление информации из онбординга. В рамках темы обучающийся выполняет задание по целеполаганию: формулирует свою цель прохождения программы, указывает желаемые навыки по окончании обучения, а также оценивает текущий уровень своих знаний по DevOps, выбрав один из предложенных вариантов.

Модуль 2. Работа с БД
Теория 3 академ. ч. Практика 5 академ. ч.
Модуль состоит из следующих тем:

Тема 1: Конфигурирование подключения к БД (Postgres)

Содержание: Поднимаем Postgres в docker-compose. Подключение к Postgres из Go. Практическое задание.

Тема 2: Работа с БД (Postgres)
Содержание: выполнение CRUD-запросов. Особенности при работе с БД в Go. Практическое задание.

Тема 3: Миграции (Goose)
Содержание: Миграции. Библиотека Goose. Миграции в sql-файле. Практическое задание.

Тема 4: Работа с базой на примере использования GORM
Содержание: ORM и GORM. Конфигурация. Практическое задание.

Тема 5: Нереляционная база данных MongoDB.
Содержание: MongoDB. Когда использовать MongoDB. BSON. Библиотека MongoDB Go Driver. Практическое задание.

Модуль 3. Сервер на Go (обработка запросов context middleware)
Теория 5 академ. ч. Практика 13 академ. ч.
Модуль состоит из следующих тем:

Тема 1: Поднимаем сервер, роутинг, первый handler
Содержание: как поднять http-сервер на Go. Практическое задание.

Тема 2: Работа с параметрами
Содержание: URL Query параметры. FormData. Http Body. Метаданные. Basic auth. Response. Практическое задание.

Тема 3: Роутер Gorilla
Содержание: Установка. Переменные. Группировка запросов. Ограничения обработчиков. Практическое задание.

Тема 4: Роутер Chi
Содержание: Установка. Группировка маршрутов. Работа с query params. Middlewares. Практическое задание.

Тема 5: Middleware
Содержание: Создание middleware-обработчика. Роутеры. Third-Party Middleware. Практическое задание.

Тема 6: Контекст запроса
Содержание: контекстом http-запросов. Практическое задание.

Тема 7: FastHTTP
Содержание: сервер для Go — Fasthttp. Пример применения быстрого сервера. Практическое задание.

Тема 8: WebSockets
Содержание: WebSocket-сервер и работа с ним средствами Go. Реализация системы мониторинга состояния сервера. Практическое задание.

Модуль 4. Низкоуровневость + продвинутая сборка

Теория 5 академ. ч. Практика 11 академ. ч.
Модуль состоит из следующих тем:

Тема 1: Продвинутая работа с модулями
Содержание: использование нескольких версий. Замена зависимостей. Использование GoProxy. Практическое задание.

Тема 2: Сборка с использованием Idflags
Содержание: Оптимизация размера бинарника. Передача флагов внешнему компоновщику. Подмена переменных в момент сборки. Практическое задание.

Тема 3: Сборка для разных ОС
Содержание: Создание программ под разные ОС. Немного о CGO. Теги сборки под разные ОС. Практическое задание.

Тема 4: Пакет unsafe
Содержание: Подводные камни unsafe. Функции unsafe. unsafe.Alignof() и unsafe.Offsetof(). Практическое задание.

Тема 5: Cgo
Содержание: GCC и gcc. Фрагменты C в Go. Библиотека на C в Go. Программы на C в Go. Недостатки cgo. Практическое задание.

Модуль 5. Микросервисная архитектура
Теория 4 академ. ч. Практика 12 академ. ч.
Модуль состоит из следующих тем:

Тема 1: Docker-образ для сервиса
Содержание: Образы Docker. Dockerfile. Собираем образ. Запуск контейнера из образа. Практическое задание.

Тема 2: Продвинутая сборка образа
Содержание: способы запуска контейнеров. Docker Compose. Многоэтапные сборки образов. Практическое задание.

Тема 3: Конфигурация приложения
Содержание: Что такое конфигурация. Конфигурация внутри файла. Конфигурация через переменные окружения. Конфигурация в удалённом хранилище «ключ-значение». Практическое задание.

Модуль 6. Межсервисное взаимодействие
Теория 3 академ. ч. Практика 14 академ. ч.
Модуль состоит из следующих тем:

Тема 1: Синхронное HTTP-взаимодействие и Swagger
Содержание: Синхронное HTTP-взаимодействие: преимущества и недостатки. Swagger: преимущества. Библиотека go-swagger. Практическое задание.

Тема 2: Фреймворк gRPC
Содержание: Практика с gRPC. Установка protobuf и плагинов. Запускаем клиент. Практическое задание.

Тема 3: gRPC Streams, interceptors
Содержание: Серверный стрим. Запустим сервер и клиент. Перехватчики. Практическое задание.

Тема 4: Асинхронное взаимодействие и Kafka
Содержание: Асинхронный вызов сервиса. Брокер сообщений. Сущности Kafka. Поднимаем Kafka. Практическое задание.

Тема 5: Работа с GraphQL в Go
Содержание: Особенности GraphQL. GraphQL-сервер. GraphQL-клиент. Query. Практическое задание.

Модуль 7. Оптимизация
Теория 4 академ. ч. Практика 8 академ. ч.
Модуль состоит из следующих тем:

Тема 1: in-memory хранение
Содержание: Map. Когда использовать in-memory хранение. Когда не использовать in-memory хранение. Практическое задание.

Тема 2: Redis
Содержание: Принципы Redis. Redis в Docker Compose. Практическое задание.

Тема 3: Алгоритм кеширования LRU
Содержание: Давно неиспользуемые данные. Практическое задание.

Модуль 8. Работа сервиса в кластере
Теория 4 академ. ч. Практика 5 академ. ч.
Модуль состоит из следующих тем:

Тема 1: Формат логов и уровни логирования
Содержание: какие форматы логирования есть, преимущества и недостатки. Уровни логирования. Практическое задание.

Тема 2: Логи в Graylog
Содержание: Graylog. Logrus в Graylog. Практическое задание.

Тема 3: Сквозное логирование
Содержание: Пример сквозного логирования. Практическое задание.

Тема 4: Метрики приложения
Содержание: Prometheus. Экспортёры. Работа с базовыми метриками. Дашборды. Практическое задание.

Тема 5: Graceful shutdown
Содержание: Как выглядит контролируемое завершение. Практическое задание.

Тема 6: Healthcheck
Содержание: Оркестратор Kubernetes. Механизм healthcheck. Виды healthcheck. Как работает liveness probe. Как работает readiness probe. Практическое задание.

Каждая тема модулей включает текстовое занятие с теоретическим материалом и пошаговыми инструкциями, после изучения которого предлагается практическое задание. Практические задания рассчитаны на 1-2 академических часа. Выполнение заданий предполагает отправку решения на проверку через личный кабинет обучающегося. Критерии оценки прописаны в описании к каждому заданию. В случае корректного выполнения выставляется зачёт. Если работа содержит ошибки, задание возвращается на доработку. При повторной неудачной попытке (после двух доработок) обучающийся получает «незачёт».

Итоговая аттестация.

Модуль посвящён выполнению финального практического задания без предварительного теоретического блока.

5. МЕТОДИЧЕСКИЕ И ОЦЕНОЧНЫЕ МАТЕРИАЛЫ

Программа обеспечена системой дистанционного обучения https://rebrainme.com/.

Педагогические технологии:

- технология дифференцированного обучения;
- технология разноуровневого обучения;
- технология развивающего обучения;
- технология проблемного обучения;
- технология дистанционного обучения.

Методы обучения:

- словесный, наглядный практический;
- объяснительно – иллюстративный;
- частично-поисковый, исследовательский проблемный;
- игровой, дискуссионный.

Электронно-библиотечные ресурсы и системы, информационно-справочные системы:

1. Научная электронная библиотека eLIBRARY.RU.
2. Собственные учебные материалы: https://rebrainme.com/golang-advanced/
3. Go database/sql tutorial [Электронный ресурс]: https://go-database-sql.org/
4. Официальная документация GORM [Электронный ресурс]: https://gorm.io/docs/
5. Официальная документация MongoDB [Электронный ресурс]: https://docs.mongodb.com/guides/
6. The WebSocket API [Электронный ресурс]: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API

6. ОЦЕНКА КАЧЕСТВА ОСВОЕНИЯ ПРОГРАММЫ

Оценочные материалы:

Для отслеживания результатов освоения программы среди слушателей проводится текущий контроль, промежуточный контроль и итоговое оценивание.

Текущий контроль

Осуществление текущего контроля проводится после занятий в виде написания практических заданий или тестирований. Тематика и условия выполнения практических заданий расписаны в личном кабинете обучающегося в СДО. Педагог проверяет решение и принимает решение о принятии решения (зачет), о необходимости доработать решение или

о незачете. Если промежуточный контроль представлен в виде тестирования, подсчет верных ответов и выставление оценки «зачёт» и «незачёт» происходят в автоматическим решиме в СДО.

Модуль 2. Работа с БД
Тема 3: Миграции (Goose)
Задание

1. Форкните репозиторий module08_03 с кодом данного задания в группу с вашими репозиториями golang_users_repos/<your_gitlab_id>.
2. Создайте у себя в проекте module08_03 из ветки master ветку 03_task.
3. Создайте в sql-файле миграцию, которая добавит в таблицу users колонку last_name varchar(100).
4. Создайте в go-файле в пакете migrations миграцию, которая разделит имя и фамилию по разным колонкам name и last_name. (После прошлого задания имя и фамилия у нас хранятся в одном столбце name).
5. При обновлении записей в базе необходимо обновлять время в колонке updated_at.
6. Миграции должны примениться автоматически при запуске приложения.
7. В ответе пришлите ссылку на merge request в ветку master своего проекта ветки 03_task.

Модуль 3. Сервер на Go (обработка запросов, context, middleware)
Тема 5: Middleware
Задание
В этом задании мы предлагаем вам на основе middleware построить базовую аутентификацию.

Условия

- Аутентификация должна быть построена на базе Basic auth.
- Для проверок используйте креды администратора из module09/internal/constants.
- Middleware должна быть активирована не на все обработчики, а только на create и delete.
- Если аутентификация не прошла, то обработчик должен отдавать статус 401 StatusUnauthorized и не пускать пользователя дальше.

Порядок действий

1. В вашем проекте module09 создайте новую ветку 05_task.
2. В пакете module09/internal/routers/chi или module09/internal/routers/gorilla создайте middleware и подключите к роутеру.
3. Проверьте работоспособность.
4. В ответе пришлите ссылку на merge request в ветку master своего проекта ветки 05_task.

Модуль 4. Низкоуровневость + продвинутая сборка
Тема 3: Сборка для разных ОС
Задание

1. Форкните репозиторий module10_03 с кодом данного задания в группу с вашими репозиториями — golang_users_repos/<your_gitlab_id>.
2. В вашем проекте module10_03 создайте новую ветку module10_03.
3. В данном репозитории содержится простейший код в виде "Hello + text". Сейчас этот код всегда выводит текст "Hello MacOS". Ваша задача — сделать так, чтобы код выводил "Hello" + любая из 3 операционных систем: Linux, MacOS, Windows (в зависимости от собираемой версии).

4. В ответ на задание пришлите:
- ссылку на merge request в ветку master вашего проекта ветки module10_03;
- три команды, компилирующие код под три операционных системы.

Модуль 5. Микросервисная архитектура
Тема 1: Docker-образ для сервиса
Задание

1. Сделайте форк проекта module11 в группу golang_users_repos/<your_gitlab_id>.
2. Создайте в своём проекте module11 ветку module11_01.
3. Напишите и закомитьте Dockerfile.
4. Запустите веб-сервис в Dicker-контейнере, чтобы он был доступен на порту 8080 (менять номер порта в коде не надо, всё делается через Docker).
5. Откройте браузер и удостоверьтесь, что веб-сервис доступен по адресу http://localhost:8080/hello.
6. В качестве ответа присылайте:

- ссылку на merge request в ветку master вашего проекта ветки module11_01;
- команду из консоли, с помощью которой вы собирали образ;
- команду из консоли, с помощью которой вы запускали контейнер из образа.

Модуль 6. Межсервисное взаимодействие
Тема 5: Работа с GraphQL в Go
Сегодня вам предстоит выполнить две задачи:

1. Реализовать GraphQL-сервер, соответствующий следующей схеме: [...]
2. Реализовать клиент, который отправит несколько запросов на GraphQL-сервер:

- создание двух пользователей,
- создание постов для этих пользователей,
- получение всех постов,
- получение постов по каждому отдельному пользователю.

В конце работы программы клиента вы должны сформировать map[int]Post с ключами UserId, и значениями в виде постов пользователей.

Второй частью ответа должен быть массив всех постов сервиса. Массив всех постов и значения в мапе должны совпадать. [...]

Модуль 7. Оптимизация
Тема 2: Redis
Задание
Ваша задача — переписать нативный кеш и использовать вместо него Redis.
Возьмите за основу код из прошлого задания. Поднимите Redis рядом с помощью Docker Compose.

1. В вашем проекте module13 сделайте новую ветку 02_task.
2. Переработайте использование нативного кеша, используйте Redis вместо него.
3. Для запуска Redis вместе с вашим сервисом используйте Docker Compose.
4. В ответе пришлите ссылку на merge request в ветку master своего проекта ветки 02_task.

Модуль 8. Работа сервиса в кластере
Тема 4: Метрики приложения
Задание

1. В репозитории проекта, который вы делали в модуле [GO-05]: Межсервисное взаимодействие, из ветки task_14_3 (или из master, если ветка была смёрджена) создайте новую ветку task_14_4.
2. Добавьте базовые метрики к этому сервису.
3. Добавьте метрики, которые считают количество вызовов ручки Send.
4. Создайте дашборд в вашей Grafana, используя https://grafana.com/grafana/dashboards/6671.
5. Сделайте скриншот вашей Grafana, добавьте в репозиторий.
6. В качестве ответа пришлите ссылку на merge request в ветку master вашего проекта ветки task_14_4.

Итоговое оценивание
В конце программы обучающиеся сдают итоговую аттестацию.

Практическое задание
До получения сертификата об окончании практикума всего один шаг — подготовить и презентовать финальный проект.

Презентация вашего проекта будет проходить на видеозвонке с одним из кураторов практикума.

О процессе подготовки и презентации
Встреча с куратором занимает два часа: один час — собеседование, один час — обсуждение задания. Порядок действий состоит из двух шагов:

1. В ответ на это сообщение пришлите три таймслота для презентации, которые соответствуют следующим условиям:
- не ранее 5 дней после текущей даты,
- не позже 14 дней после текущей даты,
- временной промежуток должен находиться в отрезке 18:00 до 20:00 по МСК с понедельника по пятницу.
2. В ответ вы получите задание, которое нужно реализовать, и письмо со ссылкой на встречу.

Об оценивании презентации
Презентация проекта состоит из трёх этапов:
1. Презентация вашего проекта оценивается от 0 до 5 баллов.
2. Три вопроса от куратора по вашему проекту, каждый оценивается от 0 до 5 баллов.
3. Три вопроса от куратора по всему практикуму, каждый оценивается от 0 до 10 баллов.

Максимальный балл, который можно получить — 50.

Результаты текущего контроля, промежуточной аттестации и итогового оценивания отображаются в личном кабинете слушателя в системе дистанционного обучения https://rebrainme.com/.

По результатам сдачи текущего контроля, промежуточного контроля и итогового оценивания педагог даёт обратную связь слушателям, отмечает их сильные стороны и обращает внимание на зоны для развития. При необходимости педагог может повторить пройденные темы со слушателями, если установлен факт плохого закрепления и усвоения темы у слушателей.