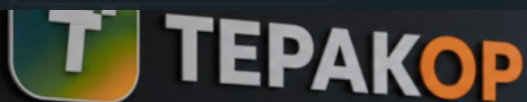


ТЕРАКОР • ПРОФЕССИЯ В IT



ОБУЧЕНИЕ. РАЗРАБОТКА. РЕЗУЛЬТАТ.



ПОДРОБНАЯ ПРОГРАММА ОБУЧЕНИЯ

Профессия Python-разработчик

От Python Core до FastAPI, PostgreSQL, Kafka и микросервисов

Python • FastAPI • Django • PostgreSQL • Kafka • Docker

Практика на проектах для портфолио

После каждого модуля - законченная работа и code review.

5 месяцев

полный маршрут

450 ак. ч.

общая нагрузка

150 ак. ч.

живые занятия

КАФКА - 20 АК. Ч.

СОБЕСЕДОВАНИЕ - 4 АК. Ч.

Преподаватель

Сергей Валерьевич Герасименко

Основатель ТЕРАКОР • практикующий full-stack разработчик

89 000 ₽

полная программа

terakor.ru

Лицензированный учебный центр • обучение онлайн

1. О профессии

Профессия Python-разработчик - цельная траектория от основ языка и SQL до backend-приложений на FastAPI, работы с Django, PostgreSQL, Docker, микросервисами и Apache Kafka. Технологии объединяются в работающие проекты, которые студент сможет представить в портфолио.

Обучение: живые занятия с преподавателем, самостоятельная практика, разбор кода и проекты после каждого модуля. Итоговая работа Commerce Hub выполняется на FastAPI; Django изучается в отдельной практике.

2. Результаты обучения

- Писать понятный Python-код и применять ООП, SOLID и подходящие паттерны GoF.
- Проектировать данные и писать SQL-запросы для PostgreSQL.
- Создавать REST API на FastAPI; понимать устройство Django ORM и Admin.
- Покрывать код unit- и интеграционными тестами, применять asyncio.
- Запускать приложение и зависимости в Docker Compose.
- Строить взаимодействие сервисов через JSON-RPC и события Kafka.
- Проектировать топики и контракты событий, обрабатывать повторы и ошибки.
- Защищать архитектуру собственного проекта и проходить техническое интервью.

3. Нагрузка и формат

Модуль	Живые занятия	Самостоятельная работа
1. Python Core, Git, SQL и PostgreSQL	38 ак. ч.	76 ак. ч.
2. ООП, SOLID, GoF, архитектура и REST	36 ак. ч.	72 ак. ч.
3. FastAPI, Django, async, тесты и Docker	44 ак. ч.	88 ак. ч.
4. Микросервисы, JSON-RPC и паттерны	8 ак. ч.	16 ак. ч.
5. Apache Kafka для Python backend	20 ак. ч.	40 ак. ч.
6. Подготовка к собеседованию	4 ак. ч.	8 ак. ч.
ИТОГО	150 ак. ч.	300 ак. ч.

Общий объем - 450 ак. ч. Ориентировочный срок полного маршрута - 5 месяцев. Самостоятельная часть включает домашние задания, code review, доработку проектов и подготовку к защите.

4. Подробная программа

Модуль 1. Python Core, Git, SQL и PostgreSQL - 38 ак. ч.

Результат: студент пишет программы на Python, ведет проект в Git, проектирует реляционную схему и извлекает данные из PostgreSQL.

- Среда разработки: Python, IDE, виртуальные окружения, зависимости, структура проекта.
- Типы данных, условия, циклы, строки, коллекции, функции, модули и пакеты.
- Файлы, JSON, CSV, регулярные выражения, исключения, отладка и логирование.
- Git: commit, ветки, merge, удаленный репозиторий и базовая командная работа.
- PostgreSQL: таблицы, ключи, ограничения, связи и нормализация.
- SQL: SELECT, JOIN, агрегаты, подзапросы, CTE, DML и транзакции.
- Индексы и диагностика запросов; подключение Python к БД и параметризованные запросы.
- PL/pgSQL и триггеры - краткий прикладной обзор.

ПРОЕКТ ПОСЛЕ МОДУЛЯ: Library Tracker

Консольная система учета книг и выдач с хранением в PostgreSQL.

Для портфолио: репозиторий, схема БД, примеры запросов, обработка ошибок, инструкция запуска и тесты ключевой логики.

Модуль 2. ООП, SOLID, паттерны GoF, архитектура и REST - 36 ак. ч.

Результат: студент проектирует предметную модель, выделяет слои приложения и объясняет выбор архитектурных решений.

- Классы, композиция, наследование, абстрактные классы, Protocol и полиморфизм.
- dataclass, аннотации типов, свойства, итераторы, генераторы и коллекции.
- Принципы SOLID: поиск нарушений, рефакторинг и проверка улучшений тестами.
- Паттерны GoF: Strategy, Factory Method, Adapter, Observer и Decorator на прикладных задачах.
- Сложность операций и выбор структур данных без избыточной теории.
- Слои API, бизнес-логики и доступа к данным; обработка ошибок и валидация.
- HTTP, REST, endpoint, статусы ответов, пагинация, контракты API и основы безопасности.

ПРОЕКТ ПОСЛЕ МОДУЛЯ: Delivery Rules

Ядро сервиса доставки с разными правилами обработки заказов и внешними интеграциями.

Для портфолио: предметная модель, SOLID, Strategy/Factory Method/Adapter, тесты и описание рефакторинга.

Модуль 3. FastAPI, Django, async, тесты и Docker - 44 ак. ч.

Результат: студент создает, тестирует и запускает backend-приложение; знает два подхода к веб-разработке на Python.

- FastAPI: маршруты, параметры, Pydantic, зависимости, обработка ошибок и OpenAPI.
- SQLAlchemy, связи, миграции, аутентификация и права доступа.
- Django: приложения, модели, ORM, миграции, Admin, формы и тестирование.
- Различия FastAPI и Django при выборе инструмента под задачу.
- async/await, event loop, таймауты и ошибки вызова внешних сервисов.
- pytest: fixtures, параметризация, mock, unit-, интеграционные и API-тесты.
- Dockerfile, Docker Compose, PostgreSQL, переменные окружения, логи, health checks.
- LLM API - дополнительный сценарий: структурированный ответ, валидация и контроль ошибок; основной проект работает без AI.

ПРОЕКТ ПОСЛЕ МОДУЛЯ: EventDesk

API регистрации на мероприятия на FastAPI; отдельное небольшое приложение на Django для практики ORM и Admin.

Для портфолио: работающий API, миграции, права доступа, тесты, Docker Compose и самостоятельный Django-пример.

Модуль 4. Микросервисы, JSON-RPC и распределенные паттерны - 8 ак. ч.

Результат: студент проектирует синхронное взаимодействие сервисов и понимает типовые проблемы распределенной системы.

- 2 ак. ч. - границы сервисов, REST и события; критерии выбора синхронного вызова.
- 2 ак. ч. - JSON-RPC: method, params, id, результат и ошибки; Python-сервер и клиент, таймауты.
- 4 ак. ч. - дубли, идемпотентность, частичные отказы; Saga, Outbox, Circuit Breaker, API Gateway и CQRS.

ПРОЕКТ ПОСЛЕ МОДУЛЯ: Inventory Link

Два небольших Python-сервиса: первый запрашивает остатки у второго через JSON-RPC.

Для портфолио: сервер, клиент, контракт методов, таймауты, обработка отказов и интеграционные тесты.

5. Apache Kafka - отдельный модуль, 20 ак. ч.

Kafka изучается как инструмент событийного взаимодействия Python-сервисов. Практика выполняется с Python-клиентами, а принципы Kafka Streams разбираются отдельно: Java/Scala-библиотека Kafka Streams не выдается за Python-библиотеку.

5.1. Архитектура и базовая модель - 3 ак. ч.

Event-driven architecture: события, команды, сообщения; когда Kafka уместна. Kafka 4 и KRaft: brokers, controller quorum, metadata. Topics, partitions, offsets, порядок, репликация, ISR, min.insync.replicas, retention и compaction. Практика: multi-broker кластер в Docker Compose и анализ партиций.

5.2. Producer API на Python - 2 ак. ч.

confluent-kafka: key/value, headers, timestamp, callback, JSON и сериализация. Ключ партиционирования, acks, retries, batching, compression, идемпотентность. Практика: публикация событий заказа с ключом order_id.

5.3. Consumer API и offsets - 3 ак. ч.

Consumer groups, масштабирование, rebalance, завершение consumer, auto/manual commit, replay. At-most-once и at-least-once: потеря и дубли. Практика: несколько consumer-инстансов, воспроизведение сбоя и защита бизнес-операции от повтора.

5.4. Admin API - 2 ак. ч.

Создание и описание топиков, конфигурация, группы и offsets; последствия изменения партиций. Практика: CLI для аудита Kafka-инфраструктуры.

5.5. Транзакции и гарантии обработки - 2 ак. ч.

Различие идемпотентности producer и транзакций. Consume-process-produce, read_committed и границы exactly-once. Почему транзакция Kafka не охватывает PostgreSQL; Outbox. Практика: транзакционный pipeline.

5.6. Schema Registry - 2 ак. ч.

Avro, Protobuf, JSON Schema, совместимость и эволюция контракта; несовместимые сообщения. Практика: версия события, которую продолжает читать прежний consumer.

5.7. Поточковая обработка на Python - 2 ак. ч.

Фильтрация, агрегаты, состояние, окна; концепции KStream/KTable и state store. Практика: Python-обработчик и обновляемая сводка событий.

5.8. Connect и эксплуатация - 4 ак. ч.

Source/sink, workers/tasks, PostgreSQL/Kafka и CDC как обзор. Lag, throughput, retry, backoff, DLQ, метрики и основы TLS/SASL/ACL. Практика: демонстрация восстановления после ошибки и диагностика задержки consumer.

ПРОЕКТ ПОСЛЕ МОДУЛЯ: Order Stream

Несколько Python-сервисов обрабатывают события заказа с версионизируемым контрактом.

Для портфолио: Producer, Consumer, Admin API, ручной commit, retry, DLQ, метрики и сценарий отказа.

6. Итоговый проект Commerce Hub

Commerce Hub - самостоятельная backend-платформа каталога и заказов. В итоговой работе используется FastAPI; Django остается отдельной учебной практикой модуля 3 и не входит в итоговую систему.

Сценарий работы

Пользователь просматривает каталог и создает заказ. Сервис сохраняет данные в PostgreSQL, обращается к отдельному сервису по JSON-RPC, публикует событие в Kafka и обновляет статус заказа. Обработчик уведомлений реагирует на событие, а отдельный компонент собирает сводку по состояниям заказов.

Обязательные элементы

- FastAPI, модели предметной области, PostgreSQL и миграции.
- Синхронное межсервисное взаимодействие через JSON-RPC.
- Kafka для событий и асинхронных обработчиков; продуманные ключи, топики и контракты.
- Retry, DLQ, защита от повторной бизнес-обработки.
- Unit-, интеграционные и API-тесты; Docker Compose.
- Логирование, health checks, базовые метрики и демонстрация одного сценария сбоя.

Результат для портфолио

Репозиторий с README, схемой сервисов и событий, инструкцией запуска, тестами, демонстрационными данными, примерами запросов и коротким видео. На защите студент объясняет архитектурные решения, путь события и восстановление после сбоя.

7. Подготовка к собеседованию - 4 ак. ч.

Заключительный модуль проводится после завершения и защиты Commerce Hub.

Время	Содержание
1 ак. ч.	Python Core, ООП, SOLID, паттерны, коллекции и асинхронность.
1 ак. ч.	SQL, PostgreSQL, FastAPI/Django, REST, JSON-RPC, тестирование, Docker и Kafka.
1 ак. ч.	Live coding: небольшая задача, чтение кода, поиск ошибок, объяснение решения.
1 ак. ч.	Пробное техническое интервью по итоговому проекту, обратная связь.

ПРОЕКТ ПОСЛЕ МОДУЛЯ: Backend Interview Case

Небольшое самостоятельное задание, разбор решения и защита архитектурных решений на пробном интервью. Для портфолио: репозиторий с кодом и тестами, описание доработок после code review; без публикации личной обратной связи.

8. Проекты в портфолио и контроль знаний

Этап	Что предъявляет студент
1. Library Tracker	Python-приложение и PostgreSQL, запросы, тесты, запуск.
2. Delivery Rules	ООП, SOLID, GoF и аргументированный рефакторинг.
3. EventDesk	FastAPI API; отдельно учебное приложение Django.
4. Inventory Link	Два сервиса и взаимодействие через JSON-RPC.
5. Order Stream	Python-клиенты Kafka, контракты, retry/DLQ и сценарий сбоя.
Итог. Commerce Hub	Связная backend-система на FastAPI, PostgreSQL, JSON-RPC и Kafka.
6. Backend Interview Case	Кодинг-задача, тесты и защита решения.

После каждого модуля - проект, code review и доработка. Минимум для репозитория: задача, README, запуск, тесты и описание решений. Для систем из нескольких сервисов добавляются схема взаимодействия и демонстрация сбоя.

9. Для кого подходит программа

- Новичкам, которым нужен полный маршрут от Python Core до backend-проектов.
- Разработчикам, знакомым с Python и SQL, которым нужны FastAPI, Django, микросервисы и Kafka.
- Специалистам, желающим систематизировать знания и подготовиться к собеседованию.

Гибкая точка входа: после диагностики можно начать с подходящего этапа. Объем, состав и стоимость сокращенного маршрута определяются отдельно. Прохождение программы само по себе не присваивает грейд Junior или Middle.

10. Преподаватель

Сергей Валерьевич Герасименко - практикующий full-stack разработчик, технический директор и основатель ТЕРАКОР и ТЕРАКОР Digital. Объясняет технологию через инженерные решения, практику разработки и разбор кода студентов.

11. Стоимость и следующий шаг

Маршрут	Стоимость
Полная программа: 5 месяцев, 450 ак. ч. общей нагрузки	89 000 ₽
Продвинутый маршрут после определения точки входа	ориентир 59 000 ₽

Пройдите бесплатный тест по Python или запишитесь на консультацию, чтобы выбрать точку входа. Точный состав и стоимость продвинутого маршрута уточняются после диагностики. Актуальные условия смотрите на странице профессии: terakor.ru/catalog/python/python-profession/.