

Мидл Python-разработчик

Пояснения к словам иностранного происхождения

Здесь собраны пояснения к словам иностранного происхождения, которые используются в программе курса, но пока не включены в словарную норму русского языка.

01	Кому подойдёт курс	Для Python-разработчиков с опытом 1-2 года: - Глубже поймёте внутреннее устройство Python - Разберётесь в лучших практиках - Научитесь писать производительный, безопасный код - Повысите свою ценность на рынке Для Python-разработчиков с опытом более 3 лет: - Систематизируете знания - Научитесь проектировать сложные системы и принимать архитектурные решения - Выйдете на новый уровень экспертизы Для опытных backend-разработчиков на других языках: - Быстро войдёте в Python и начнёте применять его в продакшене - Овладеете ключевыми инструментами - Расширите карьерные перспективы
02	Чему научитесь на курсе	- Освоите разработку сложных приложений - Создадите приложение из 10 + микросервисов с надёжным взаимодействием через OpenAPI, Protobuf и Apache Avro - Изучите технологии, которые помогут при разработке. Научитесь работать с NoSQL и реляционными базами, OLTP и OLAP, брокерами сообщений и поисковыми системами - Поймёте, как обеспечить работу приложения без сбоев - Освоите все виды тестов, трассировку и логирование, горизонтальное масштабирование и оркестрацию
03	Как проходит курс	- Сопровождение опытными наставниками - Теория на платформе Практикума - Командная или индивидуальная работа над практическими заданиями в зависимости от выбранного тарифа - Вебинары с экспертами и демо каждый спринт для обсуждения промежуточных результатов работы над проектом

Что вас ждёт

Документ о полном прохождении курса	11 практических заданий с проверкой экспертом-ревьюером	6 вебинаров с опытными наставниками и 17 демо
-------------------------------------	---	---

Мидл Python-разработчик

6 месяцев

продолжительность курса

5 ЧАСОВ 00 Интро-модуль	120 ЧАСОВ 01 Сервисы Admin Panel и ETL	80 ЧАСОВ 02 Сервис Async API
80 ЧАСОВ 03 Сервис авторизации	80 ЧАСОВ 04 Сервис UGC	60 ЧАСОВ 05 Сервис нотификации
80 ЧАСОВ 06 Дипломный сервис	9 ЧАСОВ ✱ PyTest для автоматизации тестирования (дополнительный модуль)	8 ЧАСОВ ✱ Личное развитие тимлида разработчиков (дополнительный модуль)
40 ЧАСОВ ✱ Оркестрация масштабных приложений (дополнительный модуль)		

Сравнение тарифов курса «Мидл Python-разработчик»

	Базовый тариф	Тариф с индивидуальными проектами
Длительность	6 месяцев	6 месяцев
Нагрузка в неделю	15-17 часов	15-20 часов
Количество проектных работ с проверкой ревьюером	8 командных 3 индивидуальных	0 командных 11 индивидуальных
Формат сдачи проектных работ	В команде выбирается лидер, который отправляет ревьюеру работу и получает правки по проекту. Остальные члены команды получают комментарии ревьюера через лидера команды.	Вы отправляете ревьюеру работу и получаете обратную связь, которую обрабатываете лично.
Описание	Имитация промышленной разработки, где каждый отвечает за свою зону проекта.	Имитация фриланс-разработки, где вся ответственность только на вас.
	Вы фокусируетесь только на одной области проекта, которую дорабатываете из спринта в спринт. По договорённости с командой можно поменяться ролями и выбрать новую область проекта.	Самостоятельная работа над всеми зонами проекта и большее погружение в предметную область. Но при этом более высокая загрузка.
	Прокачка навыков коммуникации и работы в команде.	Независимая работа над проектом. Не нужно согласовывать графики и подстраиваться под сокомандников.
	Всё остальное в тарифах одинаковое — материалы, воркшопы, работа наставников, кураторов и прочее.	

Бесплатное входное тестирование

2–3 часа

Серия вопросов про бэкенд-разработку.
Тест рассчитан на 2–3 часа спокойного прохождения.
Секции про Python, хранилища данных, архитектуру и DevOps.
Подробности про программу и формат курса.

Информация о курсе
Входное тестирование

Интро-модуль

00

5 часов

Модуль про требования к коду, работу с Git, формат курса и первая практика.

- Темы:
- Небольшой онбординг
 - Знакомство с Git
 - Требования
-

120 часов | 6 недель

Технологии:

- Python + UWSGI + Django
- PostgreSQL
- Nginx
- Docker
- ETL
- Elasticsearch

Создание и настройка панели администратора для онлайн-кинотеатра на Python с применением Django. Детальное изучение самых используемых компонентов Django — админки и ORM. Знакомство с индексами, миграциями и оптимизацией запросов. Проектирование реляционной базы данных. Устройство Docker и работа Nginx. Знакомство с архитектурным паттерном ETL и создание своего ETL для перекачки данных из PostgreSQL в Elasticsearch.

Проекты

Проект: админка онлайн-кинотеатра + ETL для перекачки данных.

Содержание модуля

Спринт 1	• Проектирование базы данных, знакомство с индексами и потреблением памяти • Знакомство с процессом вставки данных, dataclasses и работой со временем в базах данных • Вводная информация о базах данных без SQL, критерии выбора фреймворка и подробное изучение Django: от установки до админки	Проектная работа 1 спринта (индивидуальная в обоих тарифах) • Описание схемы базы данных PostgreSQL, с которой будут работать участники курса; • Создание скрипта для миграции данных с Sqlite на PostgreSQL; • Разработка сервиса, который позволит создавать и редактировать записи в базе данных.
Спринт 2	• Глубокое изучение технологии ORM • Реализация API в сотрудничестве с Django и тестирование результата • Подробное знакомство с Docker • Обработка запросов в production-среде и конфигурирование Nginx	Проектная работа 2 спринта (индивидуальная в обоих тарифах) • Реализация API на Django, который возвращает список фильмов; • Запуск Django-приложения в Docker; • настройка работы приложения с помощью Nginx + uWSGI.
Спринт 3	• Подробное изучение Elasticsearch, полнотекстового поиска и архитектуры ETL	Проектная работа 3 спринта (индивидуальная в обоих тарифах) • Создание отказоустойчивого переноса данных из Postgres в Elasticsearch.

80 часов | 4 недели

Технологии:

- Python
- ASYNCIO
- FastAPI
- Redis
- Итераторы и генераторы
- Корутины

Погружение в асинхронный Python и принципы его работы и применение Fast API на практике. Знакомство с итераторами, генераторами и корутинами. Разбор многопоточности и многопроцессности.

Пользователи базового тарифа работают в командах по 2–3 человека. Процесс разработки имитирует разработку в настоящей продуктовой компании — у команд есть тимлид, код-ревью и еженедельные демо с наставниками.

Пользователи индивидуального тарифа работают самостоятельно, но также проходят код-ревью и участвуют в еженедельных демо с наставниками.

Проекты

Проект: API онлайн-кинотеатра

Содержание модуля

Спринт 4

- Подробное описание общих принципов код-ревью
- Разница между солидным и несолидным кодом
- Функциональные тесты для сервиса и инфраструктура тестов
- Понятие API и инструкция по написанию документации

Проектная работа 4 спринта (индивидуальная или командная в зависимости от тарифа)

- Ревью двух работ коллег по команде / собственной работы;
- Выполнение задач по SOLID;
- Проведение функциональных тестов API.

Спринт 5

- Знакомство с асинхронным подходом, итераторами, генераторами и корутинами
- Разбор асинхронных фреймворков: Twisted, AIOHTTP и FastAPI
- Детальное изучение FastAPI

Проектная работа 5 спринта (индивидуальная или командная в зависимости от тарифа)

- Создание базовой структуры приложения FastAPI;
- Настройка Dockerfile и Docker compose;
- Создание модели для фильмов и связанных данных;
- Создание сервиса для получения информации о фильмах;
- Доработка существующего ETL;
- Добавление сервиса, который обращается к Elasticsearch и возвращает информацию о персоналиях и жанрах;
- Кеширование данных в Redis.

80 часов | 4 недели

Технологии:

- Python + Flask
- Redis
- PostgreSQL (реплики/шарды)
- OpenID
- oauth2
- jwt
- captcha + Proof-of-Work

Разработка сервиса, отвечающего за создание, хранение и логирование профиля юзера. Он позволит пользователям регистрироваться в кинотеатре и управлять своим аккаунтом. Подробный разбор, как правильно хранить пароли и обеспечить безопасность пользовательских данных. Участники курса самостоятельно проектируют архитектуру сервиса, защитив свои решения на ревью. Они продумывают систему взаимодействия между сервисами, обращая внимание на обработку ошибок и возможное отключение одного из сервисов.

Погружение в синхронные фреймворки на примере Flask и работа с реляционными базами данных на примере PostgreSQL. Фокусировка на компьютерной безопасности и правильной организации сбора, защиты и хранения данных.

Проекты

Проект: сервис авторизации пользователей с защитой данных.

Содержание модуля

Спринт 6	• Проектирование базы данных, знакомство с индексами и потреблением памяти • Знакомство с процессом вставки данных, dataclasses и работой со временем в базах данных • Вводная информация о базах данных без SQL, критерии выбора фреймворка и подробное изучение Django: от установки до админки	Проектная работа 6 спринта (индивидуальная или командная в зависимости от тарифа) Создание основы сервиса авторизации и реализация всех базовых требований к нему.
Спринт 7	• Детальный рассказ о работе OAuth 1.0, OAuth 2.0 и openID • Работа с капчей и двухфакторной авторизацией • Изучение общения между микросервисами. • Настройка устойчивости сервиса к нагрузкам и изменениям API, внедрение в AsyncAPI авторизации, управление уровнем нагрузки и быстрое нахождение проблем	Проектная работа 7 спринта (индивидуальная или командная в зависимости от тарифа) Упрощение регистрации и аутентификации пользователей с помощью входа через социальные сервисы.

80 часов | 4 недели

Технологии:

- Python
- Mongo
- ApacheKafka
- Clickhouse
- Apache Spark
- CI / CD
- ELK

Сервис разрешает пользователям создавать свой контент в вашем кинотеатре — оставлять лайки, комментарии, отзывы к фильмам. Участникам курса нужно организовать сбор и хранение данных об активности пользователей. Самостоятельный выбор асинхронного фреймворка и хранилища данных для сервиса. Проведение собственного исследования производительности разных хранилищ данных. Защита полученных результатов, выводов и принятых решений на демо с наставником.

Проекты

Организация сбора и хранения данных об активности пользователей.

Содержание модуля

Спринт 8

- Что такое UGC и зачем оно нужно. Планирование сервиса
- Описание архитектуры и полезные советы по её проектированию
- Выбор хранилища, событийная работа с данными и работа с Kafka
- Подходы к хранению и обработке данных, которые не помещаются на один сервер. Локальный подъём вычислительного кластера и работа с данными внутри него

Проектная работа 8 спринта (индивидуальная или командная в зависимости от тарифа)

- Отрисовка архитектуры сервиса;
- Работа с Kafka и Spark;
- Задача по ClickHouse;
- Создание ETL для перегрузки данных в аналитическое хранилище.

Спринт 9

- Ещё одно распределённое хранилище, хранение данных и оптимизация хранения и обработки
- CI/CD: Построение непрерывной интеграции для приложения
- Знакомство с логированием в микросервисной архитектуре

Проектная работа 9 спринта (индивидуальная или командная в зависимости от тарифа)

- Упрощение регистрации и аутентификации пользователей с помощью входа через социальные сервисы.

60 часов | 3 недели

Технологии:

- Python
- FastAPI
- RabbitMQ
- CI/CD

Сервис нотификации отвечает за коммуникацию с пользователем через Email, мобильные пуш-уведомления, websocket в браузере. Это часто решаемая бизнес-задача в коммерческой разработке — от правильной коммуникации с пользователем зависят показатели бизнеса.

Изучение архитектурных проблем веб-разработки.

Самостоятельное прохождение пути от бизнес-задачи до рабочего решения. Самостоятельный выбор стека технологий и отрисовка архитектуры сервиса.

Новые технологии: брокеры сообщений на примере RabbitMQ и Websocket.

Проекты

Проект: сервис нотификаций через email, мобильные пуш-уведомления, websocket в браузере.

Содержание модуля

Спринт 10

- Описание принципа работы сервиса нотификаций
- Механизм отправки сообщений пользователям и единая политика контактов
- Проектирование коммуникационной системы
- Устройство email и шаблонизаторов. Изучение, какие протоколы или технологии можно применить в разных ситуациях, когда нужно связаться с клиентом
- Использование брокера сообщений RabbitMQ. Сервисы транзакционной рассылки писем и сокращённые ссылки
- Возвращение к CI/CD-процессу

Проектная работа 10 спринта (индивидуальная или командная в зависимости от тарифа)

- Проектирование сервиса email-рассылок с учётом потенциального расширения количества способов связи;
- Настройка отправки одинаковых писем всем пользователям;
- Настройка отправки персонифицированных писем;
- Создание модуля для работы с Websocket;
- Приветственное письмо после регистрации пользователя.

80 часов 4 недели	
Технологии: - Технологии - Python + Framework (фреймворк участники курса выбирают сами) - Хранилище данных и остальные технологии участники курса выбирают сами	Участники курса сами выбирают бизнес-задачу или придумывают её, планируют своё время, проектируют архитектуру решения, выбирают и исследуют технологии, затем защищают разработанное приложение. Завершающая точка — защита финального проекта, где участники курса показывают результат своей работы.
	Проекты
	Проект: сервис на выбор.

Содержание модуля

Спринт 11	- Организация работы в финальном модуле - Готовые идеи для реализации в итоговом проекте - Требования для разработки и виды заказчиков - Декомпозиция проекта и самостоятельная подготовка ТЗ для заказчика - Полезные советы, как организовать рабочее время и наладить прозрачные процессы в команде - Архитектура, риски и планирование задач по проекту	Проектная работа: (индивидуальная или командная в зависимости от тарифа) Выполнение итогового проекта
-----------	---	--

PyTest для автоматизации тестирования (дополнительный модуль)



9 часов

Технологии:

- Архитектура
- CSS PyTest
- Параметризация
- Assert
- Фикстуры
- Оценка покрытия
- Unit-тесты

Узнаете, что такое юнит-тесты и правила разработки юнит-тестов, Изучите пирамиду тестирования. Рассмотрите фреймворк pytest и попрактикуетесь использовать его в разработке автотестов. Также вы изучите, как правильно выстраивать структуру автотестов и как правильно именовать их. Разберётесь, как работать с тестовыми данными в тестах с помощью параметризации, а также как оптимизировать функции в тестах с помощью фикстур. Научитесь оценивать и считать покрытие автотестами.

Личное развитие тимлида разработчиков (дополнительный модуль)



8 часов

Технологии:

- SMART
- Многозадачность
- Общение с заказчиком
- Позиционирование себя в команде
- Общение с руководителем
- Противостояние выгоранию

Погрузитесь в вопросы развития софт-скиллов для тимлидов. Вспомните, как ставить SMART-цели, оценивать свои сильные стороны и зоны роста, балансировать между кодом и управлением командой. Рассмотрите особенности продуктового мышления, приоритизацию задач и эффективную коммуникацию с разными ролями, а также научитесь гибко адаптировать планы под изменения.

Оркестрация масштабных приложений (дополнительный модуль)



40 часов

Технологии:

- Health check + K8S
- Сбор и мониторинг метрик в сервисах
- Configuration-as-Code (CaC)

Научитесь разворачивать Kubernetes кластер в облаке и локально, разберётесь с применением health check в сложных системах, освоите метрики в Python и научитесь работать с Prometheus и Grafana

Проекты

- Разработаете и опишете необходимые манифесты
- и опубликуете приложение