Você decide quem POD: empoderando a/o estudante de computação quanto à propriedade de seus dados

Oscar E. B. M. Silva, Thiaggo F. B. Souza, João M. S. Duda, Bruno S. R. Barros, Germana M. Nóbrega
{oscar.barbosa,thiaggo.bispo,joao.schmaltz,brunobarros}@aluno.unb.br,gmnobrega@unb.br
Universidade de Brasília (UnB)/Departamento de Ciência da Computação (CIC)

RESUMO

A Web atual criou oportunidades inéditas para a sociedade contemporânea, tais como as mídias sociais, que permitem uma conectividade global sem precedentes. Entretanto, o modelo de centralização de plataformas de amplo alcance popular pelas "big techs" tem sido veementemente questionado, em um movimento de descentralização da Web. O seu próprio criador, Tim Berners-Lee, lidera o projeto Solid, buscando devolver às pessoas usuárias o controle sobre seus dados. Os recursos educacionais apresentados neste artigo foram concebidos para que a/o estudante de computação tenha a oportunidade de conhecer possibilidades outras além da Web grande público. Tais recursos incluem (i) um tutorial sobre tecnologias Solid e (ii) uma aplicação Web ilustrativa, Solid compatível. Acredita-se que esses recursos podem contribuir para deslanchar discussões frutíferas, além do aspecto técnico, acerca dos impactos sociais de tecnologias de ponta, quanto à ética e à privacidade de dados.

CCS CONCEPTS

• **Social and professional topics** → Computing education.

PALAVRAS-CHAVE

Web descentralizada, Privacidade de dados, Solid PODs, Tutorial, Aplicações Web.

1 INTRODUÇÃO

Como é de conhecimento público, uma conectividade global sem precedentes vem sendo observada desde o advento da *World Wide Web* por Tim Berners-Lee no final da década de 1980, tanto entre cidadãos, quanto destes com provedores de serviços (reais ou virtuais), através de portais e de mídias sociais. Por outro lado, há vários anos Berners-Lee e equipe [24] vêm questionando o modelo centralizado que apoderou-se da Web, concentrando nas chamadas "big techs" uma parcela notável de serviços de amplo alcance popular (e.g. Instagram, X Twitter, TikTok, entre outros).

Mais recentemente, tal movimento em prol de uma descentralização da Web se faz reforçar por estudos acadêmicos que se propõem a analisar [4] ou propor [13] maneiras alternativas de controle de dados de usuária/os, ao passo que instituições de representatividade continental trazem a debate questões relevantes em torno de incidentes de impacto nesse contexto [21].

EduComp'24, Abril 22-27, 2024, São Paulo, São Paulo, Brasil (On-line)

O projeto Solid[1], liderado por Berners-Lee e empreendido também no MIT[2], surge oficialmente em 2016 com uma proposta de re-desenho da Web objetivando devolver a cada usuária/o o controle sobre seus próprios dados [14]. Desde então, várias iniciativas envolvendo tanto organizações governamentais e não-governamentais, quanto academia, em aplicações de comunicação e entretenimento [18], saúde [8, 10], educação [15] e governo eletrônico [22] vêm se fazendo registrar na literatura.

Em específico, dados de aprendizes também se fazem explorar há décadas pelos chamados Sistemas Tutores Inteligentes, a fim de se entender os processos de aprendizagem e a partir daí buscar melhorar o desempenho educacional. Entretanto, com a crescente inspeção de dados massivos, riscos éticos e de privacidade de dados vêm motivando reflexões e sistematizações no âmbito da Inteligência Artificial (IA) na Educação [11, 16], das analíticas de aprendizagem [3, 9, 19] e da formação em computação [1].

Neste artigo, apresentam-se dois recursos concebidos em trabalho local por estudantes de computação e para estudantes de computação, a saber, (i) um tutorial sobre tecnologias Solid [5] e (ii) uma aplicação Web ilustrativa, Solid-compatível [7]. A concepção dos recursos busca agregar à formação de um(a) estudante de computação em dupla perspectiva: a) como usuária/os, a fim de contribuir para consolidar uma cultura de resguardo de dados (educacionais), por questão de segurança mas também para facilitar a aprendizagem ao longo da vida; b) como futura/os profissionais de computação, propiciando conhecer outras possibilidades para além do modelo Web grande público, não somente buscando ampliar a formação técnica, mas também o senso crítico quanto ao impacto das tecnologias na sociedade.

O artigo está organizado como segue. A Seção 2 apresenta casos relevantes de utilização do projeto Solid tanto em organizações reais, quanto em âmbito acadêmico para curso de graduação em computação. A Seção 3 traz sugestões para a exploração dos recursos propostos, bem como passos empreendidos para viabilizar a concepção dos recursos que, por sua vez, são apresentados na Seção 4. E, por fim, a Seção 5 encerra o artigo.

2 TRABALHOS RELACIONADOS

Destacam-se a seguir exemplos reais de como o projeto Solid tem sido explorado em âmbito governamental e privado. Antes, entretanto, apresenta-se um maior detalhamento a respeito do projeto.

2.1 Preâmbulo sobre o projeto Solid

Como supra-mencionado, o projeto Solid busca devolver a uma pessoa usuária o controle sobre seus dados. No que segue, sumariza-se a dinâmica que viabiliza tal proposta.

[1] https://solidproject.org/
[2] https://solid.mit.edu/

De acordo com [14], Solid (*Social Linked Data*) é uma plataforma descentralizada para aplicações Web sociais, fundamentada essencialmente nos padrões do consórcio W3C e nas tecnologias da Web Semântica. Em [12], Solid é descrito como uma especificação[3] que permite às pessoas armazenarem seus dados de maneira segura e descentralizada no que é chamado de Pod (*Personal Online Datastore*). Quaisquer tipos de informação podem ser armazenados em Pods, desde arquivos usuais, até dados estruturados. A pessoa proprietária de um Pod é quem decide que dados quer compartilhar e com quem (indivíduos, organizações e/ou aplicações); Além disso, é possível conceder ou revogar acesso a qualquer instante. As aplicações, por sua vez, utilizam protocolos e formatos de dados padrões, abertos e interoperáveis para armazenar e acessar dados em um Pod, conforme supramencionado. A Figura 1 ilustra essa proposta de encapsulamento de dados.

Figura 1: Ilustração de permissão de acesso a dados em um Pod [12].

Os Pods são armazenados em Provedores de Pods. Uma pessoa tem a escolha de obter um Pod em um Provedor de terceiros ou então de montar e manter seu próprio Provedor de Pods. O projeto que abrigou a produção dos recursos ora apresentados tem como objetivo oferecer ao corpo discente de computação um Provedor de Pods que possa servir, além de espaço de armazenamento de dados educacionais, como uma "caixa de areia" para experimentos no desenvolvimento de aplicações Solid-compatíveis e verificação *hands-on* das propriedades inerentes ao Solid.

2.2 Solid em ação: aplicações em organizações

Algumas aplicações de relevância são destacadas a seguir.

2.2.1 BBC. Como menciona [20], algumas empresas, a exemplo da BBC (*British Broadcasting Corporation*), vêm reconhecendo o potencial do Solid, e investindo em suas primeiras aplicações. A aplicação em teste, denominada 'together+ Data Pod'[4], concede Pods às pessoas para armazenamento de seus históricos em *streaming*, assegurando que consentimento explícito é solicitado sempre que a empresa deseja acessar tais dados.

2.2.2 NHS. Em 2021, o Serviço Nacional de Saúde (NHS, do inglês *National Health Service*) do Reino Unido, em conjunto com a empresa Janeiro Digital, anunciaram seu trabalho na implementação de um sistema de saúde descentralizado [6], através do qual as pessoas estão tendo acesso a Pods individuais que armazenam dados relevantes sobre sua saúde. Já em [8], é apresentado um aplicativo móvel que armazena cartão de vacinas emitidas pelo NHS para fins de prova, também baseado em Solid Pod.

2.2.3 Governo eletrônico e Políticas Públicas. Conforme [22], o governo de Flanders, Bélgica, iniciou a 'Comunidade Solid', uma plataforma envolvendo academia, governo, cidadãos e indústria a fim de contribuir com o desenvolvimento em Solid. A comunidade surge consolidando algumas iniciativas anteriores de pilotos realizados com aplicações que buscavam fornecer Pods a cidadãos, como relatado em [2]. Achados do estudo indicam que Solid permite um redesenho na relação entre cidadãos, seus dados pessoais e as aplicações que utilizam esses dados, inclusive facilitando levar em conta questões legais tais como previsto pela GDPR. Ainda no âmbito da 'Comunidade Solid', encontram-se na literatura *position papers*, e.g. [23], partindo do caso de Flanders para aprofundar reflexões sobre essa nova possibilidade da relação entre dados pessoais, tecnologia e negócios, além de advogar sobre expansão da iniciativa a outras regiões.

2.3 Solid na educação em computação

Desde 2019, a disciplina de Arquitetura de Software[5], ministrada para o sexto semestre do Curso de Engenharia de Software da Universidade de Oviedo, Espanha, traz o desenvolvimento e publicação de aplicações Solid-compatíveis de forma open-source e devidamente documentado. Sob as orientações e ensinamentos do professor responsável pela disciplina, Jose Emilio Labra Gayo, os alunos se dividem em grupos e desenvolvem uma aplicação Solid por ano. Os melhores trabalhos de cada ano são destacados e enviados para concorrer no "desafio Solid", podendo aparecer destacados na página oficial do projeto Solid.

Todas as aplicações desenvolvidas são open-source e com licença MIT. Os projetos são documentados semana-a-semana pelos próprios alunos e a página da disciplina disponibiliza tanto projetos em inglês quanto em espanhol. São aplicações com viéis não-comercial, servindo apenas como demonstração dos tipos de projetos que já são possíveis de serem feitos com Solid hoje.

3 PROCEDIMENTOS METODOLÓGICOS

Nesta seção apontam-se possibilidades para a exploração dos recursos propostos utilizando estratégias pedagógicas amplamente conhecidas pela comunidade de docentes e pesquisadores, das mais variadas áreas de conhecimento. Adicionalmente, apresenta-se brevemente o procedimento prático de estudos preliminares que ancoraram a produção dos recursos, a saber, visando à implantação de um provedor de Pods próprio. Esses estudos foram determinantes para que o tutorial pudesse ser construído com mais propriedade.

3.1 Adequação pedagógica

A partir dos Referenciais de Formação da SBC [25] podem ser identificados conteúdos possíveis de serem trabalhados com os recursos propostos, ao longo dos eixos de formação dos cursos de computação, associados às competências (derivadas) explicitadas, a exemplo

[3]https://solidproject.org/TR/protocol
[4]https://www.bbc.co.uk/taster/pilots/together-pod
[5]https://arquisoft.github.io/

de "Programação de Aplicações Web", "Web Semântica e Ontologias na Educação", "Segurança da Informação", "Redes de Computadores" e "Inteligência Artificial e Computacional".

No que diz respeito à estratégia pedagógica, uma possibilidade é a utilização dos recursos propostos na dinâmica da aprendizagem por investigação (IBL, do inglês *Inquiry Based Learning*). O *framework* apresentado em [17] traz uma sistematização de fases e subfases, a saber, Orientação, Conceituação (Questionamento e Geração de Hipóteses), Investigação (Exploração, Experimentação e Interpretação de Dados), Conclusão e Discussão. Na fase de Conceituação/Questionamento podem ser incluídas, no sentido de guiar as investigações discentes, as seguintes questões orientadoras:

- Que distinções éticas e técnicas podem ser identificadas em projeto/implementação de robôs inteligentes atuando na Web centralizada *versus* descentralizada (especificamente com dados persistindo sobre Solid Pods)?
- Como tais distinções podem ser comunicadas a grande público (por exemplo, em atividades de extensão universitária)?
- Que capacidades e limites da abordagem de Solid Pods? Qual o estado-da-arte e da prática?
- Que outros conceitos e tecnologias de Web descentralizada vêm sendo combinados em projetos envolvendo Solid Pods?
- Que riscos causaram desastres que já aparecem mapeados na literatura relativamente ao modelo de Web centralizada? E na grande mídia?

A estratégia pode ser combinada com a de aprendizagem baseada em projetos (PBL, do inglês *Project Based Learning*) a fim de guiar a produção dos artefatos que efetivamente permitiriam a verificação de algumas das questões elaboradas.

3.2 Estudos preliminares: *Hands-on* Pods e Pod server

Estudos iniciais revelaram a possibilidade de obter um Pod em provedor de terceiros, ou então de montar o próprio Pod server. A Figura 2 fornece uma visão geral de um Pod server, segundo [14], por ocasião do lançamento oficial do projeto Solid em 2016.

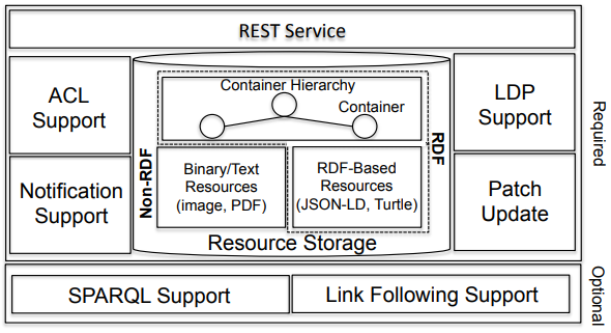

Figura 2: Visão geral de Pod server [14].

Antes de começar a implantação, de fato, de um provedor Solid próprio, foi necessário compreender capacidades e limitações. Assim, uma vez que são indicados alguns provedores de terceiros no site do projeto Solid, decidiu-se iniciar obtendo ali os primeiros Pods da equipe.

Testes foram feitos com aplicações clientes disponibilizadas no site do projeto, e para vários provedores de terceiros com implementações distintas. A maior parte das aplicações exploradas na fase de testes com provedores em produção tinha como foco o gerenciamento do Pod. Isso se deve à intenção de verificar a usabilidade do Pod de forma intuitiva, assim como avaliar o custo para utilizar e configurar as funcionalidades prometidas pelo projeto. As principais aplicações testadas foram: Inrupt PodBrowser (descontinuado e removido do GitHub), SolidOS (presente em diversos provedores em produção), Penny e Solid Filemanager. A princípio foi observado que o Inrupt Podbrowser possuía uma boa integração com Pods do servidor empresarial hospedado pela empresa Inrupt, porém foram notadas limitações de uso para Pods de outros provedores. Com relação às outras aplicações, foi notado que as aplicações ainda não oferecem controle sobre todas as funcionalidades do Pod.

Da observação dos resultados obtidos a partir dos testes iniciais, pode-se concluir que as aplicações possuem limitações a respeito de compatibilidade, ou seja, uma aplicação que é compatível com um provedor não necessariamente será compatível com outro provedor.

Conhecendo o potencial inicial das aplicações e provedores, foram iniciados os estudos em aplicações localhost com todas a implementações Solid Pod disponíveis open source: Node Solid Server (NSS); Community Solid Server (CSS); e solid-nextcloud; e PHP Solid Server. Desses o PHP Solid Server e o solid-nextcloud foram descartados devido a dificuldade de conseguir testar o servidor, em outras palavras, foi dedicado tempo a realizar os testes em localhost, porém devido a inúmeros erros, não foi possível testá-los; também vale ressaltar que o PHP Solid Server não recebia atualizações desde o início de 2022. Mas com o Node Solid Server e o Community Solid Server, foi possível completar as instruções de instalação em ambos tipos de provedores, contudo, para o Node Solid Server, apesar das inúmeras tentativas não foi obtido sucesso em fazê-lo funcionar com outras aplicações. Assim foi concluído que a melhor alternativa seria usar o Community Solid Server.

Para testar o Community Solid Server foi seguido o tutorial da página do github com pequenas modificações, obtendo sucesso para a execução no penny. Entretanto, para outras aplicações apenas o servidor localhost com um certificado de segurança sem assinatura não bastava para seu funcionamento, e em virtude desse fato, um dos autores deste trabalho utilizou de seu domínio pessoal para poder testar de forma mais fidedigna o Community Solid Server. Para adquirir os certificados foram utilizados os serviços let's encrypt e certbot. Uma vez que o certificado foi obtido, passos semelhantes foram seguidos para o teste em localhost.

4 OS RECURSOS PROPOSTOS

Considerando a potencialidade e já realidade de projetos envolvendo Solid, conforme apresentado acima, os recursos ora propostos buscam contribuir para uma disseminação da tecnologia entre estudantes de computação no Brasil. Entende-se que tais recursos podem ser explorados em cursos de computação, por estudantes em Trabalho de Conclusão de Curso, mas também em disciplinas de projeto e desenvolvimento, ou ainda em disciplinas tais como Informática e Sociedade, a fim de motivar debates que se possam fundamentar. No que segue, apresenta-se o Tutorial desenvolvido, bem como a aplicação Web Solid-compatível.

4.1 Tutorial sobre o projeto Solid

Para introduzir a/o estudante do ensino superior ao projeto Solid, foi desenvolvido um material no formato Web[6], cuja página inicial é mostrada na Figura 3. Com o material, busca-se possibilitar um aprofundamento técnico relacionado tanto ao funcionamento do projeto, quanto às disciplinas dos currículos de computação; indicam-se ainda ferramentas para gerenciar o Pod.

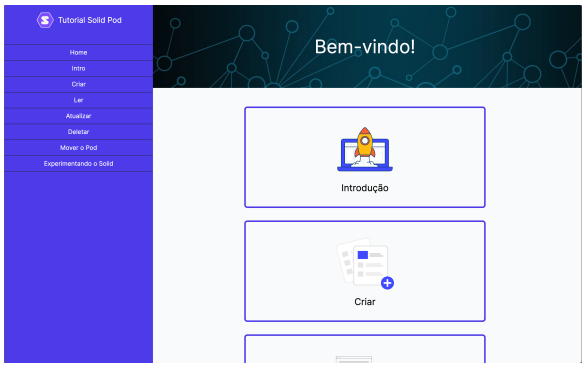

Figura 3: *Home Page* do tutorial como aplicação *Web*.

Para a confecção do tutorial foram utilizadas as seguintes tecnologias voltadas para o desenvolvimento Web:

- Typescript, uma linguagem de programação construída com base no Javascript, adicionando tipagem forte;
- ReactJS, uma biblioteca para construção de interfaces focada em *Single Page Applications*(SPAs), que buscam proporcionar aos usuários uma experiência de alta velocidade e fluidez, e que divide a aplicação em componentes;
- Tailwind CSS, um *framework* CSS que facilita a implementação de designs, utiliza a abordagem de classes utilitárias e permite uma personalização de forma flexível;
- Next.js, um *framework fullstack* para o desenvolvimento de aplicações Web com foco na produtividade. Ele é mantido pela Vercel e oferece uma arquitetura pré-definida com roteamento baseado na estrutura de pastas. Além disso, possui convenções para diversos aspectos da aplicação, como interfaces de erro e carregamento. O Next.js oferece renderização no lado do cliente e do servidor, suporte para estilização com Tailwind CSS, otimizações de imagem e fonte, e é usado por empresas como Netflix, Uber e Github para criar aplicações web interativas e rápidas;
- Vercel, como plataforma de implantação para hospedar material na web. O Vercel é uma plataforma de nuvem sem servidor que permite hospedagem instantânea e escalável, com pouca necessidade de configuração. Foi escolhido devido à sua integração eficiente com diversos frameworks, incluindo Next.js. O plano gratuito, chamado "Hobby", foi escolhido devido à sua integração com o Git, tornando o processo de implantação mais simples, adequado para o caráter educacional do projeto.

[6]https://tutorial-solid.vercel.app/

O tutorial está organizado em partes que englobam uma Introdução para situar o projeto e segmentos que abordam a utilização do Pod. Esses segmentos abrangem as etapas de Criar, Ler, Atualizar e Deletar recursos. Além disso, são tratados temas relacionados a Mover e Experimentação do Pod. Contudo, uma vez que o projeto Solid é relativamente novo, o tutorial precisará de atualizações em conformidade com as evoluções do projeto.

4.1.1 Introdução.

Na seção 'Introdução' do tutorial, é apresentado o projeto Solid. A seção explica como a abordagem corrente da Web afeta a experiência do usuário com os dados e como o Solid pode melhorá-la, como, por exemplo, no contexto de exames hospitalares. Além disso, a seção explica como os dados são tratados com o Solid e como ele se relaciona com outras tecnologias como o RDF e Linked Data. O RDF é uma linguagem para representar informações na Web, enquanto o Linked Data é uma abordagem para publicar e conectar dados na Web. O Solid utiliza essas tecnologias para permitir que os usuários controlem seus próprios dados e compartilhem com quem quiserem. Por fim, a seção apresenta um glossário com os termos principais para o entendimento do tutorial. Esses termos incluem Solid, Pod, WebId, Pod Server e URI, que são usados para explicar outros conceitos mais complexos usados na composição do Solid ou relacionados a ele. O glossário é útil para usuários sem familiaridade com esses termos.

4.1.2 Utilização por meio de CRUD.

Essa parte do tutorial aborda o tema da utilização do servidor por meio de um conjunto de seções composta pelas seções "Criar", "Ler", "Atualizar" e "Deletar". Essas subseções abordam a utilização do servidor e explicam como os usuários podem criar, ler, atualizar e deletar recursos em seus Pods.

A subseção *Create* (Criar) explica como os usuários podem criar recursos básicos em seus Pods, como pastas, arquivos, contatos e bookmarks. É explicado que os arquivos de configuração podem ser criados pelas aplicações ou manualmente pelo usuário. Além disso, é explicado como criar um Pod em uma instância do CSS por meio da interface.

A subseção *Read* (Ler) explica como os usuários podem ler recursos em seus pods. É explicado como os recursos são representados e apresentadas as funcionalidades oferecidas pelos principais podbrowsers. São discutidas as requisições que podem ser feitas ao servidor para ler recursos.

A subseção *Update* (Atualizar) explica como os usuários podem atualizar recursos em seus Pods. É explicado como fazer requisições ao servidor para atualizar recursos e quais tipos de requisições podem ser usadas para esse fim.

A subseção *Delete* (Deletar) explica como os usuários podem deletar recursos em seus pods. É explicado como fazer requisições ao servidor para deletar recursos e quais tipos de requisições podem ser usadas para esse fim.

Essas subseções são úteis para os usuários que desejam entender como criar, ler, atualizar e deletar recursos em seus pods. Além disso, as subseções apresentam informações sobre os principais *podbrowsers* recomendados para a realização dessas operações. Com essas informações, os usuários podem gerenciar seus Pods de forma mais eficiente e personalizada, dentro do que é possível no estado atual do projeto.

4.1.3 *Mover o Pod.*

Essa seção do tutorial do projeto Solid Pod aborda a questão da movimentação de Pods entre provedores. A falta de padronização e diretrizes específicas para a movimentação de pods entre provedores é um aspecto importante a ser considerado na comparação entre a proposta do projeto Solid e o estado atual da arte. A seção apresenta uma visão geral dos desafios envolvidos na movimentação de Pods.

A seção também explica como os usuários podem mover seus pods entre provedores enquanto não há uma forma de fazê-lo pelos *podbrowsers*. É explicado que a movimentação de pods pode ser feita manualmente, copiando e colando os dados de um provedor para outro. Também destaca-se que a comunidade Solid continua a evoluir e é possível que soluções para a movimentação de pods sejam desenvolvidas no futuro, oferecendo aos usuários maior flexibilidade e controle sobre seus dados.

4.1.4 *Experimentando o Solid.*

Nessa seção do tutorial, trata-se de um teste com o Solid Pod, focando no armazenamento e controle de acesso em aplicações de chat (Figura 4), especificamente LiqidChat[7] e Pod-Chat[8], analisando sua integração com os Solid Pods. Foram utilizadas duas implementações de provedores: o Community Solid Server (CSS) e o Node Solid Server (NSS). A forma de armazenamento de dados é determinada pela aplicação de chat, com diagramas e passos detalhados para explicar a interação entre usuários, provedores e servidores de chat. Foram identificados problemas de compatibilidade entre o CSS e as aplicações de chat, e estudos adicionais devem ser realizados a fim de elucidar tais problemas.

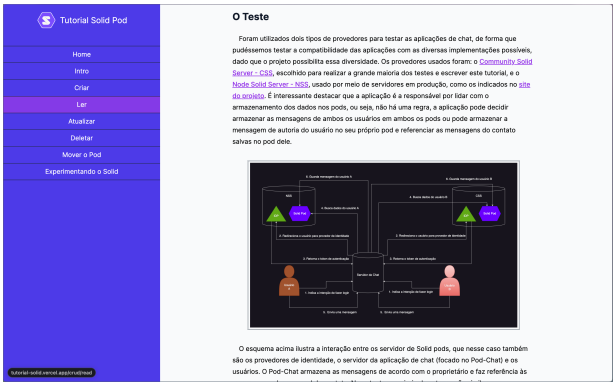

Figura 4: Página que detalha testes feitos com chats.

Na comparação das aplicações de chat em termos de armazenamento de dados e controle de acesso, o Pod-Chat mostrou estar mais alinhado com a proposta do Solid, embora ambas as aplicações exigissem muitas permissões, levantando questões de privacidade. Também houve uma comparação entre chat descentralizado e centralizado, destacando a visão inovadora dos Solid Pods, onde os usuários têm mais controle sobre seus dados, em oposição ao modelo centralizado tradicional.

Acredita-se que essa seção contribui para uma compreensão mais profunda do potencial dos Solid Pods, não apenas em aplicativos de chat, mas também na dinâmica entre as aplicações e o projeto Solid de forma mais ampla. Ela destaca os desafios atuais e as oportunidades futuras para melhorias na integração e experiência do usuário, essenciais para explorar o potencial transformador dos Solid Pods em soluções descentralizadas e centradas no usuário.

4.2 Aplicação Web Solid-compatível

Inspirados em especial pelos trabalhos realizados no Curso de Arquitetura de Software da Universidade de Oviedo, e motivados pelo objetivo de Berners-Lee para a Web do futuro com a tecnologia Solid, realizamos o desenvolvimento de uma aplicação Web Solid-compatível (e mais outra de suporte) para avaliar a experiência de desenvolver utilizando essa nova tecnologia. Esta avaliação é importante, uma vez que o desenvolvimento de aplicações Solid é uma necessidade para que a tecnologia ganhe ainda mais notoriedade.

As aplicações desenvolvidas tiveram como objetivo maior fazer uso da interoperabilidade que a tecnologia permite, que para além da privacidade e controle dos próprios dados pelos usuários, é uma das principais características que a adoção do Solid traz para a Web. A interoperabilidade permite que aplicações distintas e independentes possam trocar informações sem compartilhamento de dados diretamente entre si, o que o Solid torna possível ao estabelecer um padrão de armazenamento de dados em Pods distintos e conexão entre dados da Web, conhecido como *Linked Data*. Esse conceito de interoperabilidade é ilustrado na Figura 5.

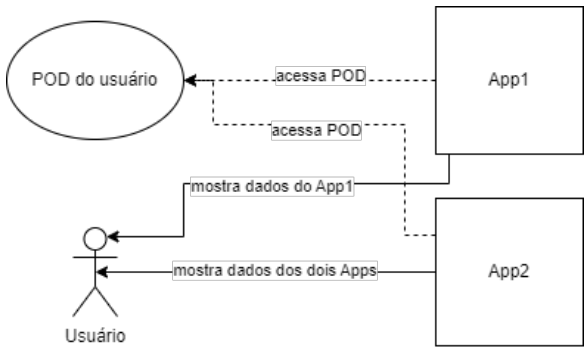

Figura 5: Ilustração de interoperabilidade no Solid.

4.2.1 *As aplicações desenvolvidas.*

As aplicações desenvolvidas têm temática voltada para o contexto educacional, ou seja, fariam sentido em fazer parte do cotidiano de alunos e professores de escolas e universidades. Sendo assim, as duas aplicações de demonstração criadas são o *RUview*[9][10] e o *Tutor*[11][12].

Começando pelo *RUview*, temos uma aplicação na qual alunos podem visualizar as refeições do RU disponíveis no dia do acesso (Figura 6).

[7]https://liqid.chat/
[8]https://pod-chat.com/

[9]RUview produção - https://ruview.vercel.app/
[10]RUview repositório - https://github.com/JomaSnow/Ruview
[11]Tutor produção - https://tutor-orcin.vercel.app/
[12]Tutor repositório - https://github.com/JomaSnow/OwnOnOwn

Figura 6: Tela com os pratos do dia do RUview

Caso autorizem o acesso a seus Pods, estes alunos podem ainda avaliar essas refeições caso gostem ou desgostem delas, e também ver se o que seus amigos pensam sobre as mesmas refeições. Existe também um sistema simples de edição de amigos em seu perfil Solid, e ainda uma área para administradores, onde é possível criar e modificar as refeições presentes no sistema e atualizar o cardápio com elas. Esta última parte não é descentralizada, com as informações das refeições, cardápios e de autenticação armazendas no Firebase, um SGBD hospedado em núvem. A Figura 7 mostra uma visão geral do funcionamento do *RUview*.

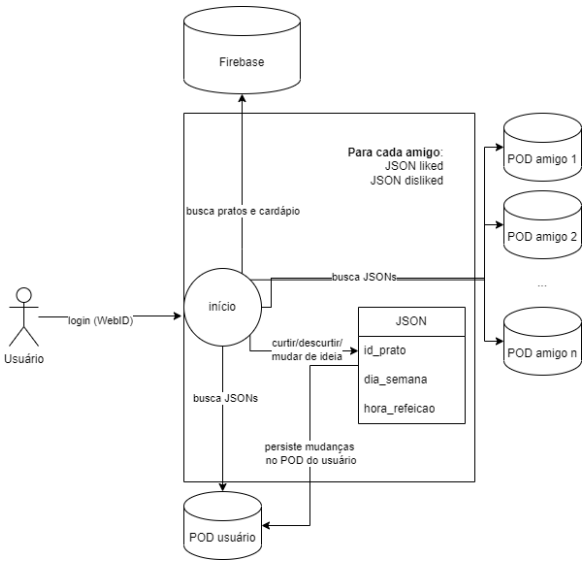

Figura 7: Diagrama Funcionamento RUview.

O usuário autoriza a aplicação a acessar seu Pod e ter acesso ao seu WebID. Uma vez autenticado, a aplicação busca as refeições e cardápios do Firebase, busca os JSONs de refeições curtidos e descurtidos do Pod do usuário, e acessa a lista de amigos do usuário. Para cada amigo nessa lista, a aplicação busca os JSONs das refeições daquele amigo para poder exibir quais amigos gostaram ou não da refeição daquele dia. Toda interação de curtida e descurtida das refeições alteram os JSONs correspondente no Pod do usuário.

Já o *Tutor* consiste em facilitar o agendamento de reuniões de estudos entre dois alunos. Uma vez que o aluno autoriza a aplicação a acessar seu Pod, ele pode criar sua agenda com seus horários livres da semana e também ver as agendas de seus amigos para enviar propostas de reunião para cada um em seus horários livres. A Figura 8 mostra um visão geral do funcionamento do *Tutor*.

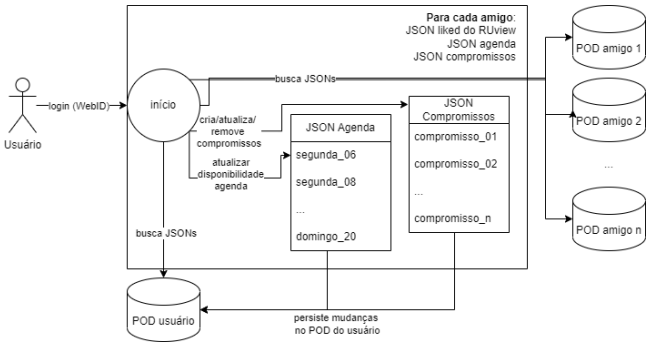

Figura 8: Diagrama Funcionamento Tutor.

A Figura 9 exibe a tela principal da aplicação.

Figura 9: Tela com as reuniões agendadas do Tutor.

O usuário autoriza a aplicação a acessar seu Pod e ter acesso ao seu WebID. Uma vez autenticado, a aplicação busca os JSONs

da agenda e dos compromissos do Pod do usuário, e acessa a lista de amigos dele. Para cada amigo nessa lista, a aplicação busca os JSONs da agenda, dos compromissos e dos pratos curtidos pelo *RUview* daquele amigo (caso existam) para poder exibir as agendas e compromissos atualizados. O JSON dos pratos curtidos é utilizado para comparar se existe algum prato do dia atual na lista e, caso exista, verifica também se o horário de algum prato coincide com um horário livre na agenda do amigo e do usuário; se este for o caso, a aplicação exibe uma mensagem para sugerir este horário para reunião. Por exemplo, na agenda do usuário e de seu amigo, o horário do almoço está livre em suas agendas, e ambos demonstraram interesse no prato do almoço deste dia (os dois possuem este prato em seus Pods); sendo assim o *Tutor* irá recomendar uma reunião na hora do almoço. Toda interação de criação, remoção ou atualização dos compromissos ou agenda alteram os JSONs correspondentes no Pod do usuário.

A interoperabilidade se mostra presente no funcionamento do *Tutor*, que procura por dados criados pelo *RUview* nos Pods dos amigos e do próprio aluno para sugerir possíveis oportunidades de reunião, com a ideia por trás sendo se ambos gostam daquela refeição, então os dois estarão pela Universidade naquele horário, sendo uma boa oportunidade para se reunirem. Ainda, são utilizados os vocabulários *vCard* e *FOAF - Friend of a Friend* nas funcionalidades de perfil e gerenciamento de amigos das aplicações.

Foram utilizadas para o desenvolvimento de ambas aplicações a bilioteca de JavaScript para desenvolvimento *frontend*, ReactJS[13] e a ferramenta PodPro[14], que permite abrir Pods e visualizar seus conteúdos para garantir que estavam sendo salvos os dados corretos, além de também permitir editar esses conteúdos para realizar diferentes testes dentro das aplicações.

4.2.2 *Implementação Solid.*

Para fazer os processos de autenticação do usuário com seu Pod, autorização de privilégios da aplicação ao Pod do usuário e as leituras e escritas em seus Pods, utilizamos as bibliotecas JavaScript da *Inrupt* (empresa de Berners-Lee): *@inrupt/solid-client*, *@inrupt/solid-client-authn-browser* e *@inrupt/vocab-common-rdf*.

Quanto aos testes, criamos Pods em 3 provedores diferentes para tentar simular 3 alunos diferentes utilizando as aplicações e interagindo entre eles. Utilizamos os seguintes provedores: *inrupt.net*, *solidcommunity.net* e *igrant.io*.

Como utilizamos as bibliotecas criadas pela *Inrupt*, precisamos nos familiarizar com os termos que eles utilizam para referenciar os dados dos Pods. Ela utiliza os conceitos de Things, DataSets e Containers, e os define da seguinte forma:

- **Thing**: Uma Thing pode ser entendida como uma referência a uma entidade do mundo real, similar ao conceito de objeto de Programação Orientada a Objeto. Os dados sobre aquela Thing são chamados de propriedades. Por exemplo, em seu Pod pode existir uma Thing "book1" com as propriedades "name" e "author". Utilizam-se os vocabulários RDF para acessar as propriedades de cada Thing.
- **DataSet**: DataSets armazenam um conjunto de Things, e toda Thing precisa necessariamente fazer parte de um DataSet. DataSets podem ser entendidos como o arquivo em uma estrutura de diretórios.
- **Container**: Containers contêm DataSets e outros recursos, como outros Containers. Eles podem ser entendidos como sendo as pastas em uma estrutura de diretórios.

4.2.3 *Limitações.*

Durante o desenvolvimento destas aplicações, algumas suposições foram feitas, e limitações observadas. Primeiro, não encontramos nenhuma informação sobre se a estrutura de um Pod deve seguir algum padrão. Nos Pods dos provedores escolhidos, todos possuiam uma estrutura similar, com um diretório (Container) "profile" que contém um arquivo (DataSet) "card" no qual as informações (Properties) sobre o usuário são agrupadas (Things) e armazenadas. Outra informação contida neste arquivo é a de amigos. Desta forma, caso o usuário utilize um Pod de um provedor que organiza os dados de forma diferente, ou caso ele crie seu próprio Pod sem seguir essa mesma estrutura, a aplicação não funcionará corretamente. A mesma suposição foi feita sobre a presença de um diretório "public" no qual arquivos salvos dentro dele estão sempre disponíveis para leitura de outras aplicações sem necessidade de autorização do usuário (sendo necessária apenas para escrita).

Segundo é sobre a impossibilidade de escrever em Pods que não os do usuário autenticado, ainda que em Containers públicos. Isso faz com que seja trabalhoso manter dados atualizados entre dois ou mais usuários, uma vez que precisam realizar todas as validações por conta ao iniciar a aplicação. Em sistemas centralizados, isso não é nenhum problema, já que o servidor sempre tem acesso aos dados atualizados de todos os usuários, mas em uma Web descentralizada isso já não é trivial para o desenvolvedor.

Ainda sobre as diferenças entre sistemas centralizados e descentralizados, temos mais um obstáculo: o acesso aos Pods podem falhar. Em aplicações centralizadas, todas as transações são atômicas, ou seja, ou uma busca/escrita de dados é feito por completo, ou nada é feito. Mas no caso de aplicações descentralizadas, como os dados estão espalhados ao longo de vários Pods, o que acontece se o acesso a deles falha? A aplicação não pode retornar somente os dados que conseguiu acessar, pois isso resultaria em exibir dados desatualizados. A aplicação pode cancelar toda a transação, similar ao que é feito em aplicações centralizadas, mas isso geraria falhas cada vez mais constantes a medida que o número de diferentes Pods necessários for aumentando. A aplicação também pode armazenar os dados em seus servidores, mas isso efetivamente vai contra os princípios do Solid.

Outra limitação sbre a forma que implementamos nossas funções é que ela são executados linearmente, um comando após o outro. Isso faz com que funções que precisam acessar o Pod de cada amigo do usuário fique cada vez mais demorada à medida que o número de amigos aumenta. Uma última limitação é assumir que o usuário não fará alterações aos dados das aplicações fora delas (diretamente em seu Pod). Por exemplo, um amigo pode recusar a reunião com o usuário, mas o usuário pode em seguida alterar o *status* em seu Pod para "confirmado", e a aplicação do amigo observará esse *status* e modificará o Pod do amigo para confirmar a reunião para ele também.

[13] https://react.dev
[14] https://podpro.dev

5 CONSIDERAÇÕES FINAIS

A aquisição de dados na Web por parte de empresas é notável, inclusive em ambientes educacionais: algumas plataformas coletam uma variedade dados, armazenam e utilizam sem que as pessoas usuárias por vezes sequer saibam quais dados seus estão sendo explorados. Nesse sentido, o projeto Solid propõe trazer o controle dos dados de volta às pessoas proprietárias, sendo uma proposta relevante a ser incorporada em cursos de computação em seus aspectos técnicos, mas sobretudo no tocante a discussões sobre ética e privacidade de dados. Acrescente-se que tais discussões vêm ganhando amplitude com a popularização da Inteligência Artificial e se fazem cada vez mais necessárias aos cursos de computação.

Os recursos ora apresentados foram elaborados por estudantes de computação e buscam contribuir com a inclusão de tais pautas nas várias disciplinas que se façam relacionar, assim como em possíveis Trabalhos de Conclusão de Curso. Esses recursos incluem um tutorial e duas aplicações Web Solid-compatíveis, que estão disponíveis para acesso público. Sendo código aberto, contribuições são bem vindas a fim de fortalecer a iniciativa no Brasil.

Uma possibilidade de trabalho futuro pela equipe de projeto é a montagem de um Pod server local, para que o corpo discente de computação possa armazenar seus dados, mas também para funcionar como caixa de areia, dando suporte a outros desenvolvimentos de aplicações Solid-compatíveis, podendo fundamentar debates.

REFERÊNCIAS

[1] Esdras L Bispo Jr, Liliane SS Fonseca, and Simone C Santos. 2021. Reflexoes e Desafios sobre a Formaçao na Etica em Pesquisa na Computaçao envolvendo Humanos. In *Anais do XXIX Workshop sobre Educação em Computação*. SBC, 488–497.

[2] Raf Buyle, Ruben Taelman, Katrien Mostaert, Geroen Joris, Erik Mannens, Ruben Verborgh, and Tim Berners-Lee. 2020. Streamlining governmental processes by putting citizens in control of their personal data. In *Electronic Governance and Open Society: Challenges in Eurasia: 6th International Conference, EGOSE 2019, St. Petersburg, Russia, November 13–14, 2019, Proceedings 6*. Springer, 346–359.

[3] Teresa Cerratto Pargman, Cormac McGrath, Olga Viberg, Kirsty Kitto, Simon Knight, and Rebecca Ferguson. 2021. Responsible learning analytics: creating just, ethical, and caring. In *Companion proceedings 11th international conference on learning analytics & knowledge (LAK21)*.

[4] Andrea De Salve, Paolo Mori, and Laura Ricci. 2018. A survey on privacy in decentralized online social networks. *Computer Science Review* 27, 154–176.

[5] Thiaggo Ferreira Bispo de Souza and Oscar Etcheaverry Barbosa Madureira da Silva. 2023. Implantação de um POD server para ecossistema educacional e sua introdução na educação superior em computação. Monografia (Graduação em Ciência da Computação). Universidade de Brasília (UnB).

[6] Janeiro Digital. 2021. *Janeiro Digital at Solid World: NHS Personal Health Stores with XFORM Health and Solid.* https://www.janeirodigital.com/blog/janeiro-digital-at-solid-world-nhs-personal-health-stores-with-xform-health-and-solid/

[7] João Marcos Schmaltz Duda. 2023. Uma Investigação Sobre o Projeto Solid: da Prospecção para Ecossistema Educacional ao Desenvolvimento de Aplicações como Prova de Conceito. Monografia (Graduação em Ciência da Computação). Universidade de Brasília (UnB).

[8] Marc Eisenstadt, Manoharan Ramachandran, Niaz Chowdhury, Allan Third, and John Domingue. 2020. COVID-19 antibody test/vaccination certification: there's an app for that. *IEEE Open Journal of Engineering in Medicine and Biology* 1, 148–155.

[9] Rebecca Ferguson. 2019. Ethical Challenges for Learning Analytics. *Journal of Learning Analytics* 6, 3, 25–30.

[10] Hemant Ghayvat, Munish Sharma, Prosanta Gope, and Pradip K Sharma. 2021. Sharif: Solid pod-based secured healthcare information storage and exchange solution in internet of things. *IEEE Transactions on Industrial Informatics* 18, 8, 5609–5618.

[11] Wayne Holmes, Kaska Porayska-Pomsta, Ken Holstein, Emma Sutherland, Toby Baker, Simon Buckingham Shum, Olga C Santos, Mercedes T Rodrigo, Mutlu Cukurova, Ig Ibert Bittencourt, et al. 2021. Ethics of AI in education: Towards a community-wide framework. *International Journal of Artificial Intelligence in Education*, 1–23.

[12] Inrupt. 2023. *Solid Project.* https://solidproject.org/about

[13] Jacob Logas, Ari Schlesinger, Zhouyu Li, and Sauvik Das. 2022. Image DePO: Towards Gradual Decentralization of Online Social Networks using Decentralized Privacy Overlays. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW1, 1–28.

[14] Essam Mansour, Andrei Vlad Sambra, Sandro Hawke, Maged Zereba, Sarven Capadisli, Abdurrahman Ghanem, Ashraf Aboulnaga, and Tim Berners-Lee. 2016. A demonstration of the solid platform for social web applications. In *Proceedings of the 25th international conference companion on world wide web*. 223–226.

[15] Alexander Mikroyannidis, Allan Third, and John Domingue. 2019. Decentralising online education using blockchain technology. In *The Online, Open and Flexible Higher Education Conference: Blended and online education within European university networks*. UNED, Madrid.

[16] Andy Nguyen, Ha Ngan Ngo, Yvonne Hong, Belle Dang, and Bich-Phuong Thi Nguyen. 2023. Ethical principles for artificial intelligence in education. *Education and Information Technologies* 28, 4, 4221–4241.

[17] Margus Pedaste, Mario Mäeots, Leo A Siiman, Ton De Jong, Siswa AN Van Riesen, Ellen T Kamp, Constantinos C Manoli, Zacharias C Zacharia, and Eleftheria Tsourlidaki. 2015. Phases of inquiry-based learning: Definitions and the inquiry cycle. *Educational research review* 14, 47–61.

[18] Hannes Ricklefs, Max Leonard, J Loveridge, Juliette Carter, Kevin Mackay, Jack Allnutt, Thomas Preece, T Nooney, Kamara Bennett, J Cox, et al. 2022. Stronger together: Cross service media recommendations. *SMPTE Motion Imaging Journal* 131, 10, 55–65.

[19] Simon J Buckingham Shum and Rosemary Luckin. 2019. Learning analytics and AI: Politics, pedagogy and practices. *British journal of educational technology* 50, 6, 2785–2793.

[20] Tim Theys, Tom Haegemans, Jelle Saldien, Lieven De Marez, and Javad Kashefi. 2023. Enhancing Users' Attitudes towards WebIDs: Exploring the Effects of Persuasive Messaging on User Adoption. In *Proceedings of Mensch und Computer 2023*. 529–533.

[21] Verica Trstenjak. [n.d.]. *Human rights in digital era. Keynote Speech at WWW 2021.* https://archives.iw3c2.org/www2021/program/keynotes/

[22] Sander Van Damme, Peter Mechant, Eveline Vlassenroot, Mathias Van Compernolle, Raf Buyle, and Dorien Bauwens. 2022. Towards a Research Agenda for Personal Data Spaces: Synthesis of a Community Driven Process. In *International Conference on Electronic Government*. Springer, 563–577.

[23] Sofie Verbrugge, Frederic Vannieuwenborg, Marlies Van der Wee, Didier Colle, Ruben Taelman, and Ruben Verborgh. 2021. Towards a personal data vault society: an interplay between technological and business perspectives. In *2021 60th FITCE Communication Days Congress for ICT Professionals: Industrial Data–Cloud, Low Latency and Privacy (FITCE)*. IEEE, 1–6.

[24] Ching-man Au Yeung, Ilaria Liccardi, Kanghao Lu, Oshani Seneviratne, and Tim Berners-Lee. 2009. Decentralization: The future of online social networking. In *W3C Workshop on the Future of Social Networking Position Papers*, Vol. 2. 2–7.

[25] Avelino Francisco Zorzo, Daltro Nunes, Ecivaldo Matos, Igor Steinmacher, Renata Mendes de Araujo, Ronaldo Correia, and Simone Martins. 2017. Referenciais de Formação para os Cursos de Graduação em Computação.