**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Бек-энд разработка

# Отчет
# Лабораторная работа №1
# “Typescript: основы языка”

Выполнил:
Стукалов Артем
K33392

Проверил:
Добряков Д. И.

Санкт-Петербург

2024 г.

**Задание:**
Нужно написать свой boilerplate на fastify + prisma + typescript + zod.
Должно быть явное разделение на:

- модели
- контроллеры
- роуты
- сервисы для работы с моделями (реализуем паттерн “репозиторий”)

**Fastify + Prisma + Zod API template**

Небольшой темплейт для написания простеньких АПИ с упором на следующие аспекты:
- Полная типобезопасность. Достигается за счет строгих правил TS и автоматической генерации типов библиотекой `Prisma`
- Удобсвто валидации данных. Используем `Zod` в качестве валидатора запросов и языка схем запросов. Для интеграции с `Prisma` используем плагин `zod-prisma-types`.
- Скорость разработки и сборки проекта. Используем `swc` для компиляции typescript и nodemon для перезапуска проекта.

**Типобезопасность и Prisma**

Typescript - прекрасная вешь, которая может зашитить от огромного кол-ва глупых ошибок вроде опечаток или забытой проверки на null. Но чтобы использовать его по максимому, нужно для начала написать хорошую систему типов.
В данном темплейте стараемся как можно больше работы переложить на компилятор и сократить до минимума использование `any` и тайпкастов через `as`.

В качестве ORM используем `Prisma`. Эта библиотека полностью типо безопасна, так как не вынуждает нас самостоятельно писать типы для моделей, миграций и подобного. `Prisma` самостоятельно генерирует типы моделей и сами модели, миграции от одной схемы БД к другой. Единственное, что на мнужно менять это файл схемы БД, который использует свой собственный синтаксис, поддержку которого очень легко добавить через плагин для VSCode(Prisma VS Code Extension).

**Zod**

Для валидации пользовательских данных используем `zod` - еще одна типобезопасная библиотека. Возможно внутри она и использует касты, но будем опираться на понятие типобезопасной абстракции.

Еще одно преимущество использования это возможность получить из `zod` схемы типы для TS. Также `zod` возможно интегрировать с `Prisma`: при генерации типов для моделей данных типы для TS будут дублироваться в виде `zod` схем для валидации.

**SWC**

SWC на данный момент самая быстрая библиотека компиляции typescript. Пример для данного темплейта на моем железе:

- Холодный старт проекта: 68мс
- Перекомпиляци я в watch режиме: 4мс

С такой скоростью мы ограничены только скоростью самой node-js.

На данный момент `swc` не поддерживает полноценный бандлинг проектов, но при желании его можно использовать в качестве лоадера для webpack или rollup, которые уже соберут проект в один файл.