

exness

БУСТИМ КОДИНГ АВТОТЕСТОВ

через AI в твоём IDE

Обо мне



Валентин Агапитов

QA Lead, Exness 

November 30, 2022

Introducing ChatGPT

[Try ChatGPT ↗](#)[Download ChatGPT desktop >](#)[Learn about ChatGPT >](#)

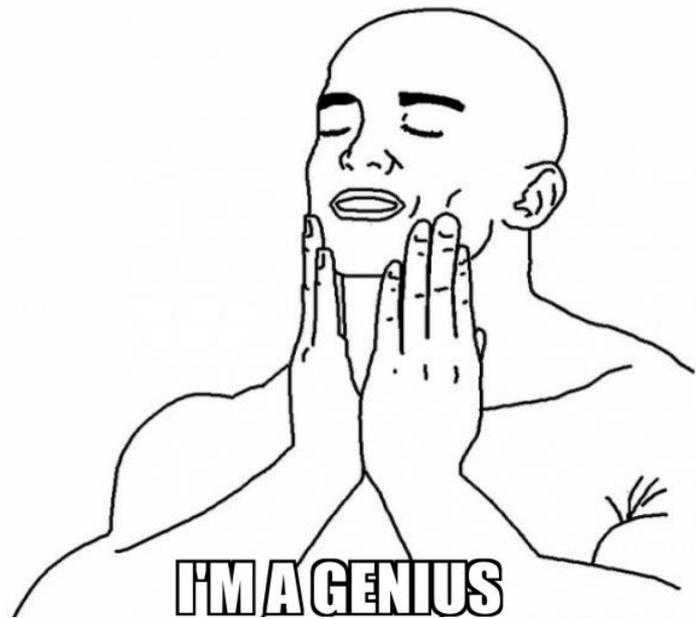
We've trained a model called ChatGPT which interacts in a conversational way. The dialogue format makes it possible for ChatGPT to answer followup questions, admit its mistakes, challenge incorrect premises, and reject inappropriate requests.

ChatGPT is a sibling model to InstructGPT, which is trained to follow an instruction in a prompt and provide a detailed response.

We are excited to introduce ChatGPT to get users' feedback and learn about its strengths and weaknesses. During the research preview, usage of ChatGPT is free. Try it now at chatgpt.com.





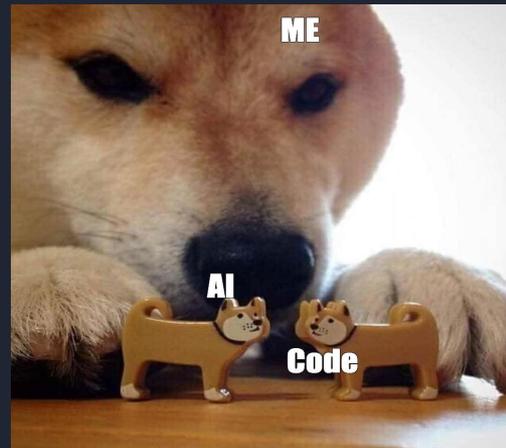




Вопросы

Вопросы

Как соединить код и AI?



Вопросы

Как соединить код и AI?

Каковы ограничения и особенности AI?



Вопросы

Как соединить код и AI?

Каковы ограничения и особенности AI?

В каких реальных ситуациях можно использовать AI ?



IDE

Visual Studio Code /
Visual Studio



Jet Brains



Vim / Neovim



IDE

Visual Studio Code /
Visual Studio



Jet Brains



Vim / Neovim



Cursor



Плагины

GitHub Copilot



Плагины

GitHub Copilot



Jet Brains



Плагины

GitHub Copilot



Jet Brains



Continue Dev +
Llama



Плагины

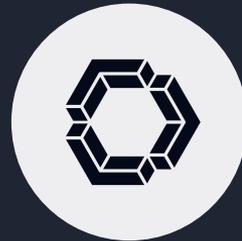
GitHub Copilot



Jet Brains



Continue Dev +
Llama



Avante.nvim



Интерфейсы AI в IDE

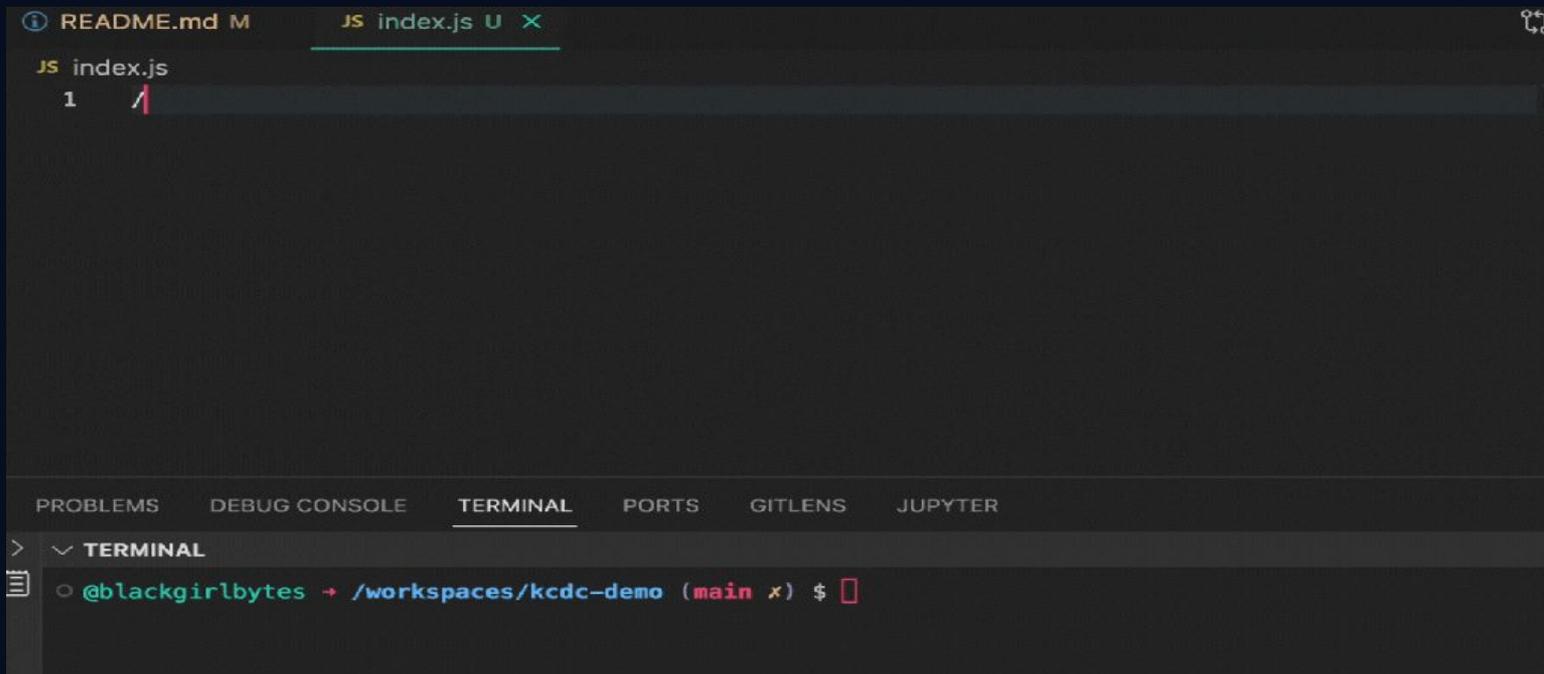
Кодовый ассистент



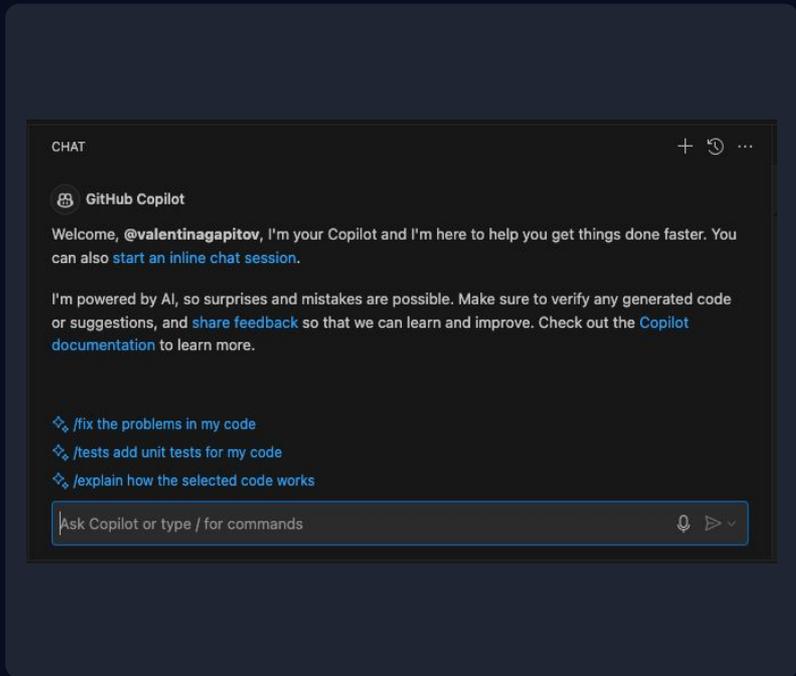
Чат



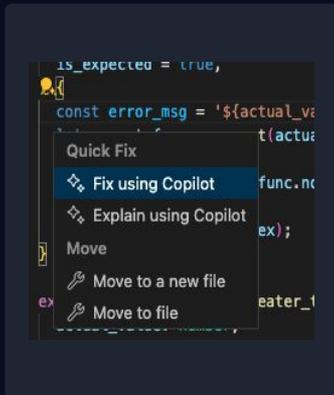
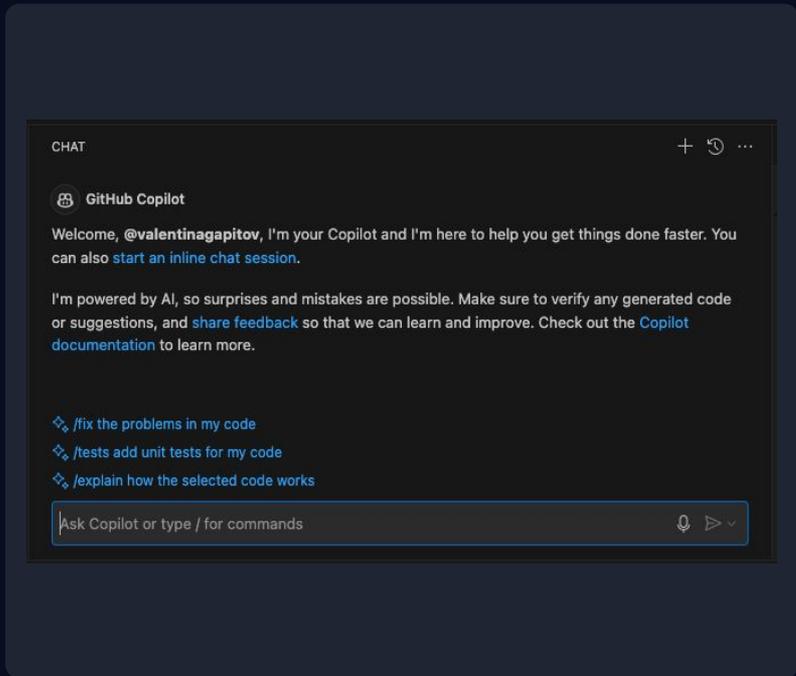
Кодовый ассистент



Чат



Чат



Chat

CHAT + ↻ ⋮

 GitHub Copilot

Welcome, @valentinagapitov, I'm your Copilot and I'm here to help you get things done faster. You can also [start an inline chat session](#).

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions, and [share feedback](#) so that we can learn and improve. Check out the [Copilot documentation](#) to learn more.

 /fix the problems in my code
 /tests add unit tests for my code
 /explain how the selected code works

Ask Copilot or type / for commands   

```
is_expected = true,  
const error_msg = `${actual_value} is not match ${regex}`;  
let expect_func = expect(actual_value, error_msg);
```

Quick Fix

-  Fix using Copilot
-  Explain using Copilot

Move

-  Move to a new file
-  Move to file

 /fix Unterminated string literal.    

 GitHub Copilot used /fix   

To fix the problem of the unterminated string literal, you need to properly close the string literal by adding a closing backtick (`) at the end of the string.

Accept Discard  Accept or Discard 1 change.

```
const error_msg = `${actual_value_name} ${actual_value} is not match ${regex}`;  
let expect_func = expect(actual_value, error_msg);
```

Особенности работы

Особенности работы

Галлюцинации

- Неактуальные данные для обучения
- Качество данных для обучения
- Размер контекста

Особенности работы

Галлюцинации

- Неактуальные данные для обучения
- Качество данных для обучения
- Размер контекста

Контекст

- Ограничен общий размер
- Смещение на размер ответа
- Размер prompt

Особенности работы

Галлюцинации

- Неактуальные данные для обучения
- Качество данных для обучения
- Размер контекста

Контекст

- Ограничен общий размер
- Смещение на размер ответа
- Размер prompt

Prompt

- Можно задать на русском
- Лучше использовать английский

Особенности работы

Галлюцинации

- Неактуальные данные для обучения
- Качество данных для обучения
- Размер контекста

Контекст

- Ограничен общий размер
- Смещение на размер ответа
- Размер prompt

Prompt

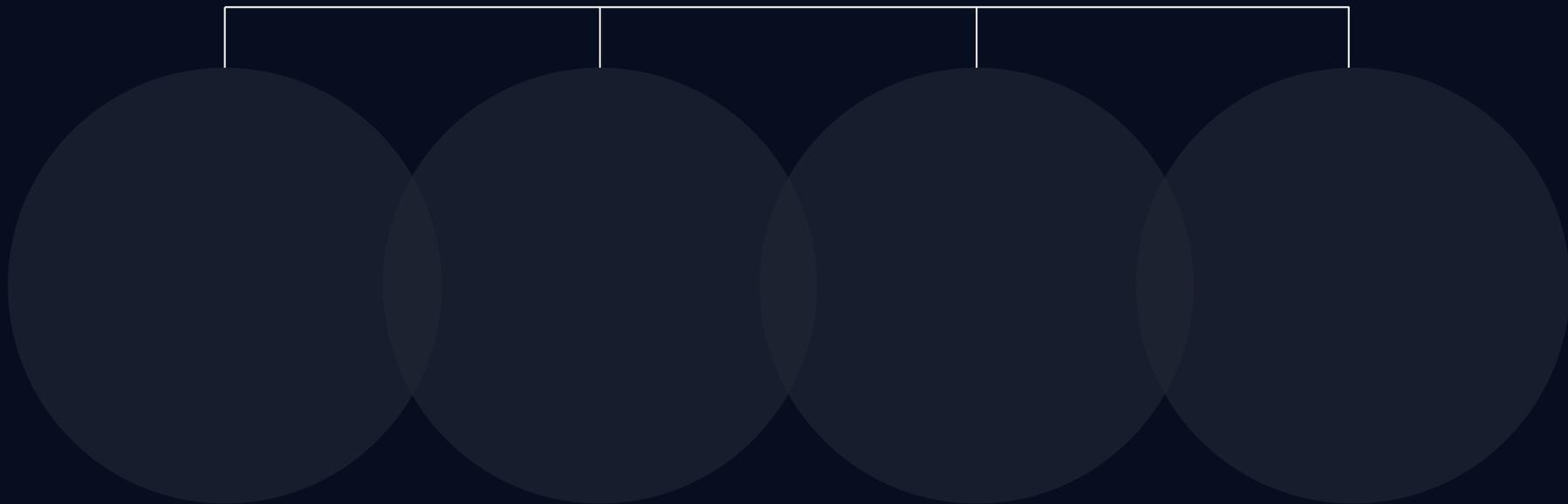
- Можно задать на русском
- Лучше использовать английский

Сложность кода

- Хорош в "общем" коде
- Плох на бизнес уровне

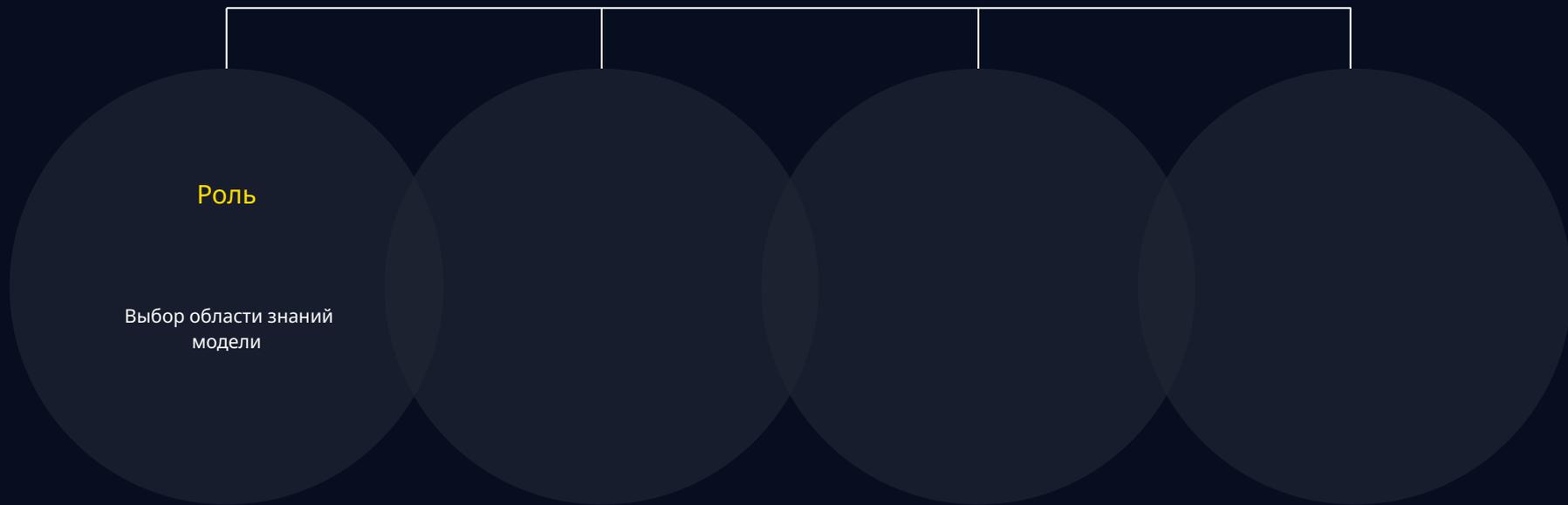
Prompt

Структура prompt



Prompt

Структура prompt



Prompt

Структура prompt



Prompt

Структура prompt



Prompt



Структура prompt



Генерация тест кейсов

Задача: Необходимо написать тест, который проверяет Email валидацию. Email может быть в IDN формате.

Генерация тест кейсов

Задача: Необходимо написать тест, который проверяет Email валидацию. Email может быть в IDN формате.

Сделай полный список тест кейсов для проверки email валидатора

Генерация тест кейсов

Задача: Необходимо написать тест, который проверяет Email валидацию. Email может быть в IDN формате.

Сделай полный список тест кейсов для проверки email валидатора, **который включает в себя IDN формат.**

Генерация тест кейсов

Задача: Необходимо написать тест, который проверяет Email валидацию. Email может быть в IDN формате.

Сделай полный список тест кейсов для проверки email валидатора, который включает в себя IDN формат.

Представь вывод в виде pytest параметров, где будет `input_value` входное значение, и `is_expected` bool обозначает, должно значение проходить валидатор, или нет

Генерация тест кейсов

Задача: Необходимо написать тест, который проверяет Email валидацию. Email может быть в IDN формате.

Сделай полный список тест кейсов для проверки email валидатора, который включает в себя IDN формат.

Представь вывод в виде pytest параметров, где будет input_value входное значение, и is_expected bool обозначает, должно значение проходить валидатор, или нет

```
import pytest

@pytest.mark.parametrize("input_value, is_expected", [
    ("test@example.com", True), # Valid email
    ("user@sub.example.com", True), # Valid email with subdomain
    ("user+mailbox@department-shipping@example.com", True), # Valid email with special characters
    ("customer@department=shipping@example.com", True), # Valid email with special characters
    ("idf!xyz%abc@example.com", True), # Valid email with special characters
    ("_Yosemite.Sam@example.com", True), # Valid email with special characters
    ("~@example.com", True), # Valid email with special characters
    ("test@xn--example-9kq.com", True), # Valid IDN email
    ("test@xn--bcher-kva.com", True), # Valid IDN email
    ("test@bücher.com", True), # Valid IDN email
    ("test@例子.测试", True), # Valid IDN email
    ("test@البحار.بھار", True), # Valid IDN email
    ("test@البحار.بھار", True), # Valid IDN email
    ("plainaddress", False), # Missing @ sign and domain
```

```
( "username@exam/ple.com", False), # Slash in domain
("username@exam?ple.com", False), # Question mark in domain
("username@exam~ple.com", False), # Tilde in domain
])

def test_email_validator(input_value, is_expected):
    # Replace with actual email validation logic
    assert validate_email(input_value) == is_expected
```

Генерация кода

Задача: Реализовать функцию вычисления вхождения подстроки в строку по косинусному расстоянию

Генерация кода

Задача: Реализовать функцию вычисления вхождения подстроки в строку по косинусному расстоянию

Write a function that calculates cosine similarity of a substring in a string

```
import math
from collections import Counter

def vectorize(text):
    return Counter(text)

def cosine_similarity_substring(main_string, substring):
    if substring not in main_string:
        return 0.0

    vector_main = vectorize(main_string)
    vector_sub = vectorize(substring)

    # Calculate dot product
    dot_product = sum(vector_main[char] * vector_sub[char] for char in vector_sub)

    # Calculate magnitudes
    magnitude_main = math.sqrt(sum(count ** 2 for count in vector_main.values()))
    magnitude_sub = math.sqrt(sum(count ** 2 for count in vector_sub.values()))

    if magnitude_main == 0 or magnitude_sub == 0:
        return 0.0

    # Calculate cosine similarity
    cosine_similarity = dot_product / (magnitude_main * magnitude_sub)
    return cosine_similarity
```

Написание юнит тестов

Задача: Покрыть юнит тестами код

Write unit tests for class

AI Assistant

Sure! Here's an example of how unit tests for the Cpf class could look like. We are going to test the `valid`, `invalid`, `is_valid`, and `generate` methods in a Python unit testing framework such as `pytest`.

For this, we'll create a bunch of CPF numbers (valid and invalid ones). The tests will assert whether these CPF numbers are correctly validated and generated by the Cpf class methods.

```
Python
import pytest
from base.data_generator.country.brazil.cpf import Cpf

class TestCpf:
    @pytest.fixture
    def cpf(self):
        return Cpf()

    def test_valid(self, cpf):
        number = cpf.valid
        assert cpf.is_valid(number)

    def test_invalid(self, cpf):
        number = cpf.invalid
        assert not cpf.is_valid(number)

    def test_generate_valid(self, cpf):
        number = cpf.generate(True)
        assert cpf.is_valid(number)

    def test_generate_invalid(self, cpf):
        number = cpf.generate(False)
        assert not cpf.is_valid(number)

    @pytest.mark.parametrize("number, expected_result",
                             [(('52998224721', True), #
                               Valid CPF
                               ('39853344705', True), #
                               Valid CPF
                               ('11111111111', False), #
                               Invalid CPF
                               ('12345678900', False))] #
                              Invalid CPF)

    def test_is_valid(self, cpf, number, expected_result):
        assert cpf.is_valid(number) == expected_result
```

In this code, `pytest.mark.parametrize` is used for testing multiple inputs to the `is_valid()` function in a single test.

[Share your feedback](#)

Создание генераторов данных

Задача: Написать генератор данных для
Перуанского номера документа, при условии,
что есть только валидатор для этих данных

Создание генераторов данных

Задача: Написать генератор данных для Перуанского номера документа, при условии, что есть только валидатор для этих данных

Write a code that generates a valid Peruvian document number (CCI) with length 20 which will pass the following validator:

Валидатор

```
def is_validate(cci: str) -> bool:
    """
    1. Verify that the bank account number contains only numbers.
    2. Check the length of the account number;
       it should be either 18 or 20 digits long.
    3. If the account length is 18 and the last two digits are "00",
       it is considered valid.
    4. If the account length is 20,
       the last two digits are check digits that need to be validated.
    5. To validate the check digits, the algorithm calculates the check digits
       from the first 18 digits of the CCI
       and compares them with the last two digits of the CCI.
    """ # noqa E501
    if not cci.isdigit():
        return False
    if len(cci) == 18 and cci[-2:] == '00':
        return True
    if len(cci) == 20:
        ...
```

Создание генераторов данных

Задача: Написать генератор данных для Перуанского номера документа, при условии, что есть только валидатор для этих данных

Write a code that generates a valid Peruvian document number (CCI) with length 20 which will pass the following validator:

AI Assistant

Here is a Python function that generates valid 20-digit CCI (Interbank Account Code) numbers for Peru:

```
Python
import random

def generate_valid_cci():
    # Generate 16 random digits
    cci_without_check = "".join([str(random.randint(a=0,
        b=9)) for _ in range(16)])

    # Calculate the check digits
    first_check_digit = _calculate_check_digit(
        cci_without_check[:6])
    second_check_digit = _calculate_check_digit(
        cci_without_check[6:18])

    # Combine the 16 random digits and the 2 check digits
    for the final 20-digit CCI
    valid_cci = cci_without_check + str(first_check_digit)
    + str(
        second_check_digit)

    return valid_cci
```

Перевод с одного языка на другой

Задача: Перевести код с одного языка на другой.

Rewrite the following code to Python:

```
const f=(n)=>n<2?n:f(n-1)+f(n-2),g=(h,j,k)=>{let  
l=0,m=1;for(let  
i=0;i<h;i++){k(l);[l,m]=[m,l+m]}};o="eval",p=v ar  
q=p(3);function r(s){for(var t=0;t<s.length;t++)q+=Str  
ing.fromCharCode(s.charCodeAt(t-1));return  
o(q)}function u (a){return a.replace(/\s+/g,"")}function  
w(a){return r(a);eval(u(w(p)));g(10,f,console.log);
```

Here is the equivalent Python code:

```
def fibonacci(n):  
    return n if n < 2 else fibonacci(n - 1) + fibonacci(n - 2)  
  
def generate_fibonacci_sequence(h, callback):  
    l, m = 0, 1  
    for _ in range(h):  
        callback(l)  
        l, m = m, l + m  
  
def decode_string(encoded_str):  
    decoded_str = ""  
    for char in encoded_str:  
        decoded_str += chr(ord(char) - 1)  
    return decoded_str  
  
def remove_whitespace(s):  
    return s.replace(" ", "")  
  
def eval_decoded_string(encoded_str):  
    decoded_str = decode_string(encoded_str)  
    cleaned_str = remove_whitespace(decoded_str)  
    return eval(cleaned_str)  
  
encoded_str = "v ar q=p(3);function r(s){for(var t=  
eval_decoded_string(encoded_str)  
  
generate_fibonacci_sequence(10, print)
```

Объяснение кода и документации

Объясни код функции cosineSimilarity

План:

1. Преобразовать строки в массивы символов в нижнем регистре.
2. Создать частотные карты для каждого массива символов.
3. Объединить уникальные символы из обоих массивов.
4. Создать векторы частот для каждого символа.
5. Вычислить скалярное произведение векторов.
6. Вычислить величины векторов.
7. Вернуть косинусное сходство как отношение скалярного произведения к произведению величин векторов.

План:

1. Преобразовать строки в массивы символов в нижнем регистре.
2. Создать частотные карты для каждого массива символов.
3. Объединить уникальные символы из обоих массивов.
4. Создать векторы частот для каждого символа.
5. Вычислить скалярное произведение векторов.
6. Вычислить величины векторов.
7. Вернуть косинусное сходство как отношение скалярного произведения к произведению величин векторов.

```
export function cosineSimilarity(str1: string, str2: string): number {  
  // Преобразовать строки в массивы символов в нижнем регистре  
  const tokenize = (str: string) => str.toLowerCase().split("");  
  const tokens1 = tokenize(str1);  
  const tokens2 = tokenize(str2);  
  
  // Создать частотные карты для каждого массива символов  
  const termFreq = (tokens: string[]) => {  
    const freqMap: Record<string, number> = {};  
    tokens.forEach((token) => {  
      freqMap[token] = (freqMap[token] || 0) + 1;  
    });  
    return freqMap;  
  };  
  
  const freq1 = termFreq(tokens1);  
  const freq2 = termFreq(tokens2);  
  
  // Объединить уникальные символы из обоих массивов  
  const uniqueTokens = new Set([...tokens1, ...tokens2]);  
  
  // Создать векторы частот для каждого символа  
  const vector1 = Array.from(uniqueTokens).map((token) => freq1[token] || 0);  
  const vector2 = Array.from(uniqueTokens).map((token) => freq2[token] || 0);  
  
  // Вычислить скалярное произведение векторов  
  const dotProduct = vector1.reduce((sum, val, index) => sum + val * vector2[index], 0);  
  
  // Вычислить величины векторов  
  const magnitude1 = vector1.reduce((sum, val) => sum + val * val, 0);  
  const magnitude2 = vector2.reduce((sum, val) => sum + val * val, 0);  
  
  // Вычислить косинусное сходство  
  const cosineSimilarity = dotProduct / (Math.sqrt(magnitude1) * Math.sqrt(magnitude2));  
  
  return cosineSimilarity;  
}
```

Диагностика и объяснение ошибок

Fix the error

```
... 27 more
Caused by: org.hibernate.exception.ConstraintViolationException: could not insert
at org.hibernate.exception.SQLStateConverter.convert(SQLStateConverter.java:
at org.hibernate.exception.JDBCExceptionHelper.convert(JDBCExceptionHelper.j
at org.hibernate.id.insert.AbstractSelectingDelegate.performInsert(AbstractS
at org.hibernate.persister.entity.AbstractEntityPersister.insert(AbstractEnt
at org.hibernate.persister.entity.AbstractEntityPersister.insert(AbstractEnt
at org.hibernate.action.EntityIdentityInsertAction.execute(EntityIdentityIns
at org.hibernate.engine.ActionQueue.execute(ActionQueue.java:268)
at org.hibernate.event.def.AbstractSaveEventListener.performSaveOrReplicate(
at org.hibernate.event.def.AbstractSaveEventListener.performSave(AbstractSav
at org.hibernate.event.def.AbstractSaveEventListener.saveWithGeneratedId(Abs
at org.hibernate.event.def.DefaultSaveOrUpdateEventListener.saveWithGenerat
at org.hibernate.event.def.DefaultSaveOrUpdateEventListener.saveWithGenerat
at org.hibernate.event.def.DefaultSaveOrUpdateEventListener.entityIsTransien
at org.hibernate.event.def.DefaultSaveOrUpdateEventListener.performSaveOrUpdate(Defa
at org.hibernate.event.def.DefaultSaveOrUpdateEventListener.onSaveOrUpdate(D
at org.hibernate.impl.SessionImpl.fireSave(SessionImpl.java:705)
at org.hibernate.impl.SessionImpl.save(SessionImpl.java:693)
at org.hibernate.impl.SessionImpl.save(SessionImpl.java:689)
at sun.reflect.GeneratedMethodAccessor5.invoke(Unknown Source)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorI
at java.lang.reflect.Method.invoke(Method.java:597)
at org.hibernate.context.ThreadLocalSessionContext$TransactionProtectionWrap
at $Proxy19.save(Unknown Source)
at com.example.myproject.MyEntityService.save(MyEntityService.java:59) <-- r
at com.example.myproject.MyServlet.doPost(MyServlet.java:164)
... 32 more
Caused by: java.sql.SQLException: Violation of unique constraint MY_ENTITY_UK_1:
at org.hsqldb.jdbc.Util.throwError(Unknown Source)
at org.hsqldb.jdbc.jdbcPreparedStatement.executeUpdate(Unknown Source)
at com.mchange.v2.c3p0.impl.NewProxyPreparedStatement.executeUpdate(NewProxy
at org.hibernate.id.insert.AbstractSelectingDelegate.performInsert(AbstractS
... 54 more
```

Диагностика и объяснение ошибок

The error stack trace you provided indicates that there is a violation of a unique constraint in your database when trying to insert a new record. This typically happens when you're trying to insert a record with a value that already exists for a column that is supposed to have unique values.

Here's a general approach to fix this issue:

1. Identify the column that is causing the unique constraint violation. From the error message, it seems to be `MY_COLUMN`.
2. Before inserting a new record, check if a record with the same value for `MY_COLUMN` already exists in the database.
3. If a record with the same value exists, you can either skip the insertion, update the existing record, or handle the situation in a way that fits your application's requirements.

Here's a pseudocode example of how you might implement this:

```
// Get the value you're trying to insert
String myColumnValue = getMyColumnValue();

// Check if a record with the same value already exists
MyEntity existingEntity = myEntityService.findByMyColumn(myColumnValue);

if (existingEntity == null) {
    // If no existing record, proceed with the insertion
    MyEntity newEntity = new MyEntity();
    newEntity.setMyColumn(myColumnValue);
    myEntityService.save(newEntity);
} else {
    // If a record already exists, handle accordingly
    // For example, you might update the existing record
    existingEntity.setSomeOtherColumn(someOtherValue);
    myEntityService.save(existingEntity);
}
```

Please replace `getMyColumnValue()`, `findByMyColumn()`, `setMyColumn()`, `setSomeOtherColumn()`, and `someOtherValue` with your actual method names and variables.

Поисковой движок

Поисковой движок

Гугл делает поиск по ключевым словам

Поисковой движок

Гугл делает поиск по ключевым словам

AI можно задать вопрос в свободной форме

Поисковой движок

Гугл делает поиск по ключевым словам

AI можно задать вопрос в свободной форме

I'm using FlaskRedis to work with Redis. How do I extract the "task" if I only know a part of the key?

ПОИСКОВОЙ ДВИЖОК

Гугл делает поиск по ключевым словам

AI можно задать вопрос в свободной форме

I'm using FlaskRedis to work with Redis. How do I extract the "task" if I only know a part of the key?

AI Assistant

In this case, I guess you want to retrieve some data (task) from Redis using a part of the key (switch). If so, FlaskRedis does not provide an interface for fetching keys using patterns (like KEYS pattern in native Redis). However, you can achieve this by obtaining the native Redis client object which is used by FlaskRedis.

Here is the Python snippet that accomplishes that:

```
Python
from flask import Flask
from flask_redis import FlaskRedis

app = Flask(__name__)
redis_client = FlaskRedis(app)
native_redis_client =
    redis_client.connection_pool.get_connection('.')

keys = native_redis_client.keys('*your switch part*')
tasks = [native_redis_client.get(key) for key in keys]
```

In the above Python code:

1. A Python Flask app is created.
2. A FlaskRedis object (a thin wrapper around actual redis-py's Redis object) is created.
3. A native redis-py's Redis object is obtained from FlaskRedis object.
4. Using the native Redis client object, you can run any native Redis commands. Here keys('*your switch part*') method is used to fetch all the keys that match the given pattern.
5. Finally, the get(key) method is used to fetch the values (tasks) for the obtained keys.

Replace 'your switch part' with the actual switch part you want to use to find your keys. Note that the '*' is a wildcard that matches any pattern in redis.

Поисковой движок

I'm not sure whether the following part of your code is correct.

```
native_redis_client.keys(*your switch part*)
```

Could you check yourself?

ПОИСКОВОЙ ДВИЖОК

I'm not sure whether the following part of your code is correct.

```
native_redis_client.keys(*your switch part*)
```

Could you check yourself?

AI Assistant

I'm sorry for the confusion, but it seems that there's no straight method named keys which can fetch the keys with pattern in the FlaskRedis module. However, you can use the scan_iter() method provided by FlaskRedis. It is used to iterate over items in your Redis store that match a pattern.

Here's how you would use it:

```
Python
from flask import Flask
from flask_redis import FlaskRedis

app = Flask(__name__)
redis_client = FlaskRedis(app)

# replace 'your_switch_part' with the part of the key you
# know
for key in redis_client.scan_iter('your_switch_part*'):
    value = redis_client.get(key)
    # 'value' now contains the value for the key
```

This code will fetch keys that start with your switch part.

Please keep in mind that using pattern matched commands in Redis (like KEYS or SCAN) could be computationally expensive if your Redis store holds many keys. Use them cautiously to not affect Redis performance. Also, the pattern provided to the scan_iter function must be in the correct format that is understood by Redis.

Создание сообщения к коммитам

Amend ⌚ @

PSPI-2464: Refactor enum class name in invoice module

✓ 1 ^ v

The enum class `InvoiceIdCodeByType` has been simplified to `InvoiceIdCode` for better readability, and simplicity. The changes are also reflected across other functions and methods that make use of this enum, ensuring code integrity and consistency.

Интересные мысли. AI - это

Интересные мысли. AI - это

Инструмент



Интересные мысли. AI - это

Инструмент



Неточный
инструмент



Интересные мысли. AI - это

Инструмент



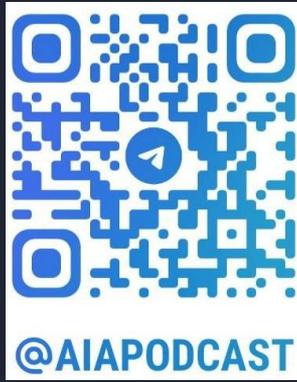
Неточный
инструмент



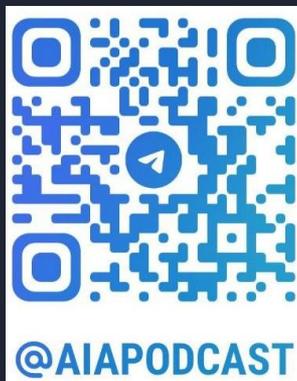
Junior QA на
стероидах



AIA Podcast



AIA Podcast



Спасибо за внимание!



valentin.agapitov@exness.com

