GetLab: Um Aplicativo para Reserva de Espaços Acadêmicos por meio de um Agente Conversacional

Bruno Brito de Morais
Instituto Federal do Ceará
Tianguá CE, Brasil
bruno.brito.morais17@aluno.ifce.edu.br
ORCID ID 0009-0008-8176-792X

Cynthia Pinheiro Santiago
Instituto Federal do Ceará
Tianguá CE, Brasil
cynthia.pinheiro@ifce.edu.br
ORCID ID 0000-0003-4013-4751

***Resumo*—A crescente utilização de aplicativos móveis tem impulsionado a busca por soluções que otimizem tarefas cotidianas com maior praticidade e eficiência. Nesse contexto, este trabalho visa melhorar o processo de reservas de espaços acadêmicos - como salas e laboratórios - por meio da criação do aplicativo GetLab. Diferentemente dos sistemas já existentes, que são limitados à plataforma Web, o GetLab foi desenvolvido para dispositivos móveis (Android e iOS) e incorpora um agente conversacional capaz de recomendar salas com base nas preferências dos usuários. Como avaliação do primeiro protótipo funcional deste aplicativo junto à comunidade acadêmica, foi conduzido um grupo focal e um teste de usabilidade, cujos resultados obtidos atestam sua eficácia e um bom nível de satisfação dos usuários.**

***Index Terms*—Reserva de espaços acadêmicos, Desenvolvimento mobile, Agente Conversacional, Usabilidade.**

I. Introdução

Os aplicativos móveis auxiliam em diversos problemas e ajudam nas tarefas do dia a dia. Essas aplicações surgem com o objetivo de automatizar e virtualizar tarefas que variam de acordo com a área de atuação. Nesse sentido, com a intenção de simplificar ao máximo a tarefa de reservar espaços acadêmicos na Instituto Federal de Educação, Ciência e Tecnologia do Ceará, utilizamos como base ferramentas já existentes e em uso na instituição que funcionam na web, porém que apresentam problemas - como a ocorrência de conflitos de reservas - ou que são excessivamente burocráticas, exigindo a necessidade de um moderador que autorize as reservas.

Com a intenção de apresentar soluções para estes problemas, este artigo apresenta o desenvolvimento de um aplicativo móvel para Android e iOS intitulado GetLab, cujo objetivo é simplificar o processo de reserva de espaços acadêmicos e melhorar a eficiência das tarefas diárias relacionadas a esse processo. Além disso, para uma maior praticidade - e como inovação - é disponibilizado um agente conversacional que aumenta a eficiência do uso ao diminuir o tempo necessário para uma reserva.

Como forma de validação, seguindo a metodologia de pesquisa-ação, foi realizado um grupo focal, com professores, servidores em cargo de gestão e da área de Tecnologia da Informação (TI) do *campus* que puderam dar seus *feedbacks* quanto às funcionalidades oferecidas em um primeiro protótipo funcional. Em um segundo momento, foi realizado um teste de usabilidade com a participação de alunos, professores e servidores. Estes avaliaram o aplicativo quanto à eficácia, eficiência e satisfação de uso através do instrumento padronizado SUS (*System Usability Scale*) [1]. Estes testes preliminares apresentaram resultados promissores, obtendo-se uma pontuação média do SUS de 89,4 considerada como "Excelente".

O restante deste artigo está organizado como se segue. Na Seção II, são descritas as ferramentas, tecnologias e instrumentos utilizados neste trabalho; na Seção III, listam-se alguns trabalhos relacionados, com propostas similares à deste trabalho; na Seção IV, a metodologia de pesquisa-ação adotada, as etapas do estudo e os instrumentos de coleta de dados são apresentados; na Seção V, detalha-se o aplicativo GetLab, com suas principais funcionalidades e características; na Seção VI, os resultados obtidos são analisados e discutidos e, por fim, na Seção VII, algumas considerações finais são apresentadas, assim como as intenções de trabalhos futuros.

II. Ferramentas, Tecnologias e Instrumentos Utilizados

Nesta seção, será apresentado o arcabouço tecnológico utilizado neste trabalho, envolvendo os conceitos de Desenvolvimento *Mobile*, o Padrão de Arquitetura utilizado, o conceito de Banco de Dados em Nuvem e o *Firebase*, a definição de Agentes Conversacionais e o instrumento padronizado para avaliação de usabilidade utilizado nesse artigo, o SUS.

A. Desenvolvimento Mobile

O mercado de dispositivos móveis passou por um processo de grande popularização com o surgimento dos *smartphones*, com sistemas operacionais (SO) mais avançados, que permitiram o desenvolvimento de aplicativos cada vez mais complexos, com mais recursos e serviços ao usuário. Sendo assim, a partir dessa popularização, o mercado de aplicativos móveis passou por um rápido crescimento, sendo disputado por diversos sistemas, que atualmente se resumem ao Android do Google e iOS da Apple [2].

O desenvolvimento de aplicações *mobile* possui duas abordagens: o desenvolvimento *cross-platform*, que utiliza uma linguagem base que pode ser executada em diferentes SO's, e as aplicações nativas que são desenvolvidas individualmente para cada SO [3]. O desenvolvimento de aplicações móveis é normalmente feito de forma nativa, com linguagens, bibliotecas e ferramentas específicas para cada SO [2].

De acordo com [4], pode-se notar algumas vantagens na escolha do desenvolvimento nativo, que são: (i) alto desempenho, devido à facilidade que o aplicativo nativo possui de se comunicar com as funcionalidades internas do SO e do dispositivo; (ii) interface mais agradável, pelo fato de que os aplicativos nativos se utilizam dos padrões de interface já padronizados e disponibilizados pelo sistema.

Por outro lado, entre as principais desvantagens do desenvolvimento *cross-platform*, estão [5]: (i) desempenho inferior por parte das aplicações não nativas, visto que elas não são projetadas para funcionar perfeitamente na sua plataforma alvo; (ii) menor integração com o *hardware* e o SO do dispositivo, o que leva a um acesso dificultado a sensores como GPS e câmeras e (iii) aplicações muito complexas podem ter dificuldades em funcionar perfeitamente em todas as plataformas, pois quanto maior a complexidade do aplicativo, mais erros estão propensos a surgir. Por estes motivos, para o desenvolvimento deste aplicativo, optou-se pelo desenvolvimento nativo.

Nesse sentido, a maneira mais direta de criar um aplicativo Android nativo é usar o Android Studio[1] com Java[2] ou Kotlin[3] [6]. Neste projeto, escolheu-se o Kotlin pois é uma linguagem mais moderna e menos verbosa que o Java e também porque é recomendada a sua utilização no desenvolvimento de aplicativos Android [7]. Kotlin é uma linguagem de programação de código aberto estaticamente tipada que tem como alvo a JVM (*Java Virtual Machine*), Android, JavaScript e Native. No desenvolvimento de um aplicativo Android com Kotlin, tem-se códigos mais seguros no sistema de tipos, onde a nulidade já é inclusa, o que ajuda a evitar muitos erros do tipo *NullPointerExceptions* [8].

Por outro lado, o Swift[4] é uma linguagem de programação consistente e intuitiva, desenvolvida pela Apple para a criação de aplicativos iOS, macOS, Apple TV e Apple Watch. No desenvolvimento deste aplicativo, foi utilizado também o *framework* SwiftUI[5] que, segundo [9], é declarativo e serve como um *framework* único para o desenvolvimento de *interfaces* entre todos os sistemas operacionais da Apple.

B. Arquitetura MVVM (Model-View-ViewModel)

De acordo com [10], a arquitetura precisa estar em conformidade com o escopo necessário para a primeira versão do sistema, mas suficientemente flexível e preparada para incorporar e absorver as evoluções subsequentes.

O MVVM (*Model-View-ViewModel*) otimiza atributos fundamentais, como a vinculação de dados, modelo, comando e comportamento [11]. No modelo MVVM, a camada *ViewModel* se comunica com a camada de visualização por meio de vinculação de dados e ligação de comandos (Figura 1). Essa camada também pode ser responsável por realizar requisições HTTP, o que torna a *View* testável e reutilizável [12].

[1]https://developer.android.com/studio
[2]https://www.java.com/pt-BR/
[3]https://kotlinlang.org/
[4]https://www.apple.com/br/swift/
[5]https://developer.apple.com/xcode/swiftui/

Figura 1: Arquitetura MVVM, adaptado de [13].

O MVVM é baseado nos seguintes elementos [14]:

- *Model*: A camada de modelo é responsável por toda a lógica que impulsiona a aplicação e os objetos de negócio relacionados.
- *View*: Nesta camada é onde está localizada a *interface* de usuário. Ela inclui qualquer código específico da plataforma necessário para conduzir a interação do usuário com a aplicação.
- *ViewModel*: Esta camada é responsável por enviar os dados diretamente para a camada de *View*. Assim, atua como um intermediário entre a *View* e o *Model*, proporcionando uma desacoplação do código e facilitando testes futuros.

A responsabilidade da *ViewModel* no contexto do MVVM, é disponibilizar para a *View* uma lógica de apresentação, não tendo nenhum conhecimento específico sobre a *View* ou como ela é implementada. A *ViewModel* implementa propriedades e comandos, para que a *View* possa preencher seus controles e notifica a mesma, caso haja alteração de estado, seja através de eventos ou de notificação de alteração. A *ViewModel* é peça fundamental no MVVM, por que é ela quem vai coordenar as iterações da *View* com o *Model*, considerando que ambas não têm conhecimento uma da outra. Além disso, a *ViewModel*, também pode implementar a lógica de validação, para garantir a consistência dos dados. A camada *Model* é responsável por fornecer os dados e pode conter a lógica de negócio para o domínio do problema [15].

C. Agentes Conversacionais

Um agente conversacional (também chamado de *chatbot*, *chatterbot*, agente de conversação ou sistema de diálogo) é - segundo [16] - uma aplicação que simula uma conversa humana por meio de uma interação textual entre um usuário humano, que fornece a entrada, e o agente que responde a ele, fornecendo respostas ou formulando perguntas. Considerando sua grande utilização em sistemas web, os agentes conversacionais podem ser considerados como uma forma de interface, que complementa ou substitui outras formas de acesso ao computador [17]. Há dois tipos de agentes conversacionais: aqueles baseados em regras e os que usam Inteligência Artificial. Os baseados em regras possuem comandos específicos e pré-definidos, o que significa que, caso o usuário pergunte algo fora do contexto para o qual o agente foi desenvolvido, ele não saberá como agir adequadamente. Já os agentes baseados em inteligência artificial possuem a capacidade de aprender a

linguagem natural ao longo do tempo e com base nos dados fornecidos [18] .

Neste projeto, devido à simplicidade do contexto, escolheu-se utilizar um agente baseado em regras para recomendar aos usuários qual sala melhor se adequaria ao seu perfil e às suas necessidades.

D. Banco de Dados em Nuvem e Firebase

Os Bancos de Dados (BD) surgiram como uma alternativa mais robusta e eficiente para o gerenciamento de dados, que anteriormente eram persistidos em sistemas de arquivos pelas aplicações orientadas a objetos [19]. No entanto, segundo [20], os sistemas relacionais tradicionais, devido a sua complexidade de operação e implementação, são mais custosos em termos de processamento e são normalmente executados sobre um único sistema computacional.

Em geral, os recursos de computação e *hardware* são propensos a ficarem obsoletos rapidamente e a utilização de plataformas computacionais de terceiros é uma solução inteligente para os usuários lidarem com a infraestrutura de TI. Nesse sentido, a Computação em Nuvem é uma tendência recente de tecnologia cujo objetivo é proporcionar serviços de TI sob demanda, com pagamento baseado no uso [21].

Sistemas de Gerenciamento de Banco de Dados (SGBDs) são candidatos potenciais para a implantação em nuvem já que, em geral, as instalações destes sistemas são complexas e envolvem uma grande quantidade de dados, ocasionando um custo elevado, tanto em *hardware* quanto em *software* [21]. Além disso, um BD nativo em nuvem é, segundo [22], uma classe de SGBD que não atende ao modelo relacional, sendo esse tipo de SGBD também chamado NoSQL (*Not only* SQL). Este é um termo genérico para uma classe definida de SGBDs não-relacionais que foram projetados para gerenciar grandes volumes de dados e, em geral, utilizam estruturas e interfaces simples [21].

Nesse contexto, o Firebase[6] é uma plataforma de computação em nuvem, criada pela Google, como uma solução para desenvolvedores de *software*. A combinação de recursos no Firebase acelera a integração do BD em nuvem simultaneamente na web e em aplicativos móveis, auxiliando ainda na resolução de tarefas recorrentes a serem realizadas pelos desenvolvedores [23]. O Firestore[7] é o BD do Firebase, sendo um SGBD NoSQL hospedado em nuvem, flexível e escalonável específico para desenvolvimento *mobile* e web, sendo portanto escolhido para ser o BD em nuvem do aplicativo retratado neste trabalho.

O Firebase possui ainda várias outras ferramentas, sendo uma delas o Firebase Auth[8], também utilizado neste aplicativo. O objetivo do Firebase Auth, segundo [24], é fornecer serviços de *back-end* e bibliotecas prontas para a autenticação de usuários, usando senhas, números de telefone, provedores de identidade conhecidos como o Google, Facebook, Twitter, entre outros. Inclui também um sistema de gestão de utilizadores através do qual os desenvolvedores podem habilitar a autenticação do usuário com e-mail e login de senha armazenados no Firestore [25].

[6]https://firebase.google.com/

[7]https://firebase.google.com/docs/firestore?hl=pt-br

[8]firebase.google.com/products/auth

E. System Usability Scale (SUS)

A norma ISO 9241-11, define “usabilidade” como sendo o grau em que um produto é usado por usuários específicos para atingir seus objetivos com eficácia, eficiência e satisfação em um contexto de uso específico. A eficácia relaciona-se com a capacidade de se alcançar os objetivos conforme o esperado. A eficiência está relacionada com os recursos necessários para os usuários alcançarem seus objetivos. O grau de satisfação diz respeito à experiência de usar o sistema interativo no contexto de uso para o qual foi projetado [26].

Atualmente, o questionário padronizado mais amplamente utilizado para avaliação da usabilidade é o System Usability Scale (SUS) [1]. Além disso, é o questionário que apresenta os resultados mais confiáveis, em todos os tamanhos de amostra [27], mesmo com um número relativamente pequeno de participantes [28].

De uma forma geral, o SUS consiste em um questionário composto por dez assertivas em escala Likert de cinco pontos, variando desde “Discordo Fortemente” a “Concordo Fortemente” [29]. As assertivas alternam-se entre positivas e negativas, onde há uma diferença em relação ao cálculo do *score* de cada uma.

Para as assertivas ímpares (positivas) o *score* é dado pela subtração de 1 à resposta do usuário. Já para as assertivas pares (negativas), o *score* é obtido por 5 menos a resposta do usuário. Após obter o *score* de cada assertiva, todos os *scores* são somados e ajustados, multiplicando-se o resultado desta soma por 2,5. Assim, o resultado final será um índice variando de 0 a 100, onde pontuações mais altas indicam melhor usabilidade [29].

Para avaliar se a pontuação do SUS é aceitável, torna-se necessário algum tipo de comparação. Sendo assim, em [30] desenvolveu-se uma escala de notas em que as pontuações do SUS abaixo de 60 representavam “F”, entre 60 e 69 eram “D”, entre 70 e 79 eram “C”, entre 80 e 89 eram “B” e 90 e acima de 90 eram “A”, conforme mostra a Figura 2.

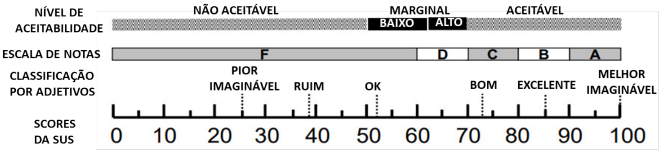

Figura 2: Classificação de Bangor, adaptado de [30]

Destaca-se que está se tornando uma meta comum na indústria atingir um *score* médio na SUS de 80, como evidência de uma experiência de usuário acima da média, pois 80 é um “B” na classificação de Bangor [31].

III. Trabalhos Relacionados

Entre os trabalhos previamente publicados sobre o tema, destacam-se estudos que envolvem ambientes de reserva de

espaços físicos utilizando tecnologias de desenvolvimento *mobile* e web. Todos eles visam reduzir conflitos e facilitar essas práticas. Alguns desses trabalhos são apresentados a seguir.

No trabalho de [32], os autores criaram uma aplicação para reservar salas conforme os horários disponíveis. A lógica se resume da seguinte forma: se a reserva para a sala escolhida - com duração de até 2 horas - estiver disponível, a requisição por este espaço ocorrerá instantaneamente. No entanto, se a reserva for superior a 2 horas, o administrador será notificado e o usuário deverá aguardar a sua resposta. As vantagens mencionadas pelos autores são: redução dos conflitos nas horas de pico; redução de *spammers*, pois permite apenas usuários verificados; uma vez que uma reserva é feita, não é permitido que outro usuário reserve a mesma sala no mesmo dia e horário. No entanto, ainda é necessário que o administrador autorize a reserva, se esta ultrapassar 2 horas.

Em [33], os autores criaram um aplicativo *mobile* para reservas de salas. No sistema desenvolvido, para realizar uma reserva, são necessárias algumas informações, como motivo, e-mail, função do usuário no sistema, data, horário e nome do local, sendo estas obrigatórias para prosseguir com a reserva. Neste caso, os autores evidenciaram o problema de que, para realizar a reserva de alguma sala ou espaço, era necessário que a pessoa enviasse um e-mail, o que ocasionava atrasos e conflitos.

No trabalho de [34], os autores criaram um sistema web com a finalidade de agendamento e reserva de salas. Embora seja um *software* web, ele pode ser utilizado em dispositivos móveis por meio de um navegador. Esse sistema possui dois tipos de usuários: administradores e usuários comuns. Os administradores podem criar novas salas e horários, enquanto os usuários comuns podem apenas reservar as salas disponíveis.

Em [35] é relatado o desenvolvimento de um sistema web para a automatização no processo de alocação de salas. Apesar disso, neste *software* é apresentado uma tela de *login*. Logo em seguida, apresentam-se as salas disponíveis para reserva e a indicação dos blocos aos quais as salas pertencem, além de ser exibido se a sala está disponível ou não.

No trabalho de [36], foi criado um sistema web onde os autores desenvolveram uma plataforma na qual é possível reservar salas. Além disso, disponibiliza reservas no ramo empresarial para empresas cadastradas na plataforma.

No trabalho de [37], os autores criaram uma plataforma web para alocação de reserva de espaços acadêmicos. Neste trabalho, existem três tipos de usuários, sendo eles: funcionário, gestor de espaços e gestor do sistema. As solicitações de reserva podem ter três cores diferentes, sendo elas: verde, que significa que a notificação pode ser aceita e não possui conflito com nenhuma outra solicitação; vermelha, que significa que a notificação não pode ser aceita de imediato, pois há um pedido de reserva em conflito e por último, amarela, que significa que a solicitação está em conflito com uma reserva já aceita.

Em [38], os autores desenvolveram um protótipo com o objetivo de desenvolver futuramente uma ferramenta na forma de uma plataforma web, que poderá ser utilizada por alunos e professores, para reservas de equipamentos e de laboratórios.

Tabela I: Comparativo de trabalhos relacionados

Características	Get Lab	[32]	[33]	[34]	[35]	[36]	**[37]**	**[38]**
Definição de uma arquitetura	**X**			**X**	**X**	**X**	**X**	
Versão para Android	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**
Versão para iOS	**X**	**X**	**X**	**X**	**X**	**X**	**X**	**X**
Versão para Web				**X**	**X**	**X**	**X**	**X**
Uso do Firebase	**X**	**X**	**X**					
Auxílio de um agente conversacional	**X**							

A Tabela I sumariza os trabalhos mencionados nesta Seção em comparação com o trabalho apresentado neste artigo. Embora os sistemas tratados nestes trabalhos tenham trazido vantagens de reservas de espaços aos seus usuários, em nenhum deles há o uso de um agente conversacional e nem o desenvolvimento *mobile* nativo, que é o diferencial da presente pesquisa.

IV. Metodologia

A. Caracterização da Pesquisa

Este estudo ocorreu entre Junho/2022 e Junho/2023, tendo sido desenvolvido na forma de uma pesquisa aplicada de tipo exploratória, com abordagem quali-quantitativa e com delineamento transversal. O procedimento metodológico adotado foi uma pesquisa-ação [39] e teve, como instrumento de coleta de dados, tanto para dados quantitativos como qualitativos, um questionário.

Os dados obtidos no questionário foram tabulados[9]. Para as análises quantitativas, foi empregada a análise do questionário SUS, segundo a proposta de [40]. Os dados qualitativos foram coletados a partir das questões abertas do questionário.

Quanto ao procedimento metodológico, a pesquisa-ação pode ser entendida como um método de pesquisa de natureza participativa, cujo objetivo é buscar uma solução coletiva para uma determinada situação-problema [41]. Na área de desenvolvimento de sistemas, é adequada para investigações que envolvem a avaliação destes ao longo do seu desenvolvimento ou durante sua implantação em um ambiente real [42].

Tal pesquisa é tipicamente realizada em ciclos iterativos que sucessivamente refinam o conhecimento adquirido nos ciclos anteriores. Um ciclo é formado por várias etapas, a saber [43]: “Diagnosticar”, “Planejar a Ação”, “Intervir”, “Avaliar” e “Refletir”, conforme ilustrado na Figura 3.

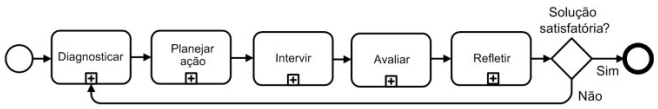

Figura 3: Etapas de um ciclo da pesquisa-ação [42].

No caso desta pesquisa, para obter protótipo funcional, foi executado o primeiro ciclo da pesquisa-ação, com as seguintes etapas [42]:

[9]https://encurtador.com.br/kADG3

1) Na etapa “Diagnosticar”, é feita a identificação e análise dos problemas que motivam a organização a realizar ações para melhorar algum aspecto de seu funcionamento. Esta etapa está descrita na Seção IV-C, com o Levantamento de Requisitos do aplicativo.
2) A etapa “Planejar a Ação” envolve o planejamento das intervenções que serão realizadas para solucionar ou, pelo menos, reduzir os problemas identificados. Esta etapa está descrita na Seção IV-D, com o detalhamento da implementação do GetLab.
3) Na etapa “Intervir” são executadas ações que podem ocorrer de diferentes formas, por exemplo, implantando ou modificando um sistema, envolvendo todos de uma só vez ou apenas um grupo de usuários. Paralelamente, dados qualitativos de diversas fontes são coletados por meio de entrevistas, grupos focais, reuniões etc. A Seção IV-E descreve a primeira avaliação do aplicativo através de um Grupo Focal.
4) A etapa “Avaliar” é aquela em que pesquisadores e demais pessoas envolvidas avaliam os resultados diante dos objetivos, buscando identificar os efeitos decorrentes das ações e até que ponto os problemas foram resolvidos. A avaliação do aplicativo foi feita através da aplicação do questionário com o SUS, descrito na Seção IV-F.
5) A última etapa do ciclo, “Refletir”, é aquela em que é feita uma reflexão das atividades e dos resultados obtidos na pesquisa-ação até o momento. Esse tema é tratado na Seção VI.

B. Público Alvo e Objetivos

No público-alvo da pesquisa estão todos os estudantes, professores, técnicos administrativos e gestores da Instituto Federal de Educação, Ciência e Tecnologia do Ceará, uma vez que todos podem realizar reservas de espaços acadêmicos.

Neste contexto, os objetivos desta pesquisa são desenvolver e avaliar um primeiro protótipo funcional do GetLab, um aplicativo para reserva de espaços acadêmicos por meio de um agente conversacional, de maneira a atender aos usuários de acordo com seus respectivos perfis e necessidades.

Nas próximas seções, serão detalhadas as etapas do desenvolvimento e avaliação deste aplicativo.

C. Levantamento de Requisitos

Esta etapa, que iniciou-se em setembro/2022 e finalizou em fevereiro/2023, teve um caráter exploratório, visando entender e esclarecer as necessidades dos usuários, reunir os requisitos necessários para a implementação do aplicativo GetLab e delimitar o escopo de desenvolvimento. Esse procedimento teve duas etapas: na primeira, foi necessário entender as características dos sistemas anteriores de reserva de salas em funcionamento no campus. Em um segundo momento, foram analisados aplicativos e sistemas disponíveis no mercado e na academia que desempenham funções semelhantes.

O primeiro sistema utilizado no *campus* foi o Google Agenda[10], por meio do qual apenas os professores tinham permissão para agendar salas e laboratórios. No entanto, o sistema não bloqueava as salas já reservadas de maneira que era possível a reserva do mesmo espaço acadêmico no mesmo horário por dois professores distintos, causando conflitos.

O segundo sistema utilizado - e atualmente em vigor - é o Sistema Unificado de Administração Pública (SUAP)[11], onde o problema anterior de conflito de reservas não existe, sendo ainda possível que uma reserva seja feita por qualquer aluno ou funcionário, desde que estes tenham as permissões necessárias. Porém, este é um sistema web em que os usuários do *campus* relatam dificuldades quanto ao acesso das funcionalidades. Além disso, o fato de ser preciso que os administradores aprovem as solicitações de reserva torna o processo burocrático e lento, além de sobrecarregar alguns servidores com esta tarefa adicional.

Em um segundo momento, buscou-se na literatura *softwares* semelhantes, onde os mais relevantes estão listados e comparados na Seção III. No entanto, na maioria dos casos, a marcação de reservas é manual, sem o auxílio de agentes conversacionais para recomendar a sala que melhor se adequa às necessidades dos usuários, sendo esta uma das principais contribuição deste trabalho.

Com isso, após o levantamento dos requisitos nas duas etapas mencionadas, elaborou-se as seguintes funcionalidades a estarem presentes no escopo do GetLab:

- Autenticação via *login*;
- Cadastro e atualização de usuários (alunos, professores e demais servidores);
- Exibição dos espaços acadêmicos e seus horários de disponibilidade para seleção;
- Criação do ambiente “Minha Agenda”, para exibição das salas reservadas pelo usuário;
- Implementação de um Agente Conversacional para recomendação de salas de acordo com as necessidades e perfil do usuário;
- Disponibilização de um ambiente para o administrador criar e atualizar espaços acadêmicos e gerenciar perfis de usuários.

Uma vez definido o escopo, o seguinte passo foi a implementação de um protótipo funcional passível de validação, descrito na próxima Seção.

D. Implementação do GetLab

Nesta etapa - que iniciou em setembro/2022 e finalizou em junho/2023 - foram criados, em paralelo ao levantamento de requisitos, duas versões do GetLab: uma para Android e outra para iOS. A escolha para o desenvolvimento destas duas versões deveu-se a um requisito não-funcional de compatibilidade, ou seja, de ser compatível com vários SO de forma a atender a todos os usuários. Para tanto, utilizou-se duas tecnologias *mobile* nativas: Kotlin (Android) e Swift (iOS).

A ideia do Agente Conversacional surgiu com o estudo de trabalhos relacionados, quando identificou-se uma lacuna ao

[10]https://workspace.google.com/intl/pt-BR/products/calendar/

[11]https://suap.ifce.edu.br/accounts/login/?next=/

constatar que nenhum deles continha essa tecnologia. Este *chat* está disponível a todos os perfis de usuário.

No aplicativo GetLab, decidiu-se que seria necessário implementar outras opções pelas quais o usuário pode realizar a reserva em paralelo ao chat, então implementou-se a reserva seguindo uma seleção de blocos e o agendamento rápido, existindo assim na plataforma três opções para agendar um espaço. Esses fluxos serão mais detalhados na seção V.

E. Grupo Focal

O objetivo desse grupo focal foi promover um *brainstorm* para melhorar a experiência do usuário e identificar novas funcionalidades para o GetLab. Um Grupo Focal é considerado uma técnica de coleta de dados qualitativos, proporcionando a interação em grupo para a produção de dados que seriam menos acessíveis fora do contexto interacional. A partir dessa técnica, é possível coletar dados, diretamente, dos depoimentos de um grupo, que relata suas experiências e percepções, em torno de um tema de interesse coletivo [44].

Para este primeiro momento de coleta de dados, foram convidados todos os servidores do *campus* - entre eles professores, técnicos administrativos e servidores em cargo de gestão - via e-mail e grupos de redes sociais, durante as duas semanas anteriores à intervenção, que ocorreu no dia 12/06/2023.

Nesta fase, houve três etapas importantes para avaliar o protótipo funcional do GetLab, sendo elas: (i) o planejamento da reunião; (ii) a realização do grupo focal, propriamente dito, e (iii) a análise de dados. Para maior conveniência de todos, a reunião deu-se de forma *online* por meio da plataforma Microsoft Teams[12], o que permitiu gravar e transcrever todas as falas dos participantes. Nesta ocasião, participaram 4 professores, 1 técnico-administrativo e 1 servidor em cargos de gestão em uma reunião que se estendeu por 1:30h.

No planejamento da reunião, realizou-se o *setup* prévio para que os servidores pudessem testar o aplicativo, na forma de protótipo funcional, em seus respectivos dispositivos. Para tanto, foi disponibilizado acesso livre à plataforma *Sauce Labs*[13], que ofereceu um ambiente virtual para testes.

Durante o grupo focal, explicou-se como seria conduzida a reunião, apresentando aos participantes como eles poderiam usar o *Sauce Labs* para emular o GetLab. Além disso, o próprio aplicativo e suas funcionalidades foram apresentadas por um dos autores deste artigo, para um primeiro contato com a ferramenta. Também foi explicado que os participantes poderiam anotar suas dúvidas e possíveis melhorias durante a apresentação para que, ao final, pudessem manifestar sua opinião e sugestões de melhoria.

Após a reunião, durante a etapa de análise de dados, as principais dúvidas e sugestões foram listadas e analisadas. Entre as principais melhorias sugeridas, destacam-se: alterar a paleta de cores do aplicativo para contemplar as cores da instituição; importar os dados de *login*/senha do Sistema Acadêmico em vigor na instituição para evitar o cadastro individual de usuários; incluir a possibilidade de cadastrar usuários em lote; validar os *e-mails* durante o cadastro de usuários para evitar usuários que não são do *campus*; incluir confirmação do cancelamento de uma reserva; dar a opção de atalho para incluir salas logo após a inclusão de um novo bloco no sistema e integrar com os demais sistemas utilizados no *campus*, como o SUAP.

Após a análise das sugestões, como melhoria neste primeiro momento, a paleta de cores foi alterada para contemplar as cores da instituição. As demais melhorias seriam contempladas em um segundo ciclo da pesquisa-ação a ser realizado como trabalho futuro.

Após a inclusão desta alteração, foi realizado um teste de usabilidade incluindo também o corpo discente, como detalhado na próxima Seção.

F. Avaliação do Aplicativo

Para a avaliação do aplicativo, utilizamos o questionário SUS, traduzido para o português [45]. O teste de usabilidade consiste de 10 assertivas (Tabela II), adaptadas para o contexto da avaliação do GetLab e disponibilizadas por meio de um formulário Google Forms[14].

Tabela II: Assertivas do SUS adaptadas para o GetLab

01	Acho que gostaria de utilizar o GetLab com frequência.
02	Considerei o GetLab mais complexo do que necessário.
03	Achei o GetLab fácil de utilizar.
04	Acho que necessitaria de ajuda de um técnico para conseguir utilizar o GetLab.
05	Considerei que as varias funcionalidades do GetLab estavam bem integradas.
06	Achei que o GetLab tinha muitas inconsistências.
07	Suponho que a maioria das pessoas aprenderia a utilizar o GetLab.
08	Considerei o GetLab muito complicado de utilizar.
09	Senti-me muito confiante a utilizar o GetLab.
10	Tive que aprender muito antes de conseguir lidar com o GetLab.

Além das perguntas do SUS, o formulário incluiu, no início, um Termo de Consentimento Livre e Esclarecido (TCLE) no qual foi informado aos respondentes que a sua participação não era obrigatória e que o seu anonimato e o de suas respostas seria preservado, bem como os benefícios e riscos de se participar desta pesquisa.

Além disso, no final do questionário foram incluídas três perguntas abertas e opcionais onde cada respondente poderia informar: (1) o que mais gostou no GetLab; (2) o que poderia melhorar e (3) se haveria alguma sugestão a ser incorporada no aplicativo.

Este questionário esteve disponível durante um período de 3 dias, de 14 a 17/06/2023. Para este público, foi disponibilizada apenas a versão para Android do aplicativo. Foi então enviado um *e-mail* convidando os discentes e servidores a participarem da avaliação do GetLab, em que neste e-mail estavam anexados o APK (*Android Package*) do aplicativo, um vídeo sobre como realizar a instalação do APK no *smartphone* e um formulário Google com a avaliação de usabilidade,

[12]https://www.microsoft.com/pt-br/microsoft-teams/

[13]https://saucelabs.com/

[14]https://encurtador.com.br/ciyF9

incluindo as questões do SUS. Os usuários foram convidados a navegar livremente na aplicação e, em seguida, responder ao questionário de avaliação.

V. Aplicativo GetLab para Reserva de Espaços Acadêmicos

O GetLab é um aplicativo Android/iOS para reserva de espaços compartilhados como laboratórios, salas de aula, auditórios, quadras de esporte e quaisquer outros ambientes de instituições educacionais que sejam abertos ao uso da comunidade acadêmica. O objetivo principal desta ferramenta é oferecer um melhor planejamento do uso coletivo dos espaços físicos. Para tanto, contribui para o gerenciamento eficiente destes recursos por estudantes, professores e demais funcionários da instituição, permitindo verificar a disponibilidade dos espaços em tempo real e recomendando o mais adequado, de acordo com as necessidades e perfil do usuário.

Nas seções seguintes serão melhor detalhadas as funcionalidades disponíveis no aplicativo, bem como algumas de suas regras de negócio.

A. Perfis de Usuários

O fluxo dos usuários no GetLab depende do perfil de cada usuário, que podem ser de três tipos: estudante, professor e administrador, conforme mostrado na Figura 4.

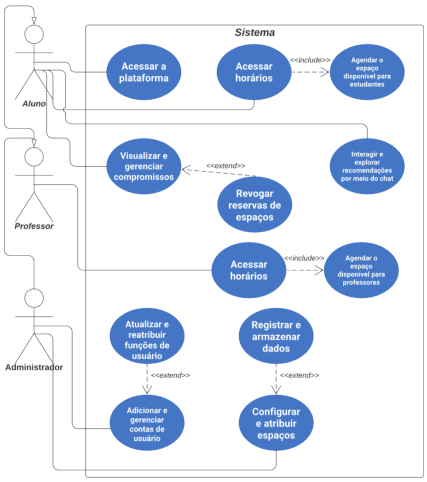

Figura 4: Diagrama de Casos de Uso do GetLab

Dependendo do perfil do usuário, ele terá permissão para reservar espaços específicos. Por exemplo, o estudante só poderá reservar espaços designados para a reserva de estudantes. O professor terá a possibilidade de reservar espaços designados tanto para a reserva de estudantes como para a de professores. Já o administrador, além de poder reservar espaços destinados para a reserva de alunos e professores, também terá a possibilidade de gerenciar espaços acadêmicos e alterar os perfis de usuários.

B. Fluxos Possíveis de Reserva

O fluxo que os usuários podem percorrer no GetLab é definido pelas formas pelas quais podem-se realizar reservas no GetLab, que são: (i) Fluxo de Reserva escolhendo-se o bloco; (ii) Agendamento Rápido, sem a escolha do bloco, e (iii) por meio do Agente Conversacional (*Chat*), conforme a tela inicial do aplicativo (Figura 5). Na Figura 6, são ilustradas todas as possibilidades de fluxo de reserva do GetLab nesta primeira versão.

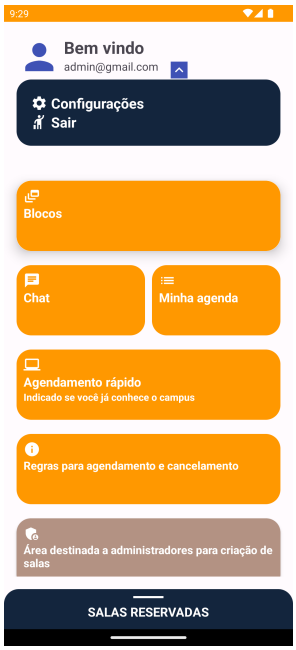

Figura 5: Tela inicial do GetLab

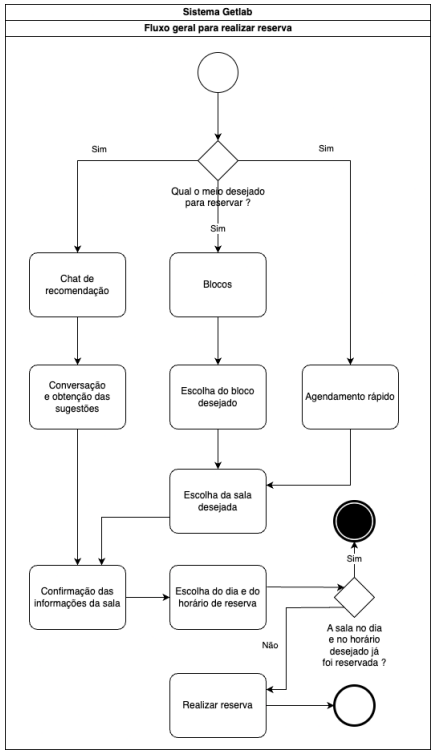

Figura 6: Fluxos possíveis de reserva do GetLab.

No Fluxo de Reserva padrão (por Blocos), logo após o *login*, o usuário escolhe a opção "Blocos" na tela inicial e, em seguida, é direcionado a uma tela para a escolha do bloco didático desejado. Na sequência, é solicitado que se escolha o espaço físico - pertencente ao bloco escolhido previamente - de sua preferência. Em seguida, é exibida uma tela com as características do espaço, para que o usuário possa conferir e confirmar. Em seguida, é levado a escolher o dia e o horário da reserva. Se escolher um dia e horário já reservado previamente, receberá uma mensagem informando que a reserva não poderá ser efetuada. Caso contrário, poderá concluir a realização da reserva. No Agendamento Rápido e por meio do Agente Conversacional, pula-se a etapa de escolha dos blocos, indo diretamente para a escolha de salas.

No fluxo de reserva via Agente Conversacional, ao clicar na opção de *Chat* (Figura 7), inicia-se uma interação baseada em regras. Neste primeiro momento, a interação corresponde a duas perguntas. A primeira é: "Olá, deixe-me ajudar você a escolher uma sala. Você deseja um espaço com quantos lugares?". Em seguida, são apresentadas opções para o usuário escolher a quantidade de assentos desejada. Na segunda pergunta do *chat*, segue a interação: "Ótimo, agora escolha a quantidade de computadores que a sala possui?". O usuário pode então escolher a quantidade de computadores desejada, e em seguida são mostradas sugestões de espaços acadêmicos com base nas informações fornecidas pelo usuário e de acordo com as possibilidades de reserva para o seu perfil, permitindo que este siga o processo de reserva de salas. Quando o usuário clicar em uma das salas recomendadas pelo *Chat*, segue-se o mesmo fluxo de reserva do Agendamento Rápido.

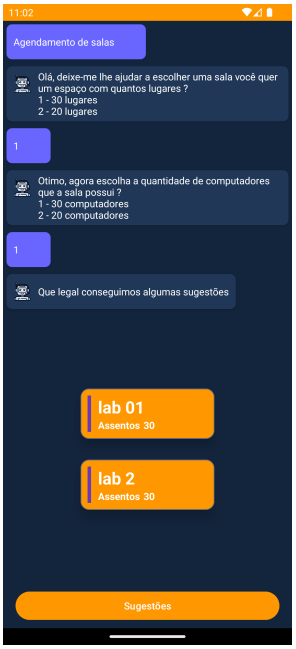

Figura 7: Sugestões do Agente Conversacional

Com o Agente Conversacional, o tempo necessário para agendamento da sala diminui, pois o usuário não precisa consultar os espaços disponíveis um a um antes de conseguir efetivar uma reserva. Nesse caso, os espaços passíveis de agendamento já serão recomendados, de acordo com as respostas fornecidas às perguntas, e o usuário poderá escolher entre as opções possíveis a que mais lhe convier.

VI. Resultados e Discussões

Esta seção tem como objetivo relatar e discutir os resultados obtidos após a aplicação do Teste de Usabilidade na forma de um questionário com as assertivas do instrumento SUS. Como resposta a este questionário, obteve-se 18 respostas: 15 de estudantes, 2 de docentes e 1 de servidor ocupante de cargo de gestão (Figura 8). Este quantitativo foi suficiente, uma vez que o SUS tem se mostrado robusto, mesmo com um número relativamente pequeno de participantes, como 8 ou 10 [28].

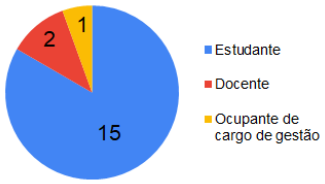

Figura 8: Público de participantes do Teste de Usabilidade

O cálculo da pontuação individual do SUS para cada participante foi calculado da seguinte forma: Para os itens 1, 3, 5, 7 e 9 (ímpares), a contribuição da pontuação é a posição da escala menos 1. Para os itens 2, 4, 6, 8 e 10 (pares), a contribuição é 5 menos a posição da escala. Em seguida, é somada a pontuação de todos os itens e o valor resultante é ajustado, multiplicando-se por 2,5. Assim, obtemos um valor de 0 a 100 na escala SUS [29].

Para obter o *score* total da aplicação, fazemos uma média dos resultados de todos os participantes. Para o GetLab, foi obtida uma pontuação de 89,4 (Figura 9) o que, segundo a Classificação de Bangor [30], apresentada na Seção II-E, corresponde a um sistema tido como "Aceitável", com nota "B" e sendo classificado como "Excelente".

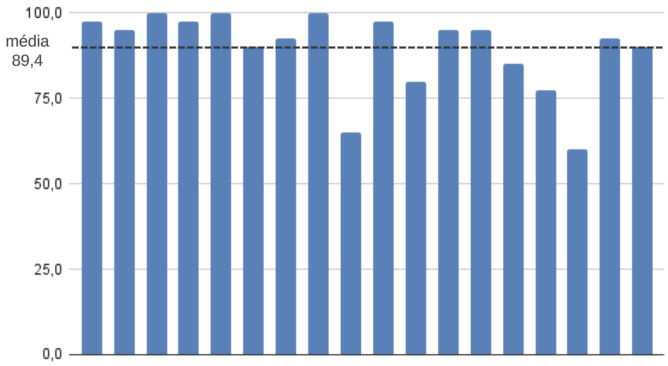

Figura 9: Gráfico da pontuação SUS para o GetLab

Para entender melhor este resultado, analisamos também as respostas abertas do questionário. Identificando os 18 partici-

pantes com rótulos de P01 a P18, encontramos as seguintes opiniões:

- Para a pergunta "O que mais gostou no GetLab?", os usuários responderam que: acharam a *interface* fácil de usar e/ou intuitiva (P02, P05, P06, P08, P11, P14, P17 e P18) mesmo no primeiro uso (P12); também acharam que o Agente Conversacional facilita o processo de reserva (P03); comentam que o *design* da interface está bem organizado (P08, P18); gostaram da possibilidade de agendar salas antecipadamente (P07) e que acharam que o aplicativo poderia otimizar as reservas de salas e laboratórios (P08).
- Para a pergunta "O que poderia melhorar no GetLab?", os respondentes mencionaram que: as cores da paleta poderiam ter combinado melhor entre si (P18); o Assistente Conversacional não retornou resultados corretos (P07, P14); a *interface/design* poderia ser melhor (P06, P14, P15, P17); poderia adicionar mais recursos (P02); o aplicativo travou algumas vezes (P09, P14), apresentou erros (P16) ou tem partes que não funcionam (P14, P17).
- Para a pergunta "Você teria alguma sugestão a ser incorporada no GetLab?", alguns usuários solicitaram: que houvesse um tutorial para explicar as funções do aplicativo (P08); melhorar a responsividade para a tela do celular na horizontal (P09); integrar com os sistemas já existentes no *campus* (P10, P11) e ter a opção de arrastar as opções com o dedo (P14, P17).

Considerando que este é um primeiro protótipo funcional, as sugestões serão consideradas para uma segunda versão do aplicativo.

VII. Conclusões e Trabalhos Futuros

Neste trabalho, apresentamos o desenvolvimento e avaliação do GetLab, um aplicativo para reserva de espaços acadêmicos com o auxílio de um agente conversacional, idealizado de maneira a atender à comunidade acadêmica de acordo com seus respectivos perfis e necessidades.

Como principais resultados, foi considerado "Excelente" após o teste de usabilidade constituído das assertivas do SUS, um instrumento padronizado amplamente utilizado para este fim, embora - segundo o relato dos usuários - ainda precise de correções e melhorias, que serão tratadas nas próximas versões.

Como trabalhos futuros, em um segundo ciclo do método de pesquisa-ação, serão incluídas as sugestão de *interface* fornecidas durante o grupo focal e nas respostas às questões abertas do questionário avaliativo, bem como serão disponibilizados tutoriais e telas de ajuda para suporte aos usuários. Também serão corrigidos alguns *bugs* identificados pelos usuários e mais funcionalidades serão incorporadas, assim como a integração com outros sistemas do *campus*, para uma maior interoperabilidade.

Referências

[1] J. R. Lewis, "The system usability scale: past, present, and future," *International Journal of Human–Computer Interaction*, vol. 34, no. 7, pp. 577–590, 2018.

[2] V. G. C. Nunes, "Avaliação de abordagens e definição de diretrizes para o desenvolvimento de aplicativos mobile: um relato de experiência utilizando aabordagem não nativa de desenvolvimento.," 2021.

[3] L. Delia, N. Galdamez, L. Corbalan, P. Pesado, and P. Thomas, "Approaches to mobile application development: Comparative performance analysis," in *2017 Computing conference*, pp. 652–659, IEEE, 2017.

[4] P. Nawrocki, K. Wrona, M. Marczak, and B. Sniezynski, "A comparison of native and cross-platform frameworks for mobile applications," *Computer*, vol. 54, no. 3, pp. 18–27, 2021.

[5] P. Que, X. Guo, and M. Zhu, "A comprehensive comparison between hybrid and native app paradigms," in *2016 8th International Conference on Computational Intelligence and Communication Networks (CICN)*, pp. 611–614, IEEE, 2016.

[6] "Introdução ao desenvolvimento nativo do android no windows." https://learn.microsoft.com/pt-br/windows/android/native-android, note = Acessado em: 25-06-2023.

[7] "Guia inicial desenvolvimento android." https://www.alura.com.br/artigos/o-guia-inicial-desenvolvimento-android. Acessado em: 21-06-2023.

[8] "Conheça a linguagem de programação kotlin." https://developer.android.com/kotlin/learn?hl=pt-br-null-safety. Acessado em: 21-06-2023.

[9] G. F. Paim, I. J. B. Santos Júnior, *et al.*, "Comparativo de performance entre aplicativos móveis nativos e multiplataforma," 2022.

[10] A. C. Varoto, "Visões em arquitetura de software," Master's thesis, Instituto de Matemática e Estatística, University of São Paulo, São Paulo, 2002.

[11] X. Li, D. Chang, H. Pen, X. Zhang, Y. Liu, and Y. Yao, "Application of mvvm design pattern in mes," pp. 1374–1378, 06 2015.

[12] C. Eidhof, M. Gallagher, and F. Kugler, *App Architecture: IOS Application Design Patterns in Swift*. North Charleston, SC, USA: CreateSpace Independent Publishing Platform, 1st ed., 2018.

[13] "Model-view-viewmodel(mvvm)." https://github.com/ahmedeltaher/MVVM-Kotlin-Android-Architecture. Acessado em: 21-06-2023.

[14] A. S. R. Barbosa *et al.*, "Análise comparativa entre os padrões mvc, mvp, mvvm e mvi na plataforma android," 2022.

[15] A. F. Cesar, "Uma análise comparativa entre os padrões mvp e mvvm na plataforma android," 2019.

[16] L. M. R. Tarouco, P. F. d. Silva, and F. Herpich, "Cognição e aprendizagem em mundo virtual imersivo," 2020.

[17] P. C. d. A. R. Tede and F. de Almeida Barros, "Agentes inteligentes conversacionais: conceitos básicos e desenvolvimento," *Sociedade Brasileira de Computação*, 2016.

[18] R. Kar and R. Haldar, "Applying chatbots to the internet of things: Opportunities and architectural elements," *arXiv preprint arXiv:1611.03799*, 2016.

[19] C. J. Date, *Introdução a sistemas de bancos de dados*. Elsevier Brasil, 2004.

[20] C. A. Crummenaurer, "Motivações para o uso de sistemas nosql," 2020.

[21] F. R. Sousa, L. O. Moreira, J. A. F. d. Macêdo, and J. C. Machado, "Gerenciamento de dados em nuvem: Conceitos, sistemas e desafios," *Topicos em sistemas colaborativos, interativos, multimidia, web e bancos de dados, Sociedade Brasileira de Computacao*, pp. 101–130, 2010.

[22] P. J. Sadalage and M. Fowler, *NoSQL distilled: a brief guide to the emerging world of polyglot persistence*. Pearson Education, 2013.

[23] M. Tram, "Firebase," 2019.

[24] P. F. Peixoto, "Moita redonda: um estudo sobre o desenvolvimento cross-platform no estímulo de produção e comércio de artesanato no ceará," 2020.

[25] C. Khawas and P. Shah, "Application of firebase in android app development-a study," *International Journal of Computer Applications*, vol. 179, no. 46, pp. 49–53, 2018.

[26] S. Barbosa, B. Silva, M. Silveira, I. Gasparini, T. Darin, and G. Barbosa, "Interação humano-computador e experiencia do usuário. autopublicação," 2021.

[27] T. Tullis and J. Stetson, "A comparison of questionnaires for assessing website usability abstract: Introduction," in *Usability Prof. Assoc. Conf*, pp. 1–12.

[28] B. Albert and T. Tullis, *Measuring the user experience: collecting, analyzing, and presenting usability metrics*. Newnes, 2013.

[29] J. Brooke *et al.*, "Sus-a quick and dirty usability scale," *Usability evaluation in industry*, vol. 189, no. 194, pp. 4–7, 1996.

[30] A. Bangor, P. Kortum, and J. Miller, "Determining what individual sus scores mean: Adding an adjective rating scale," *Journal of usability studies*, vol. 4, no. 3, pp. 114–123, 2009.
[31] J. Sauro and J. R. Lewis, *Quantifying the user experience: Practical statistics for user research*. Morgan Kaufmann, 2016.
[32] A. Praveen, K. Nanda, N. Rajith, N. Giriraj, R. Radhika, N. Mahesh, K. Vishnu, T. Anjali, and S. Sarath, "Conference room booking application using flutter," in *2020 International Conference on Communication and Signal Processing (ICCSP)*, pp. 0348–0350, IEEE, 2020.
[33] H. Singh and R. Shah, "Bookiiit - designing a venue booking system (technical demo)," pp. 287–291, 09 2020.
[34] R. Oliveira, "Um aplicativo online para reservas de sala: o case do cinted/ufrgs. universidade federal do rio grande do sul," *Monografia. Disponível em:¡ https://www. lume. ufrgs. br/handle/10183/185080¿. Acesso em*, vol. 8, 2019.
[35] P. F. F. Lima, "Desenvolvimento de sistema para automatização de alocação de salas no centro de tecnologia da universidade federal do ceará," 2020.
[36] J. Cruz and D. Marques, "Reserva já: Solução em reservas de salas," Master's thesis, Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, São Paulo, 2018.
[37] C. C. H. Nakai, J. E. Fernandes, and M. J. Pereira, "Plataforma de gestão de espaços numa escola de ensino superior," in *16th Iberian Conference on Information Systems and Technologies (CISTI 2021)*, pp. 1–6, 2021.
[38] P. C. A. Barzaga, J. D. German, G. O. Binoya, S. Bucao, S. Ibe, and D. Yap, "ereserba cardinal: An integrated room reservation system for higher education institutions," in *Proceedings of the International Conference on Industrial Engineering and Operations Management, Dubai*, pp. 10–12, 2020.
[39] M. Thiollent, *Metodologia da pesquisa-ação*. Cortez editora, 2022.
[40] R. P. da Costa, A. F. S. dos Santos, and C. P. Santiago, "Análise de usabilidade do sistema q-acadêmico utilizando o método system usability scale (sus): Um estudo de caso," in *Anais do XIV Encontro Unificado de Computação do Piauí e XI Simpósio de Sistemas de Informação*, pp. 231–238, SBC, 2021.
[41] D. Filippo, G. Roque, and S. Pedrosa, "Pesquisa-ação: possibilidades para a informática educativa," *Metodologia de Pesquisa Científica em Informática na Educação: Abordagem qualitativa de Pesquisa*, vol. 3, 2018.
[42] D. Filippo, "Pesquisa-ação em sistemas colaborativos," *Sistemas Colaborativos*, vol. 1, 2011.
[43] R. M. Davison, "An action research perspective of group support systems: how to improve meetings in hong kong," *Unpublished PhD Dissertation, City University of Hong Kong*, 1998.
[44] D. S. Backes, J. S. Colomé, R. H. Erdmann, and V. L. Lunardi, "Grupo focal como técnica de coleta e análise de dados em pesquisas qualitativas," *O mundo da saúde*, vol. 35, no. 4, pp. 438–442, 2011.
[45] A. I. Martins, A. F. Rosa, A. Queirós, A. Silva, and N. P. Rocha, "European portuguese validation of the system usability scale (sus)," *Procedia computer science*, vol. 67, pp. 293–300, 2015.