FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA

PRIMEIRA SPRINT DO CHALLENGE

MATÉRIA: DOMAIN DRIVEN DESIGN USING JAVA

Professor: Leonardo Gasparini Romão

Nome do Grupo: RailAI

Bruno Eleuterio RM: 560150

Júlia Oizumi RM: 559303

Marcello Coelho RM: 559778

Turma: 1TDSPB

Indíce

[Descritivo do Projeto 4](#__RefHeading___Toc280_506570180)

[Descrição do fluxo e regras de negócio 4](#__RefHeading___Toc282_506570180)

[Entidades 5](#__RefHeading___Toc284_506570180)

[Coordinate 5](#__RefHeading___Toc286_506570180)

[Location 5](#__RefHeading___Toc288_506570180)

[PointOfInterest 5](#__RefHeading___Toc290_506570180)

[WeatherData 5](#__RefHeading___Toc292_506570180)

[Camera 5](#__RefHeading___Toc294_506570180)

[Route 6](#__RefHeading___Toc296_506570180)

[Train 6](#__RefHeading___Toc298_506570180)

[Permission 6](#__RefHeading___Toc300_506570180)

[Role 6](#__RefHeading___Toc302_506570180)

[User 6](#__RefHeading___Toc304_506570180)

[MaintenanceTask 6](#__RefHeading___Toc306_506570180)

[VideoRecording 6](#__RefHeading___Toc308_506570180)

[TrackFailure 6](#__RefHeading___Toc312_506570180)

[TrackFailureType 6](#__RefHeading___Toc314_506570180)

[Serviços 7](#__RefHeading___Toc284_506570180_Copy_1)

[WeatherService 7](#__RefHeading___Toc314_506570180_Copy_1)

[VideoAnalysisService 7](#__RefHeading___Toc314_506570180_Copy_2)

[CameraService 7](#__RefHeading___Toc314_506570180_Copy_3)

[MaintenanceTaskService 7](#__RefHeading___Toc314_506570180_Copy_2_C)

[AlertService 7](#__RefHeading___Toc314_506570180_Copy_2_1)

[UserService 7](#__RefHeading___Toc314_506570180_Copy_2_2)

[RoleService 7](#__RefHeading___Toc314_506570180_Copy_2_3)

# Descritivo do Projeto

Para combater a superlotação gerada pelas avarias nos trilhos e a demora nos consertos delas, elaboramos um projeto para implantação de câmeras nos trens coletando imagens, armazenando essas imagens, realizando análise prévia por Inteligência Artificial e análise posterior por analistas e técnicos, de modo a tornar a identificação dos problemas rápida e eficiente, como também a manutenção dos trilhos e o conserto de falhas.

O desenvolvimento da inteligência artificial se dará através do desenvolvimento da inteligência artificial em Python, tendo em vista todo o ecossistema já existente na linguagem. Em Java, a ideia é desenvolver a API que atuará como interface entre a unidade de processamento de imagens, o sistema embarcado nos trens e o usuário do sistema, através de um servidor HTTP.

Desta forma, optamos por neste entregável, realizar de forma inicial a modelagem de classes das entidades que serão controladas por este servidor.

## Descrição do fluxo e regras de negócio

A aplicação de câmeras será realizada durante as viagens dos trens, com o objetivo de aumentar o monitoramento minucioso dos trilhos, armazenando também as imagens para que possam ser realizados levantamento de dados, análises e previsão de falhas e avarias de todas as formas como por exemplo: ferrugem, desgaste, quebras etc.

Em primeiro momento a análise das imagens coletadas será realizada por meio de Inteligência Artificial. Ao identificar uma possível falha através das imagens, serão coletados dados de localização e meteorologia, para que além de localizada e corrigida, a concessionária seja também capaz de traçar padrões e atacar diretamente a causa do problema.

Além da captura e armazenamento de imagens e da análise realizada pela Inteligência Artificial, o sistema também prevê o envio de alertas para os usuários, para que todos sejam alertados uma vez que uma possível falha seja encontrada e para que a ordem de manutenção seja emitida conforme a necessidade.

# Entidades

## Coordinate

Entidade para controle de posição das análises. Utilizada para definir a posição de GPS e facilitar a manutenção.

## Location

Entidade que acumula a posição, informação meteorológica e linha (para evitar confusão em locais onde duas linhas se conectam).

## PointOfInterest

Entidade que estende *Location*, permite a criação de pontos específicos no trajeto que podem ou não acumular ocorrências a fim de encontrar padrões.

## WeatherData

Entidade para obtenção de dados meteorológicos de um certo local.

## Camera

Entidade da câmera que obtém as imagens para análise.

## Route

Entidade que determina a linha onde o trem está transitando.

## Train

Entidade para o trem que carrega a câmera.

## Permission

Entidade de permissões para cargos.

## Role

Entidade para cargos de usuário.

## User

Entidade de usuários.

## MaintenanceTask

Entidade para ordens de manutenção de um ou mais defeitos nos trilhos.

## VideoRecording

Entidade correspondente a gravação do vídeo a ser analisado.

## TrackFailure

Entidade para os possíveis defeitos encontrados durante a análise.

## TrackFailureType

Enum para classificação do tipo de defeito.

# Serviços

## WeatherService

Serviço para obtenção e atualização de dados meteorológicos e uma dada localização ou ponto de interesse.

## VideoAnalysisService

Serviço para processamento de vídeo e sinalização de potenciais defeitos encontrados.

## CameraService

Serviço para controle da câmera de gravação.

## MaintenanceTaskService

Serviço para controle das ordens de serviço.

## AlertService

Serviço para métodos relacionados a alertas enviados.

## UserService

Serviço para métodos relacionados a usuários e suas interações com o sistema.

## RoleService

Serviço para métodos relacionados aos cargos e permissões.

# Diagrama UML

O diagrama estará também disponível como arquivo PDF anexo ao relatório.

