# Антипаттерны разработки

### Магические числа (Magic numbers) — не обраружил в коде Спагетти-код (Spaghetti Code) — не обраружил (субьективная оценка) Лазанья-код (Lasagna Code) — не обраружил, каждый слой изолирован функционально Слепая вера (Blind Faith) — не обнаружил. В проете такого на нашел.

### Шифрованный код (Cryptic code) — такогого за собой не замечаю

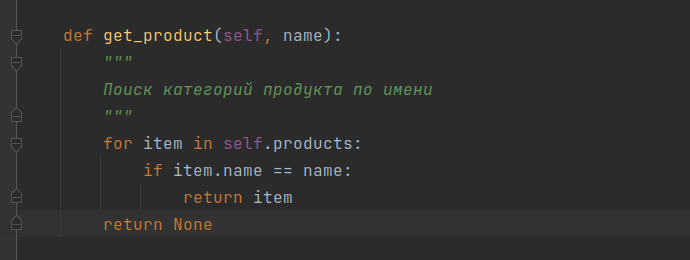
### Жёсткое кодирование (Hard code) — Такое водится за мной первоночальном этапах проекта (реализация прототипа), т. к. хочется быстрее сделать готовое решение, потом можно подумать об «универсализме». На самом деле ранее не считал использование файлов ini антипаттерном, сейчас буду знать.

### Мягкое кодирование (Soft code) — иногда, не могу остановится, когда хочется реализовать какой-то универсальный модуль, в проекте я этого не увидел

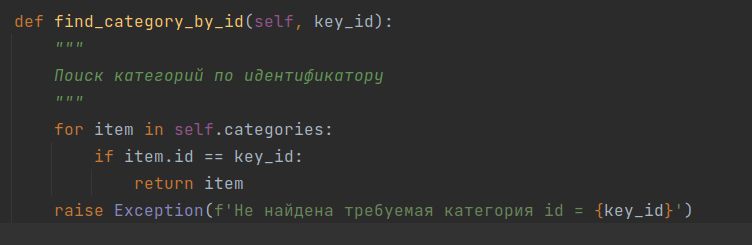
### Поток лавы (Lava flow) — в данной проекте данный антиаттерн думаю есть, т. к. нет жётского валидатора, который пропускал данные между уровней (например это создание объектов, если перредать невалидные параметры, т.о. следует обрабатывать исключительные ситуацции). Большую роль, как мне кажется, влияет уровень изоляции того или иного модуля, По сути если мы пишем каждый модуль воспринимаем входные данные как «чёрный ящик», то даного паттерна не должен возникнуть.

### Специализированные антипаттерны в Python:

* + Обработка ошибок и исключительных ситуаций через if – такой антипаттер есть:



* + Решение можем реализовать аналогично:



### Создание словарей более сложным способом - в проекте его нет, сам придерживаюсь данному паттерну

### Отдельное присваивание значений вместо распаковки

– спорный паттерн, может усложнить читаемость кода. В проекте его нет. Считаю его можно применять, когда переменные имеет одну и ту же природу и принимают начальное значение например: “index1, index2 = index2, index2+1”

### Использование map и фильтров вместо списковых включений

### Придерживаюсь данному паттерну, запомнил еще по вашему курсу алгоритмы. В проекте его нет

### В ООП Боязнь размещать логику в объектах предметной области

Если я правильно понял, то к данному паттерну мы можем отнести модуль «engine.py», также некоторую функциональность контроллера.  
Т.о. такой антипаттерн есть в проекте, нужно перенести логику работы из этих моделей с моделями в модуль модели.

### Божественный объект (God Object)

## Да, такой антираттерн есть это «engine.py», исправить егго можно разделить функциональность на отдельные классы. (например ProductEngine, CategoryEngine).

### Полтергейст (Poltergeist)

В текущем решении данный паттерн наверняка не проявит себя, но в большом проекте, нужно реализовывать тестирование

### Сплошное одиночество (Singletonitis)

В текущем прокте данного паттерна нет

### Приватизация (Privatization)

### В проекте данного паттерна нет

## Методологические антипаттерны

Такого нет в текущем проекте

### Золотой молоток (Golden Hammer)

Такого нет в текущем проекте

### Фактор невероятности (Improbability factor)

Наличие такого паттерна в проекте можно рассмотреть как отсудствии валидации при создании объектов.

### Преждевременная оптимизация (Premature optimization)

Я грешу переодически этим паттерном, работаю над этим. В проекте его нет.

### Изобретение велосипеда (Reinventing the wheel)

Данный паттерн есть в проекте соотвественно — это наличие собственного FrameWork’а

### Изобретение квадратного колеса (Reinventing the square wheel)

Этот паттерн тоже можно отнести к даннному проекту, т. к. конечно же Djangо, Flask, работает лучше. Но т. к. проект не завершен и мы его делали в учёбных целях, то данный паттерн для даннного проекта не актуален.

# Архитектурные антипаттерны

Переходим к архитектурным антипаттернам

## Инверсия абстракции (Abstract Inversion)

Этот паттена я не нашел в проекте

## Большой комок грязи (Big ball of mud)

Этот паттена я не нашел в проекте

Затычка на ввод (Input kludge)

Данный паттерн можно обнаружить, т.к. валидация данных при создании объектов нет

## Волшебная кнопка (Magic pushbutton)

Данный паттерн можно обнаружить, требуется перенсти некоторую функциональность из контроллеров и модуля, разделить класс Еngine на классы

## Членовредительство (Mutilation)

Данного антипаттерна я не обнаружил в проекте

## Дымоход предприятия (Stovepipe Enterprise)

Данный антипаттерн не применим к данному проекту, т.к. он является учебным

## Дымоход системы (Stovepipe System)

Данный антипаттерн не применим к данному проекту, т.к. он является учебным

## Путаница (Jumble)

Данный антипаттерн не обнаружил в проекте.