Documentação do Projeto - SeaCare

1. Introdução

Objetivo do Projeto

O projeto SeaCare visa desenvolver uma plataforma digital multifuncional dedicada à mitigação da poluição marinha. O objetivo principal é fornecer ferramentas eficientes para registrar, mapear e combater a poluição nos oceanos, além de engajar a comunidade em ações de limpeza e conscientização ambiental. A plataforma também inclui um modelo de inteligência artificial (IA) para a detecção de lixo marinho e uma loja sustentável para arrecadar fundos e promover práticas ecologicamente corretas.

Apresentação do Problema

A poluição marinha é uma ameaça crítica ao meio ambiente, afetando a biodiversidade, os ecossistemas marinhos e a saúde humana. Atualmente, há uma carência de sistemas integrados que permitam o registro preciso de incidentes de poluição, a organização eficaz de eventos de limpeza e o engajamento da comunidade de forma coordenada. Além disso, a detecção e mapeamento de lixo marinho apresentam desafios significativos devido à ausência de tecnologias avançadas e acessíveis para essa finalidade.

Solução Proposta

A solução proposta pelo SeaCare envolve o desenvolvimento de uma plataforma digital completa que aborda os desafios da poluição marinha através dos seguintes componentes principais: registro e mapeamento de incidentes, organização e gestão de eventos de limpeza, engajamento e coordenação de voluntários, modelo de IA para detecção de lixo marinho, e loja sustentável. A plataforma será composta por duas partes principais: um back-end em Java utilizando o Spring Framework e um back-end em .NET, cada um com seus requisitos específicos.

2. Funcionalidades e Ferramentas

Registro e Mapeamento de Incidentes

Utilizando .NET, o back-end será responsável por esta funcionalidade. Será desenvolvida uma aplicação MVC que separa corretamente as regras de negócios nas controllers e views. A funcionalidade incluirá um sistema para a

criação e submissão de formulários de incidentes de poluição marinha, com campos obrigatórios como título e descrição, e a possibilidade de upload de fotos. A persistência dos dados será gerenciada com o banco de dados Oracle, assegurando o mapeamento de relacionamentos entre tabelas. Para a validação dos dados de entrada, serão utilizados componentes de validação do .NET, garantindo a integridade dos dados. Além disso, haverá a integração com a API do Google Maps para exibir os incidentes reportados.

Organização e Gestão de Eventos de Limpeza

O back-end em Java será responsável por esta funcionalidade, desenvolvendo uma API que permitirá a criação de eventos de limpeza, com formulários que permitem a definição de nome, data, local, descrição e detalhes adicionais. Haverá notificações para lembrar os participantes sobre eventos futuros, usando Firebase Cloud Messaging e Email Services. Os voluntários poderão se inscrever nos eventos, que terão controle de número de vagas disponíveis, e poderão visualizar eventos próximos. Feedback e avaliações pós-evento serão coletados para melhorar futuras ações. A persistência dos dados será gerenciada com Spring Data JPA e a aplicação seguirá o nível 3 de maturidade HATEOAS, incluindo documentação com SWAGGER.

Engajamento e Coordenação de Voluntários

Também sob a responsabilidade do back-end em Java, esta funcionalidade permitirá o cadastro de novos voluntários por meio de formulários, com campos obrigatórios como nome, senha, CEP, e-mail e telefone. Os dados serão armazenados e gerenciados em um banco de dados Oracle. Notificações informando sobre novos eventos e incidentes serão configuráveis de acordo com as preferências do usuário. Tarefas específicas poderão ser atribuídas aos voluntários, e um sistema de permissões classificará os usuários como voluntários ou coordenadores, garantindo um controle eficaz das atividades.

Modelo de IA para Detecção de Lixo Marinho

Esta funcionalidade será gerida pelo back-end em Java, utilizando bibliotecas Python como TensorFlow, Keras e OpenCV para o desenvolvimento do modelo de IA. Os dados históricos e imagens subaquáticas serão coletados e pré-processados. O modelo será treinado e validado, e posteriormente integrado à plataforma SeaCare. Uma API será desenvolvida para comunicação com o modelo de IA, permitindo o processamento em tempo real. Os resultados serão visualizados em um dashboard desenvolvido em React Native.

Loja Sustentável

A funcionalidade de loja sustentável será integrada ao mobile usando React Native, com um direcionamento para uma loja em um site externo, responsável pelas vendas e distribuições dos produtos. Esta abordagem permitirá que a SeaCare promova produtos ecologicamente corretos e arrecade fundos para apoiar suas iniciativas ambientais.

3. Entregas

Desenvolvimento Mobile (React Native)

Aplicativo móvel para registrar incidentes, inscrever-se em eventos, receber notificações e visualizar atividades. Integração com back-end do Firebase e back-ends Java e .NET.

Desenvolvimento Backend em Java (Spring Framework)

API RESTful para criação e gestão de eventos e cadastro de voluntários. Utiliza Spring Boot, Spring Data JPA, Bean Validation, HATEOAS e Swagger.

Desenvolvimento Backend em .NET (MVC)

Aplicação MVC conectada a banco de dados Oracle. Inclui registro e mapeamento de incidentes e uso do design pattern Repository.

Disruptive Architectures & IoT

Desenvolvimento de modelo de IA usando bibliotecas Python (TensorFlow, Keras). Coleta de dados, treinamento, validação e integração com a plataforma via API.

Quality Assurance & Compliance

Análise e design do sistema, incluindo requisitos, modelagem de dados, diagramas UML e planejamento de sprints. Documentação detalhada.

Banco de Dados

Criação e gerenciamento do banco de dados Oracle para os backends em .NET e Java. Implementação do esquema do banco, mapeamento de entidades e otimização de consultas.

DevOps e Implantação

Implementação usando Docker Compose para orquestrar containers, incluindo o backend em Java. Foco na eficiência de desenvolvimento e implantação.

4. Plano de Backlog de Produto

Épico 1: Registro e Mapeamento de Incidentes

Funcionalidade 1.1: Criação de Formulário de Registro de Incidentes

- **Item 1.1.1: Implementação da API de Registro**
  - **Descrição:** Permitir que usuários registrem incidentes de poluição marinha através de um formulário. Necessário para registrar e gerenciar dados de incidentes, garantindo integridade e usabilidade.
  - **Critério de Aceite:** Implementado em .NET MVC. API deve aceitar dados como título, descrição, e fotos, validando entradas.
- **Item 1.1.2: Validação de Dados de Registro**
  - **Descrição:** Garantir que os dados inseridos pelos usuários sejam válidos e completos.
  - **Critério de Aceite:** Implementação de validação com Entity Framework. Testes unitários para todas as regras de validação.

Funcionalidade 1.2: Mapeamento de Incidentes

- **Item 1.2.1: Integração com Google Maps**
  - **Descrição:** Exibir incidentes reportados em um mapa para facilitar a visualização e análise geoespacial.
  - **Critério de Aceite:** Incidentes devem ser plotados corretamente no Google Maps via integração com a API. Testes de integração e documentação Swagger incluídos.

Funcionalidade 1.3: Paginação de Resultados

- **Item 1.3.1: Implementação de Paginação na API**
  - **Descrição:** Melhorar a performance ao listar muitos registros de incidentes.
  - **Critério de Aceite:** API deve suportar paginação usando Spring Data JPA. Testes unitários e documentação Swagger atualizada.

Épico 2: Organização e Gestão de Eventos de Limpeza e Palestras Educacionais

Funcionalidade 2.1: Criação de Eventos

- **Item 2.1.1: Formulário de Criação de Eventos**
  - **Descrição:** Permitir a criação de eventos de limpeza e palestras educacionais capturando informações como nome, data, local, e descrição.

    - **Critério de Aceite:** Implementado em Java Spring. Formulário funcional e dados armazenados no banco de dados Oracle. Testes unitários e de integração realizados.

Funcionalidade 2.2: Notificações para Eventos

- **Item 2.2.1: Sistema de Notificações**
    - **Descrição:** Enviar notificações para participantes sobre novos eventos e lembretes.
    - **Critério de Aceite:** Integração com Firebase Cloud Messaging e serviço de email. Testes unitários para notificações. Documentação incluída.

Funcionalidade 2.3: Inscrição em Eventos

- **Item 2.3.1: Sistema de Inscrição**
    - **Descrição:** Permitir que voluntários se inscrevam em eventos, controlando o número de vagas.
    - **Critério de Aceite:** Implementado em Java Spring. Sistema funcional com controle de vagas. Testes unitários e de integração realizados.

Épico 3: Engajamento e Coordenação de Voluntários

Funcionalidade 3.1: Cadastro de Voluntários

- **Item 3.1.1: Formulário de Cadastro**
    - **Descrição:** Permitir que novos voluntários se cadastrem, fornecendo informações básicas como nome, senha, CEP, e-mail, e telefone.
    - **Critério de Aceite:** Implementado em Firebase. Formulário funcional com dados armazenados no banco de dados Firebase. Testes unitários e de integração realizados.

Funcionalidade 3.2: Sistema de Notificações Personalizadas

- **Item 3.2.1: Configuração de Notificações**
    - **Descrição:** Enviar notificações personalizadas aos voluntários sobre novos eventos e incidentes, conforme suas preferências.
    - **Critério de Aceite:** Implementado em Java Spring com notificações configuráveis. Testes unitários para validar notificações. Documentação incluída.

Funcionalidade 3.3: Sistema de Permissões e Tarefas

- **Item 3.3.1: Atribuição de Tarefas e Permissões**
    - **Descrição:** Permitir a atribuição de tarefas específicas a voluntários e classificação dos usuários como voluntários ou coordenadores.
    - **Critério de Aceite:** Implementado em Java Spring. Sistema funcional com gerenciamento de permissões. Testes unitários e de integração

realizados.

Épico 4: Modelo de IA para Detecção de Lixo Marinho

Funcionalidade 4.1: Desenvolvimento de Modelo de IA

- **Item 4.1.1: Coleta e Pré-processamento de Dados**
  - **Descrição:** Coletar e pré-processar dados históricos e imagens subaquáticas para treinar o modelo de IA.
  - **Critério de Aceite:** Implementado em Python com TensorFlow, Keras e OpenCV. Dados processados e testes de qualidade realizados.

Funcionalidade 4.2: Treinamento e Validação do Modelo

- **Item 4.2.1: Treinamento e Validação**
  - **Descrição:** Treinar e validar o modelo de IA para detecção de lixo marinho.
  - **Critério de Aceite:** Modelo treinado e validado, com resultados documentados. Testes de validação realizados.

Funcionalidade 4.3: Integração do Modelo de IA

- **Item 4.3.1: Desenvolvimento da API de IA**
  - **Descrição:** Desenvolver uma API para comunicação com o modelo de IA, permitindo processamento em tempo real.
  - **Critério de Aceite:** API funcional e integrada ao back-end em Java. Testes de integração e documentação incluída.

Épico 5: Loja Sustentável

Funcionalidade 5.1: Integração com Loja Externa

- **Item 5.1.1: Integração via API com Loja Externa**
  - **Descrição:** Direcionar usuários para uma loja externa para a venda e distribuição de produtos ecologicamente corretos.
  - **Critério de Aceite:** Implementado em React Native. API configurada para redirecionamento. Testes de integração realizados.

5. Diagrama de Arquitetura do Projeto

6. Conclusão

O projeto SeaCare representa uma iniciativa abrangente e inovadora para enfrentar o desafio da poluição marinha. Com uma plataforma digital integrada, o SeaCare facilitará o registro e mapeamento de incidentes de poluição, a

organização de eventos de limpeza e o engajamento da comunidade. O uso de inteligência artificial para a detecção de lixo marinho e a implementação de uma loja sustentável irão reforçar os esforços de preservação ambiental e conscientização. Através de tecnologias modernas e práticas ecologicamente corretas, o SeaCare contribuirá significativamente para a proteção dos oceanos e a promoção de um futuro sustentável.