Аринов Данияр Рустемович

Мужчина, 21 год, родился 17 января 2004

+7 (910) 0053539 — Telegram: @darinovyo
daniararinov995@gmail.com — предпочитаемый способ связи
Другой сайт: https://github.com/vegitobluefan

Проживает: Москва, м. Войковская
Гражданство: Россия, есть разрешение на работу: Россия
Готов к переезду, готов к командировкам

Желаемая должность и зарплата

Golang-разработчик

Специализации:
— Программист, разработчик
Занятость: полная занятость, частичная занятость, проектная работа, стажировка
График работы: полный день, сменный график, гибкий график, удаленная работа
Желательное время в пути до работы: не имеет значения

Опыт работы — 3 года 4 месяца

Май 2022 — настоящее время
3 года 2 месяца

Фриланс, Проектная деятельность

Москва, github.com/vegitobluefan

Python/Golang разработчик

- Список используемых технологий:
Python: фреймворк Django, DRF, PostgreSQL, SQLite, Docker,
RestAPI, nginx, Яндекс облако(Ubuntu), git.

Golang: net/http, gin, goroutines
Базы данных: PostgreSQL, Redis, MongoDB
Тестирование: Go testing package
DevOps: Docker, Git, CI/CD (GitHub Actions / GitLab CI), Nginx, базово Kubernetes
Инфраструктура: Linux, systemd, Prometheus, Grafana
API: REST, gRPC, WebSocket
Прочее: микросервисы, асинхронщина (asyncio), написание CLI-утилит, парсеры

- Список инструментов: VScode, Docker desktop, Postman, Git bash.
Ознакомиться с предложенным ниже проектами: https://github.com/vegitobluefan/

1. Полностью реализовал Rest API сервис, в котором пользователи могут делиться своими рецептами. Настроил регистрацию и авторизацию с помощью Djoser. Реализовал возможность подписок, добавления рецепта в избранное, корзину. Возможность скачивать корзину. Также был выполнен деплой на удалённый сервер с использованием Docker и nginx. Настроил github actions, выполняющий проверку кода на соответствие pep8, а также отправляющего сообщение в Telegram.
Ссылка: https://github.com/vegitobluefan/Foodgram_project.git

2. Разработал API для генерации изображений с помощью модели Stable Diffusion. API обрабатывает запросы, ставит задачи в очередь через Celery и выполняет их асинхронно. Реализовано логирование, обработка ошибок и масштабируемая архитектура.
Стек технологий:
- Backend: Python, FastAPI

- Очередь задач: Celery, Redis
- Генерация изображений: Stable Diffusion (AUTOMATIC1111 API)
- Обработка данных: Pydantic, Base64
- Инфраструктура: Docker, Uvicorn
- Аппаратное ускорение: NVIDIA CUDA, Tensor Cores
Ссылка: https://github.com/vegitobluefan/StableDiffusionML

3. Реализовал API для реферальной системы (Django, DRF). Сервис позволяет пользователям регистрироваться, использовать уникальные invite-коды и отслеживать рефералы.
Ссылка: https://github.com/vegitobluefan/ReferralSystem_API.git

4. Был реализован REST API для сервиса, учитывающего рейтинг объектов (книг, фильмов и т.д.). Работал в команде из 3-х человек. Выполнял роль тимлида: распределял задачи, помогал участникам решать вопросы, организовал коммуникацию в команде. В итоге проект был выполнен в срок, заказчик остался доволен.
Ссылка: https://github.com/vegitobluefan/API_yamdb.git

5. Реализовал backend социальной сети, где пользователи делятся своими питомцами. Аутентификация, комментарии, подписки, разбиение по местоположениям и категориям. Возможность переключаться между светлой и тёмной темой.
Ссылка: https://github.com/vegitobluefan/Doggygram.git

6. TaskFlow — микросервис для отложенного выполнения задач
Ссылка: https://github.com/vegitobluefan/TaskFlow.git
- Описание:
Небольшой, но расширяемый HTTP-сервис, реализующий отложенное выполнение задач (job queue imitation) с REST API. Пользователь может создать задачу и получить её статус через уникальный ID. Задачи выполняются асинхронно с симуляцией долгой обработки (3–5 минут), как пример I/O-bound процесса. Реализована внутренняя очередь с мьютекс-защитой и встроенным логированием событий.
- Функции:
1. Создание задач (POST /task)
2 .Получение статуса задачи (GET /task/{id})
3. Асинхронное выполнение с внутренней очередью
4. Простая и масштабируемая архитектура
5. Логирование каждого этапа задачи

7. Orders Management System — микросервисный проект на Go, реализующий архитектуру распределённой системы для управления заказами, кухней, складом и платежами с использованием gRPC.
Система разбита на независимые сервисы и использует взаимодействие через gRPC-интерфейсы. Каждый модуль обрабатывает свою часть логики.
Ссылка: https://github.com/vegitobluefan/OrdersManagementSystem.git
- Технологический стек
- Модули:
1. orders: приём и маршрутизация заказов
2. kitchen: обработка заказов на кухне
3. stock: управление запасами и контроль доступности ингредиентов

8. LibraryAPI — RESTful сервис управления книгами.
Разработал RESTful API для управления библиотечным каталогом. Реализованы основные CRUD-операции с поддержкой фильтрации по автору и симуляцией выдачи/возврата книг. Проект структурирован по слоям (модели, маршруты, утилиты) и легко масштабируется.
Ссылка: https://github.com/vegitobluefan/LibraryAPI.git
- Функциональность:
1. Получение всех книг с фильтрацией по автору
2. Получение книги по ID

3. Добавление новой книги
4. Выдача книги (уменьшение количества)
5. Возврат книги (увеличение количества)
6. Обработка ошибок и валидация ввода
4. payments: обработка и проверка платежей

Март 2022 — Июнь 2025
3 года 4 месяца

T1

t1.ru

Информационные технологии, системная интеграция, интернет

- Разработка программного обеспечения
- Системная интеграция, автоматизации технологических и бизнес-процессов предприятия, ИТ-консалтинг

Middle Backend разработчик

Работал в стабильной кросс-функциональной команде:
3 backend-разработчика,
2 бизнес-аналитика, тимлид.
Активно взаимодействовали с отделами frontend и DS/ML разработки.

!Первый проект : Разработка корпоративного хранилища данных (КХД) и модуля отчётности с нуля.
Участвовал в создании КХД и системы формирования отчетов. Основной фокус — оптимизация процессов подготовки ведомостей по результатам работы подразделений.

Достижения:
1. Спроектировал и реализовал микросервисы координации процедур согласования и управления файлами, сократив срок утверждения ключевых договоров в 2,5 раза.
2. Внедрил асинхронную нотификацию об изменениях в БД через Kafka, что ускорило реакцию менеджеров на внештатные ситуации.
3. Разработал backend для UI поиска файлов по дате и содержанию; интегрировался с ML-моделями, которые извлекали теги из документов.
4. Участвовал в покрытии кода тестами, написал техническую документацию, занимался деплоем в тестовом контуре (CI/CD: Docker, k8s, Jenkins).

Технологии: Go, Python, gRPC, REST API, PostgreSQL, ClickHouse, Amazon S3, Docker, Kubernetes, Jenkins, Kafka, Prometheus, Grafana, Kibana

! Второй проект: Миграция менеджерского Python-монолита на микросервисную архитектуру
Работал над выносом функциональности в отдельные сервисы. Особое внимание уделялся чату с документ-редактором, который заинтересовал одного из заказчиков.

Достижения:
1. Внедрил хранение документов и чатов с использованием MongoDB и Redis. Документировал маршруты API через OpenAPI (Swagger), упростив интеграцию с frontend.
2. Самостоятельно выделил модуль личного кабинета в несколько микросервисов: аутентификация, профиль пользователя, сервис аналитики, справочники.
3. Перевёл взаимодействие между сервисами с REST на gRPC, что дало прирост производительности и уменьшение сетевых издержек.

Технологии: Go, Python, REST API, gRPC, MongoDB, Redis, RabbitMQ, Swagger

!Третий проект: Разработка высоконагруженного API для агрегации финансовых данных.
Компании потребовалось создать унифицированный API-шлюз для агрегации данных из различных внутренних и внешних финансовых систем (банковские транзакции, биллинг, CRM). Задача — обеспечить высокую доступность, низкие задержки и безопасность при

обработке запросов.

 Достижения:
1. Разработал ядро API на Go с кешированием в Redis, что сократило время ответа с 800 мс до 120 мс.
2. Реализовал балансировку нагрузки через Nginx + Kubernetes, что позволило выдерживать до 15 000 RPS.
3. Интегрировал систему мониторинга (Prometheus + Grafana) и алертинга, что снизило время реакции на инциденты с 30 минут до 3 минут.
4. Оптимизировал запросы к БД (PostgreSQL + ClickHouse), внедрив пагинацию и материализованные представления, что уменьшило нагрузку на СУБД на 40%.
5. Автоматизировал CI/CD (GitLab CI + ArgoCD), сократив время деплоя с 20 минут до 3 минут.

Технологии:
Go, Python, gRPC, REST API, PostgreSQL, ClickHouse, Redis, Nginx, Kubernetes, Prometheus, Grafana, GitLab CI, ArgoCD

Образование

Высшее

2025 **Московский авиационный институт (национальный исследовательский университет), Москва**
Информатика и вычислительная техника (бакалавр техники и технологии)

Повышение квалификации, курсы

2025 **Быстрый старт в FastAPI Python**
Stepik, Backend-development

2024 **Яндекс Практикум**
Образовательные технологии Яндекса, Python-разработчик

Навыки

Знание языков
Русский — Родной
Английский — C1 — Продвинутый

Навыки
Python GitHub REST API Pytest ООП CI/CD Docker Nginx Git HTML Unittest Kanban FastAPI Pydantic Redis Celery Django PostgreSQL Django ORM Linux SQL JavaScript Java Golang SQLAlchemy gRPC Numpy Kubernetes RabbitMQ Ruby On Rails

Дополнительная информация

Обо мне
Английский язык: C1 (продвинутый уровень)
Люблю спорт (тяжёлая атлетика), увлекаюсь разработкой CLI-утилит и оптимизацией кода.
Активно изучаю Kubernetes и продвинутую работу с gRPC.