UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

CAMILA EMANUELE DE SOUZA

ATUALIZAÇÃO DO BACKEND DO SISTEMA OPEN SOCIAL CARE: MIGRANDO DA ARQUITETURA SERVERLESS PARA UMA API EM LARAVEL

GUARAPUAVA

2023

CAMILA EMANUELE DE SOUZA

ATUALIZAÇÃO DO BACKEND DO SISTEMA OPEN SOCIAL CARE: MIGRANDO DA ARQUITETURA SERVERLESS PARA UMA API EM LARAVEL

Social Care system backend update: migrating from serverless architecture to an API in Laravel

Proposta de Trabalho de Conclusão de Curso de Graduação apresentado à disciplina de Trabalho de Conclusão de Curso 1, do Curso Superior de Tecnologia em Sistemas para Internet da Universidade Tecnológica Federal do Paraná, Campus Guarapuava, como requisito parcial para obtenção do título de Tecnólogo em Tecnologia em Sistemas para Internet.

Orientador: Prof. Dr. Andres Jessé Porfirio

Coorientador: Gustavo Vicari Duarte

GUARAPUAVA

2023

LISTA DE FIGURAS

Figura 1 – Tela de gerenciamento de banco de dados do Open Social Care no Firebase. **9**
Figura 2 – Diagrama de casos de uso para o Administrador do sistema **13**
Figura 3 – Quadro kanban das atividades no github **14**

LISTA DE ABREVIATURAS E SIGLAS

Siglas

API	*Application Programming Interface*
CCG	Conselho da Comunidade da Comarca de Guarapuava
ER	Entidade relacionamento
HTML	*HyperText Markup Language*
IU	Interface do usuário
PHP	*Hypertext Preprocessor*
PPLs	Pessoas Privadas de Liberdade
SDK	*Software Development Kit*
SQL	*Structured Query Language*
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1	**INTRODUÇÃO**	**4**
1.1	**Considerações iniciais**	**4**
1.2	**Objetivos**	**5**
1.2.1	Objetivo geral	5
1.2.2	Objetivos específicos	5
1.3	**Justificativa**	**6**
1.4	**Estrutura do trabalho**	**7**
2	**O OPEN SOCIAL CARE**	**8**
3	**MATERIAIS E MÉTODOS**	**10**
3.1	**Materiais**	**10**
3.1.1	Git e Github	10
3.1.2	PHP e Laravel	10
3.1.3	Docker	11
3.1.4	PostgreSQL	11
3.2	**Métodos**	**11**
3.2.1	Análise e coleta de requisitos	11
3.2.2	Organização de tarefas para o desenvolvimento	13
3.2.3	Planejamento do banco de dados	14
4	**CONSIDERAÇÕES FINAIS**	**15**
	REFERÊNCIAS	**16**

1 INTRODUÇÃO

Esta sessão explana uma breve descrição do sistema, quais seus problemas e logo propondo uma solução.

1.1 Considerações iniciais

O Conselho da Comunidade é um dos órgãos da Execução Penal, regulado pela Lei 7.210, de 11/07/1984, - Lei de Execução Penal - Estas instituições representam a real possibilidade de intervir nas relações sociais dentro e fora da prisão, trazendo à tona a necessidade de modificar o modelo de convivência individualizador, referente a forma como as pessoas interagem e se relacionam em sociedade em relação às Pessoas Privadas de Liberdade (PPLs), promovendo a aproximação da comunidade com a prisão e vice-versa [...] (OLIVEIRA, 2012). Essa aproximação, pode ajudar a preparar os detentos para a vida após prisão, tornando mais fácil para se reintegrarem na sociedade.

O Conselho da Comunidade existe para auxiliar juízes durante a execução penal, gerenciando documentação e garantindo os direitos às PPLs, como o recebimento de roupas e sapatos, como também que os direitos humanos sejam respeitados. Os conselhos também prestam atendimentos para a emissão de antecedentes criminais, obtenção de documentação civil, busca de materiais e recursos solicitados por/para PPLs ou familiares, além de elaborar projetos para remição de pena voltado para atividades de caráter humanizado. Na cidade de Guarapuava - PR, uma das atribuições do Conselho da Comunidade é o serviço social, onde as assistentes sociais realizam atendimentos às PPLs, sendo de grande importância o registro e o armazenamento dessas informações de forma segura, eficiente, disponível e gerenciável.

Atualmente a gestão dos atendimentos realizados pelas assistentes sociais têm muitas brechas, pois são registrados em papel ou em arquivos de texto. Estes procedimentos são falhos, gerando dificuldades na busca por informações, como também a perda de documentos. O Open Social Care é um sistema que visa auxiliar nas atividades desenvolvidas pelas assistentes sociais, de forma a evitar os problemas citados, além de prover um melhor gerenciamento das informações coletadas nos atendimentos das assistentes sociais e eventuais outros usuários.

O sistema teve sua primeira fase de desenvolvimento na disciplina de Sistemas Distribuídos da Universidade Tecnológica Federal do Paraná (UTFPR), câmpus Guarapuava, onde foi proposto uma plataforma básica (um protótipo) apenas para verificar os requisitos do sistema, bem como se a proposta do *software* seria útil para os usuários, neste caso, às assistentes sociais. Com a aplicação finalizada e apresentada às assistentes sociais, foi possível identificar algumas melhorias para o projeto como, por exemplo, sobre a arquitetura do sistema.

Durante esta fase inicial do desenvolvimento, o sistema apresentou potencial para ser usado em outras instituições e, também, por um assistente social sem necessariamente pertencer a um Conselho da Comunidade. Então, como o desenvolvimento foi feito voltado somente

ao Conselho da Comunidade da Comarca de Guarapuava (CCG), é proposto tornar a aplicação mais robusta para que outros perfis de usuário ou instituições possam também fazer uso. Atualmente como o sistema segue um modelo de arquitetura *Serverless*[1], fortemente atrelado à plataforma onde ele foi construído, o que causa dificuldades de generalização e restrições em relação à normas de segurança e privacidade. Diante disso, considera-se necessária uma refatoração da arquitetura do sistema, optando então por tecnologias que permitam uma gestão transparante do sistema e dos dados gerados por ele.

A nova versão do sistema proporcionará aos usuários a personalização de formulários para cadastro de atendimentos, não ficando mais acoplado a Conselhos da Comunidade. Também irá contar com cadastros de organizações, podendo atender várias instituições em um servidor, fazendo com que as assistentes sociais e demais usuários tenham um único acesso e possam participar dentro de outras entidades. Outro ponto a destacar é que a nova versão irá contar com diferentes tipos de permissões para os usuários, onde cada perfil tenha acessos a diferentes lugares da aplicação, tornando-a mais segura e menos suscetível ao erro. A alternativa a ser explorada é a construção de uma *Application Programming Interface* (API)[2] que visa fornecer os recursos atualmente disponibilizados pela arquitetura *Serverless*.

1.2 Objetivos

Esta seção apresenta os objetivos do trabalho.

1.2.1 Objetivo geral

Desenvolver uma API para o sistema Open Social Care e aplicá-la em substituição à infraestrutura proprietária usada no modelo *Serverless*.

1.2.2 Objetivos específicos

- Avaliação do *feedback* (parecer) obtido com o protótipo anterior;
- Revisão dos requisitos do software dadas as sugestões apontadas pelas assistentes sociais;
- Planejamento, na forma de diagramas e *mockups*, das alterações a serem realizadas na nova versão do *software*;
- Elaboração de um modelo para o banco de dados;

[1] Arquitetura sem um servidor próprio, em geral, utilizando infraestrutura em nuvem gerenciada e fornecida por empresas privadas.

[2] API (*Application Programming Interface*): conjunto de funções e procedimentos que permitem a integração de sistemas.

- Implementação de uma API *Restful* que substitua os acessos realizados na plataforma proprietária do modelo *Serverless*;
- Implementação de testes para as funcionalidades desenvolvidas;
- Configuração de um servidor de *staging* para demonstração e coleta de *feedback* da nova versão do sistema;
- Escrita da documentação do *software*.

1.3 Justificativa

No CCG são realizados atendimentos semanalmente, gerando muitos arquivos e dados para a instituição, todos esses atendimentos tem um padrão a ser seguido e atualmente as assistentes sociais, para realizarem um atendimento, utilizam arquivos de texto ou são feitos na mão em papel sendo armazenados no *Google Drive* ou *One Drive*. Esse método de trabalho gera vários problemas, como ao levantar dados ou relatórios, estes precisam ser feitos manualmente, como também podem gerar a perda de documentos ou dificuldade em encontrar os arquivos, isso faz com que o trabalho se torne exaustivo, demorado e suscetível ao erro.

A estrutura do Open Social Care foi desenvolvida em um modelo em nuvem que não contemplava o gerenciamento de um servidor. A aplicação inicial foi construída utilizando o *Firebase*, considerado como um *Backend*, um modelo de serviço que oferece toda a infraestrutura voltada para o funcionamento interno do *software*, como sistemas, banco de dados, envio e recebimento de informações, armazenamento, entre outros. Isso quer dizer que o profissional não precisará desenvolver todo o sistema de forma manual, uma vez que o *Firebase* oferece esse serviço de forma mais automatizada (REMESSA, 2021).

O problema do protótipo ter sido feito dessa forma, é que fica obrigatório tudo estar centralizado na infraestrutura e nas tecnologias da *Google* (proprietária do *Firebase*), cada instituição que fosse utilizar a aplicação precisaria ter uma conta *Google* e realizar a configuração e manutenção do sistema por conta própria, gerando responsabilidade sobre o armazenamento e segurança das informações na conta pessoal de um único usuário (quem implementou e configurou o servidor do sistema). Ademais, a personalização das configurações na plataforma *Firebase* é mais limitada se comparada à executar a aplicação com servidor gerenciado por conta própria, onde a instituição pode ter amplos poderes para gerir a forma como os dados são armazenados.

Outro ponto a ressaltar, é que o sistema inicial (protótipo) foi desenvolvido exclusivamente para atender à demanda do CCG, sendo limitado em relação ao uso em outras instituições que também realizam atendimentos sociais, dificultando a expansão do sistema para demais áreas que realizam atendimento social. Logo uma das melhorias propostas, é fazer um *software* que possa ser usado de forma genérica, isso é, a inclusão de cadastro de múltiplas ins-

tituições, onde também cada uma poderá realizar personalizações a fim de atender diferentes demandas em seus cadastros de atendimentos, possibilitando o uso do sistema para diferentes tipos de instituições.

1.4 Estrutura do trabalho

O trabalho está estruturado em quatro seções principais. No Capítulo 1, foi apresentado uma descrição do projeto de forma geral, apresentando também seus problemas e proposto soluções, além de os objetivos do trabalho. No Capítulo 2, é abordado sobre o protótipo Open Social Care, destacando as suas funcionalidades de infraestrutura. Em seguida, no Capítulo 3, é descrito os Materiais e Métodos que serão usados para o desenvolvimento das atividades. Finalmente no Capítulo 4, é concluído o trabalho mostrando os pontos importantes de desafios e soluções.

2 O OPEN SOCIAL CARE

O Open Social Care é um sistema para gerenciamento de atendimentos sociais, ele foi criado para auxiliar o dia a dia de trabalho de assistentes sociais do CCG. Devido aos detalhes supracitados que dificultam a realização do trabalho destes funcionários sendo importante a utilização de um sistema para suprir as necessidades do cadastro de atendimentos.

O sistema surgiu como um projeto de disciplina de Sistemas Distribuídos da UTFPR, câmpus Guarapuava, quando uma assistente social do CCG entrou em contato com o professor da disciplina e sugeriu a ideia de informatização do processo até então utilizado no CCG. Através de reuniões e discussões foi feita a coleta dos requisitos e planejado como o sistema teria que funcionar, por fim, foi desenvolvido um protótipo como um objeto de estudos na disciplina.

Dada a característica e os tópicos da disciplina, optou-se pelo estudo e experimentação da abordagem *Serverless* de modo que cada aluno pudesse configurar a infraestrutura e trabalhar na aplicação sem precisar de um servidor *Backend*. Na ocasião, as principais tecnologias utilizadas foram *Firebase* e *React.JS*.

Com o *Firebase* foi possível fazer uso de serviços prontos, como o banco de dados, a autenticação de usuários e a hospedagem de arquivos e das páginas. Conforme ilustrado na Figura 1 foi usado o *Firestore Database*, que é um banco de dados *NoSQL*, nele é possível armazenar dados em documentos que contêm mapeamentos de campos para valores. Esses documentos são armazenados em coleções, que são contêineres de documentos que podem ser usados para organizar dados e criar consultas. Os documentos suportam muitos tipos de dados diferentes, desde *strings*[1] e números simples a objetos complexos e aninhados. Também é possível criar subcoleções dentro dos documentos e criar estruturas de dados hierárquicas que podem ser escalonadas à medida que o banco de dados cresce (FIREBASE, 2023a).

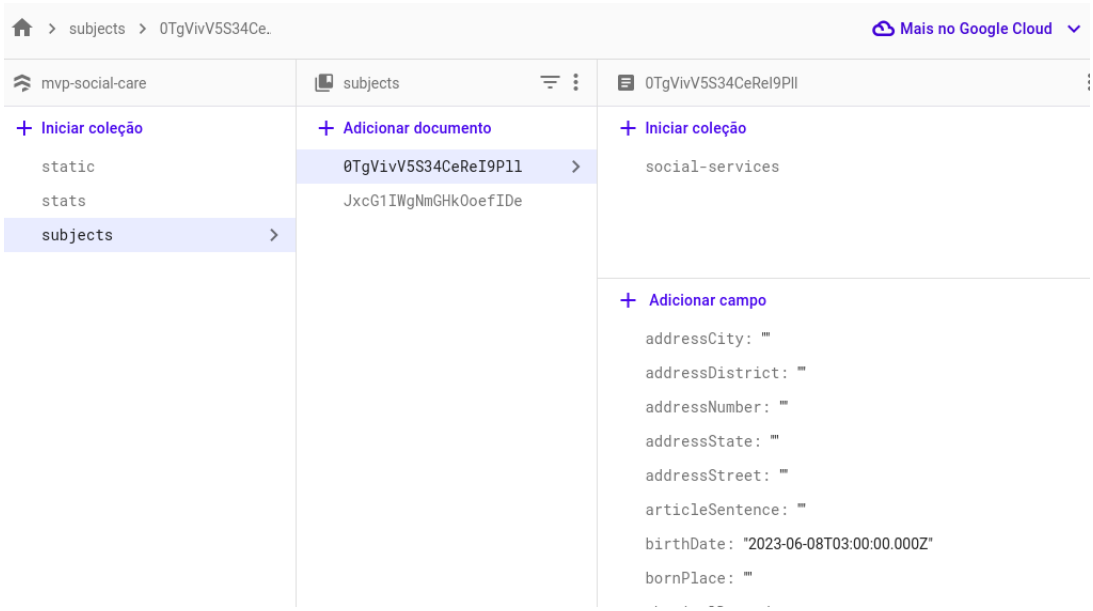
Para autenticação foi usado o *Authentication* que fornece serviços de *Backend*, *Software Development Kit* (SDK) e bibliotecas de Interface do usuário (IU) prontas para autenticar usuários no seu aplicativo. Ele oferece suporte à autenticação usando senhas, números de telefone, provedores de identidade federados conhecidos, como *Google*, *Facebook* e *Twitter*, entre outros (FIREBASE, 2023c). E também foi usado para salvar imagens e arquivos o *Firebase Storage*, um serviço de armazenamento de objetos avançado, simples e econômico criado para a escala do *Google*. Com os SDKs do *Firebase* para *Cloud Storage*, é possível usar a segurança do *Google* para fazer *upload* e *download* de arquivos nos aplicativos do *Firebase*, independentemente da qualidade da rede (FIREBASE, 2023b).

De modo geral, o experimento com a arquitetura *Serverless* foi bastante válido no contexto da disciplina, entretanto, conforme já mencionado, o modelo sofre limitações em relação ao seu uso prático em cenário real. Ademais, a generalização do sistema, que irá possibilitar a utilização dele para demais instituições além do CCG e a possibilidade de hospedá-lo em um

[1] Sequências de caracteres alfanuméricos (letras, números e/ou símbolos)

servidor próprio, sem a dependência de tecnologias de terceiros, aumenta a contribuição e as possibilidades de aplicação do projeto.

Figura 1 – Tela de gerenciamento de banco de dados do Open Social Care no Firebase.



Fonte: Autoria própria (2023).

3 MATERIAIS E MÉTODOS

A ênfase deste capítulo está em descrever as principais ferramentas utilizadas no projeto e o processo de organização utilizado. Este capítulo está subdividido em duas seções, 3.1 Materiais e 3.2 Métodos.

3.1 Materiais

Nesta seção são apresentadas as principais ferramentas e tecnologias utilizadas para o desenvolvimento da API do sistema.

3.1.1 Git e Github

O *Git* e o *Github* são duas ferramentas bastante utilizadas pelos desenvolvedores ao trabalhar em conjunto, com estas ferramentas, é possível armazenar o código como também o histórico dele. O *Git* se trata de um sistema de controle de versionamento de código, possibilitando que possua diferentes versões e funcionalidades sendo desenvolvidas simultaneamente e independentes entre si (GIT, 2023). Já o *Github* é uma aplicação web que possibilita a hospedagem de repositórios *Git* (AQUILES, 2014).

3.1.2 PHP e Laravel

O PHP é uma linguagem de código aberto de uso geral, muito utilizada, e especialmente adequada para o desenvolvimento web e que pode ser embutida dentro do *HyperText Markup Language* (HTML) (PHP, 2023). O *Laravel* é um *framework* feito para o *Hypertext Preprocessor* (PHP), [...] um *framework* compreende um conjunto de classes ou funções implementadas em uma linguagem de programação especifica usadas para auxiliar o desenvolvimento de um *software* (GABARDO, 2017).

A utilização da linguagem de programação PHP para desenvolvimento de aplicações *Web* é facilitada com o uso do *Laravel* uma vez que diversos recursos são fornecidos já implementados, além disso, a adoção do *framework* contribui para a organização do projeto visto que padrões e nomenclaturas precisam ser adotados e padronizados. Outro beneficio do uso do *framework* é a garantia da segurança da aplicação, visto que procedimentos de autenticação já são fornecidos e testados pela comunidade de desenvolvedores.

3.1.3 Docker

O Docker é um projeto de software livre para automatizar a implantação de aplicativos como contêineres autossuficientes portáteis que podem ser executados na nuvem ou localmente. O Docker também é uma empresa que promove e aprimora essa tecnologia, trabalhando em colaboração com fornecedores de nuvem, do Linux e do Windows, incluindo a Microsoft (MICROSOFT, 2023).

3.1.4 PostgreSQL

PostgreSQL é um sistema de banco de dados com código aberto, altamente estável que fornece suporte a diferentes funções de *Structured Query Language* (SQL), como chaves estrangeiras, sub consultas, *triggers* (eventos), e diferentes tipos e funções definidas pelo usuário. Ele aumenta ainda mais a linguagem SQL oferecendo várias características que meticulosamente escalam e reservam cargas de trabalho de dados. É usada principalmente para armazenar dados para muitas aplicativos móveis, web, geoespaciais e analíticas (KINSTA, 2023).

3.2 Métodos

Nesta seção são apresentados os métodos de planejamento para o desenvolvimento do projeto.

3.2.1 Análise e coleta de requisitos

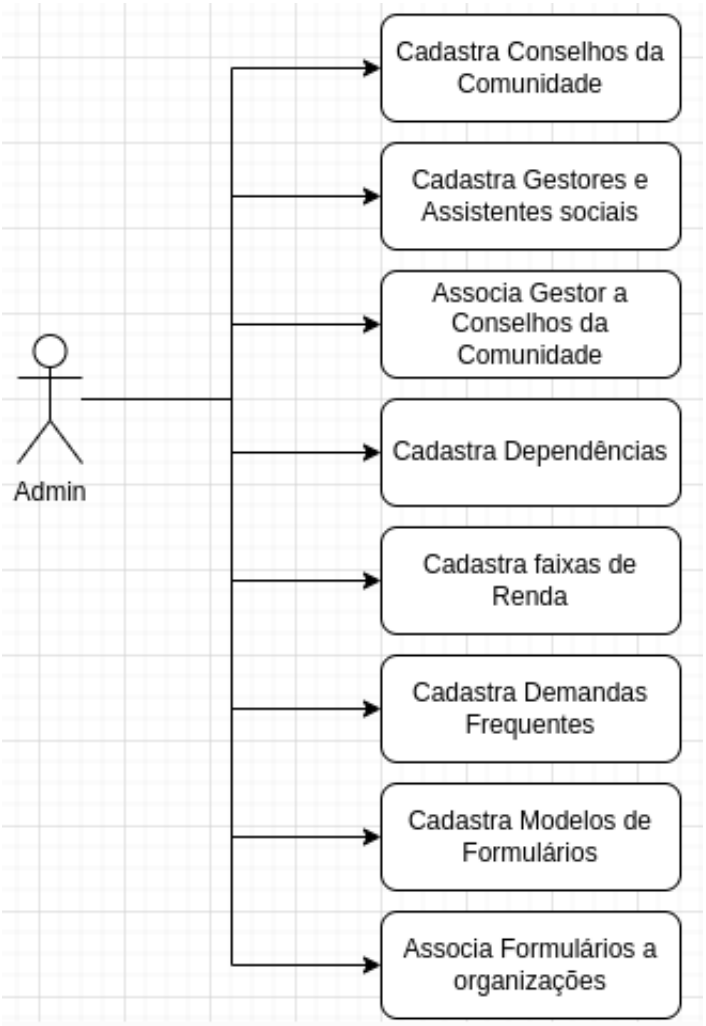
Como o Open Social Care é um projeto que foi apresentado para as assistentes sociais, o protótipo inicial teve uma breve fase de testes, e no decorrer da utilização do sistema, foram reveladas algumas dificuldades ao realizar alguns passos, além das preocupações técnicas, como permissão dos usuários, acessos ao sistema e a infraestrutura. Portanto, através disso foram feitas reuniões para validar os requisitos e coletar funcionalidades que o sistema poderia suprir. Além disso, foram feitas reuniões mais técnicas para validar a estrutura do sistema para chegar em um melhor resultado. Conforme o levantamento dos requisitos, separamos com base em requisitos funcionais, técnico e operacional e utilizamos da metodologia MoSCoW. O método MoSCoW é uma técnica de priorização usada na gestão de projetos e desenvolvimento de softwares com o intuito de encontrar um entendimento em comum entre as partes interessadas sobre a importância que elas atribuem a cada requisito (PIRES, 2019), ele se baseia em etapas, o que o sistema precisa ter, deveria ter, poderia ter e não teria.

Requisitos principais coletados:

- Área de atendimentos:
  - Formulário de cadastro seguindo os campos necessários conversado em reuniões (exemplo: motivo do atendimento e data);
  - Possibilidade de adicionar campo para salvar arquivos ou imagens no cadastro;
  - Listagem com base no sujeito e último atendimento realizado.
- Área de sujeitos:
  - Formulário de cadastro e edição seguindo os campos necessários conversado em reuniões (exemplo: Nome, data de nascimento e dados de familiares);
  - Listagem de visualização com base nos últimos atendimentos realizados;
  - Caixa de busca em tempo real, sem a necessidade de precisar apertar botão para buscar.

Além disso, fluxos de casos de uso foram pensados para melhor entender as funcionalidades que o sistema deveria ter a fim de validar a utilização e o comportamento da aplicação com base nos diferentes perfis de usuário, um exemplo de diagrama de casos de uso pode ser visto na Figura 2. O caso de uso descreve o comportamento do sistema sob diversas condições conforme o sistema responde a uma requisição de um chamado ator primário. O ator primário inicia uma interação com o sistema para atingir um objetivo. O sistema responde, protegendo o interesse de todos (COCKBURN, 2005).

Figura 2 – Diagrama de casos de uso para o Administrador do sistema

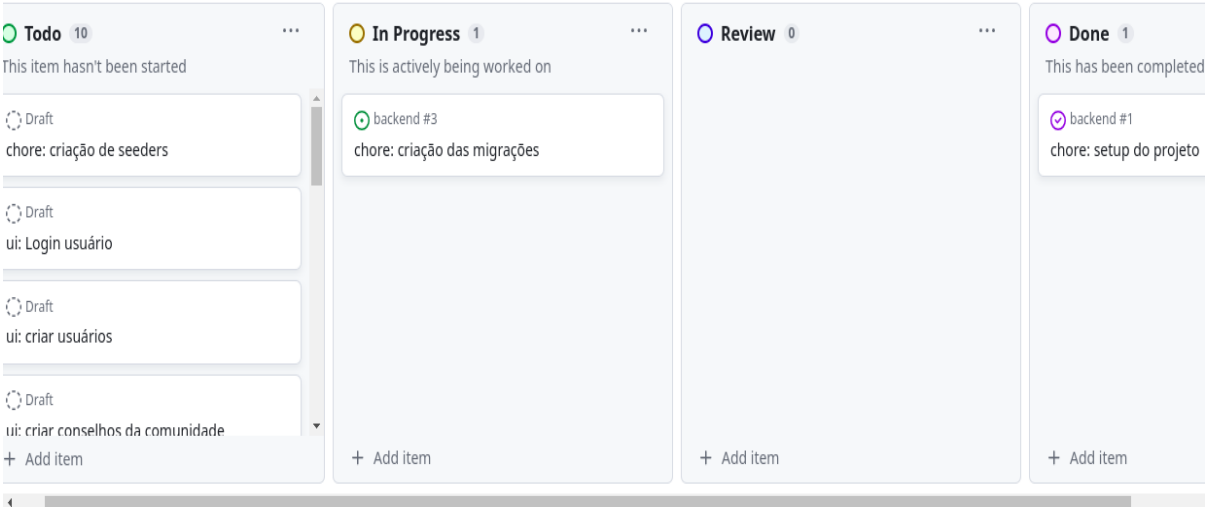


Fonte: Autoria própria (2023).

3.2.2 Organização de tarefas para o desenvolvimento

Para o desenvolvimento das atividades, é feita a utilização da metodologia Kanban. A metodologia Kanban pode ser resumida em uma maneira de organizar os processos que envolvem as equipes de uma organização, ressaltando a priorização das tarefas e tornando o foco bem definido para todos. Dessa maneira, é possível identificar e resolver problemas no fluxo de trabalho (AMORIM, 2023). Na separação das atividades, foi utilizado o recurso *projects* do *Github*, onde é possível separar as atividades em *cards* e organizar com base na sua etapa, *To do* (para fazer), *In Progress* (Em progresso), *Review* (Revisão) e *Done* (Feito), conforme expresso na Figura 3.

Figura 3 – Quadro kanban das atividades no github



Fonte: Autoria própria (2023).

3.2.3 Planejamento do banco de dados

Devido às características do problema, optou-se pelo uso de um modelo relacional de banco de dados. Um banco de dados armazena e fornece acesso a pontos de dados relacionados entre si. O modelo relacional é uma maneira intuitiva e direta de representar dados em tabelas. Em um banco de dados relacional, cada linha na tabela é um registro com uma ID exclusiva chamada chave. As colunas da tabela contêm atributos dos dados e cada registro geralmente tem um valor para cada atributo, facilitando o estabelecimento das relações entre os pontos de dados (ORACLE, 2023).

Também foi feito como base uma modelagem do banco de dados, onde foi utilizada a abordagem de modelo Entidade relacionamento (ER), um diagrama ER é um tipo de fluxograma que ilustra como “entidades”, p. ex., pessoas, objetos ou conceitos, se relacionam entre si dentro de um sistema. Diagramas ER são mais utilizados para projetar ou depurar bancos de dados relacionais nas áreas de engenharia de software, sistemas de informações empresariais, educação e pesquisa. Também conhecidos como DERs, ou modelos ER, usam um conjunto definido de símbolos, tais como retângulos, diamantes, ovais e linhas de conexão para representar a interconectividade de entidades, relacionamentos e seus atributos. Eles espelham estruturas gramaticais, onde entidades são substantivos e relacionamentos são verbos (LUCID, 2023).

4 CONSIDERAÇÕES FINAIS

Neste trabalho é abordado o desenvolvimento de uma API para substituir a infraestrutura *backend* do protótipo Open Social Care. Diante desse desafio, foram descritas as principais ferramentas e métodos a serem utilizadas para alcançar os objetivos propostos.

O protótipo inicial é fortemente dependente de serviços prontos, diante disso, acredita-se que propor uma nova arquitetura com infraestrutura passível de gerenciamento favorecerá a implantação e a manutenção do sistema em cenários reais. Além disso, destaca-se que a infraestrutura inicial não é fácil de ser configurada, precisando ser feito individualmente para cada instituição que irá utilizar o sistema podendo tornar o protótipo inviável para algumas finalidades.

Outro ponto importante para o projeto, é a abordagem de tornar a aplicação mais genérica onde poderá ser utilizada por diversas instituições. Desta maneira, não sendo mais voltada somente a Conselhos da Comunidade, fazendo o sistema ter um maior alcance para sua utilização e, consequentemente, ampliando a contribuição do presente trabalho.

REFERÊNCIAS

AMORIM, S. **O que é Kanban e como usar essa metodologia**. 2023. Disponível em: https://enotas.com.br/blog/kanban/#:~:text=O%20que%20%C3%A9%20a%20metodologia%20Kanban%3F,problemas%20no%20fluxo%20de%20trabalho. Acesso em: 12 set. 2023.

AQUILES, A. **Controlando Versões com Git e GitHub**. São Paulo, SP - Brasil: Casa do Código, 2014. 199 p.

COCKBURN, A. **Escrevendo Casos de Usos Eficazes: Um guia prático para desenvolvedores de software**. Porto Alegre, RS - Brasil: Bookman Editora, 2005. 256 p.

FIREBASE. **Cloud Firestore**. 2023. Disponível em: https://firebase.google.com/docs/firestore?hl=pt-br#:~:text=O%20Cloud%20Firestore%20%C3%A9%20um%20banco%20de%20dados%20NoSQL%20hospedado,em%20APIs%20REST%20e%20RPC. Acesso em: 11 set. 2023.

FIREBASE. **Cloud Storage para Firebase**. 2023. Disponível em: https://firebase.google.com/docs/storage?hl=pt-br#:~:text=download%20de%20objetos.-,Quer%20armazenar%20outros%20tipos%20de%20dados%3F,Firebase%20e%20o%20Google%20Cloud. Acesso em: 11 set. 2023.

FIREBASE. **Firebase Authentication**. 2023. Disponível em: https://firebase.google.com/docs/auth?hl=pt-br#:~:text=O%20Firebase%20Authentication%20fornece%20servi%C3%A7os,Facebook%20e%20Twitter%2C%20entre%20outros. Acesso em: 11 set. 2023.

GABARDO, A. C. **Laravel para Ninjas**. São Paulo, SP - Brasil: Novatec Editora Ldta, 2017. 184 p.

GIT, I. **Git is a free and open source distributed version control system.** 2023. Disponível em: https://git-scm.com. Acesso em: 12 set. 2023.

KINSTA, I. **O que é PostgreSQL?** 2023. Disponível em: https://kinsta.com/pt/base-de-conhecimento/o-que-e-postgresql/. Acesso em: 13 jan. 2023.

LUCID, S. I. **https://www.lucidchart.com/pages/pt/o-que-e-diagrama-entidade-relacionamento**. 2023. Disponível em: https://www.lucidchart.com/pages/pt/o-que-e-diagrama-entidade-relacionamento. Acesso em: 13 set. 2023.

MICROSOFT. **O que é o Docker?** 2023. Disponível em: https://learn.microsoft.com/pt-br/dotnet/architecture/microservices/container-docker-introduction/docker-defined. Acesso em: 12 set. 2023.

OLIVEIRA, A. de F. G. **O conselho da comunidade e suas interfaces**. 2012. Disponível em: https://meuartigo.brasilescola.uol.com.br/atualidades/o-conselho-comunidade-suas-interfaces.htm. Acesso em: 22 ago. 2023.

ORACLE. **O que é um banco de dados relacional (RDBMS)?** 2023. Disponível em: https://www.oracle.com/br/database/what-is-a-relational-database/. Acesso em: 13 set. 2023.

PHP, T. G. **O que é o PHP?** 2023. Disponível em: https://www.php.net/manual/pt_BR/intro-whatis.php. Acesso em: 12 set. 2023.

PIRES, R. **Aprenda a usar a técnica MoSCoW nos projetos da sua agência**. 2019. Disponível em: https://rockcontent.com/br/blog/metodo-moscow/. Acesso em: 12 set. 2023.

REMESSA, O. **Firebase: descubra para que serve, como funciona e como usar**. 2021. Disponível em: https://www.remessaonline.com.br/blog/firebase-descubra-para-que-serve-como-funciona-e-como-usar/. Acesso em: 05 set. 2023.