**Pontifícia Universidade Católica do Rio Grande do Sul**

**Verificação e Validação de Software**

**Engenharia de Software**

**Carolina Ferreira, Felipe Freitas, Mateus Caçabuena e Murilo Kasperbauer**

***Relatório do Trabalho 2***

**Porto Alegre**

**2024**

**Introdução**

Neste relatório, iremos abordar a aplicação de testes em um sistema de informação de complexidade média, denominado **Veículos via Montadora**. Este sistema consiste no backend de um projeto desenvolvido em Python, utilizando o framework FastAPI. O banco de dados utilizado é o MongoDB, e a gerência de dependências é feita através do Poetry. Além disso, o sistema é executado em cima de uma função AWS Lambda, garantindo escalabilidade e eficiência.

**Descrição do Sistema**

O sistema **Veículos via Montadora** tem como objetivo gerenciar e facilitar o cadastro, consulta e atualização de informações sobre veículos, através de uma interface de programação de aplicações (API) robusta e eficiente. Dentre as funcionalidades principais, destacam-se a gestão de dados de veículos e montadoras, integração com serviços externos e a aplicação de regras de negócio específicas do domínio automotivo.

**Repositório de Código**

O código-fonte do sistema está disponível em um repositório público, o qual pode ser acessado através do link: [Repositório Veículos via Montadora](https://tools.ages.pucrs.br/veiculos-via-montadora/backend" \t "_new)

**Linguagem de Programação**

O sistema é desenvolvido em Python, utilizando o framework FastAPI, conhecido por sua performance e simplicidade na construção de APIs RESTful.

**Instruções de Compilação e Execução**

Para compilar e executar o sistema, é necessário seguir os seguintes passos:

1. **Requisitos**:
   * Python ^3.9 (Recomendado: 3.10.4)
   * Docker
2. **Instalação do Python**:
   * Verifique a versão instalada do Python:

bash

Copiar código

python --version

* + Caso necessário, instale a versão correta do Python a partir do [site oficial](https://www.python.org/downloads/" \t "_new).

1. **Instalação do Docker**:
   * Siga as instruções de instalação no site oficial do Docker.
2. **Configuração do Ambiente de Desenvolvimento**:
   * Clone o repositório do projeto:

bash

Copiar código

git clone https://tools.ages.pucrs.br/veiculos-via-montadora/backend.git

* + Acesse a pasta do projeto:

bash

Copiar código

cd backend

1. **Iniciar o Servidor API com FastAPI**:
   * Instale o gerenciador de dependências Poetry na versão 1.4.1:

bash

Copiar código

pip install "poetry==1.4.1"

* + Configure o Poetry para criar o ambiente virtual dentro da pasta do projeto:

bash

Copiar código

poetry config virtualenvs.in-project true

* + Instale as dependências do projeto:

bash

Copiar código

poetry install

* + Inicie o servidor localmente:

bash

Copiar código

poetry run uvicorn app.main:app --reload

* + O servidor estará disponível em https://localhost. Para que o HTTPS funcione, é necessário colocar um certificado e uma chave privada válida dentro do diretório certs, conforme as instruções no arquivo certs/README.md.

1. **Iniciar o Servidor API com Docker**:
   * Para fazer o deploy do projeto utilizando Docker, o endpoint da API na máquina host será https://localhost:8080.
2. **Configuração do Banco de Dados MongoDB**:
   * Execute o comando abaixo para iniciar o container do MongoDB:

bash

Copiar código

docker compose up mongodb

* + Para parar o container, execute:

bash

Copiar código

docker compose down

Com esses passos, o ambiente estará preparado para a execução dos testes e a análise crítica da qualidade dos mesmos, conforme os objetivos estabelecidos para este trabalho.