

- RM559064 - Pedro Henrique dos Santos 2TDSPJ
- RM556182 - Vinícius de Oliveira Coutinho 2TDSPV
- RM557992 - Thiago Thomaz Sales Conceição 2TDSPJ

Link GitHub

<https://github.com/MottuChallenge/Devops.git>

Link Video no youtube

<https://youtu.be/YxOf4qUlqF4>

Link do projeto no Azure DevOps

<https://dev.azure.com/MottuGridChallenge/Sprint%204%20-%20Azure%20DevOps>

Descrição da solução

Este projeto implementa um sistema de **gestão de pátio (Yard)**, **setores (Sector)** e **vagas (Spots)** para organização e alocação de motos.

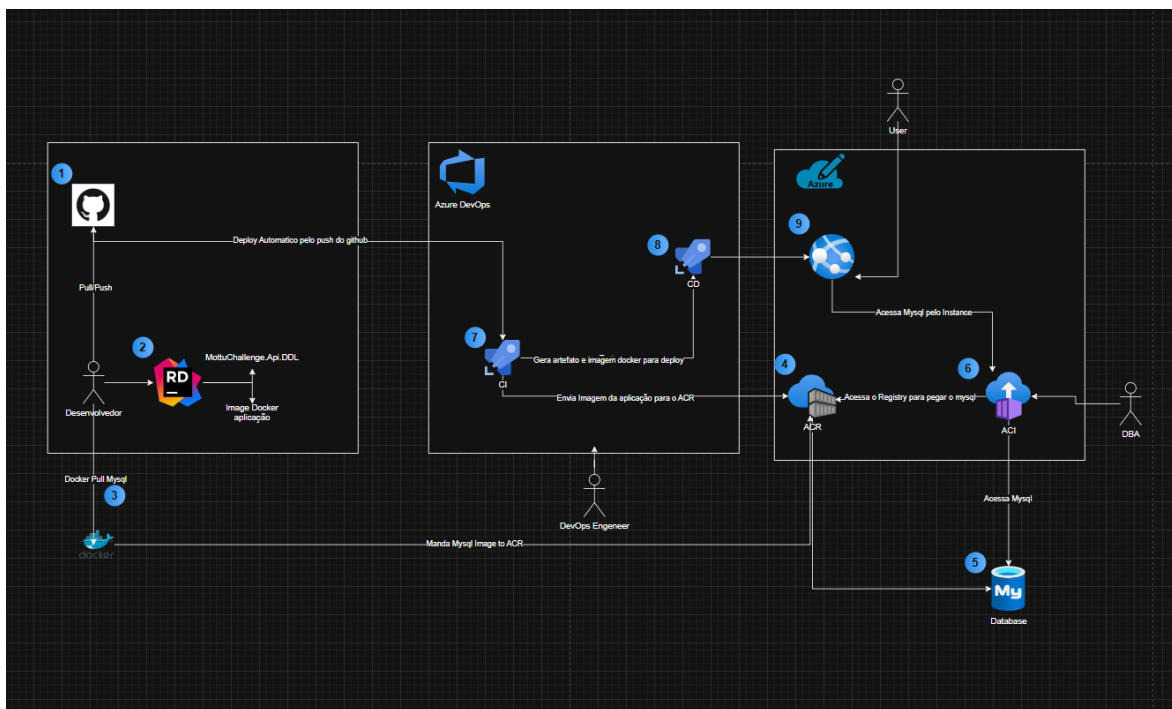
O objetivo é permitir que filiais da Mottu consigam estruturar seus pátios em setores e, automaticamente, gerar as vagas disponíveis para as motos.

Desenvolveremos uma API RESTful para registrar, atualizar e consultar a localização das motos em tempo real nos pátios da Mottu. O sistema permitirá:

- Cadastro e atualização de motos, pátios, seções e filiais.
- Consulta rápida da localização de cada moto.
- Integração com cameras e IA para verificar se um setor esta cheio e com base nisso aconselhar a criar outros setores ou mudar de patio as motos que chegaram com esse modelo especifico, tambem para localizar uma moto especifica
- Facilidade de integração com outros sistemas internos da Mottu.
- Tera um sistema alerta onde quando uma moto estiver perto de sua revisão avisara a um prestador de serviço da mottu para adicionar essa moto a um setor de revisão

Essa solução trará mais agilidade, precisão e controle para a operação, reduzindo erros e otimizando o uso dos recursos.

Diagrama da Arquitetura + Fluxo CI/CD



Detalhamento dos componentes

	A	B	C	D
1	Nome do componente	Tipo	Descrição funcional	Tecnologia/Ferramenta
2	GitHub	SCM (Source Code Management)	Armazena e versiona o código-fonte da aplicação, acionando o pipeline automaticamente via push.	GitHub
3	Desenvolvedor	Usuário	Responsável por realizar alterações no código, criar Dockerfile e fazer push/pull no repositório.	IDE (Rider), Git
4	Docker Local	Container Engine	Responsável por gerar e testar a imagem Docker localmente antes do envio para o pipeline.	Docker Desktop
5	Azure DevOps (Pipeline CI)	Orquestrador CI	Compila o código, executa testes e gera a imagem Docker da aplicação. Em seguida, envia a imagem para o Azure Container Registry (ACR).	Azure DevOps Pipelines
6	Azure DevOps (Pipeline CD)	Orquestrador CD	Recupera a imagem Docker do ACR e realiza o deploy automático da aplicação no Azure Container Instance (ACI).	Azure DevOps Pipelines
7	Azure Container Registry (ACR)	Registro de imagens	Armazena as imagens Docker da aplicação e do MySQL, permitindo que o ACI as consuma.	Azure Container Registry
8	Azure Container Instance (ACI)	Ambiente de execução	Executa os containers da aplicação e do MySQL em instâncias isoladas na nuvem.	Azure Container Instance
9	MySQL Database	Banco de Dados	Banco relacional utilizado pela aplicação, acessado a partir do container ACI.	MySQL
10	Usuário (Cliente)	Usuário final	Interage com a aplicação implantada no ambiente em nuvem.	—
11	DBA	Usuário administrativo	Responsável por gerenciar e monitorar o banco de dados MySQL hospedado.	MySQL Workbench / Azure Portal
12	Azure Web App / Endpoint	Serviço Web	Fornecer acesso ao serviço implantado (API ou site) para usuários e sistemas externos.	Azure App Service / ACI endpoint
13	DevOps Engineer	Profissional de DevOps	Cria e mantém os pipelines, configura o ACR e o ambiente de deploy no Azure.	Azure DevOps / Azure CLI
14				
15				
16				
17				