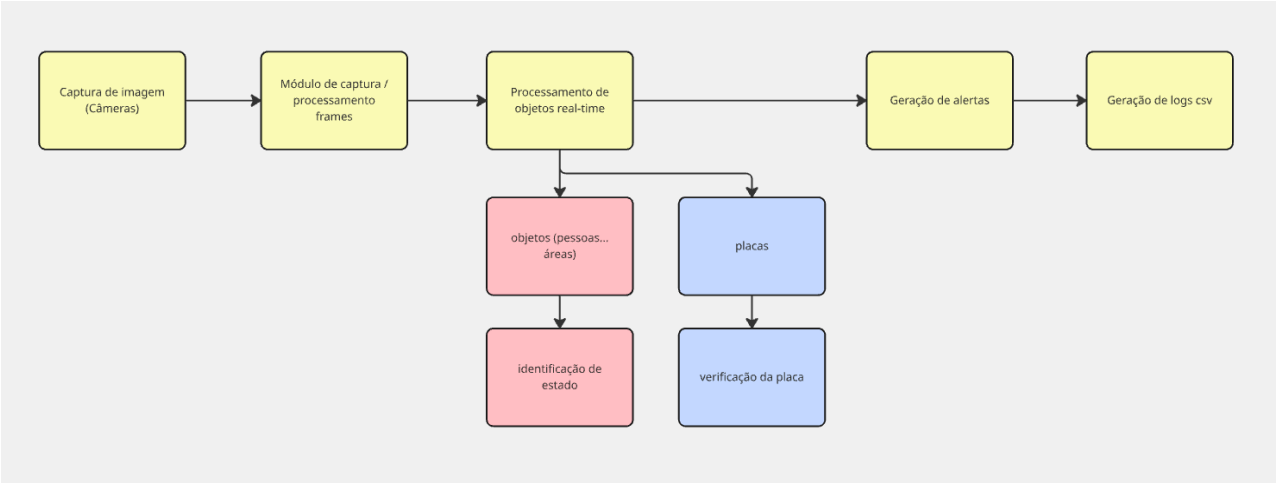
Desafio Capiva

Sirius Zevallos

Fluxo de Funcionamento

1. **Captura de Imagens por Câmeras**
    - o posicionamento das câmeras precisa cobrir os canteiros e as baias por cima, e a portaria/entrada pela frente
    - o vídeo das câmera é transmitido para o servidor pelo protocolo RTSP/RTP
2. **Módulo de Captura**
    - Coleta dos frames; o input do vídeo é realizado por OpenCV e FFmpeg
    - Frames são enviados para uma pipeline de análise
3. **Processamento Real-Time**
    - um modelo YOLO v8 detecta os objetos
    - as detecções de objetos (com exceção placa) são averiguadas nas regras do sistema; se uma regra é desrespeitada, o objeto que a desrespeitou muda de estado para “não-seguro”
    - paralelamente, um módulo EasyOCR recebe um objeto placa, e verifica se ele foi autorizado
    - as placas autorizadas ficam em um arquivo .csv
    - uma placa não-reconhecida é alterada para o estado “não-seguro”
4. **Geração de Alertas**
    - quando um objeto se torna “não-seguro”, é gerado um alerta
    - o formato do alerta gerado depende do nível do alerta, 2 (crítico) ou 1 (menor)
    - o alerta nível 2 gera uma sirene no local, enviado pelo backend para um microcontrolador Raspberry PI
    - o alerta nível 1 gera uma notificação push através do Whatsapp integrado com o backend
5. **Geração de Logs**
    - quando um objeto se torna “não-seguro”, é gerado um log da ocorrência
    - o log é adicionado em um arquivo .csv
    - é feita uma análise de dados utilizando Pandas, visando avaliar:
        i. áreas com maior concentração de alertas;
        ii. índice de reincidência por zona;
        iii. dia da semana e hora do dia com maior frequência;
        iv. EPIs menos utilizados/respeitados;
        v. correlação entre colaboradores por área e infrações;
        vi. predição de acidentes;
        vii. entre outras métricas.



Estrutura da Solução

Nós temos os seguintes **objetos**:

- pessoas (colaboradores)
- EPIs
- máquinas pesadas
- caminhões
- áreas
- placas

Onde:

- EPIs contém:
  - capacetes
  - coletes
  - luvas
  - óculos
  - botas
- áreas contêm:
  - baias de caminhão
  - zona perigosas
  - zonas seguras

E cada um desses objetos com exceção de “áreas” pode estar no **estado**:

- seguro
- não-seguro, que contêm os níveis:
  - 2 — violação crítica
  - 1 — violação menor

Em relação às **zonas perigosas** e **zonas seguras**:

- Uma área é perigosa quando:
  - Está ocupada por pessoas sem os EPIs exigidos
  - Tem circulação indevida (ex: pedestre em zona de caminhões/máquinas)
  - Foi detectada uma violação ativa de regra

- Uma área é "segura" quando:
    - Não contêm máquinas pesadas
    - Está ocupada por pessoas corretamente equipadas/autorizadas
    - Está operando dentro dos parâmetros definidos

Em relação às **violações**; o nível 2 — violação crítica:

- É determinado por:
    - Pessoa sem capacete em área com movimentação de cargas elevadas ou guindastes
    - Pessoa dentro de zona de operação de máquinas pesadas (ex: escavadeira em movimento)
    - Pessoa em área restrita com risco elétrico ou químico
    - Dois veículos manobrando na mesma baia ao mesmo tempo
    - Pessoa entre caminhão e parede (zona de esmagamento)
- E ocasiona:
    - Disparo de alerta sonoro/visual no local
    - Disparo de alerta sonoro/visual para o time de segurança

O nível 1 — violação menor:

- É determinado por:
    - Pessoa sem colete refletivo em área com veículos.
    - Caminhão manobrando fora da baia designada, mas sem pessoas próximas.
    - Caminhão estacionado desalinhado
    - Falha de reconhecimento de placa de veículo na entrada
- E ocasiona:
    - Notificação push para o time da obra e time de segurança
    - Sugestão de inspeção manual ou auditoria

Uma violação gera uma **ocorrência** a ser enviada para um banco de dados. Uma ocorrência contêm:

- data/horário
- qual câmera detectou
- área em que ocorreu o incidente
- número de pessoas envolvidas no incidente
- número de pessoas em estado “não-seguro”
- máquinas envolvidas
- caminhões envolvidos
- regra ativada
- nível da violação (1 ou 2)
- ação tomada

Escolha de Tecnologias

1. OpenCV e FFmpeg
   A escolha do OpenCV se dá pelo seu módulo de Input/Output que tem várias funções para ler (e escrever) vídeos ou sequência de imagens. É uma biblioteca fácil de ser utilizada e que cumpre a função. Além de tudo, é compatível nativamente

com todos os outros modelos escolhidos. Já o FFmpeg é comumente usado para manejar e converter vídeos, e oferece um suporte robusto para o OpenCV. Dessa forma, utilizaremos o FFmpeg para preparar o conteúdo do stream, e o OpenCV para ler frame-a-frame.

2. YOLO v8
   A versão mais atual do YOLO foi escolhida pela sua velocidade (consegue processar 30 ou mais FPSs quando executado em uma GPU), além de ter um desempenho esperado muito alto quando bem treinado, mesmo tratando de imagens ruidosas. Além disso, a versão 8 do YOLO é focada em detecção, sendo ideal para a função de detectar objetos.
3. EasyOCR
   Foi escolhido por ter uma capacidade de leitura mais confiável em ambientes reais (como à noite ou através de lentes borradas, poeira, etc). Além disso, é altamente configurável, possui suporte em várias línguas, incluindo portuguÊs, e pode ser associado com o YOLO v8 utilizado na etapa de detecção.
4. Raspberry Pi
   Para acionar a sirene audível, que é a forma mais rápida e eficiente de alertar tanto os colaboradores quanto a equipe de segurança, é essencial contar com um dispositivo de borda simples, barato e fácil de configurar. Por conta disso, o Raspberry Pi é uma ótima escolha, já que além de acessível, ele pode ser acionado remotamente por um simples POST com o protocolo HTTPS, o que facilita sua integração com o restante do sistema.
5. WhatsApp
   A escolha do Whatsapp é devido ao fato de que ele está presente em 99% dos celulares do Brasil, de acordo com o Tecnoblog, o que praticamente garante que todos os membros da equipe da obra e de vão ter acesso imediato às notificações, sem depender de apps adicionais. A integração pode ser feita utilizando o WhatsApp Web automatizado via bibliotecas como whatsapp-web.js, e o backend geraria os alertas, com base nas detecções do sistema, disparando mensagens automáticas para grupos ou contatos específicos através dessa conexão.
6. Pandas + csv
   O combo possibilita análises reais em produção com baixo volume de dados. Dado que uma obra não é permanente, é esperado uma produção relativamente baixa de dados de ocorrência — especialmente já que acidentes não devem ser eventos frequentes. Por isso, o uso de arquivos CSV atende perfeitamente à necessidade de registro local e organizado das ocorrências. Outra vantagem é que quase todos os softwares de planilha, bancos de dados e outras ferramentas de análise de dados podem importar arquivos CSV.No mais, esses arquivos podem, futuramente, ser mantidos pela Capiva e cruzados com dados de outras obras, ampliando a capacidade preditiva e analítica da empresa.
   Quanto ao Pandas, ele permite realizar operações como análise temporal, estatísticas descritivas, limpeza, manipulação e visualização de dados — cobrindo de forma eficiente todas as necessidades analíticas.

Processamento em Tempo Real com Múltiplas Câmeras

Existem algumas alternativas para rodar com múltiplas câmeras evitando que o sistema quebre.

- **Controlar a taxa de leitura dos frames**
  Pular a exibição de alguns frames para manter uma taxa de quadros por segundo (FPS) mais baixa, garantiria que o sistema não fique sobrecarregado. Nem todos os frames precisam ser processados para garantir a eficácia do sistema, pois a movimentação no mundo real é mais lenta do que a da análise computacional.Assim sendo, reduzir a taxa padrão dos vídeos (normalmente de 30 FPS) para dois frames (o 1º e o 15º frame de cada segundo, por exemplo), já reduziria significativamente o custo computacional, mantendo uma amostragem mínima necessária para detecção de quaisquer situações.
- **Controle de Pipeline por Câmera**
  O sistema poderia ser estruturado de forma que cada câmera opere em seu próprio pipeline, ou seja, cada processo executa localmente todas as etapas: captura do vídeo, detecção de objetos, verificação de regras, geração de alertas e registro de logs, onde apenas os resultados do processo são enviados para um backend comum. Tal abordagem permite **escalabilidade horizontal**, facilitando o controle de carga e a manutenção do sistema.

Integração com Legados

Dado que o sistema gera arquivos .csv, o objetivo da integração é garantir que esses dados possam ser utilizados por sistemas legados de forma segura e simples. O CSV é um formato amplamente suportado e quase que universal, já que a maioria dos softwares de planilha, bancos de dados e ferramentas de análise (até os mais antigos) consegue importar arquivos nesse formato. Porém, sem uma especificação clara do sistema legado, é difícil definir uma estratégia única de integração. Assim, algumas das possibilidades seriam:

- **Importação direta**
  Sistemas legados que não possuem APIs podem importar os arquivos CSV manualmente ou através de rotinas internas.
- **Conversão via middleware**
  Ferramentas como MCP, n8n ou Node-RED podem ser utilizadas para monitorar uma pasta onde os arquivos CSV são gerados, ou reagir a um webhook que indique a criação de um novo arquivo. Após a detecção, o middleware pode:
    - Ler e validar o conteúdo do CSV;
    - Realizar transformações necessárias;
    - Converter os dados em comandos SQL ou outro formato compatível com o sistema legado;
    - Executar a integração de forma automatizada.

Ideias Extras

- **Ativação inteligente de câmeras com sensor de movimentos**
  Principalmente para a escalabilidade, sensores de movimento posicionados em áreas estratégicas da obra — como baías, zonas de risco ou áreas de construção — podem ser utilizados para indicar quando há necessidade de ativar a captura de

vídeo. A partir desses sinais, o sistema decide quais câmeras devem ser ativadas em tempo real e quais podem permanecer inativas, economizando processamento e largura de banda.
Outra opção seria manter esse padrão somente em áreas consideradas mais seguras, não arriscando nenhuma falta de supervisão em áreas perigosas. Também seria possível, ao invés de utilizar sensores, usar algoritmos como diferença entre frames para detectar se houve mudança significativa de um frame para outro, demonstrando movimento.

- **Reconhecimento facial como complemento**
  Integrar um sistema de reconhecimento facial tornaria o processo mais completo, pois permitiria associar infrações a colaboradores específicos, permitindo identificar padrões de reincidência e avaliar se há indícios de negligência, estresse, e/ou necessidade de reorientação, reforçando a responsabilidade e o bem estar individual no ambiente.
- **Otimização de leitura de placas, por distância**
  Como (até onde eu entendi) o EasyOCR tende a tentar ler qualquer elemento textual presente no frame, uma melhoria possível seria condicionar a ativação da leitura de placas à proximidade do veículo em relação à câmera — por exemplo, somente quando o carro estiver a uma distância inferior a 1,5 metros. Isso evitaria leituras irrelevantes de textos ao fundo, em placas de sinalização ou anúncios de ônibus, otimizando o desempenho do sistema.

Fim :)