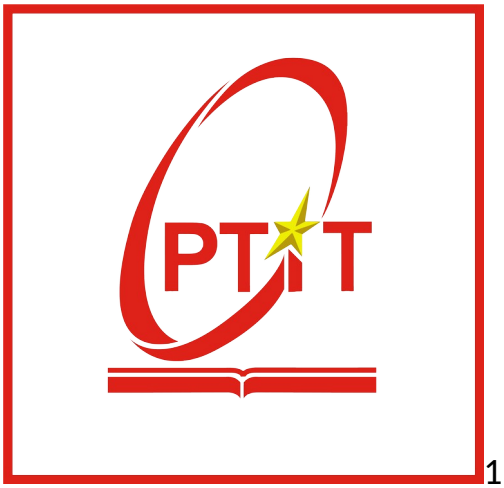
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG



1

Báo cáo hàng tuần

Môn học: Thực tập cơ sở

Giảng viên: Kim Ngọc Bách

Họ và tên: Nguyễn Hữu Phúc

Mã SV: B22DCAT224

Lớp: E22CQCN04-B

Báo cáo tuần 6

Sửa lại file code để cho chuyên nghiệp hơn

1. Cấu hình file .env

```
# MONGODB DB
MONGO_URI=mongodb://nguyenphuc:123456@localhost:27017
MONGO_DB_NAME=Glamira
```

2. Cấu hình database config

```
import os
from dataclasses import dataclass

from dotenv import load_dotenv
from typing import Dict


class DatabaseConfig():
    def validate(self) -> None:
        for key, value in self.__dict__.items():
            if value is None:
                raise ValueError(f"----------Missing config for
{key}-------------")

@dataclass
class MongoDBConfig(DatabaseConfig):
    uri : str
    db_name : str

def get_database_config() -> Dict[str,DatabaseConfig]:
    load_dotenv()
    config = {
        "mongodb" :  MongoDBConfig(
            uri = os.getenv("MONGO_URI"),
            db_name = os.getenv("MONGO_DB_NAME")
        )
    }

    for db, setting in config.items():
        print(type(setting))
        setting.validate()

    return config
```

```
db_config = get_database_config()
print(db_config)
```

3. Cấu hình mongodb_connect

```
from pymongo import MongoClient
from pymongo.errors import ConnectionFailure
from config.database_config import get_database_config
from database.schema_manager import create_mongodb_schema,
validate_mongodb_schema


class MongoDBConnect:
    def __init__(self, mongo_uri, db_name):
        self.mongo_uri = mongo_uri
        self.db_name = db_name
        self.client = None
        self.db = None

    def connect(self):
        try:
            self.client = MongoClient(self.mongo_uri)
            self.client.server_info()  # Test connection
            self.db = self.client[self.db_name]
            print(f"Connected to MongoDB: {self.db_name}")
            return self.db
        except ConnectionFailure as e:
            raise Exception(f"Failed to connect to MongoDB:
{e}") from e

    def close(self):
        if self.client:
            self.client.close()
            print("MongoDB connection closed")

    def __enter__(self):
        self.connect()
        return self

    def __exit__(self, exc_type, exc_val, exc_tb):
        self.close()


def query_product_views(db):
```

```
    # Query to retrieve specific fields, limited to 5 documents
    projection = {
        "_id": 1,
        "time_stamp": 1,
        "current_url": 1,
        "referrer_url": 1,
        "collection": 1,
        "product_id": 1,
        "option": 1
    }
    documents = db.summary.find({}, projection).limit(5)
    return list(documents)


def main():
    configMongo = get_database_config()
    with MongoDBConnect(configMongo["mongodb"].uri,
configMongo["mongodb"].db_name) as mongo_client:
        db = mongo_client.db
        # Ensure schema exists and validate
        create_mongodb_schema(db)
        validate_mongodb_schema(db)

        # Query and display documents
        documents = query_product_views(db)
        print("Retrieved documents:")
        for doc in documents:
            print(doc)


if __name__ == "__main__":
    main()
```

4. Cấu hình schema_manager

```
from pymongo import MongoClient
from pymongo.errors import CollectionInvalid

def create_mongodb_schema(db):
    # Check if collection exists; create it with schema only if
it doesn't
    if "summary" not in db.list_collection_names():
        try:
            db.create_collection("summary", validator={
                "$jsonSchema": {
```

```
                    "bsonType": "object",
                    "required": ["_id", "time_stamp",
"current_url", "collection", "product_id"],
                    "properties": {
                        "_id": {"bsonType": ["objectId",
"string"]},
                        "time_stamp": {"bsonType": ["int",
"long"]},
                        "ip": {"bsonType": ["string", "null"]},
                        "user_agent": {"bsonType": ["string",
"null"]},
                        "resolution": {"bsonType": ["string",
"null"]},
                        "user_id_db": {"bsonType": ["string",
"null"]},
                        "device_id": {"bsonType": ["string",
"null"]},
                        "api_version": {"bsonType": ["string",
"null"]},
                        "store_id": {"bsonType": ["string",
"null"]},
                        "local_time": {"bsonType": ["string",
"null"]},
                        "show_recommendation": {"bsonType":
["string", "bool", "null"]},
                        "current_url": {"bsonType": "string"},
                        "referrer_url": {"bsonType": ["string",
"null"]},
                        "email_address": {"bsonType":
["string", "null"]},
                        "recommendation": {"bsonType": ["bool",
"null"]},
                        "utm_source": {"bsonType": ["bool",
"string", "null"]},
                        "utm_medium": {"bsonType": ["bool",
"string", "null"]},
                        "collection": {"bsonType": "string"},
                        "product_id": {"bsonType": "string"},
                        "option": {
                            "bsonType": ["array", "null"],
                            "items": {
                                "bsonType": "object",
                                "properties": {
```

```
                                    "option_label":
{"bsonType": ["string", "null"]},
                                    "option_id": {"bsonType":
["string", "null"]},
                                    "value_label": {"bsonType":
["string", "null"]},
                                    "value_id": {"bsonType":
["string", "null"]}
                                }
                            }
                        }
                    }
                }
            })
            # Create an index on product_id
            db.summary.create_index("product_id")
            print("Created summary collection with schema")
        except CollectionInvalid:
            print("summary collection already exists")
    else:
        print("summary collection already exists, skipping
creation")

def validate_mongodb_schema(db):
    collections = db.list_collection_names()
    print("Collections:", collections)
    if "summary" not in collections:
        raise ValueError("Missing summary collection in
MongoDB")
    # Check for documents, but don't fail if empty
    document_count = db.summary.count_documents({})
    if document_count == 0:
        print("Warning: summary collection is empty")
    else:
        print(f"summary collection contains {document_count}
documents")
    print("Validated schema for summary collection")
```

5. Viết file main

```
from database.mongodb_connect import MongoDBConnect,
query_product_views
from database.schema_manager import create_mongodb_schema,
validate_mongodb_schema
```

```python
from config.database_config import get_database_config


def main():
    configMongo = get_database_config()
    with MongoDBConnect(configMongo["mongodb"].uri,
configMongo["mongodb"].db_name) as mongo_client:
        db = mongo_client.db
        # Ensure schema exists and validate
        create_mongodb_schema(db)
        validate_mongodb_schema(db)

        # Query and display documents
        documents = query_product_views(db)
        print("Retrieved documents:")
        for doc in documents:
            print(doc)


if __name__ == "__main__":
    main()
```

6. Kết quả

```
Connected to MongoDB: Glamira
summary collection already exists, skipping creation
Collections: ['summary', 'ProductViews']
summary collection contains 41432473 documents
Validated schema for summary collection
Retrieved documents:
{'_id': ObjectId('5ed8cb2bc671fc36b74653ad'), 'time_stamp': 1591266092, 'current_url': 'https://www.glamira.fr/glamira-pendant-viktor.html?alloy=yellow-375', 'referrer_ur
{'_id': ObjectId('5ed8cb2b4c9ffc36e828fabe'), 'time_stamp': 1591266092, 'current_url': 'https://www.glamira.com.au/customcheckout/onepage/payment/', 'referrer_url': 'http
{'_id': ObjectId('5ed8cb2cfeae1f377811165c'), 'time_stamp': 1591266092, 'current_url': 'https://www.glamira.es/alianzas/?gclid=CjwKCAjwt-L2BRA_EiwAacX32XIKXImV_rFgP1XrUgK
{'_id': ObjectId('5ed8cb2a56b3dd3742261b8c'), 'time_stamp': 1591266091, 'current_url': 'https://www.glamira.com.au/glamira-diamonds-ohrstecker-louisa-skug100735.html?allo
{'_id': ObjectId('5ed8cb2a796d133700cefa27'), 'time_stamp': 1591266091, 'current_url': 'https://www.glamira.bg/', 'referrer_url': '', 'collection': 'view_home_page'}
MongoDB connection closed

Process finished with exit code 0
```