FECAP

PROJETO SyP

Requisitos da disciplina Modelagem de Software e Arquitetura de Sistemas

São Paulo

2024

INTEGRANTES DO PROJETO e RA'S

Arthur Felipe Alves Nunes	-	24026007
Diogo Bonfim Moreira	-	24026300
Érika Santana da Silva	-	24026205
Guilherme Luis Martins Passos	-	24025973

Sumário

1 INTRODUÇÃO

A SyP tem como objetivo o desenvolvimento de uma plataforma digital focada em doações de roupas e livros, promovendo a solidariedade e a sustentabilidade ao conectar doadores e receptores de forma prática e eficiente. Inspirada em conceitos de economia circular e consciente, a plataforma permite que usuários registrem, filtrem e agendem a entrega ou retirada de itens, garantindo uma experiência intuitiva e segura. A solução busca otimizar o processo de doações, com funcionalidades de geolocalização, avaliações de transações e notificações em tempo real, além de ser projetada para ser escalável, segura e acessível em diversos dispositivos.

2. DOCUMENTO DE ABERTURA DO PROJETO

Prefácio

Este documento descreve o desenvolvimento de uma plataforma inovadora de doação de livros e roupas, conectando doadores a quem precisa, de maneira simples e acessível. Inspirado em modelos de economia colaborativa, o projeto busca contribuir para a redução de desperdícios e para o impacto social positivo, promovendo o reaproveitamento e a solidariedade.

Introdução

O projeto proposto visa criar um site de doação onde usuários possam oferecer livros e roupas que não utilizam mais para pessoas que necessitam desses itens. Diferente de modelos tradicionais de e-commerce, a plataforma será totalmente dedicada a doações, facilitando a conexão entre doadores e beneficiários. O sistema será fácil de usar, intuitivo e acessível, promovendo uma cultura de compartilhamento e reutilização.

Glossário

Doação: Ato de entregar itens (livros ou roupas) para outra pessoa, sem expectativa de pagamento.

- **Usuário:** Pessoa cadastrada na plataforma, seja como doador ou beneficiário.
- **Item:** Objeto disponibilizado para doação (livro ou roupa).
- **Beneficiário:** Usuário que recebe as doações.
- **Sistema:** A plataforma online que conecta doadores e beneficiários.

Definição de requisitos de usuário

Os usuários da plataforma deverão ser capazes de se cadastrar, oferecer itens para doação, visualizar itens disponíveis, manifestar interesse em receber itens e combinar a forma de entrega. Eles também poderão avaliar as interações, contribuindo para um ambiente seguro e colaborativo.

Arquitetura do sistema

A arquitetura do sistema será baseada em um modelo cliente-servidor, onde a interface do usuário (frontend) será acessível via navegadores, conectada a um backend que gerenciará os dados e as operações de doação. O sistema utilizará APIs para comunicação com o banco de dados e integração com serviços externos, como autenticação de usuários e geolocalização.

Especificação de requisitos do sistema

O sistema deve ser capaz de realizar o cadastro e login de usuários, cadastrar itens para doação, permitir a pesquisa e visualização de itens, e possibilitar a interação entre doadores e beneficiários. Deve garantir segurança dos dados e a escalabilidade para suportar o aumento de usuários e itens ao longo do tempo.

Modelos do sistema

O sistema de doação de livros e roupas é composto por três modelos principais: *Usuário*, *Doação* e *Item*. O modelo *Usuário* armazena as informações dos doadores e dos beneficiários, incluindo nome, email, localização e histórico de doações. O modelo *Doação* registra as ofertas de doação feitas pelos usuários, enquanto o modelo *Item* especifica os tipos de itens doados, como livros ou roupas, incluindo descrição, estado e categoria. Esses modelos se relacionam para permitir que os usuários visualizem e recebam doações de forma eficiente.

Evolução do sistema

O sistema evoluirá a partir de uma versão inicial com foco em funcionalidades básicas, como cadastro de usuários, listagem de itens para doação e controle de solicitações. Em futuras versões, pretende-se integrar um sistema de reputação e feedback, bem como melhorar a experiência do usuário com funcionalidades como filtros avançados, recomendações personalizadas e integração com plataformas de transporte para facilitar o envio das doações. Com o tempo, o sistema poderá expandir-se para incluir outros tipos de itens e doações.

Apêndices

No apêndice do projeto, serão incluídas informações complementares e documentos de apoio, como detalhes sobre tecnologias utilizadas (ex.: frameworks web, banco de dados, APIs de integração), instruções de instalação do sistema para desenvolvedores, bem como guias de uso para os administradores do site. Além disso, exemplos de telas, wireframes, e diagramas de arquitetura estarão presentes para apoiar a implementação e a compreensão do sistema.

3. REQUISITOS DE SISTEMA

3.1 REQUISITOS FUNCIONAIS DE SOFTWARE

Necessários 6 requisitos

RFS01	
Função	Cadastro de usuários e doadores
Descrição	Permitir o cadastro de novos usuários na plataforma
Entradas	Nome, e-mail, senha, telefone, endereço.
Fonte	Requisitos do sistema, usuário
Saídas	Confirmação de cadastro, e-mail de verificação
Ação	O sistema armazena os dados do usuário e envia um e-mail de verificação

RFS02	
Função	Validar o acesso de usuários cadastrados
Descrição	Os usuários devem autenticar-se usando e-mail e senha ou através de redes sociais (Google, Facebook)
Entradas	E-mail, senha ou credenciais de redes sociais
Fonte	Requisitos de segurança
Saídas	Acesso à plataforma ou mensagem de erro (credenciais incorretas)
Ação	O sistema verifica as credenciais e permite ou nega o acesso

RFS03	
Função	Facilitar o agendamento de entrega ou retirada entre doador e receptor
Descrição	Após solicitar um item, o receptor e o doador podem agendar um horário para retirada ou entrega
Entradas	Data e horário disponíveis, localização, preferências do usuário
Fonte	Doador e receptor
Saídas	Confirmação do agendamento via notificação ou e-mail
Ação	O sistema envia confirmações e registra o agendamento no perfil de ambos

RFS04	
Função	Exibir uma listagem dos itens disponíveis para doação
Descrição	Os doadores podem listar itens (roupas e livros) disponíveis, e os receptores podem visualizá-los com filtros
Entradas	Filtros (categoria, estado de conservação, localização)
Fonte	Base de dados de itens
Saídas	Lista de itens filtrada com imagens, descrição e localização
Ação	O sistema busca e exibe os itens disponíveis de acordo com os filtros

RFS05	
Função	Permitir que usuários cadastrem itens para doação
Descrição	O doador insere informações sobre o item, incluindo fotos e descrição
Entradas	Descrição do item, fotos, categoria (roupa/livro), estado de conservação, localização
Fonte	Requisitos de usuário
Saídas	Item listado na plataforma
Ação	O sistema armazena os dados do item e o exibe na listagem pública

RFS06	
Função	Permitir que doadores e receptores avaliem a experiência da doação
Descrição	Após a entrega, os usuários podem deixar uma avaliação e comentário sobre a transação
Entradas	Nota (1 a 5 estrelas), comentário
Fonte	Usuário
Saídas	Avaliação registrada e visível no perfil do usuário avaliado
Ação	O sistema armazena as avaliações e as exibe para os outros usuários

3.2 REQUISITOS NÃO FUNCIONAIS DE SOFTWARE

Necessários 6 requisitos

RFS01	
Função	Garantir que a plataforma possa lidar com um grande volume de usuários e itens
Descrição	O sistema deve ser capaz de crescer conforme a demanda aumenta
Entradas	Volume de dados e acessos simultâneos
Fonte	Requisitos de sistema e negócios
Saídas	Desempenho consistente, sem queda de velocidade
Ação	Implementar uma arquitetura em nuvem com balanceamento de carga

RFS02	
Função	Proteger dados pessoais e transações
Descrição	A plataforma deve garantir a segurança dos dados dos usuários
Entradas	Dados pessoais (e-mail, senha, informações de doações)
Fonte	Requisitos de privacidade e legislação (LGPD, GDPR)
Saídas	Dados criptografados e protegidos
Ação	Implementar criptografia de dados, autenticação em dois fatores e práticas seguras de armazenamento

RFS03	
Função	Garantir alta disponibilidade do sistema
Descrição	A plataforma deve estar disponível 24/7, com no máximo 1% de tempo de inatividade
Entradas	Demandas dos usuários e transações
Fonte	Requisitos de negócios e experiência do usuário
Saídas	Plataforma sempre acessível
Ação	Implementar servidores redundantes e monitoramento em tempo real

RFS04	
Função	Garantir tempos de resposta rápidos
Descrição	O sistema deve carregar páginas e realizar buscas em menos de 3 segundos
Entradas	Consultas, requisições de usuários
Fonte	Padrões de usabilidade
Saídas	Respostas rápidas, sem atrasos
Ação	Otimização de código, banco de dados eficiente e caching

RFS05	
Função	Suportar acesso em diferentes dispositivos
Descrição	O sistema deve funcionar bem em desktops, tablets e smartphones
Entradas	Resoluções de tela, diferentes navegadores e sistemas operacionais
Fonte	Requisitos de design responsivo
Saídas	Interface adaptável e otimizada para todos os dispositivos
Ação	Implementar design responsivo com frameworks como Bootstrap

RFS06	
Função	Garantir que a plataforma seja intuitiva e fácil de usar
Descrição	A interface deve ser simples, com fluxo de navegação claro
Entradas	Ações do usuário
Fonte	Requisitos de UX/UI
Saídas	Satisfação do usuário, facilidade de navegação
Ação	Conduzir testes de usabilidade e ajustar a interface com base no feedback

4. CASOS DE USO

Registrar Doação de Item

- **Ator:** Doador.
- **Descrição:** Um doador acessa a plataforma, seleciona a opção de doar um item, insere as informações necessárias e o item é listado para outros usuários.
- **Ações:**
    - O doador faz login na plataforma.
    - Seleciona a opção "Doar Item".
    - Preenche as informações (descrição, fotos, estado de conservação, etc.).
    - O item é registrado no sistema e exibido para receptores.

2. Buscar e Solicitar Item para Doação

- **Ator:** Receptor.
- **Descrição:** Um receptor acessa a plataforma, navega pelos itens disponíveis, filtra por interesse e solicita a doação de um item.
- **Ações:**
    - O receptor faz login na plataforma.
    - Utiliza os filtros de busca (roupas, livros, estado, localização).
    - Visualiza um item de interesse e solicita a doação.
    - O doador é notificado e ambos combinam a retirada/entrega.

3. Avaliar Transação

- **Ator:** Doador ou Receptor.
- **Descrição:** Após a transação, os usuários podem avaliar e comentar a experiência da doação.
- **Ações:**
    - O usuário faz login na plataforma.
    - Acessa o histórico de transações e seleciona a opção de avaliação.
    - Insere uma nota e um comentário sobre a experiência.
    - O sistema armazena e exibe a avaliação no perfil do outro usuário

5. ARQUITETURA DO SISTEMA

1. Arquitetura em Camadas

A arquitetura do sistema será dividida nas seguintes camadas:

1.1 Camada de Apresentação (Frontend)

- **Função:** Interface com o usuário, responsável por exibir as páginas, permitir interações e realizar as requisições para a camada de controle.
- **Tecnologias:** HTML5, CSS3 (com frameworks como Bootstrap), JavaScript (React.js, Vue.js ou Angular).
- **Componentes:**
    - **Páginas de Cadastro/Login:** Formulários para registro e autenticação de usuários.

    - **Listagem de Itens:** Interface de exibição de roupas e livros para doação com filtros.
    - **Formulário de Doação:** Permite aos usuários cadastrarem novos itens para doação.
    - **Sistema de Agendamento:** Interface para agendar retirada ou entrega de itens.
    - **Sistema de Avaliação:** Interface para avaliação da experiência de doação.

1.2. Camada de Controle (Backend)

- **Função:** Gerencia as regras de negócios, responde às requisições do frontend e interage com o banco de dados.
- **Tecnologias:**
    - **Linguagem:** C# (ASP.NET Core).
    - **API RESTful:** Para comunicação entre frontend e backend.
    - **Framework:** ASP.NET Web API.
- **Componentes:**
    - **Controller de Autenticação:** Gerencia login, registro e autenticação de usuários (JWT para autenticação segura).
    - **Controller de Itens:** Manipula a criação, listagem e edição de itens de doação.
    - **Controller de Agendamentos:** Gerencia agendamentos entre doadores e receptores.
    - **Controller de Avaliações:** Permite usuários avaliarem a experiência e gerencia as avaliações.

1.3. Camada de Serviços (Business Logic)

- **Função:** Implementa as regras de negócios, como validação de dados, lógica de agendamento, e envio de notificações.
- **Componentes:**
    - **Serviço de Autenticação:** Valida e gera tokens JWT para segurança nas requisições.
    - **Serviço de Notificações:** Gera e envia notificações (e-mail/SMS) para os usuários sobre novas doações ou agendamentos.
    - **Serviço de Filtro de Itens:** Aplica filtros baseados em categorias, localização e estado de conservação dos itens.
    - **Serviço de Agendamento:** Regras para validação de datas e horários de agendamento.

1.4. Camada de Persistência (Database Layer)

- **Função:** Armazena e recupera dados do banco de dados, incluindo usuários, itens, agendamentos e avaliações.
- **Tecnologias:**
    - **Banco de Dados Relacional:** SQL Server, PostgreSQL ou MySQL.
    - **ORM:** Entity Framework Core para mapeamento objeto-relacional.
- **Componentes:**
    - **Tabelas de Usuários:** Armazena dados pessoais e de login.
    - **Tabelas de Itens:** Armazena detalhes dos itens para doação (descrição, categoria, estado, fotos, localização).
    - **Tabelas de Agendamentos:** Registra informações sobre as retiradas ou entregas de itens.
    - **Tabelas de Avaliações:** Registra avaliações de doações entre usuários.

1.5. Camada de Integração (APIs Externas)

- **Função:** Gerencia a comunicação com serviços externos como gateways de pagamento, APIs de geolocalização ou sistemas de envio de mensagens.
- **Componentes:**
    - **API de Geolocalização (Google Maps API):** Para calcular distâncias entre doador e receptor.
    - **API de Notificações (SendGrid, Twilio):** Para envio de e-mails ou SMS para os usuários.
    - **API de Redes Sociais:** Para login via Google ou Facebook.

---

2. Arquitetura Física (Infraestrutura)

A arquitetura física envolve a escolha de servidores e serviços em nuvem para garantir escalabilidade, segurança e alta disponibilidade.

2.1. Servidores em Nuvem

- **Provedor de Nuvem:** AWS, Microsoft Azure ou Google Cloud Platform.
- **Componentes:**
    - **Serviços de Computação (EC2 ou App Services):** Hospedagem da aplicação backend (API).
    - **Banco de Dados Gerenciado (RDS ou Cosmos DB):** Serviço de banco de dados relacional em nuvem.

    - **Serviço de Armazenamento (S3 ou Blob Storage):** Para armazenar imagens dos itens doados.
    - **Serviço de Balanceamento de Carga (Load Balancer):** Distribui o tráfego entre diferentes instâncias da aplicação para alta disponibilidade.

2.2. CDN (Content Delivery Network)

- **Função:** Garantir o carregamento rápido de conteúdos estáticos (CSS, imagens, JavaScript) para os usuários, independentemente da localização.
- **Provedor:** Cloudflare ou AWS CloudFront.

2.3. Monitoramento e Logs

- **Ferramentas:**
    - **Monitoramento:** Prometheus, Grafana, ou AWS CloudWatch para monitorar a saúde e desempenho do sistema.
    - **Logging:** ELK Stack (Elasticsearch, Logstash e Kibana) ou serviços gerenciados para armazenar e analisar logs.

---

3. Arquitetura Lógica (Fluxo de Dados)

O fluxo de dados é projetado para que as requisições dos usuários fluam de maneira eficiente entre as camadas.

3.1. Fluxo de Cadastro/Autenticação

1. O usuário preenche os dados de cadastro no frontend.
2. A requisição é enviada via API ao controller de autenticação no backend.
3. O backend valida os dados e armazena as informações no banco de dados.
4. Um token JWT é gerado e enviado de volta ao frontend para manter a sessão do usuário.

3.2. Fluxo de Doação de Item

1. O doador insere os detalhes do item (descrição, fotos, etc.) no frontend.
2. O frontend envia a requisição ao controller de itens.
3. O backend valida os dados, armazena o item no banco de dados e o torna disponível para visualização por outros usuários.
4. O sistema notifica receptores próximos com base na localização.

3.3. Fluxo de Agendamento

1. O receptor seleciona um item e solicita a doação.
2. O backend valida a solicitação e cria um registro de agendamento.
3. O doador é notificado e confirma a data e hora de entrega.
4. Ambas as partes recebem confirmações por e-mail ou SMS.

3.4. Fluxo de Avaliação

1. Após a transação, o usuário acessa a página de avaliação no frontend.
2. A avaliação é enviada ao backend, que a armazena no banco de dados.
3. O perfil do usuário avaliado é atualizado com a nova pontuação.

---

4. Segurança

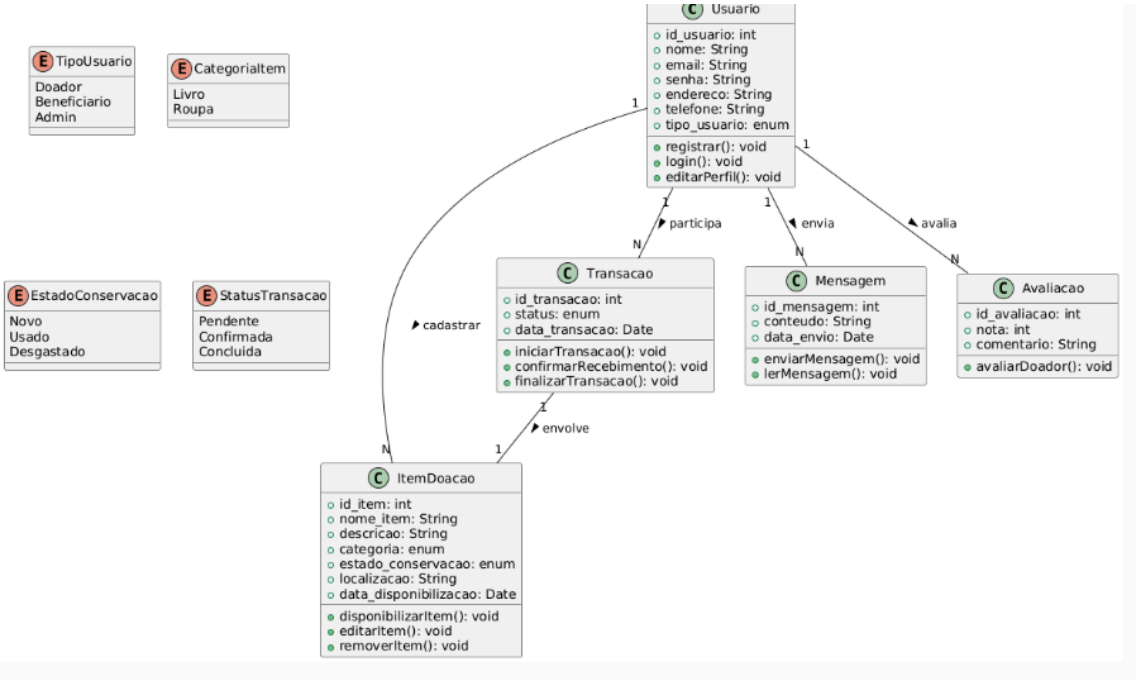
4.1. Proteção de Dados

- **Autenticação em Dois Fatores (2FA):** Para usuários que desejarem aumentar a segurança de suas contas.
- **Criptografia de Dados:** Todos os dados sensíveis são criptografados usando TLS/SSL.
- **Controle de Acesso Baseado em Papéis (RBAC):** Diferenciar permissões para doadores, receptores e administradores.

4.2. Mitigação de Ataques

- **Proteção contra CSRF e XSS:** Implementação de tokens para evitar ataques CSRF e sanitização de entradas de usuários.
- **Rate Limiting:** Limitar o número de requisições para evitar ataques de força bruta.

6. Diagrama de Classes



feito com PlantUML

código:

```
@startuml
!define Enum enum

class Usuario {
  +id_usuario: int
  +nome: String
  +email: String
  +senha: String
  +endereco: String
  +telefone: String
  +tipo_usuario: Enum
  +registrar(): void
  +login(): void
  +editarPerfil(): void
}

class ItemDoacao {
  +id_item: int
```

```
  +nome_item: String
  +descricao: String
  +categoria: Enum
  +estado_conservacao: Enum
  +localizacao: String
  +data_disponibilizacao: Date
  +disponibilizarItem(): void
  +editarItem(): void
  +removerItem(): void
}

class Transacao {
  +id_transacao: int
  +status: Enum
  +data_transacao: Date
  +iniciarTransacao(): void
  +confirmarRecebimento(): void
  +finalizarTransacao(): void
}

class Mensagem {
  +id_mensagem: int
  +conteudo: String
  +data_envio: Date
  +enviarMensagem(): void
  +lerMensagem(): void
}

class Avaliacao {
  +id_avaliacao: int
  +nota: int
  +comentario: String
  +avaliarDoador(): void
}

Enum TipoUsuario {
  Doador
  Beneficiario
  Admin
}

Enum CategoriaItem {
  Livro
  Roupa
}
```

```
Enum EstadoConservacao {
  Novo
  Usado
  Desgastado
}

Enum StatusTransacao {
  Pendente
  Confirmada
  Concluida
}

Usuario "1" -- "N" ItemDoacao : cadastrar >
Transacao "1" -- "1" ItemDoacao : envolve >
Usuario "1" -- "N" Transacao : participa >
Usuario "1" -- "N" Mensagem : envia >
Usuario "1" -- "N" Avaliacao : avalia >
@enduml
```

7. REFERÊNCIAS BIBLIOGRÁFICAS

SOMMERVILLE, I. **Engenharia de Software.** 11ª Edição. São Paulo: Pearson Addison-Wesley, 2017.