Especificação de Sistema
DoATask

Grupo No. 7

David Carvalho Nº27973
Diogo Marques Nº.27972
Gabriel Fortes Nº27976
Gonçalo Vidal Nº27984
Diogo Caldas Nº27951
Vasco Gomes Nº27955

Licenciatura em Engenharia Sistemas Informáticos

2ºano

Barcelos |, 2024

Índice de tabelas

Índice de Figuras

Índice

1. Introdução

DoATask é uma plataforma desenvolvida na qual facilita e promove a realização de voluntariado dentro das comunidades.

Através da **DoATask** os utilizadores podem criar ou se juntarem-se a comunidades, sejam elas focadas em apoio a instituições de solidariedade social, limpeza de espaços públicos, ou qualquer outra iniciativa de caráter voluntário. Cada comunidade funciona como um espaço virtual onde é possível criar e realizar tarefas voluntarias. Para valorizar a participação e tornar o processo mais motivador, cada tarefa concluída concede ao voluntário pontos e moedas virtuais que podem ser trocadas na loja da Comunidade, oferecendo recompensas e reforçando o ciclo de colaboração.

1.1. Objetivo do projeto

O **DoATask** tem como objetivo facilitar o acesso ao voluntariado e incentivar as pessoas a envolverem-se no mesmo. A plataforma foi desenvolvida para simplificar todas as etapas do processo desde encontrar uma tarefa até à sua realização, permitindo que qualquer pessoa, possa contribuir para a sua comunidade. Para incentivar o envolvimento foi desenvolvido um sistema de recompensa com pontos e moedas que podem ser trocados por itens na loja, criando um ambiente que valoriza cada ação solidária e encoraja uma participação das pessoas.

1.2. Problemas a resolver

Atualmente, uma das maiores dificuldades para quem quer fazer voluntariado é a falta de um ponto central onde possam ser consultadas, de forma clara e organizada, as várias oportunidades disponíveis. As tarefas voluntárias estão frequentemente espalhadas por diferentes meios, redes sociais, sites de associações ou grupos informais, o que obriga os interessados a procurar ativamente em múltiplas fontes, muitas vezes sem saber por onde começar. Esta dispersão dificulta a adesão, reduz a visibilidade de muitas iniciativas e acaba por afastar potenciais voluntários que, apesar da vontade de ajudar, não encontram facilmente onde ou como fazê-lo. Além disso, a ausência de um sistema comum que valorize as contribuições de cada pessoa contribui para uma menor motivação e continuidade nas ações de voluntariado.

Por outro lado, também não existem ferramentas para quem quer criar uma comunidade de voluntariado ou criar as suas próprias tarefas de voluntariado. Muitos grupos informais ou pessoas com ideias para ajudar acabam por não avançar por falta de uma plataforma que ajude a centralizar as atividades de voluntariado. Esta ausência de suporte tecnológico compromete a mobilização de novas iniciativas e limita o crescimento de redes de solidariedade locais.

1.1. Solução proposta

O **DoATask** resolve estes problemas ao oferecer uma plataforma que centraliza todas as oportunidades de voluntariado e simplifica a criação de novas comunidades e tarefas. Os utilizadores encontram, num só lugar, um catálogo organizado de ações disponíveis, filtrável por comunidade. Quem pretende criar iniciativas tem acesso a ferramentas intuitivas para abrir comunidades. Um sistema de notificações que informa sobre os vários estados das atividades de voluntario. Além disso, o **DoATask** valoriza cada contribuição através da atribuição de pontos e moedas que podem ser trocadas na loja da comunidade valorizando o esforço de cada voluntário e incentivando a participação contínua. Desta forma, o **DoATask** oferece uma solução completa e acessível tanto para quem quer ajudar como para quem organiza iniciativas solidárias, fomentando a criação e o fortalecimento de redes de voluntariado locais.

2. Processos de negócio

2.1. Processo da tarefa de voluntariado

1.1.1. Etapas

- **Criar uma tarefa:** O utilizador cria uma tarefa na plataforma, especificando os detalhes da mesma, titulo, descrição, localização, dificuldade, comunidade na qual a tarefa será criada e opcionalmente imagens.
- **Eliminar ou não a tarefa:** Caso o utilizador quiser eliminar a tarefa criada pode elimina-la, mas somente se nenhum utilizador aceitou a mesma.
- **Aceitar a tarefa:** Se o utilizador não tiver outra tarefa em curso nessa comunidade, pode aceitar a tarefa.
- **Cancelar ou não a tarefa:** Caso o utilizador quiser cancelar a tarefa pode cancela-la.
- **Terminar a tarefa:** O utilizador termina a tarefa
- **Avaliar a tarefa:** O utilizador avalia a tarefa
- **Receber recompensa:** O utilizador recebe as recompensas em pontos e moedas.

1.1.2. Descrição do processo

Um utilizador pode criar uma tarefa, a tarefa fica então visível para todos os membros da comunidade. Enquanto ninguém tiver aceite a tarefa, o criador pode apagá-la. No entanto, assim que for aceite por outro utilizador, já não pode ser eliminada. Quando a tarefa for terminada o utilizador avalia a mesma.

Os voluntários podem ver as tarefas disponíveis e, se não tiverem nenhuma em andamento nessa comunidade, podem aceitar uma. Se mudarem de ideia antes de a realizar, podem cancelar a aceitação. Depois de concluírem a tarefa, marcam-na como terminada e têm a opção de a avaliar. Ao finalizar, recebem automaticamente pontos e moedas como recompensa.

2.2. Processo da comunidade

1.1.3. Etapas

- **Criar uma comunidade:** Um utilizador cria uma comunidade especificando o nome e a localidade da mesma, sendo que o utilizador têm que pertencer a localidade que especificar.
- **Entrar na comunidade:** Um utilizador entra numa comunidade caso possuía morada na localidade a que a comunidade pertence.

1.1.4. Descrição do processo

O utilizador pode criar uma nova comunidade ao indicar o nome e a localidade da mesma. Para isso, é necessário que o utilizador pertença à localidade que está a registar. Outros utilizadores podem entrar nessa comunidade desde que também tenham morada na mesma localidade.

2.3. Processo da loja

1.1.5. Etapas

- **Criar item:** Utilizador cria um item para ser disponibilizado na loja da sua comunidade
- **Esconder ou Mostrar item:** Utilizador pode esconder o item caso o mesmo esteja disponível na loja ou pode mostrar o item caso o mesmo não esteja disponível na loja.
- **Comprar o item:** Utilizador compra o item caso possuía moedas suficientes

1.1.6. Descrição do processo

Um utilizador pode criar um item para a loja da sua comunidade, depois de criado, o item pode ser colocado visível ou invisível na loja, o utilizador pode escondê-lo a qualquer momento, ou voltar a mostrá-lo quando quiser que fique novamente disponível. Qualquer membro da comunidade pode comprar o item, desde que tenha moedas suficientes.

3. Requisitos funcionais

Os requisitos funcionais descrevem o comportamento esperado da aplicação e as funcionalidades que devem ser implementadas para satisfazer as necessidades dos utilizadores.

Código	Regra
RF 01	O sistema deve permitir que um utilizador crie tarefas dentro de uma comunidade à qual pertence.
RF 02	O sistema deve permitir que o criador elimine uma tarefa, desde que ainda não tenha sido aceite por nenhum voluntário.
RF 03	O sistema deve permitir que um utilizador aceite uma tarefa, desde que não tenha outra tarefa em andamento na mesma comunidade.
RF 04	O sistema deve permitir que o utilizador cancele uma tarefa aceite, antes da sua conclusão.
RF 05	O sistema deve permitir que o utilizador marque a tarefa como concluída.
RF 06	O sistema deve permitir que o utilizador avalie a tarefa após a sua conclusão.
RF 07	O sistema deve atribuir pontos e moedas automaticamente ao utilizador após concluir uma tarefa.
RF 08	O sistema deve permitir que um utilizador crie uma comunidade, definindo o nome e a localidade da mesma.
RF 09	O sistema deve verificar se o utilizador pertence à localidade indicada antes de permitir a criação da comunidade.
RF 10	O sistema deve permitir que um utilizador entre numa comunidade, caso a sua morada corresponda à localidade da mesma.
RF 11	O sistema deve listar as comunidades disponíveis de acordo com a morada do utilizador.
RF 12	O sistema deve permitir que o utilizador crie um item na loja da sua comunidade.
RF 13	O sistema deve permitir que o utilizador esconda ou mostre um item na loja.
RF 14	O sistema deve permitir que o utilizador compre um item, desde que tenha moedas suficientes e pertença a comunidade da loja que o item pertence.
RF 15	O sistema deve atualizar automaticamente o saldo de moedas do utilizador após a compra de um item.

RF 16	O sistema deve contabilizar os pontos e moedas recebidos por cada utilizador após completar tarefas.
RF 17	O sistema deve permitir ao utilizador visualizar o seu saldo de pontos e moedas.

Tabela 1- Requisitos Funcionais

4. Requisitos não funcionais

Os requisitos não funcionais descrevem as características de qualidade que a aplicação deve cumprir, sem estarem diretamente relacionadas com funcionalidades específicas.

Código	Regra
RNF 01	A interface do sistema deve ser simples, intuitiva e acessível para todos utilizadores.
RNF 02	O sistema deve disponibilizar mensagens de erro claras e orientadoras sempre que ocorrer uma falha.
RNF 03	O sistema deve validar os dados introduzidos pelo utilizador para prevenir injeções de código malicioso.
RNF 04	O sistema deve garantir que apenas utilizadores autenticados possam criar, aceitar ou gerir tarefas e comunidades.
RNF 05	O sistema deve ter um mecanismo de recuperação de conta.
RNF 06	O sistema deve possuir autenticação.

Tabela 2 - Requisitos não funcionais

5. Project Backlog

O Project Backlog representa as funcionalidades, requisitos e tarefas que devem ser implementadas na aplicação e tem como objetivo organizar e descrever, de forma clara, as ações que o sistema deve permitir, baseando-se nas necessidades dos utilizadores.

ID	Como Utilizador	Pretendo
1	Utilizador não autenticado	Registar
2	Utilizador não autenticado	Login
3	Utilizador autenticado	Atualizar os dados da minha conta
4	Utilizador autenticado	Criar uma comunidade
5	Utilizador autenticado	Entrar numa comunidade
6	Utilizador autenticado	Criar uma tarefa
7	Utilizador autenticado e criador da tarefa	Eliminar uma tarefa
8	Utilizador autenticado	Aceitar tarefa
9	Utilizador autenticado	Cancelar Tarefa
10	Utilizador autenticado	Terminar Tarefa
11	Utilizador autenticado e criador da tarefa	Avaliar tarefa
12	Utilizador autenticado e dono da comunidade	Criar um item para a loja
13	Utilizador autenticado e dono da comunidade	Esconder ou mostrar item na loja
14	Utilizador autenticado	Comprar um item na loja
15	Utilizador autenticado	Consultar o meu saldo de moedas e pontos

Tabela 3 - Project Backlog

6. Arquitetura do sistema

6.1. Frontend

O **frontend** foi construído com Next.js, uma framework baseada em React, a frontend comunica-se com a backend chamando os endpoitnts para obter e manipular os dados.

6.2. Backend

A backend é constituída por vários serviços que em conjunto garantem a funcionalidade do sistema:

- **NestJs**: Tem como função processar os pedidos recebidos do frontend, tratar da lógica da aplicação, fazer a gestão dos dados e comunicar-se com a base de dados
- **Supabase Auth**: Responsável pela criação de contas, login e gestão das sessões dos utilizadores.
- **Supabase Storage:** Responsável por guardar ficheiros. Estes ficheiros são armazenados e acedidos através de URLs gerados automaticamente.
- **Prisma:** Utilizado em conjunto com o NestJS, para gerir todas as interações com a base de dados.

6.3. Base de Dados PostgreSQL

A aplicação utiliza uma base de dados PostgreSQL, fornecida pelo serviço Supabase, para armazenar todas as informações da aplicação. O PostgreSQL é um sistema de gestão de base de dados relacional, que permite realizar operações complexas de forma eficiente, garantindo a integridade dos dados e o bom desempenho da aplicação.

1.2. Fluxo do funcionamento da arquitetura

1. **Interação do utilizador:** O utilizador acede á aplicação, carregando paginas e componentes contruídos com NextJS.
2. **Pedido REST ao Backend:** Quando o utilizador por exemplo cria uma tarefa, Nextjs envia uma requisição HTTP para o endpoint correspondente no NestJS, incluindo o token JWT em cookie para autenticação.
3. **Autenticação:** O Nestjs recebe o pedido, valida o JWT junto ao serviço de autenticação do Supabase, garantindo que o utilizador está autenticado e autorizado para executar aquela ação.
4. **Acesso à base de dados:** Ainda no NestJs, o serviço correspondente utiliza o Prisma para interagir com a base de dados PostgreSQL, executando operações de leitura ou escrita.
5. **Armazenamento de Ficheiros:** O NestJS recebe o ficheiro, caso o mesmo exista, e utiliza o serviço de Storage do Supabase para guarda-lo, se o pedido da frontend for para receber o serviço gera e retorna um URL.
6. **Resposta ao frontend:** O resultado do pedido é devolvido pelo prisma ou pelo serviço Supabase, dependendo do pedido do utilizador, o NestJS formata uma resposta JSON adequada e envia-a de volta á NextJS.

7. Diagramas casos de uso

Neste capítulo são apresentados os diagramas de casos de uso da aplicação DoATask, que representam as principais interações entre os utilizadores e o sistema.

7.1. Interações do Sistema

O diagrama de casos de uso "Interações do Sistema" representa as principais funcionalidades disponíveis na aplicação DoATask, divididas consoante o tipo de utilizador. Este diagrama mostra de forma clara quais as ações que um utilizador não autenticado e autenticado pode realizar. Este diagrama ajuda a visualizar os diferentes níveis de acesso dentro da aplicação e a organizar as interações consoante o estado de autenticação do utilizador.

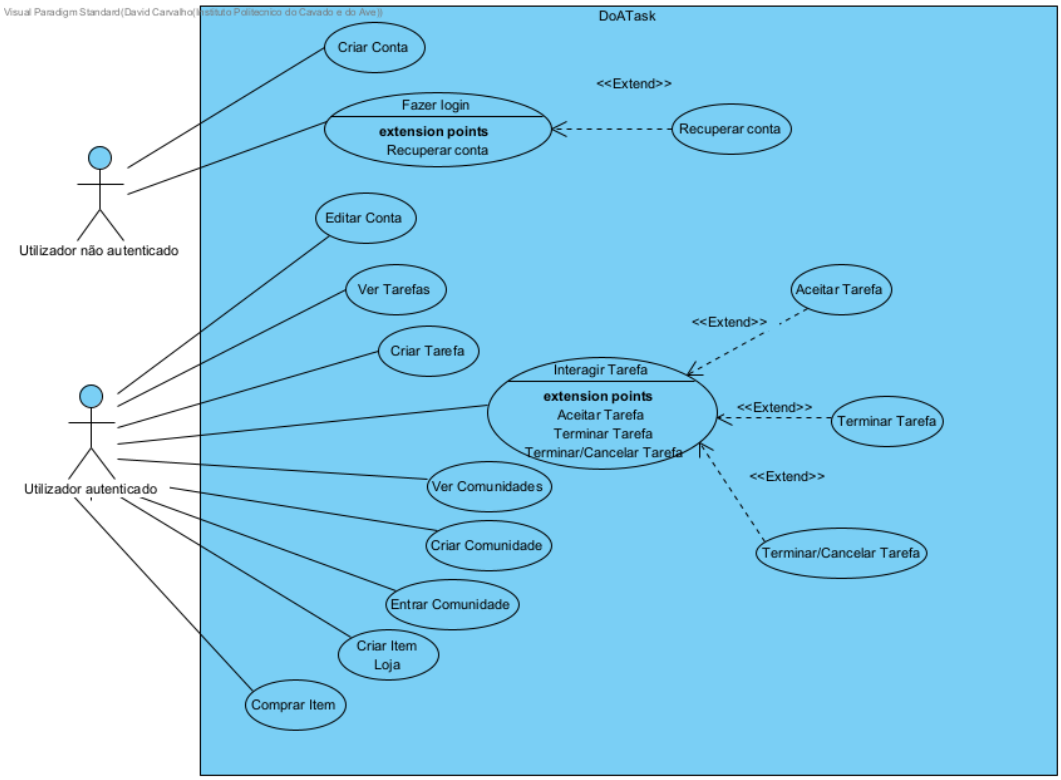

Figura 1- Diagrama Caso de Uso Interações do Sistema

7.1. Utilizador não autenticado

Um **utilizador não autenticado** tem acesso limitado à aplicação e apenas pode realizar ações básicas relacionadas com a entrada no sistema.

- **Criar Conta:** Permite ao utilizador registar-se na plataforma.
- **Fazer Login:** Permite iniciar sessão na conta existente.
- **Recuperar Conta:** Caso o utilizador tenha esquecido as credenciais, pode aceder à funcionalidade de recuperação de conta.

7.2. Utilizador autenticado

Depois de iniciar sessão, o utilizador ganha acesso a um conjunto mais alargado de funcionalidades. Estas incluem:

- **Editar Conta**: Alterar os dados do seu perfil.
- **Ver Tarefas**: Ver as tarefas disponíveis por comunidade.
- **Criar Tarefa**: Criar uma nova tarefa voluntária numa comunidade.
- **Interagir com Tarefa**: Envolve diferentes ações:
- **Aceitar Tarefa:** Aceitar tarefa voluntaria
- **Terminar Tarefa:** Terminar tarefa voluntaria
- **Cancelar Tarefa**: Cancelar tarefa voluntaria
- **Ver Comunidades**: Ver as comunidades disponíveis.
- **Criar Comunidade**: Criar uma nova comunidade, caso o utilizador pertença à mesma localidade.
- **Entrar Comunidade**: Juntar-se a uma comunidade da localidade do utilizador.
- **Criar Item**: Adicionar um item à loja da comunidade do utilizador.
- **Comprar Item**: Comprar itens na loja de uma comunidade.

8. Conclusão

9. Bibliografia

- Material fornecido pela professora no moodle.