

SafeWork

Plataforma Inteligente de Segurança e Bem-Estar no Trabalho

Relatório Final – Compliance, Quality Assurance & Tests

Integrantes:

- Angello Turano da Costa – RM 556511
- Cauã Sanches de Santana – RM 558317
- Gustavo de Souza Amorim – RM 556999

Instituição: FIAP
Ano: 2025

1. Introdução e Objetivo

O projeto SafeWork foi desenvolvido com o objetivo de criar uma plataforma integrada que fortalece a segurança, o compliance e o bem-estar dos colaboradores em ambientes de trabalho. A solução combina visão computacional, gestão operacional, indicadores de conformidade e monitoramento de bem-estar em um único ecossistema tecnológico.

A plataforma utiliza inteligência artificial para análise automática de uso de EPIs (equipamentos de proteção individual), alertas de não conformidade, dashboards gerenciais e check-ins de bem-estar, apoiando decisões rápidas e redução de riscos operacionais. Este relatório consolida a visão funcional, arquitetura, estratégia de testes, automação em Azure DevOps e documentação utilizada ao longo da Global Solution.

2. Arquitetura da Solução

A arquitetura da SafeWork foi pensada para ser modular, escalável e aderente às boas práticas de mercado. Ela é dividida em camadas bem definidas, que se comunicam por meio de APIs REST e serviços especializados.

2.1 Camadas Principais

- Camada de Front-End: responsável pela interface com o usuário (WebApp), formulários de cadastro, upload de mídias, visualização de alertas e dashboards.
- Camada de Back-End: implementada em Java Spring Boot, concentra a lógica de negócio, regras de validação, autenticação, gestão de funcionários, EPIs, alertas e check-ins de bem-estar.
- Camada de IA (Visão Computacional): responsável por processar imagens e vídeos para detectar presença ou ausência de EPIs, utilizando modelos como YOLO/OpenCV.
- Camada de Dados: banco de dados relacional (SQL) que armazena cadastros, regras, eventos, alertas, resultados de análises e registros de bem-estar.

2.2 Fluxo Resumido da Solução

1. O usuário acessa o sistema via WebApp e realiza login seguro.
2. Supervisores cadastram funcionários, setores e regras de EPIs.
3. Vídeos e imagens são enviados para análise automática de conformidade.
4. A IA de visão computacional processa as mídias e identifica possíveis não conformidades.
5. Alertas são gerados automaticamente e exibidos em telas específicas e dashboards.
6. Colaboradores podem registrar diariamente seu bem-estar (humor, fadiga, estresse).
7. Gestores acompanham tudo por meio de dashboards e relatórios de compliance.

3. Funcionalidades Principais

A solução foi estruturada em Features, cada uma representando um conjunto de funcionalidades entregues ao usuário. Abaixo estão as principais funcionalidades da plataforma SafeWork:

F01 – Autenticação e Controle de Acesso

Permite login seguro, controle de sessão, recuperação de senha e proteção de rotas internas conforme o perfil de acesso.

F02 – Cadastro e Gestão de Funcionários

Gerencia os colaboradores, com cadastro, edição, listagem, filtros e vínculo com setores da empresa.

F03 – Cadastro de EPIs e Regras de Conformidade

Registra EPIs e define regras obrigatórias por setor, servindo como base para as validações da IA.

F04 – Upload de Mídia para Análise

Permite o envio de imagens e vídeos que serão processados pela visão computacional para análise de uso de EPIs.

F05 – IA de Detecção de EPIs

Responsável por detectar a presença ou ausência de EPIs em colaboradores presentes nas mídias analisadas.

F06 – Alertas de Não Conformidade

Gera alertas automáticos quando uma não conformidade é identificada, mantendo histórico para auditoria.

F07 – Dashboard de Compliance

Apresenta indicadores de conformidade por setor, período e tipo de EPI, facilitando a tomada de decisão.

F08 – Check-in de Bem-Estar do Funcionário

Permite que colaboradores registrem diariamente seu humor, fadiga e estresse, apoiando ações preventivas.

4. Backlog – Features e Critérios de Aceite

O backlog foi estruturado no Azure Boards utilizando o processo Scrum. Cada Feature possui descrição clara, critérios de aceite e vínculo com Sprints. A seguir, um resumo dos critérios de aceite utilizados como base para os testes de QA.

F01 – Autenticação e Controle de Acesso

- Critérios de aceite definidos e cadastrados no Azure DevOps (Description + Acceptance Criteria).
- Vinculada a pelo menos um caso de teste funcional dentro do Test Plan.

F02 – Cadastro e Gestão de Funcionários

- Critérios de aceite definidos e cadastrados no Azure DevOps (Description + Acceptance Criteria).
- Vinculada a pelo menos um caso de teste funcional dentro do Test Plan.

F03 – Cadastro de EPIs e Regras de Conformidade

- Critérios de aceite definidos e cadastrados no Azure DevOps (Description + Acceptance Criteria).
- Vinculada a pelo menos um caso de teste funcional dentro do Test Plan.

F04 – Upload de Mídia para Análise

- Critérios de aceite definidos e cadastrados no Azure DevOps (Description + Acceptance Criteria).
- Vinculada a pelo menos um caso de teste funcional dentro do Test Plan.

F05 – IA de Detecção de EPIs

- Critérios de aceite definidos e cadastrados no Azure DevOps (Description + Acceptance Criteria).
- Vinculada a pelo menos um caso de teste funcional dentro do Test Plan.

F06 – Alertas de Não Conformidade

- Critérios de aceite definidos e cadastrados no Azure DevOps (Description + Acceptance Criteria).
- Vinculada a pelo menos um caso de teste funcional dentro do Test Plan.

F07 – Dashboard de Compliance

- Critérios de aceite definidos e cadastrados no Azure DevOps (Description + Acceptance Criteria).
- Vinculada a pelo menos um caso de teste funcional dentro do Test Plan.

F08 – Check-in de Bem-Estar do Funcionário

- Critérios de aceite definidos e cadastrados no Azure DevOps (Description + Acceptance Criteria).
- Vinculada a pelo menos um caso de teste funcional dentro do Test Plan.

Prints do Backlog e das Features foram gerados diretamente do Azure DevOps e anexados abaixo como evidência visual.

5. Testes – Test Plans, Execuções e Métricas

A qualidade da solução foi garantida por meio de uma bateria de testes funcionais, planejados e executados no Azure DevOps Test Plans. Foram criadas suítes de testes para cada Feature (F01 a F08), totalizando 8 casos de teste principais, todos executados com sucesso.

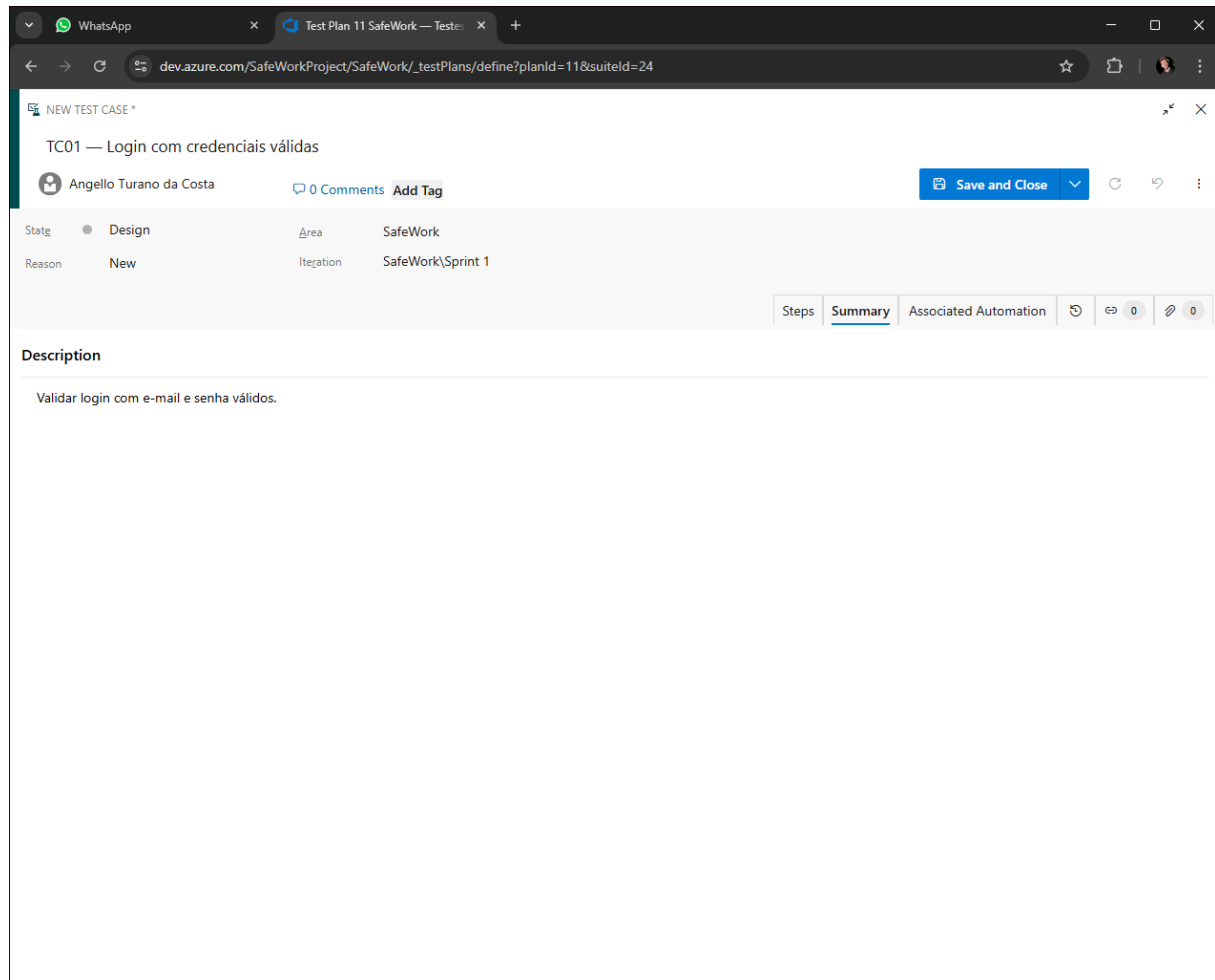
Cada caso de teste contém: objetivo (Summary), pré-requisitos, passos detalhados, resultados esperados e uso de parâmetros (Param Values), permitindo reutilização em diferentes combinações de dados. As execuções foram realizadas com o Web Runner, marcando passo a passo como Passed, e gerando relatórios automáticos de progresso.

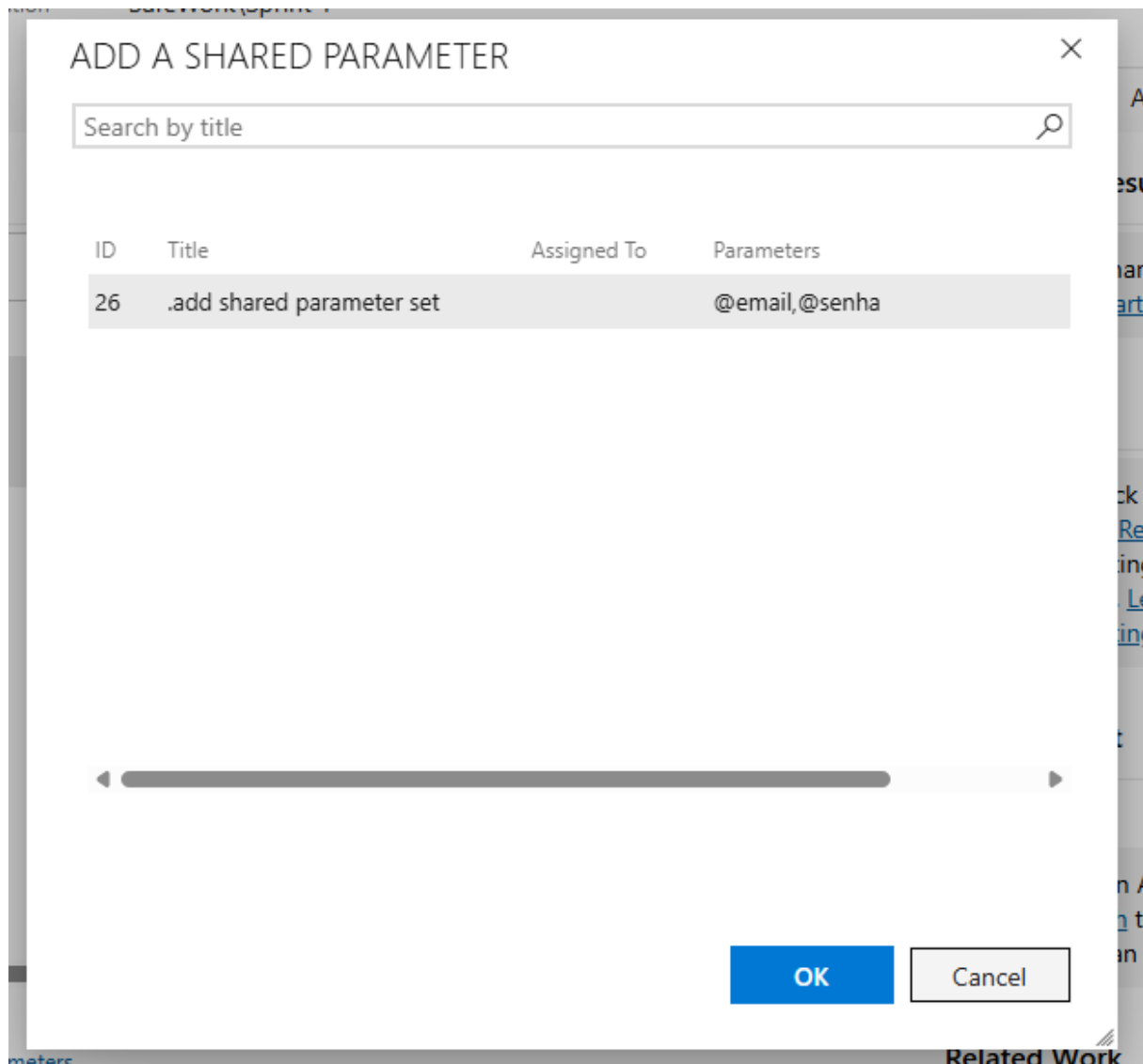
5.1 Estrutura dos Casos de Teste

- 8 Test Cases – um para cada Feature (F01 a F08).
- Uso de parâmetros como , , , , , e .
- Execução 100% Passed, com gráfico de Outcome indicando sucesso total.

5.2 Evidências Visuais dos Testes (Azure DevOps)

A seguir, alguns prints extraídos do Azure DevOps, demonstrando a configuração dos Test Plans, casos de teste e execuções.

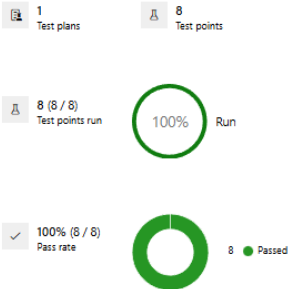


[illegible]

Progress report

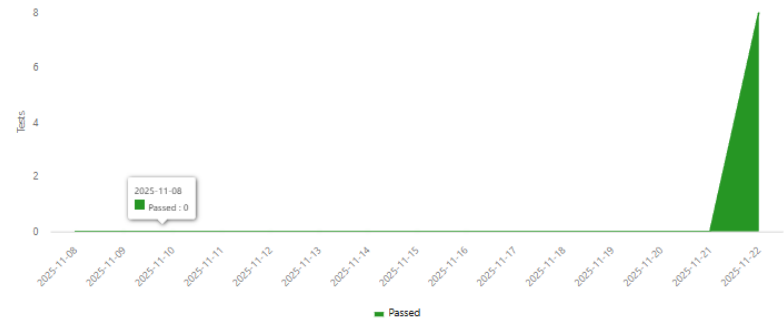
SafeWork — Testes Funcionais Test Suites Outcome Configuration Tester Priority Assigned To

Summary



Outcome trend

Last 14 Days



Details

Test plan name	Test points	Run % ↓	Passed %	Failed %	Not run count
SafeWork — Testes Funcionais	8	100	100	0	0
> F08 — Bem-Estar	1	100	100	0	0
> F07 — Dashboard	1	100	100	0	0
> F06 — Alertas	1	100	100	0	0
> F05 — IA de Detecção	1	100	100	0	0
> F04 — Upload	1	100	100	0	0
> F03 — EPis	1	100	100	0	0
> F02 — Funcionários	1	100	100	0	0
> F01 — Autenticação	1	100	100	0	0

WhatsApp

Test Plan 11 SafeWork — Teste

dev.azure.com/SafeWorkProject/SafeWork/_testPlans/define?planId=11&suiteId=24

NEW TEST CASE *

TC01 — Login com credenciais válidas

Angello Turano da Costa

0 Comments

Add Tag

Save and Close

State

Design

Area

SafeWork

Reason

New

Iteration

SafeWork\Sprint 1

Steps

Summary

Associated Automation

0

0

Steps

Recent test results

Deployment

Development

Related Work

Details

1. Acessar a página de Login

Página exibida com campos de e-mail e senha

2. Informar <email> e <senha> válidos

Sistema aceita as credenciais

3. Clicar em "Entrar"

Usuário é redirecionado ao Dashboard

Click or type here to add a step

Parameter values

Add a shared parameter set

Convert to shared parameters

Run manual tests with web runner. [Learn how to get started](#)

To track releases associated with this work item, go to [Releases](#) and turn on deployment status reporting for Boards in your pipeline's Options menu. [Learn more about deployment status reporting](#)

Add link

Link an Azure Repos [commit](#), [pull request](#) or [branch](#) to see the status of your development. You can also create a branch to get started.

Add link

[Add an existing work item as a parent](#)

Priority

S

+

Project Settings

SafeWork

General

Overview

Teams

Permissions

Notifications

Service hooks

Dashboards

Boards

Project configuration

Team configuration

GitHub connections

Pipelines

Agent pools

Parallel jobs

Settings

Test management

Service connections

XAML build services

Repos

Repositories

Artifacts

Storage

Project details

Name

SafeWork

Description

Process

Basic

Save



Project administrators

- Angello Turano da Costa**
RM556511@fap.com.br
- Cauã Sanches de Santana**
RM558317@fap.com.br
- Gustavo de Souza Amorim**
RM556999@fap.com.br
- + 1 more

Add administrator

Azure DevOps services



COMPLIANCE, QUALITY ASSURANCE & TESTS

Crie um plano de projeto Scrum, no Azure DevOps Organizations com:

- 1- **(30%)** Features, todas elas com descrição e critério de aceite documentados. Tire um "print screen" da tela com a relação de Features e as telas com os detalhes de cada Feature, colando esses "prints" em um documento de resposta da prova. Inclua no documento de resposta da prova o link para a página inicial do seu projetos no Azure DevOps Services (Menu lateral -> Overview -> Summary)
- 2- **(40%)** Crie os casos de testes de sistema manuais dentro do Azure, controlando os dados de input e output esperados de cada teste e indicando a preparação a ser feita para que cada teste execute como previsto. Cada Feature deve ter ao menos 1 caso de teste de sistema manual. As entradas e saídas devem usar a funcionalidade "Param Values". O sumário deve conter os pré requisitos do teste (preparação a ser feita para que cada teste execute como previsto). Tire um "print screen" das telas que mostram a configuração do sumário e dos passos dos testes criados, colando esses "prints" em um documento de resposta da prova.

7d485e97-73aa-46c7-b44e-3915a73e7b90.png

SafeWork — Teste... Current
nov. 22 - nov. 29
0% run. [View report](#)

Test Suites

Filter suites by name

SafeWork — Testes Funcionais

- F01 — Autenticação (1)
- F02 — Funcionários (1)
- F03 — EPis (1)
- F04 — Upload (1)
- F05 — IA de Detecção (1)
- F06 — Alertas (1)
- F07 — Dashboard (1)
- F08 — Bem-Estar (1)

More options

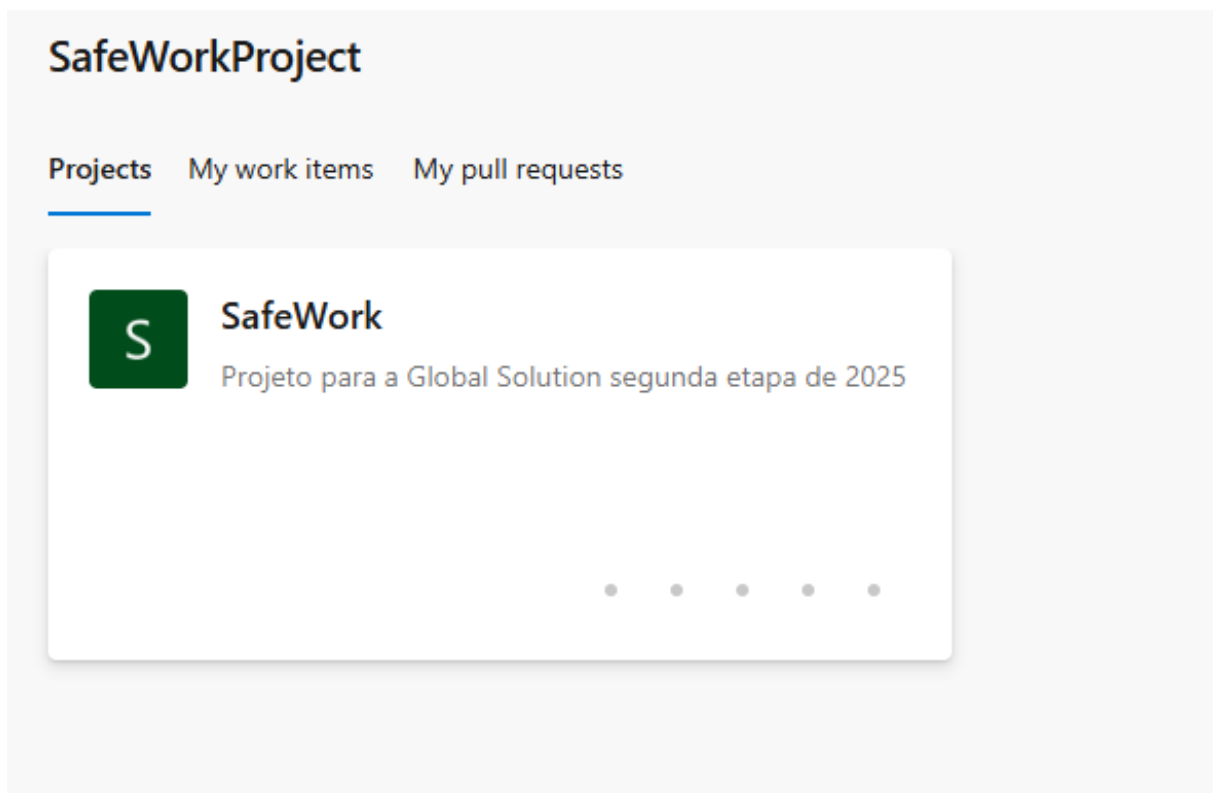
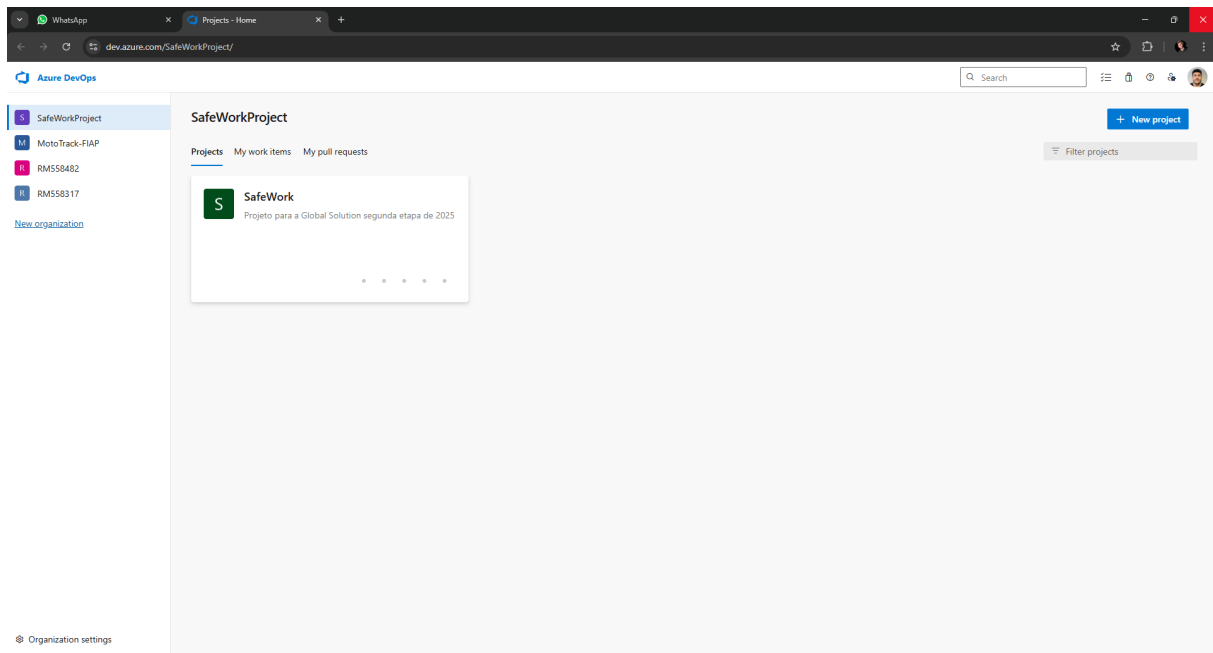
F08 — Bem-Estar (ID: 17) Help

Define Execute Chart

Test Cases (1 item) New Test Case

- ☐ Title
- ☐ TC08 — Realizar check-in de bem-estar diário

8a6adf6b-0a52-42b5-8483-c6b04520756f.png



(1) WhatsApp

SafeWork Team Issues Board - 1

+

dev.azure.com/SafeWorkGS/SafeWork/_boards/board/t/SafeWork%20Team/Issues?workitem=1

☆

Azure DevOps

SafeWorkGS / SafeWork / Boards / Boards

Search

ISSUE 1

1 F01 — Autenticação e Controle de Acesso

No one selected

Save and Close

Follow

0 Comments

Add Tag

State To Do

Area SafeWork

Reason Added to backlog

Iteration SafeWork\Sprint 1

Updated by Angello Turano da Costa: 3m ago

Details

0

0

Description

Permitir que usuários acessem a plataforma com login seguro, controle de sessão e recuperação de senha. Garante segurança e rastreabilidade.

Discussion

Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request.

[switch to Markdown editor](#)

Planning

Priority 2

Effort

Deployment

Development

Related Work

▲ All Parameter Sets

- ▼ .add shared parameter set 1
- .add shared parameter set 2
- .add shared parameter set 3
- .add shared parameter set 4
- .add shared parameter set 5
- .add shared parameter set 6
- .add shared parameter set 7
- .add shared parameter set 8

(1) WhatsApp

SafeWork Team Issues Backlog

dev.azure.com/SafeWorkGS/SafeWork/_backlogs/backlog/SafeWork%20Team/Issues?workitem=2

SafeWorkGS / SafeWork / Boards / Backlogs

Search

SafeWork

2 F01 — Autenticação e Controle de Acesso

No one selected

0 Comments

Add Tag

Save and Close

Follow

Updated by Angello Turano da Costa: 12m ago

Details

State: To Do

Area: SafeWork

Reason: Added to backlog

Iteration: SafeWork\Sprint 1

Description

Permitir que usuários acessem a plataforma com login seguro, controle de sessão e recuperação de senha. Garante segurança e rastreabilidade.

Planning

Priority: 2

Effort

Deployment

To track releases associated with this work item, go to [Releases](#) and turn on deployment status reporting for Boards in your pipeline's Options menu. [Learn more about deployment status reporting](#)

Development

Add link

Link an Azure Repos [commit](#), [pull request](#) or [branch](#) to see the status of your development. You can also [create a branch](#) to get started.

Related Work

Add link

[Add an existing work item as a parent](#)

Discussion

Add a comment. Use # to link a work item, @ to mention a person, or ! to link a pull request.

[switch to Markdown editor](#)

Project settings

6. Pipeline CI – Integração Contínua

Para apoiar o ciclo de desenvolvimento e garantir a integridade do código, foi configurada uma pipeline de Integração Contínua (CI) utilizando Azure DevOps Pipelines. A pipeline é responsável por compilar o projeto Java com Maven, executar testes automatizados (quando presentes) e publicar artefatos (.jar) para uso posterior.

6.1 Pipeline YAML Utilizada

```
trigger:
- main

pool:
vmImage: 'ubuntu-latest'

steps:

- task: Maven@3
inputs:
mavenPomFile: 'pom.xml'
goals: 'clean install'
publishJUnitResults: true
testResultsFiles: '**/surefire-reports/*.xml'
displayName: 'Build com Maven'

- task: CopyFiles@2
inputs:
SourceFolder: '$(System.DefaultWorkingDirectory)'
Contents: '**/*.jar'
TargetFolder: '$(Build.ArtifactStagingDirectory)'
displayName: 'Copiar arquivos .jar'

- task: PublishBuildArtifacts@1
inputs:
PathtoPublish: '$(Build.ArtifactStagingDirectory)'
ArtifactName: 'safework-artifact'
publishLocation: 'Container'
displayName: 'Publicar Artefato'
```

7. Tecnologias Utilizadas

A escolha das tecnologias levou em consideração robustez, suporte da comunidade, aderência ao mercado e compatibilidade com o ecossistema de DevOps. A seguir, um resumo das principais tecnologias utilizadas.

- Java 17 e Spring Boot para desenvolvimento do back-end e APIs REST.
- Maven para gerenciamento de dependências e build.
- Banco de dados SQL (PostgreSQL/MySQL/SQL Server).
- Ferramentas de visão computacional como OpenCV/YOLO para detecção de EPIs.
- Azure DevOps (Boards, Repos, Pipelines, Test Plans) para gestão, versionamento e QA.
- Git como sistema de controle de versão.

8. Conclusão

O projeto SafeWork demonstrou, na prática, a integração entre desenvolvimento de software, inteligência artificial, qualidade de software e práticas de DevOps. A solução proposta é capaz de contribuir significativamente para a segurança no ambiente de trabalho, permitindo monitoramento automatizado de EPIs, geração de alertas, acompanhamento de bem-estar e visualização de indicadores de conformidade.

Do ponto de vista acadêmico, o projeto consolidou conhecimentos em arquitetura de software, desenvolvimento com Java Spring Boot, gestão ágil com Scrum, uso do Azure DevOps (Boards, Test Plans, Pipelines), além de boas práticas de documentação técnica. O resultado final é um conjunto coeso de artefatos (código, testes, pipelines e documentação) pronto para apresentação e evolução futura.

Links Importantes:

- Repositório de Código (GitHub): <https://github.com/AngelloTDC/SafeWork-Q.A.>
- Azure DevOps – Projeto SafeWork: <https://dev.azure.com/SafeWorkProject/SafeWork>