UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

Rafael Lazzaretti Madalóz

Desenvolvimento de aplicativo móvel para consulta de informações acadêmicas para alunos de graduação da UFSC

Florianópolis - SC
2023

Rafael Lazzaretti Madalóz

Desenvolvimento de aplicativo móvel para consulta de informações acadêmicas para alunos de graduação da UFSC

Trabalho de conclusão de curso apresentado como parte dos requisitos para obtenção do grau de Bacharel em Ciências da Computação.
Orientador: Prof. Frank Augusto Siqueira, Dr.

Florianópolis - SC
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Madalóz, Rafael
Desenvolvimento de aplicativo móvel para consulta de informações acadêmicas para alunos de graduação da UFSC / Rafael Madalóz ; orientador, Frank Siqueira, coorientador, Mateus Silva, 2023.
90 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Ciências da Computação, Florianópolis, 2023.

Inclui referências.

1. Ciências da Computação. 2. Aplicativo Móvel. 3. Integração de Sistemas. 4. Usabilidade. 5. Experiência do Usuário. I. Siqueira, Frank. II. Silva, Mateus. III. Universidade Federal de Santa Catarina. Graduação em Ciências da Computação. IV. Título.

Rafael Lazzaretti Madalóz

Desenvolvimento de aplicativo móvel para consulta de informações acadêmicas para alunos de graduação da UFSC

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Bacharel em Ciências da Computação" e aprovado em sua forma final pelo Curso de Graduação em Ciências da Computação.

Florianópolis - SC, 29 de Junho de 2023.

Prof. Álvaro Júnior Pereira Franco, Dr.
Coordenador do Curso

Banca Examinadora:

Prof. Frank Augusto Siqueira, Dr.
Orientador

Prof. Jean Carlo Rossa Hauck, Dr.
Avaliador
Universidade Federal de Santa Catarina

Prof. Ricardo Pereira e Silva, Dr.
Avaliador
Universidade Federal de Santa Catarina

Aos meus pais, pelo apoio incondicional ao longo da minha jornada acadêmica.

AGRADECIMENTOS

Gostaria de expressar meus sinceros agradecimentos a todas as pessoas que contribuíram de forma significativa para a realização deste trabalho de conclusão de curso.

Primeiramente, gostaria de agradecer aos meus pais, Dilva Lazzaretti Madalóz e Natal Paulo Madalóz, por todo o apoio incondicional ao longo de minha jornada acadêmica.

Gostaria também de estender meu agradecimento ao Francisco Camerini, por ceder as chaves de autenticação que foram essenciais para os testes no desenvolvimento do aplicativo. Sua colaboração foi de extrema importância para o sucesso deste projeto.

Em memória do professor Dr. Leandro José Komosinski, meu orientador na disciplina de Introdução ao TCC, gostaria de prestar uma homenagem especial. Sua dedicação, conhecimento e apoio foram fundamentais para o início deste trabalho. Infelizmente, ele nos deixou prematuramente devido à Covid-19, mas sua influência e ensinamentos continuam vivos em nossas memórias.

Agradeço também ao professor Dr. Mateus Grellert Da Silva, meu orientador durante o segundo semestre de 2021 na disciplina de TCC1. Sua orientação foi essencial para o progresso do trabalho.

Por fim, gostaria de expressar minha gratidão ao professor orientador Dr. Frank Augusto Siqueira, que gentilmente aceitou dar continuidade como meu orientador na disciplina de TCC2. Sua orientação, expertise e paciência foram inestimáveis para a conclusão deste trabalho.

“A usabilidade deve ser vista como uma parte integrante da engenharia de software, buscando entregar soluções eficientes e amigáveis aos usuários.“
(Jakob Nielsen, 1993)

RESUMO

Com o avanço constante da tecnologia, o desenvolvimento de aplicativos vem se tornando uma tendência cada vez mais presente no cotidiano. Essas soluções digitais têm como objetivo simplificar e otimizar tarefas do dia a dia, proporcionando uma experiência mais conveniente e eficiente para os usuários. No contexto da Universidade Federal de Santa Catarina (UFSC) existem diversos sistemas dispersos em diferentes domínios que auxiliam os alunos de graduação no gerenciamento de suas atividades acadêmicas e rotinas. No entanto, nem todos esses sistemas foram projetados e adaptados para dispositivos móveis, o que acaba limitando a acessibilidade e a usabilidade para os usuários que dependem de seus smartphones para acessar informações e realizar tarefas importantes relacionadas à vida acadêmica. Dessa forma, este projeto propõe o desenvolvimento de um novo aplicativo que aborde essa lacuna existente, com uma solução capaz de oferecer uma interface gráfica intuitiva, amigável e adaptada para dispositivos móveis, integrando-se harmoniosamente com os sistemas e serviços já existentes na universidade para consultas de informações importantes do dia-a-dia do estudante acadêmico, proporcionando uma experiência mais fluida e completa. Além disso, uma pesquisa para avaliar a usabilidade e a satisfação dos usuários com o novo aplicativo desenvolvido foi realizada, sendo possível compreender as percepções e preferências dos mesmos, fundamental para aprimorar continuamente a solução e garantir que ela atenda efetivamente às demandas e expectativas dos usuários.

Palavras-chave: Aplicativo móvel. Integração de sistemas. Usabilidade. Experiência do usuário.

ABSTRACT

With the constant advancement of technology, application development has become an increasingly present trend in everyday life. These digital solutions aim to simplify and streamline day-to-day tasks, providing a more convenient and efficient experience for users. In the context of UFSC there are several systems dispersed in different domains that help undergraduate students in the management of their academic activities and routines. However, not all of these systems were designed and adapted for mobile devices, which ends up limiting accessibility and usability for users who depend on their smartphones to access information and perform important tasks related to academic life. In this way, this project proposes the development of a new application that addresses this existing gap, with a solution capable of offering an intuitive, user-friendly graphical interface adapted for mobile devices, harmoniously integrating with the existing systems and services at the university to consultations of important information of the academic student's day-to-day, providing a more fluid and complete experience. In addition, a survey was carried out to assess the usability and satisfaction of users with the newly developed application, making it possible to understand their perceptions and preferences, which is essential to continuously improve the solution and ensure that it effectively meets the demands and expectations of users.

Keywords: Mobile app. Systems integration. Usability. User experience.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

AOT	*Ahead Of Time*
CAGR	Sistema de Controle Acadêmico da Graduação
CDN	Content Delivery Network
CRUD	*Create, Read, Update, Delete*
DOM	*Document Object Model*
IU	Interface de Usuário
JIT	*Just In Time*
JSON	Notação de Objetos JavaScript
PWA	*Progressive Web App*
RSS	*Rich Site Summary*
RU	Restaurante Universitário
SDK	Kit de Desenvolvimento de Software
SSL	Secure Sockets Layer
SUS	System Usability Scale
UFSC	Universidade Federal de Santa Catarina
URI	*Uniform Resource Identifier*
URL	*Uniform Resource Locator*
VM	Máquina Virtual

SUMÁRIO

1 INTRODUÇÃO

A internet vem revolucionando o dia a dia de todas as pessoas, e nos últimos anos os dispositivos móveis estão se destacando cada vez mais como principal forma de acesso a essa ferramenta. No Brasil, 79,1% dos domicílios têm acesso à rede, e destes, 98,1% utilizam dispositivos móveis para se conectar na internet, seja para trocar e-mails ou assistir vídeos (IBGE, 2019).

Dentro dos campus da UFSC a realidade não é diferente, e os alunos possuem uma gama de sistemas na internet para controlar as principais informações necessárias da universidade. Para que possam verificar seus horários de aula, é necessário entrarem e fazerem a autenticação no Sistema de Controle Acadêmico da Graduação (CAGR). Já se quiserem ver o cardápio do almoço do Restaurante Universitário (RU), precisam acessar o site do restaurante. Caso necessitem ver a data da próxima prova de uma determinada disciplina ou saberem a nota de uma prova, é necessário utilizar o Moodle. Se quiserem saber as novidades que estão acontecendo, precisam encontrar o site de notícias da UFSC.

Por mais que existam vários sistemas funcionais para diversas tarefas, alguns não são responsivos e possuem uma interface antiga, ou seja, existe uma dificuldade de ler o conteúdo a partir de um dispositivo móvel. Portanto, uma aplicação que una as funcionalidades desses diversos sistemas com uma interface responsiva seria benéfica para os estudantes.

Já existiram alguns aplicativos desenvolvidos para mitigar esse problema. Um dos mais famosos, Minha UFSC, registra uma marca de mais de 10 mil instalações na loja de aplicativos da Google Play Store. Entretanto, todos tiveram problemas e hoje nenhum está disponível e funcional.

Um trabalho de conclusão de curso recente, conduzido pelos ex-alunos Adson Perereira Leal e Angelo Manoel De Matos Leal (LEAL; LEAL *et al.*, 2019), resultou no desenvolvimento de um aplicativo denominado Scholar, voltado para os estudantes de graduação. Esse projeto trouxe consigo um conjunto de funcionalidades relevantes, como a visualização da grade de horários diretamente do CAGR, acesso ao cardápio do RU e recebimento de notificações. Nesse contexto, o presente trabalho tem como base o projeto Scholar, com o propósito de solucionar os problemas identificados, que levaram à inoperância do aplicativo, e ainda implementar novos recursos que aprimorem a experiência dos usuários.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O objetivo deste trabalho é desenvolver um aplicativo móvel abrangente que ofereça aos estudantes de graduação da UFSC acesso a diversas informações relevantes obtidas do *webservice* do CAGR, que inclui dados das disciplinas matriculadas no semestre atual do aluno, e também do *webservice* do Moodle, que inclui dados das notas das avaliações e dados de

eventos do calendário, sendo necessário criar um sistema integrado que possibilite a coleta e a atualização regular dos dados para consulta dos alunos.

Também, através do aplicativo, os estudantes devem visualizar de forma prática e conveniente os cardápios dos restaurantes de todos os campi, permitindo que planejem suas refeições de acordo com suas preferências e necessidades alimentares.

Uma limitação identificada em aplicações anteriores é a ausência de integrações com outros sistemas, o que resultava na necessidade dos usuários cadastrarem manualmente eventos e datas de provas. Essa tarefa demandava tempo e esforço dos estudantes, além de aumentar o risco de erros e esquecimentos.

Com o objetivo de superar essa dificuldade, o aplicativo desenvolvido neste trabalho deve fazer com que os dados sejam automaticamente sincronizadas para consulta dos usuários.

Além das funcionalidades mencionadas anteriormente, este trabalho também tem como objetivo uma avaliação de usabilidade do aplicativo, por meio de uma pesquisa de satisfação, na qual os estudantes são convidados a fornecer suas opiniões e experiências com relação à usabilidade e à eficácia do aplicativo.

1.1.2 Objetivos Específicos

a) Desenvolver um aplicativo utilizando boas práticas de usabilidade;

b) Integrar informações das disciplinas vinculadas ao sistema CAGR, como disciplinas e grade de horários;

c) Integrar informações das disciplinas vinculadas ao sistema Moodle, como notas e calendário de eventos;

d) Introduzir as notícias vinculadas ao *Rich Site Summary* (RSS) da UFSC, para que o aluno sempre fique informado das novidades acadêmicas;

e) Hospedar o aplicativo na loja Google Play Store, para que possa ser encontrado com maior facilidade;

f) Criar e aplicar um questionário aos usuários do aplicativo para avaliar a usabilidade.

1.2 JUSTIFICATIVA

Atualmente a UFSC utiliza mais de um sistema para que os alunos possam gerenciar sua graduação, os mais utilizados são: CAGR, onde ocorrem as matrículas e o aluno acessa sua grade de horários; Moodle, utilizado para acessar informações de cada disciplina, como provas e trabalhos e suas respectivas notas;

Diante disso, já houveram tentativas no desenvolvimento de ferramentas que integrassem esses principais sistemas, porém todos necessitavam que o usuário cadastrasse informações manualmente, sem sincronização automática com outros sistemas. Vale destacar que o CAGR é

um sistema que funciona bem em computadores, mas não é responsivo para dispositivos móveis, o que dificulta o acesso.

2 FUNDAMENTAÇÃO TEÓRICA

Neste Capítulo, são abordados os conceitos essenciais que foram utilizados no desenvolvimento do trabalho. Cada subseção define e descreve as características necessárias para o entendimento dos mesmos.

2.1 DESENVOLVIMENTO DE APLICATIVOS MÓVEIS

A maneira de desenvolver aplicativos mudou drasticamente ao longo dos anos. Inicialmente, não havia mercados de onde os usuários pudessem obter aplicativos, no entanto, em março de 2008, a App Store foi lançada pela Apple e, em outubro do mesmo ano, a Play Store foi lançada para Android. Essas lojas separaram sites da web e aplicativos nativos e os vincularam às suas respectivas plataformas. Conforme as tecnologias progrediram, uma nova forma de desenvolvimento foi apresentada ao mundo na forma do Apache Cordova em 2009. O Apache Cordova foi posteriormente vendido, renomeado e se tornou uma das plataformas de desenvolvimento mais populares para aplicativos híbridos (ADETUNJI *et al.*, 2020).

Atualmente existem quatro (4) maneiras de desenvolver um aplicativo móvel, levando a quatro tipos diferentes de aplicativos: nativo, web, híbrido e *Progressive Web App* (PWA).

2.1.1 Aplicativo Nativo

São aplicativos desenvolvidos com ferramentas e linguagens de programação dedicadas a uma determinada plataforma móvel (EL-KASSAS *et al.*, 2017), portanto, os programadores devem estar em conformidade com as linguagens e ferramentas específicas necessárias para desenvolver o aplicativo com sucesso.

Uma grande desvantagem desta abordagem de desenvolvimento é a fragmentação da plataforma móvel, o que significa que para uma empresa de desenvolvimento atingir mais público em várias plataformas, deve haver o 're-desenvolvimento' do mesmo aplicativo em diferentes tecnologias e ferramentas específicas para cada plataforma desejada. Isso leva a um aumento no tempo, custo, esforço, manutenção e baixa portabilidade (MALAVOLTA *et al.*, 2017).

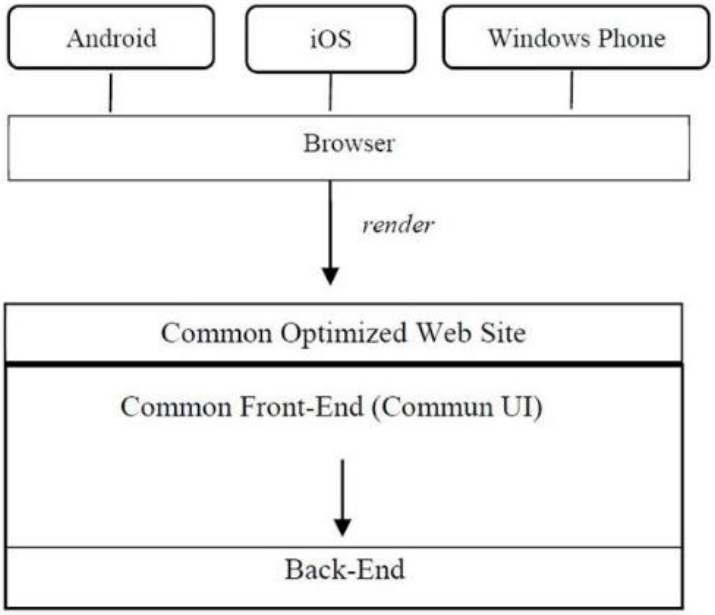
Em contrapartida, aplicativos nativos têm acesso total aos recursos e sensores do dispositivo móvel, o que os tornam mais eficientes (ADETUNJI *et al.*, 2020).

2.1.2 Aplicativo Web

São aplicativos otimizados para celular desenvolvidos com base em tecnologias da *Web*, como HTML, CSS e JavaScript. Eles são hospedados em servidores remotos e são acessados por meio de uma *Uniform Resource Locator* (URL) específica a partir de navegadores instalados no dispositivo móvel do usuário, tornando o aplicativo independente porque o navegador atua como seu ambiente de tempo de execução (EL-KASSAS *et al.*, 2017). Esta abordagem impõe a

otimização do aplicativo, levando em consideração os tamanhos de tela de vários dispositivos. Os aplicativos web adotam o modelo cliente-servidor, onde um solicitante de serviço (cliente) faz certas chamadas ou solicitações a um provedor de serviços (servidor) que, por sua vez, responde a solicitação do cliente. A comunicação de ida e volta é tratada por um protocolo de nível de aplicativo (HTTP) (LATIF; LAKHRISSI; ES-SBAI *et al.*, 2016).

Figura 1 – Abordagem do desenvolvimento de aplicativos web.



Fonte: Latif, Lakhrissi, Es-Sbai *et al.* (2016)

Apesar do desenvolvimento do aplicativo ser feita uma única vez e poder ser executado em qualquer plataforma, possuem acesso limitado aos recursos e funcionalidades nativos ou de baixo nível do dispositivo, além de um desempenho inferior em comparação com aplicativos nativos devido ao HTML e JavaScript que são interpretados através de navegadores (CHANG; OH, 2015).

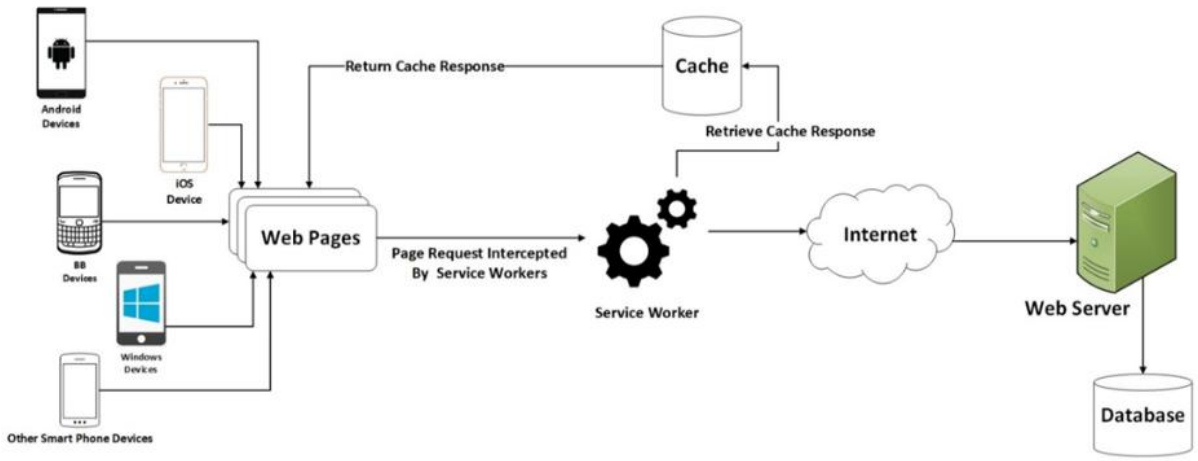
2.1.3 Aplicativo PWA

PWA não é uma nova tecnologia ou estrutura, mas sim melhores práticas que foram adotadas pela web para dar uma sensação de aplicativo nativo ao usuário. A adoção dessa abordagem produz tipos especiais de aplicativos web que não requerem instalação antes de serem usados e são servidos por um servidor remoto por meio de um protocolo de transferência de hipertexto (HTTPS) seguro, ao contrário dos aplicativos da web regulares que podem ser servidos usando HTTP (BEHL; RAJ, 2018).

Ele pode ser instalado na tela inicial do usuário a partir do navegador. O usuário recebe uma experiência de aplicativo nativo com suporte em tela cheia (sem navegador) após decidir instalar o PWA no dispositivo, baseado nos conceitos de um único aplicativo para todas as plata-

formas. No entanto, ele possui recursos distintos, como carregamento instantâneo e notificação push, mesmo no estado offline (KHAN; AL-BADI; AL-KINDI, 2019). A Figura 2 mostra em diagrama a abordagem de desenvolvimento do PWA.

Figura 2 – Abordagem do desenvolvimento de aplicativos PWA.



Fonte: Adetunji *et al.* (2020)

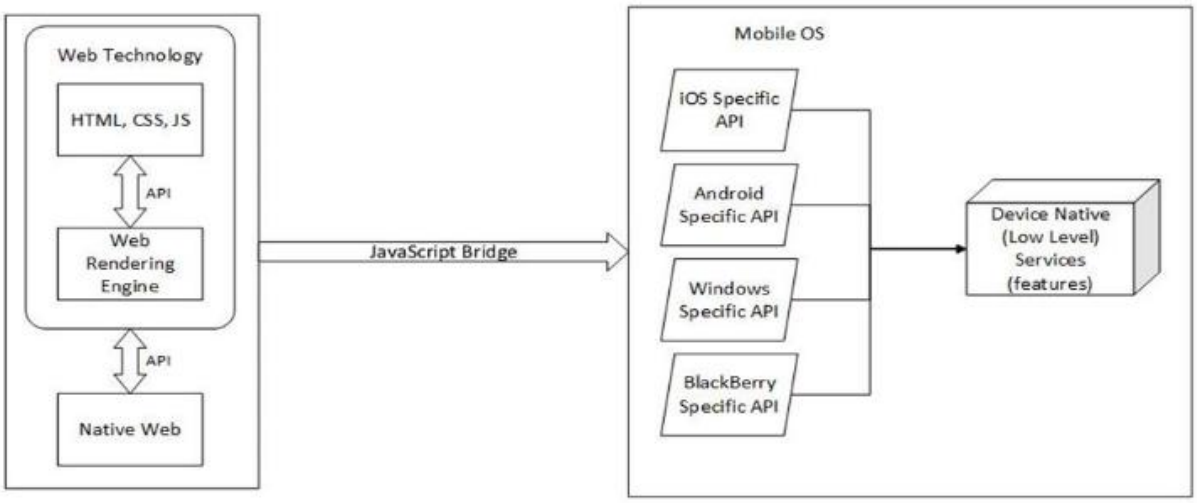
2.1.4 Aplicativo Híbrido

A abordagem um aplicativo híbrido fortalece os benefícios da abordagem nativa e da web, superando, assim, algumas limitações impostas por ambas, permitindo aos desenvolvedores criar uma base de código para todas as outras plataformas diferentes, sem a necessidade de se especializar em nenhuma plataforma em particular, por exemplo, Android ou iOS. O princípio é que se escreva o código como para qualquer site normal, o código é então empacotado em um *shell* nativo que permite que se direcione qualquer plataforma conforme for necessário.

Aplicativos híbridos cresceram em popularidade em comparação com aplicativos web, pois eles também conseguiram oferecer uma experiência *offline*, o que significa que o aplicativo pode funcionar sem uma conexão com a internet e, essencialmente, deixando de limitar um aplicativo à web para ser um aplicativo real. Um aplicativo híbrido é executado dentro do *webview* existente que está localizado dentro do aplicativo nativo. A ponte entre a parte nativa e o *webview* é o que, por sua vez, dá ao aplicativo acesso a recursos do tipo nativo, como sensores e notificações. Ele faz isso fornecendo um *wrapper* nativo onde o código baseado na web está contido, bem como uma API JavaScript que conecta as solicitações de serviço da tecnologia baseada na web para a API da plataforma (ADETUNJI *et al.*, 2020). A Figura 3 mostra uma visão esquemática da abordagem híbrida.

O desenvolvimento desse projeto foi feito utilizando essa abordagem, com o uso do *framework* Flutter, que utiliza a linguagem de programação Dart e atualmente pode ser compilado para as plataformas Android, iOS, Windows, Mac, Linux, Google Fuchsia e Web.

Figura 3 – Abordagem do desenvolvimento de aplicativos híbridos.



Fonte: Latif, Lakhrissi, Es-Sbai *et al.* (2016)

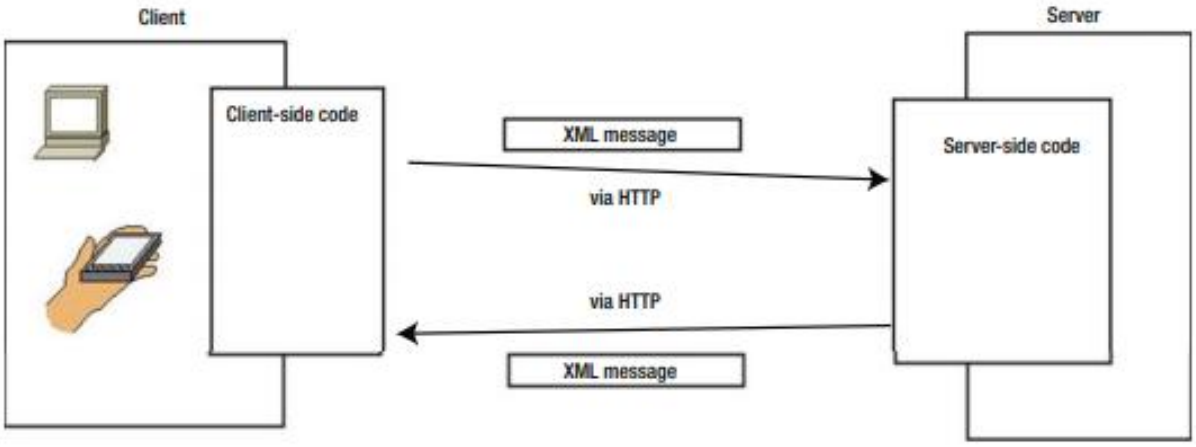
O Flutter oferece um ambiente de desenvolvimento altamente produtivo, permitindo a criação de aplicativos a partir de um único código-base. Isso significa que é possível economizar tempo e esforço ao desenvolver e manter um único aplicativo em vez de criar e gerenciar aplicativos separados para cada plataforma. Também, ao contrário das abordagens PWA e Web, um aplicativo híbrido com Flutter é capaz de oferecer uma experiência de usuário nativa. O Flutter utiliza widgets personalizados que se assemelham aos elementos da interface de usuário nativa de cada plataforma, garantindo uma aparência e sensação autênticas, semelhantes a um aplicativo nativo.

2.2 API RESTFUL

Uma API RESTful é composta por vários serviços da Web, cada um deles implementando uma ou mais operações *Create, Read, Update, Delete* (CRUD) sobre um recurso específico. Essas operações são geralmente mapeadas para os métodos HTTP POST, GET, PUT e DELETE. Um recurso é qualquer tipo de informação que pode ser exposta na Web (por exemplo, uma foto, um documento HTML, informação de um aluno). Os recursos são endereçáveis por um *Uniform Resource Identifier* (URI). Um caminho (também chamado de rota ou ponto final) representa um recurso sobre o qual as operações podem ser executadas. Além disso, as operações aceitam parâmetros. Um parâmetro é uma informação que pode ser passada junto com a solicitação para diversos fins, como resultados de filtragem e classificação. Várias respostas esperadas podem ser especificadas para cada operação. Uma resposta é identificada pelo código de status retornado e pode opcionalmente incluir um corpo. O código de status determina o resultado da operação (por exemplo, se foi bem-sucedida ou não) e o corpo de resposta inclui informações adicionais (MARTIN-LOPEZ; SEGURA; RUIZ-CORTÉS, 2019).

A Figura 4 mostra o diagrama de uma interação de uma API RESTFul.

Figura 4 – Interação com API RESTful.



Fonte: Doglio (2015)

Por meio de APIs RESTFul foi realizada a integração de dois *web services*: CAGR e Moodle.

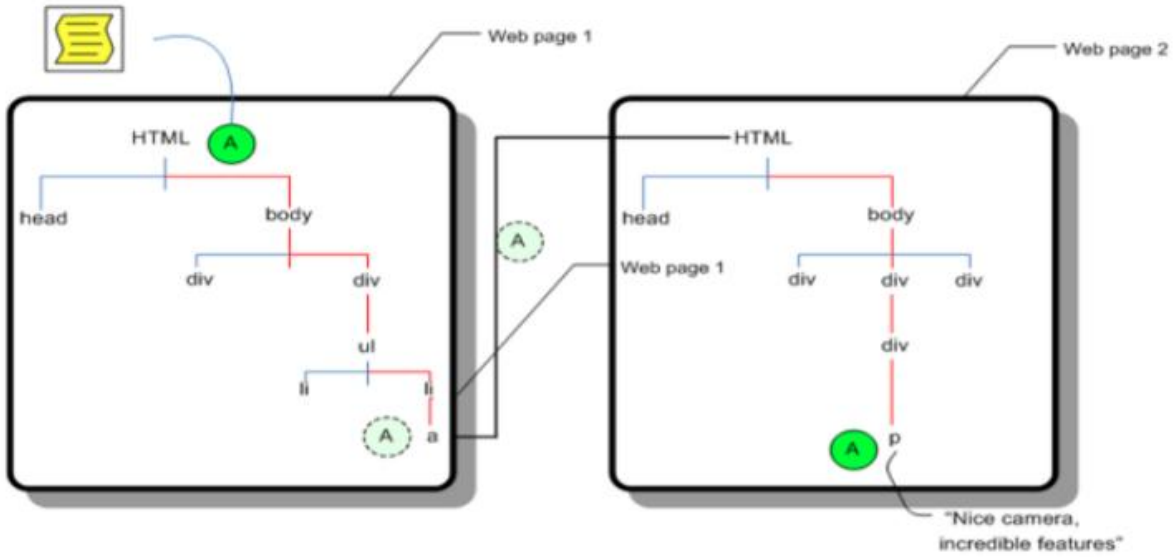
2.3 WEB SCRAPING

Web Scraping, ou também conhecido como extração de dados da web, é uma forma de mineração de dados. O objetivo básico e importante do processo é extrair informações de sites diferentes e não estruturados e transformá-los em uma estrutura compreensível como planilhas ou banco de dados. Quando um agente está seguindo cada link de uma página da web, ele está, na verdade, executando a mesma operação que um ser humano normalmente faria ao interagir com um site da web. Este agente pode seguir links (emitindo solicitações HTTP GET) e enviar formulários (por meio de HTTP POST), navegando por diversas páginas da web (GLEZ-PEÑA *et al.*, 2014).

A próxima etapa é seguir caminhos especificados pelo usuário dentro do documento para recuperar as informações desejadas com base nos dados recuperados na etapa anterior. Esses caminhos são especificados por seletores CSS. Eles usam caminhos relativos e absolutos (com base na árvore *Document Object Model* (DOM)) para apontar o analisador para um elemento específico dentro de um documento da web. Normalmente, as operações de web scraping usam expressões regulares para restringir ou cortar as informações localizadas, a fim de recuperar dados com um tamanho especificado pelo usuário (SAURKAR; PATHARE; GODE, 2018). A Figura 5 esquematiza um agente de Web Scraping que coleta informações de páginas da web.

A extração de dados da web está sendo realizada para coletar dados dos cardápios dos restaurantes universitários.

Figura 5 – Coleta de informações com base na árvore DOM.



Fonte: Mattosinho (2010)

2.4 OAUTH 2.0

O *framework* OAuth 2.0 define um protocolo baseado na *Web* que permite a um usuário conceder a sites acesso aos seus recursos (dados ou serviços) em outros sites (autorização). Os primeiros sites são chamados de partes confiáveis (RP) e os últimos são chamados de provedores de identidade (IdP). Na prática, o OAuth 2.0 também é frequentemente usado para autenticação. Ou seja, um usuário pode fazer login em um RP usando sua identidade gerenciada por um IdP (Single Sign-on, SSO) (FETT; KÜSTERS; SCHMITZ, 2016).

Se um usuário deseja autorizar um RP a acessar alguns dos dados do usuário em um IdP, o RP redireciona o usuário (ou seja, o navegador do usuário) para o IdP, onde o usuário se autentica e concorda em conceder ao RP acesso a alguns de seus dados do usuário no IdP. Então, junto com algum token (um código de autorização ou um token de acesso) emitido pelo IdP, o usuário é redirecionado de volta ao RP. O RP pode então usar o token como uma credencial no IdP para acessar os dados do usuário no IdP (DENNISS; BRADLEY, 2017).

O OAuth também é comumente usado para autenticação, embora não tenha sido projetado com a autenticação em mente. Um usuário pode, por exemplo, usar sua conta do Facebook, sendo o Facebook o IdP, para fazer login na rede social Pinterest (o RP). Normalmente, para fazer login, o usuário autoriza o RP a acessar um identificador de usuário exclusivo no IdP. O RP então recupera esse identificador e considera este usuário como conectado(FETT; KÜSTERS; SCHMITZ, 2016).

Neste trabalho, está sendo utilizado para realizar a autenticação dos usuários no aplicativo Gradufsc através do *framework* Firebase Authentication, desenvolvido pela Google.

2.5 FLUTTER

Flutter é um Kit de Desenvolvimento de Software (SDK) moderno e ágil, que possui uma estrutura de Interface de Usuário (IU) móvel para a construção de aplicativos híbridos lançados em 2018 pelo Google e é um SDK de código aberto, o que significa que a comunidade pode ajudar a desenvolver e melhorar a estrutura.

Da mesma forma como outras estruturas de construção de aplicativos híbridos, Flutter utiliza uma única base de código para diferentes plataformas, como web, móvel e Windows. O objetivo do Google é fornecer uma estrutura capaz de melhorar a velocidade de desenvolvimento e oferecer aplicativos híbridos com desempenho e IU semelhantes aos aplicativos nativos (OLSSON, 2020). Além disso, o Flutter tem uma linguagem de renderização e codificação diferente de outras estruturas, que geralmente utilizam JavaScript, (DAHL, 2019). No caso do Flutter, a linguagem de programação usada é Dart.

Este trabalho utiliza esse framework pela suas vantagens, que são a velocidade de desenvolvimento, pois ele utiliza uma linguagem de programação única e um conjunto de widgets personalizáveis que facilitam a criação de interfaces de usuário atraentes e responsivas, e também pela capacidade de alcançar múltiplas plataformas.

2.5.1 Dart

Dart é uma linguagem de programação lançada em 2011, desenvolvida e mantida pelo Google. O objetivo do Google era substituir a linguagem de programação JavaScript por sua nova linguagem Dart. Uma das características mais significativas do Dart é que a linguagem é tanto *Ahead Of Time* (AOT), em que o código é traduzido em arquivos assembly e o assembler converte para código binário para as diferentes arquiteturas, e *Just In Time* (JIT), em que o código é executado através de ferramentas de linha de comando da Máquina Virtual (VM) do Dart. Dessa forma, evita pontes de JavaScript para que a execução seja mais rápida e propõe um recurso chamado de *Hot reload*, que permite acelerar o ciclo de desenvolvimento, pois carrega as alterações de código na VM e reconstrói a árvore do *widget*, preservando o estado do aplicativo (TRAN, 2020).

Ao utilizar Dart, o Flutter não usa *widgets* do fabricante de equipamento original, evitando os problemas de desempenho que outras estruturas enfrentam acessando frequentemente recursos das plataformas iOS e Android (DAHL, 2019).

2.5.2 Widgets

Os *widgets* são todos os componentes necessários para construir a IU, que utilizam duas bibliotecas para o design: *Cupertino* para componentes iOS e *Material Design* para componentes Android. Assim, usando o Flutter, é possível ter um aplicativo com uma interface de usuário diferente dependendo da plataforma que o cliente escolhe usar. A Figura 6 compara a estrutura HTML/CSS com a estrutura Flutter.

Figura 6 – Estrutura HTML/CSS vs Flutter.

Fonte: Allain *et al.* (2020)

O Flutter classifica os *widgets* em duas categorias: *StatelessWidget* (sem estado) e *StatefulWidget* (com estado). O estado se refere aos dados que mudam ao longo do aplicativo de ciclo de vida. Quando um *widget* contém dados que podem mudar ao longo do tempo, ele precisará ser reconstruído para representar as alterações mais recentes no dispositivo do usuário. Portanto, este widget mantém um estado e pertence à categoria *StatefulWidget*. *StatelessWidget* não têm um estado e não requerem uma reconstrução (ALLAIN *et al.*, 2020).

2.6 FIREBASE

Firebase é uma estrutura útil para a construção de aplicativos portáteis e da web para quem requer um banco de dados em tempo real, o que implica que quando um usuário atualiza um registro no banco de dados, a atualização deve ser transmitida a cada usuário instantaneamente. Ele fornece uma plataforma básica e unificada para muitos aplicativos, juntamente com uma série de outros recursos do Google incluídos no serviço. O Firebase lida com a maior parte do trabalho do lado do servidor quando se trata de desenvolvimento de aplicativos. Existem inúmeros elementos que tornam o Firebase uma ferramenta essencial no desenvolvimento do ponto de vista do desenvolvedor (CHATTERJEE *et al.*, 2018).

Os principais componentes/serviços oferecidos pelo Firebase são os seguintes (CHATTERJEE *et al.*, 2018):

1. Autenticação: O recurso de autenticação no firebase permite que apenas usuários autorizados acessarem o aplicativo. Ele oferece suporte à autenticação por meio de senhas, e-mail ou nome de usuário, número de telefone, ou através de contas associadas ao Gmail, Github, Twitter, Facebook e também permite que o desenvolvedor crie autenticação personalizada.

2. Hospedagem: É possível enviar aplicativos da web e conteúdo estático de forma rápida e eficaz para uma Content Delivery Network (CDN) com um único comando. Ele contém suporte a domínio personalizado, CDN Global e Certificado Secure Sockets Layer (SSL) com aprovisionamento automático.

3. Mensagens: Solução de mensagens de plataforma cruzada que permite transmitir mensagens de forma confiável sem custo. É possível enviar mensagens de notificação para orientar o usuário.

4. Análise: Este recurso permite que o desenvolvedor do aplicativo entenda como os usuários estão usando o aplicativo. O SDK captura eventos e propriedades por conta própria e também permite que se obtenha dados personalizados. O painel fornece detalhes como usuário mais ativo ou qual recurso do aplicativo é mais usado. Ele também fornece dados resumidos.

5. Armazenamento: O Firebase também oferece facilidade de armazenamento. Ele pode armazenar e recuperar conteúdo como imagens, vídeos e áudio diretamente do SDK do cliente. O upload e o download são feitos em segundo plano. Os armazenamentos de dados são seguros e somente o usuário autorizado pode acessá-los.

6. Banco de dados em tempo real: É um banco de dados em nuvem e não precisa de consultas baseadas em SQL para armazenar e buscar dados. É altamente confiável, portanto, mesmo se a conexão for perdida, os dados serão mantidos. Os dados são armazenados como Notação de Objetos JavaScript (JSON) e sincronizados continuamente para cada usuário associado.

7. Relatórios de falhas: Cria relatórios de erros em do aplicativo após o lançamento. Os erros são agrupados em grupos diferentes de acordo com a gravidade do erro. Também pode-se criar eventos personalizados para capturar as etapas que levam ao travamento do aplicativo.

O Firebase foi utilizado pois possuí uma solução completa e integrada para o armazenamento de dados, que serão tanto para informações do usuário, como grade de horários e disciplinas, quanto para os dados dos menus dos restaurantes. Além disso possui facilidade na integração com o desenvolvimento de aplicativos móveis pois fornece uma API simples e intuitiva que permite a comunicação direta entre o aplicativo e o banco de dados, tornando o processo de armazenamento e recuperação de dados rápido e eficiente.

2.7 AMAZON LAMBDA

O Amazon Lambda é um serviço de computação sem servidor fornecido pela Amazon Web Services (AWS). Ele permite que os desenvolvedores executem código de forma escalável

e eficiente, sem a necessidade de provisionar e gerenciar servidores. O Lambda é especialmente útil para aplicações que exigem respostas rápidas a eventos, como a execução de funções em resposta a gatilhos específicos. O sistema é baseado em funções, e uma função é um código escrito pelo programador. Um evento aciona a função e o trecho de código será executado toda vez que o gatilho for acionado (ALTALEB; KHALAF *et al.*, 2022).

Esse serviço oferece uma variedade de funcionalidades e recursos que simplificam o desenvolvimento de aplicações. Entre eles, destacam-se:

- Suporte a várias linguagens de programação, incluindo Python, Node.js, Java, C# e Go. Isso permite que os desenvolvedores usem a linguagem de programação de sua preferência.
- Capacidade de resposta a eventos em tempo real. O Amazon Lambda pode ser configurado para executar funções automaticamente em resposta a eventos específicos, como atualizações de banco de dados ou envio de mensagens.
- Escalabilidade automática, permitindo que as funções do Lambda sejam dimensionadas automaticamente de acordo com a carga de trabalho.

Esse serviço foi escolhido como parte do projeto para agendar e executar o programa responsável pela coleta de dados dos restaurantes universitários. Sua utilização se baseia na necessidade de manter os menus atualizados diariamente de forma automatizada.

3 REVISÃO DA LITERATURA

Tendo como objetivo desmistificar os benefícios para integralização de sistemas e também realizar um estudo sobre as métricas de usabilidade de um aplicativo, foi realizado um estudo exploratório utilizando técnicas adaptadas/simplificadas, baseadas nas diretrizes de Kitchenham *et al.* (2009). Embora o presente estudo não tenha seguido exatamente todos os passos e detalhes de uma Revisão Sistemática de Literatura, é importante ressaltar que foram realizados esforços para obter uma visão abrangente das informações relevantes.

3.1 DEFINIÇÃO DA BUSCA

As questões de pesquisa abordadas por este estudo são:

1. Que tipos de benefícios a integração de sistemas para aplicativos podem trazer a um usuário?
2. Como e quais as melhores técnicas para mensurar a satisfação dos usuários sobre a usabilidade de uma aplicação?

Objetivo: Identificar e analisar vantagens na integração de sistemas com aplicações desenvolvidas com o propósito de unir funcionalidades de diversos sistemas e definir métricas de usabilidade para cálculo de satisfação dos usuários.

Controle: Através de uma revisão exploratória serão apurados artigos relacionados ao objetivo da pesquisa.

População: Publicações que têm relação com desenvolvimento de aplicativos responsivos, que calculam a satisfação de usuários quanto a usabilidade e que também integram funcionalidades de aplicações externas.

Resultados: Técnicas utilizadas para integração de sistemas e métricas para cálculo de usabilidade da aplicação.

Aplicações: Projetos de desenvolvimento de aplicações com ênfase na integração de sistemas.

3.2 ANÁLISE DO ESTADO DA ARTE

Em contraste com o aumento no número de aplicativos móveis, muitos usuários não conseguem encontrar os aplicativos apropriados para atender aos seus requisitos, tendo que fazer o uso combinado de várias aplicações para atender às suas necessidades específicas (ZHAI *et al.*, 2016). O que geralmente acontece são usuários consultarem separadamente serviços dedicados, e em seguida, alimentar o resultado de uma pesquisa como entrada para outra, ou ainda compará-los manualmente. No entanto, várias aplicações podem ser modeladas como uma composição de aplicativos/serviços preexistentes, permitindo ao usuário definir aplicativos mais complexos que atendam melhor as suas necessidades (BOULAKBECH *et al.*, 2017).

Tabela 2 – Definições da busca.

Fontes
1. Google Acadêmico (Google Scholar)
Critérios de Inclusão
1. Artigos disponíveis integralmente em bases de dados científicas digitais. 2. Artigos publicados a partir do ano de 2015. 3. Artigos publicados nos idiomas inglês ou português. 4. Artigos que apresentem o desenvolvimento de sistemas integrados utilizando web services. 5. Artigos que apresentem métricas de usabilidade.
Critérios de Exclusão
1. Artigos relacionados com desenvolvimento de sistemas voltados para área da saúde. 2. Artigos que tem por objetivo realizar a integração entre hardware e software.

Fonte: Desenvolvido pelo autor.

Tabela 3 – Strings de busca avançada.

Fonte	Google Acadêmico
String de busca	(("integrated system") OR ("integrated application") OR ("system unification") OR ("application unification")) AND ("web services") AND (usability) -health)
Filtros	Intervalo: 2015-2021. Busca em títulos e resumos.

Fonte: Desenvolvido pelo autor.

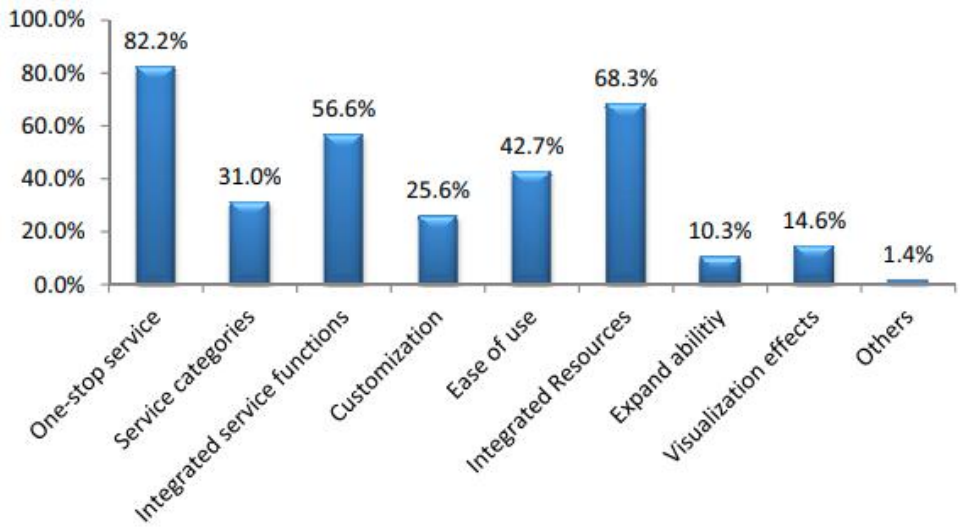
Para realizar essa integração geralmente utiliza-se APIs, que estão cada vez mais sendo publicados pelas principais plataformas da web, como eBay, Amazon e Google. Aplicativos da web que consumem essas APIs são chamados coletivamente de *mashups* e oferecem uma experiência interessante para os usuários da web (GUTIÉRREZ-CARREÓN; DARADOUMIS; JORBA, 2015).

Os *mashups* foram inicialmente desenvolvidos no campo da música, criando uma composição através da combinação de duas ou mais músicas pré gravadas, geralmente sobrepondo a trilha vocal de uma música perfeitamente sobre a faixa instrumental de outra (WOODLAND; KLASS, 2005). Atualmente, a palavra *mashup* é amplamente aplicada a aplicativos da web, permitindo aos usuários integrar produtos e serviços para criação de novos e exclusivos, que implica na combinação de duas ou mais fontes de dados baseados na web e tem capacidade de reutilizar ou combinar dados ou informações com um aplicativo para criar e produzir resultados de valor agregado. Eles atraem amplo interesse e continuam a crescer em popularidade (GRAMMEL; STOREY, 2010).

Um estudo sobre as vantagens de um *mashup* de serviço de informação em um uma biblioteca digital (YAN; DENG; ZHANG, 2016) foi realizada e apresenta os resultados na Figura 7.

Neste estudo, mais de quatro quintos (82,2 por cento) dos entrevistados concordaram que o serviço de centralização de dados único é um ponto forte óbvio. Além disso, as características

Figura 7 – Vantagens do uso de *mashup* em uma biblioteca digital.



Fonte: Yan, Deng e Zhang (2016)

integradas de recursos (68,3 por cento) e funções (56,6 por cento) também receberam mais da metade da aprovação dos entrevistados. Facilidade de uso (42,7 por cento), categorias de serviço (31,0 por cento) e personalização (25,6 por cento) dos *mashups* também foram reconhecidas por uma parcela considerável dos entrevistados. Os efeitos de visualização (14,6 por cento) e a capacidade de expansão (10,3 por cento) dos *mashups* foram menos valorizados em comparação com outras vantagens.

Este estudo mostra que a integração percebida tem efeito semelhante na intenção de adoção do serviço. Os autores destacam que a razão pode ser que as características essenciais de um *mashup* de serviço de informação são a reutilização e a síntese de componentes de informação, assim, os usuários valorizam a integração ao considerar se devem ou não utilizar essa tecnologia.

Um outro experimento, com o desenvolvimento de um mashup semântico para integrar recursos do Google Apps, com o sistema de gestão de aprendizagem chamado Chamilo foi realizado em uma turma de alunos de graduação matriculados no curso "Sistemas de Informação de Gestão" (GUTIÉRREZ-CARREÓN; DARADOUMIS; JORBA, 2015) da Open University of Catalonia. Em geral, o sistema Chamilo oferece suporte a tipos diferentes de aprendizagem e atividades colaborativas. Permite especificar os objetivos do curso, identificar unidades de aprendizagem, desenvolver materiais e apresentações de conteúdo, construir exercícios de avaliação, enquanto fornece ferramentas que contêm comunicação síncrona e assíncrona (chat e fóruns). No entanto, de acordo com os autores, os alunos possuem necessidades específicas, a fim de atender objetivos específicos de um um curso, necessitando de ferramentas que estão além escopo do Chamilo. Essas ferramentas são oferecidas pelo Google Apps e incluem Google Drive para armazenamento e criação de documentos, Youtube para conteúdo audiovisual geren-

ciamento, Google Hangouts para lidar com mensagens eletrônicas instantâneas e recursos de videoconferência ao vivo, entre outros.

Os alunos que participaram da experiência foram divididos em dois grupos: um grupo experimental e um grupo de controle. Alunos no grupo experimental usaram o Chamilo integrado com serviços em nuvem, já o grupo de controle foi solicitado a acessar as duas ferramentas separadamente, realizando o acesso aos serviços em nuvem manualmente. Ambos os grupos de alunos foram instruídos pelo mesmo professor, que atribuiu aos alunos os mesmos problemas específicos para resolverem.

O resultado analisado pelos autores teve um impacto positivo na carga cognitiva e de usabilidade. No nível de carga cognitiva, foi examinado se os alunos que têm acesso à funcionalidade integrada têm menos sobrecarga cognitiva e, portanto, concentram sua atenção melhor no aprendizado. Em segundo lugar, no nível de usabilidade.

No caso do indicador de consciência, com base nas respostas de cada um dos grupos, determinou-se que o grupo experimental apresentou um melhor desempenho para encontrar e organizar soluções. Em comparação com o grupo de controle, o sistema integrado aumentou a motivação dos alunos. Em suma, concluiu-se que os alunos que usam o sistema integrado tiveram benefícios significativos em respeito à relação "carga cognitiva-satisfação"em relação aos alunos do grupo de controle que usaram o serviços do Google de forma independente.

Gao (2019) diz que o sucesso de uma aplicação *mashup* é alcançado por um princípio centrado no usuário a todo o processo de desenvolvimento, com enfoque em características e comportamento humano, bem como fatores que irão aprimorar mais a experiência do usuário. Diante disso quatro pontos importantes foram destacados para o desenvolvimento de um *mashup*.

1. **Facilidade de uso**: Terminologias adequadas e representações visuais podem ajudar os usuários finais a compreender facilmente as funções de cada componente e módulo, para que possam executar seus objetivos com mais eficiência. O design também desempenha um papel importante no aumento da facilidade de uso. Uma interface amigável com um design de navegação explícito e uma categorização bem definida de conteúdo vai reduzir muito tempo e esforço que os usuários finais precisam ter na ação de tomada de decisões. A interface também deve permitir que os usuários interajam de forma fácil e livre.

2. **Facilidade de aprendizagem**: Deve-se fornecer elementos de treinamento integrados, como tutoriais, instruções, exemplos, dicas e sugestões.

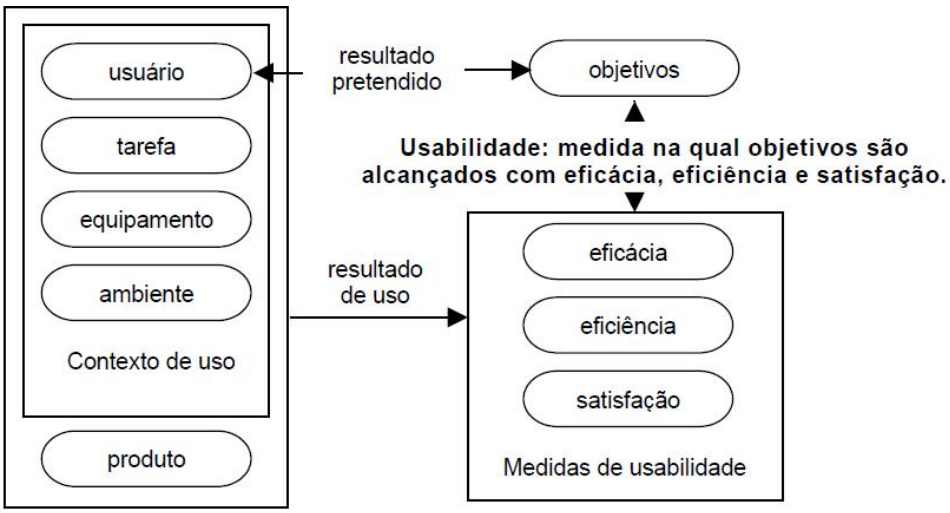
3. **Tolerância a erros**: Normalmente, os usuários finais não reservam tempo para aprender todas as funcionalidades e operações antes de começar a usar a ferramenta levando a uma grande possibilidade de cometer erros. Além disso, sempre ocorrem interpretações erradas e negligência, como cliques errados e erros ortográficos. Então é importante aumentar a tolerância a erros.

4. **Satisfação**: A experiência de uso individual é diferente para todos. A maneira mais direta e eficaz de aumentar a satisfação de uma ferramenta de mashup é criar uma interface amigável e um layout limpo. Especialmente para a página inicial, painel frontal e menu de navegação, porque a primeira impressão tem muito peso. Além disso, existem algumas outras maneiras que podem tornar a ferramenta mais agradável. Por exemplo, com uso de mensagens de sucesso após finalizar tarefas.

Usabilidade é definida pela ISO 9241-11 (ISO, 1998) como "a extensão em que um produto pode ser usado por usuários específicos para atingir objetivos específicos com eficácia, eficiência e satisfação em um contexto de uso especificado" e ainda define as medidas citadas a seguir, estruturadas na Figura 8.

1. Eficácia: Acurácia e completude com as quais usuários alcançam objetivos específicos.
2. Eficiência: Recursos gastos em relação à acurácia e abrangência com as quais usuários atingem objetivos.
3. Satisfação: Ausência do desconforto e presença de atitudes positivas para com o uso de um produto.
4. Contexto de Uso: Usuários, tarefas, equipamento (hardware, software e materiais), e o ambiente físico e social no qual um produto é usado.
5. Sistema de Trabalho: Sistema, composto de usuários, equipamento, tarefas e o ambiente físico e social, com o propósito de alcançar objetivos específicos.

Figura 8 – Estrutura da usabilidade



Fonte: Schneider (2017)

Um dos aspectos importantes a serem considerados é o número de cliques necessários para realizar determinadas tarefas. Pesquisas recentes têm explorado como a redução do número de cliques pode influenciar a satisfação do usuário e melhorar a experiência de uso.

Segundo (SMITH; JOHNSON, 2021), uma redução significativa no número de cliques pode levar a uma experiência mais eficiente e eficaz para os usuários. Em seu estudo comparativo, eles descobriram que participantes que interagiram com uma versão de um aplicativo com menos cliques relataram menor esforço percebido, maior satisfação e conclusão mais rápida de tarefas em comparação com aqueles que usaram uma versão com mais cliques.

Também, segundo (BROWN; DAVIS, 2022), que conduziram um estudo sobre aplicações web, descobriram que os usuários que interagiram com uma versão que exigia menos cliques demonstraram maior satisfação e concluíram as tarefas de forma mais rápida em comparação com aqueles que usaram uma versão com mais cliques. Esses resultados ressaltam a importância de uma interface com menos cliques para melhorar a experiência do usuário.

Para Cybis, Betiol e Faust (2017) a avaliação de usabilidade deve considerar qual técnica será utilizada, quem serão os avaliadores e em que nível o projeto se encontra, e para isso dividiu as técnicas de avaliação em três grupos:

1. Técnicas Prospectivas: São baseadas na aplicação de entrevistas e questionários aos usuários para que seja avaliado a satisfação dos mesmos.

2. Técnicas Preditivas ou Diagnósticas: Não possuem a participação direta dos usuários para prever os erros de projeto e são classificadas em: avaliações heurísticas, que consistem em realizar uma inspeção do sistema para detectar problemas ou as barreiras que os usuários provavelmente encontrarão durante a interação e já fornecer possíveis soluções; e *checklist*, técnica baseada em lista de verificação, em que qualquer profissional faz o diagnóstico rápido de problemas gerais e repetitivos das interfaces.

3. Técnicas Objetivas ou Empíricas: É realizada através de ensaios de interação e monitoramento diretamente com o usuário.

Kaya, Ozturk e Gumussoy (2019) dizem que um sistema deve satisfazer as expectativas e necessidades dos usuários com suas características físicas e técnicas e que o sistema de escala de usabilidade, conhecido como System Usability Scale (SUS) é um dos sistemas mais populares e fáceis de usar um questionário para medir o nível de usabilidade de qualquer produto.

John Brooke desenvolveu o SUS em 1996. Ele contém dez questões básicas e simples sobre a usabilidade de um sistema, sendo uma ferramenta útil para entender os problemas que os usuários enfrentam enquanto usam o sistema, fornecendo uma pontuação única para usabilidade, projetada como uma medida unidimensional (BROOKE *et al.*, 1996).

Dez declarações são apresentadas ao usuário, que se relacionam a vários aspectos da usabilidade em uma escala Likert de 5 pontos, variando de forte desacordo (1) a forte acordo

(5). A pontuação final para o SUS varia de 0 a 100, com pontuações mais altas indicando maior percepção de usabilidade (MOL *et al.*, 2020).

Os dez itens da escala são os seguintes:

1. Acho que gostaria de utilizar este sistema com frequência.
2. Considerei o sistema mais complexo do que necessário.
3. Achei o sistema fácil de utilizar.
4. Acho que necessitaria de ajuda de um técnico para conseguir utilizar este sistema.
5. Considerei que as várias funcionalidades deste sistema estavam bem integradas.
6. Achei que este sistema tinha muitas inconsistências.
7. Suponho que a maioria das pessoas aprenderia a utilizar rapidamente este sistema.
8. Considerei o sistema muito complicado de utilizar.
9. O sistema pareceu confiável ao utilizar.
10. Precisei aprender várias coisas novas antes de conseguir usar o sistema.

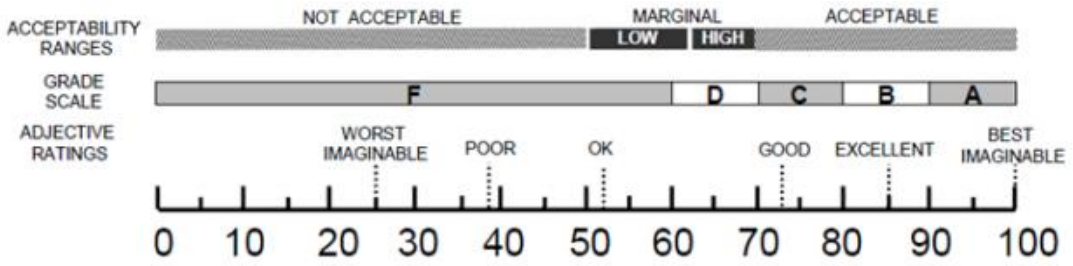
Esta escala de dez itens tem sido referida como um padrão da indústria e é considerada um dos questionários mais amplamente usados na prática de usabilidade (LEWIS; SAURO, 2017).

Como se pode verificar no SUS, as perguntas ímpares têm significados positivos e as perguntas pares têm significados negativos. A pontuação das perguntas positivas é feita da seguinte forma: A pontuação do usuário é reduzida em um ponto. Por exemplo, se a pontuação do usuário for 4 para a pergunta 5, a pontuação do resultado será 3. A pontuação das perguntas negativas é feita da seguinte forma: A pontuação do usuário é subtraída de 5. Por exemplo, se a pontuação do usuário for 3 para a questão 4, então a nova pontuação será 2. Após todas as pontuações serem determinadas, a soma das pontuações é multiplicada por 2,5 para fazer o intervalo entre 0 e 100.

O SUS possui excelentes propriedades psicométricas, pois de acordo com pesquisas tem mostrado consistentemente que possui confiabilidade igual ou superior a 90% (LEWIS; BROWN; MAYES, 2015). De acordo com esses dados normativos, uma pontuação de 68 pode ser interpretado como um resultado aceitável.

Para entender melhor o significado de uma pontuação SUS, Bangor, Kortum e Miller (2009) propõem uma escala de classificação adjetiva, ao invés de uma escala numérica, que se assemelha a uma classificação escolar americana, com conceitos A (pontuação entre 90 e 100), B (80 à 89), C (70 à 79), D (60 à 69), e F (0 à 59). Neste caso a pontuação de 0 à 25 seria considerado pior do que imaginado, de 26 à 38 ruim, de 39 à 52 razoável, de 53 à 73 bom, de 74 à 85 excelente e de 85 à 100 melhor do que imaginado. A Figura 9 representa essa escala.

Figura 9 – Escala de classificação adjetiva



Fonte: Bangor, Kortum e Miller (2009)

3.3 DISCUSSÃO SOBRE O ESTADO DA ARTE

Para um usuário que necessita utilizar diversas aplicações com objetivos em comum, a integração das funcionalidades em um único aplicativo *mashup* traz a centralização de dados, melhorando as suas necessidades, pois facilita a sua organização, trazendo uma melhor usabilidade e ele acaba tendo uma carga cognitiva superior.

Todavia, para que o usuário se sinta satisfeito, alguns requisitos de usabilidade devem ser cumpridos. O sistema deve ser fácil de navegar, deve ter uma interface simples e objetiva, em que o usuário tenha certeza do que acontece nos eventos gerados e que falhas sejam prevenidas, quando existe algum processo mais complexo de navegação, um passo a passo deve ser exibido. Esses requisitos podem ser apurados durante o desenvolvimento da aplicação por meio de *checklists* e uma das melhores e mais utilizadas formas de mensurar a usabilidade com o usuário é aplicando um questionário, sendo o SUS uma boa estratégia, pois possui uma classificação confiável e respeitada de pontuação.

Tabela 4 – Principais pontos da pesquisa.

Autor	**Pontos Principais**
Yan, Deng e Zhang (2016)	Traz como benefícios de *masups*: centralização e visualização de dados, integração de recursos, maior facilidade no uso, capacidade de expansão e personalização.
Gutiérrez-Carreón, Daradoumis e Jorba (2015)	Aponta que *masups* trazem para os usuários um impacto positivo na carga cognitiva e de usabilidade além de aumentar desempenho para encontrar e organizar soluções.
Gao (2019)	Define que para uma aplicação ser satisfatória ao usuário deve ser fácil de usar e de aprender, além de possuir tolerância a falhas e um layout amigável.
Cybis, Betiol e Faust (2017)	Apresenta a avaliação de usabilidade em três técnicas: prospectivas, preditivas e objetivas.
Kaya, Ozturk e Gumussoy (2019)	Avalia o questionário SUS como um dos sistemas mais populares e fáceis de usar para medir o nível de usabilidade de qualquer produto.
Lewis, Brown e Mayes (2015)	Pesquisa que diz que o SUS possui excelentes propriedades psicométricas com confiabilidade superior a 90%.

Fonte: Autor.

4 PROPOSTA DE SOLUÇÃO

Neste capítulo são apresentados uma visão geral do que este trabalho se propõe a resolver, soluções disponíveis e um levantamento de requisitos.

4.1 MOTIVAÇÃO E NECESSIDADE DA PROPOSTA

A UFSC possui diversos sistemas muito úteis para os acadêmicos gerenciarem a graduação, sendo o CAGR e o Moodle muito utilizados. O CAGR é onde o aluno encontra a sua grade de horários, com as informações da disciplina, o nome dos professores que ministrarão as aulas, os horários e locais. Também é lá que o aluno gera documentos importantes, como o atestado de matrícula. Já o Moodle permite que educadores criem ambientes virtuais de aprendizagem, onde podem disponibilizar materiais didáticos, tarefas, entre outras atividades, para que os alunos possam acessar, interagir com colegas, enviar trabalhos, realizar testes, entre outras atividades relacionadas ao ensino e aprendizagem.

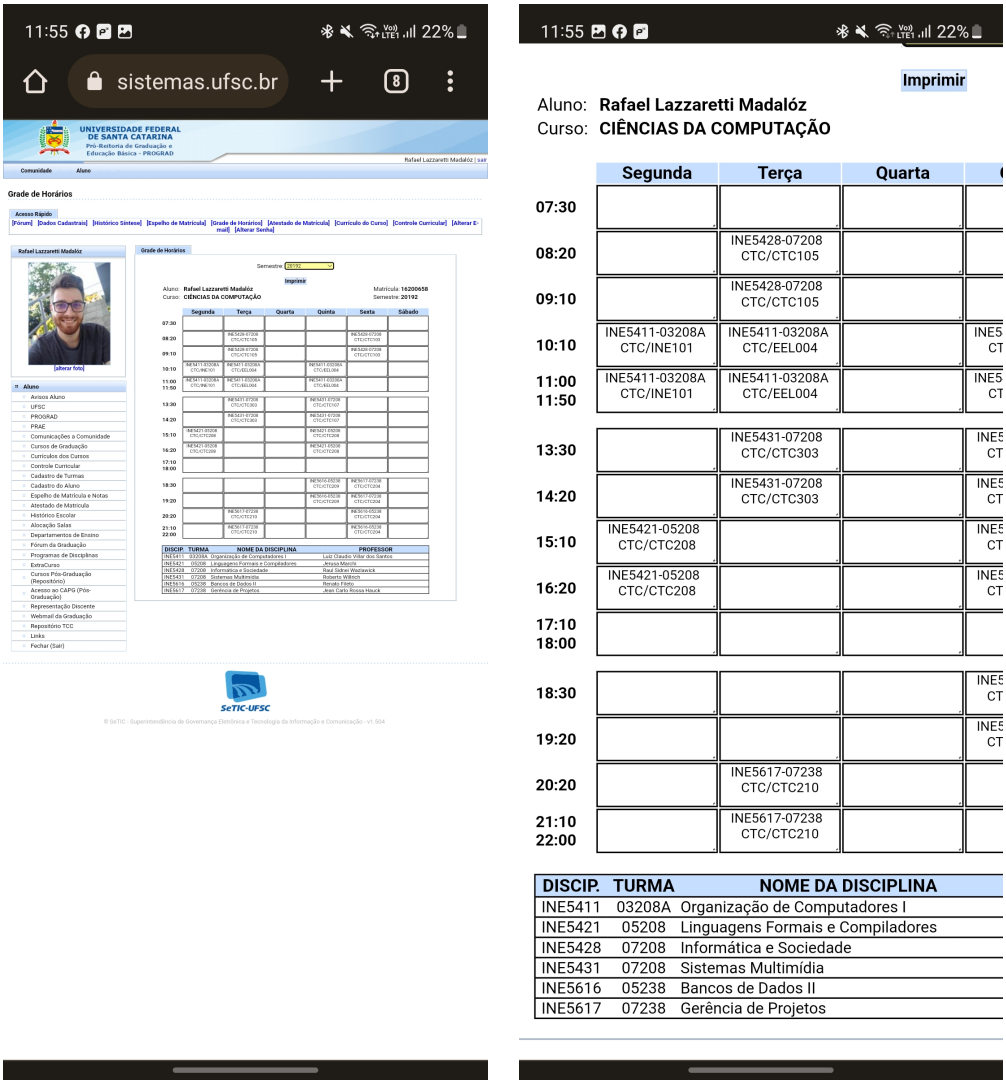
Ao ser acessado por dispositivos móveis, o CAGR apresenta problemas de responsividade, ou seja, o conteúdo é projetado para telas maiores e não fica adequado em telas menores, além de que para tarefas muito simples, como conferir a grade de horários para acessar qual a próxima aula do dia são necessários os seguintes passos:

1. Acessar a página do CAGR.
2. Acessar a sessão Aluno.
3. Realizar a autenticação com o CAS, inserindo as credenciais para fazer login na página.
4. Acessar o menu Grade de Horários.
5. Dar zoom na página para poder ler a informação.

A Figura 10 mostra como é renderizada a página de grade de horários na tela de um dispositivo móvel, mostrando que como a fonte fica muito pequena, é necessário dar um zoom na página para poder ler seu conteúdo.

Além disso, pelo fato de serem utilizados múltiplas aplicações, faz com que o usuário tenha que acessar diferentes páginas na internet para encontrar informações sobre as disciplinas, fazendo com que o número de cliques necessários aumente para realizar as tarefas desejadas.

Portanto, foram integradas funcionalidades das duas aplicações principais da UFSC: CAGR e Moodle, com o objetivo do usuário consultar as principais informações desses serviços. Além disso, também foi adicionado o cardápio do Restaurante Universitário de todos os campi, e as notícias vinculadas ao RSS de notícias da UFSC.



(a) Tamanho real (b) Zoom aplicado

Figura 10 – Página de grade de horários do CAGR

4.2 SOLUÇÕES EXISTENTES

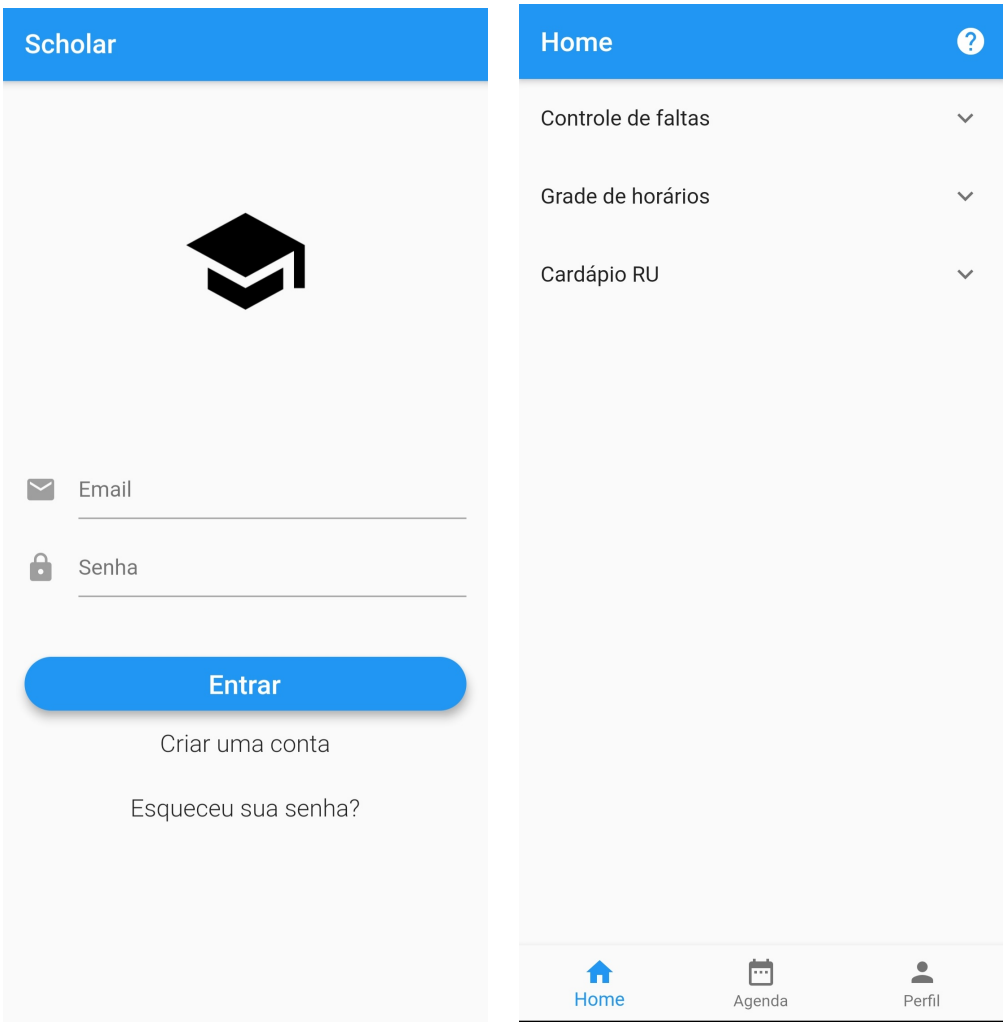
4.2.1 Aplicativo Scholar

O aplicativo Scholar foi um aplicativo lançado em 2019 como trabalho de conclusão de curso dos alunos Adson Leal e Angelo Leal do curso de ciências da computação da UFSC e serviu como fonte de inspiração para o desenvolvimento deste trabalho, sendo utilizado as mesmas chaves de autenticação para o CAGR. O aplicativo possuía as seguintes funcionalidades:

1. Exibir a grade de horários (disponibilizada pela API do CAGR);
2. Exibir o cardápio do RU (disponibilizado por API que coleta dados web desenvolvido pelos autores do aplicativo);
3. Cadastrar manualmente tarefas (provas ou trabalhos);

4. Cadastrar manualmente a sua frequência em disciplinas.

O aplicativo teve falhas com a autenticação com o CAGR e deixou de funcionar no início do ano de 2020. A figura 11 mostra a página de login e a página inicial do aplicativo, que possuía recursos simples, porém úteis, e que tiveram um grau alto de satisfação com relação às funcionalidades pela comunidade acadêmia na época. A Figura 12, por sua vez, mostra a modelagem do banco de dados do aplicativo Scholar.



(a) Página de Login (b) Página Inicial

Figura 11 – Capturas de Tela do Aplicativo Scholar

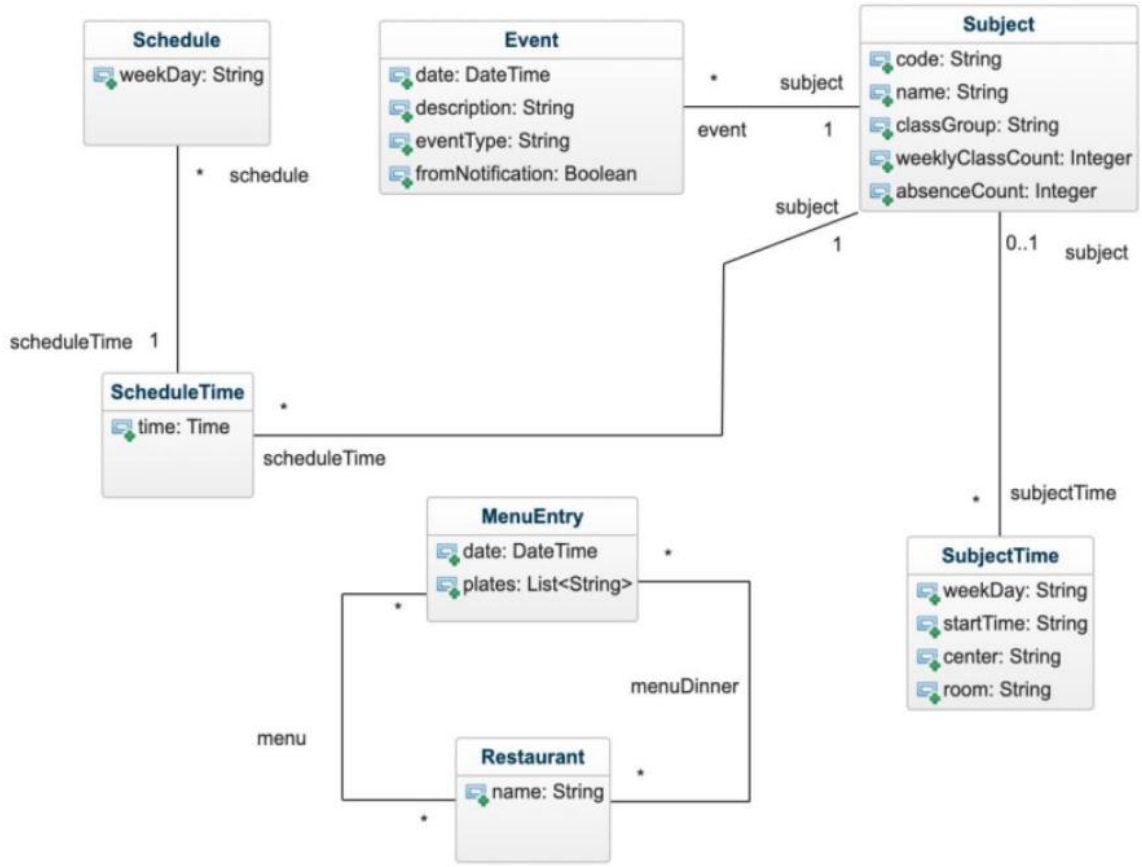
Fonte: (LEAL; LEAL *et al.*, 2019)

4.2.2 Aplicativo Nossa UFSC

O aplicativo Nossa UFSC foi lançado em Maio de 2022 por Guilherme Ferraz, de forma independente, e possui os seguintes recursos:

1. Cadastrar as disciplinas por meio de submissão de uma captura de tela da grade de horário do CAGR;

Figura 12 – Modelagem de banco de dados Scholar.



Fonte: Leal, Leal *et al.* (2019)

2. Cadastro de eventos por disciplina;
3. Exibir o cardápio do Restaurante Universitário (apenas do campus Trindade);
4. Informações sobre festas universitárias submetidas manualmente pelos usuários.

Para a utilização desse aplicativo é necessário que os alunos façam o cadastro dos eventos e não possui nenhuma integração com o Webservice do CAGR e nem com o do Moodle.

O aplicativo pode ser encontrado na loja de aplicativos para Androoid, Play Store[1].

4.3 LEVANTAMENTO DE REQUISITOS

O levantamento dos requisitos mencionados a seguir foi feito com base na vivência e nas necessidades identificadas pelo autor desse trabalho como aluno da UFSC, além de uma análise dos recursos disponíveis pelos webservices do Moodle e do CAGR para os requisitos funcionais. O programa que coleta os dados dos restaurantes está sendo referenciado como

[1] Aplicativo Nossa UFSC: `play.google.com/store/apps/details?id=com.portal.ufsc`.

"programa scraper", enquanto a aplicação de interface gráfica para dispositivos móveis está como "aplicativo móvel".

4.3.1 Requisitos Não-Funcionais

Os requisitos não-funcionais estabelecem as características e restrições do sistema que não se relacionam diretamente com suas funcionalidades específicas. Eles são fundamentais para garantir o desempenho, a confiabilidade e a segurança do sistema. Os requisitos não-funcionais identificados para o sistema são:

- RNF1: O programa scraper deve ser hospedado em servidor da Amazon, no serviço Amazon Lambda;
- RNF2: O programa scraper deve ser executado diariamente em horários pré-determinados;
- RNF3: O programa scraper deve ser desenvolvimento utilizando a linguagem de programação Python;
- RNF4: O programa scraper deve utilizar Docker para facilitar a instalação das dependências de bibliotecas Python no servidor;
- RNF5: O programa scraper deve armazenar os dados no banco de dados na nuvem Firebase.
- RNF6: O aplicativo móvel deve ser compatível com Android na versão 5 ou superior;
- RNF7: O aplicativo móvel deve ser desenvolvido utilizando o framework Flutter;
- RNF8: O aplicativo móvel deve armazenar os dados no banco de dados na nuvem Firebase.

4.3.2 Requisitos Funcionais

Os requisitos funcionais descrevem as funcionalidades específicas que o sistema deve oferecer para atender às necessidades dos usuários. Os requisitos funcionais identificados para o sistema são:

- RF1: O aplicativo móvel deve exibir as três próximas disciplinas matriculadas do usuário na página inicial, informando o nome da disciplina, o local e o horário;
- RF2: O aplicativo móvel deve exibir a grade de horários completa separado por dia da semana, exibindo o nome da disciplina e o horário;
- RF3: O aplicativo móvel deve exibir uma lista com todas as disciplinas matriculadas, informando o nome, código e o nome dos professos que ministram as aulas;

- RF4: O aplicativo móvel deve exibir o cardápio do Restaurante Universitário, separado por dia da semana, do campus selecionado pelo usuário;
- RF5: O aplicativo móvel deve exibir um botão para o usuário acessar o PDF do cardápio do Restaurante Universitário do campus selecionado;
- RF6: O aplicativo móvel deve exibir os eventos cadastrados do calendário do usuário no Moodle, exibindo a informação do título, nome da disciplina e o horário do evento;
- RF7: O aplicativo móvel deve exibir todas as notas cadastradas nas disciplina no Moodle;
- RF8: O aplicativo móvel deve exibir as manchetes mais recentes do portal de notícias da UFSC;
- RF9: O aplicativo móvel deve permitir o download do histórico síntese do aluno;
- RF10: O aplicativo móvel deve permitir o download do atestado de matrícula do aluno;
- RF11: O aplicativo móvel deve permitir o download do currículo do curso do aluno;
- RF12: O aplicativo móvel deve notificar o usuário sobre o horário das aulas, enviando uma mensagem do aplicativo com o nome da disciplina, o local e o horário;
- RF13: O aplicativo móvel deve notificar o usuário sobre o horário dos eventos do calendário, enviando uma mensagem no aplicativo com o título do evento, e horário de entrega.

4.4 CASOS DE USO

Os casos de uso representam as principais funcionalidades que o sistema oferece ao usuário. Nesse contexto, três atores foram identificados no sistema, sendo eles:

- Usuário: É a entidade que interage com o aplicativo. Ele será um aluno de graduação da UFSC.
- Moodle Webservice: É um serviço web no qual o aplicativo interage para obter informações sobre eventos (provas, trabalhos, atividades) de disciplinas e recuperar notas do usuário.
- CAGR Webservice: É outro serviço web no qual o aplicativo interage para obter informações sobre as disciplinas do usuário.

Com base nos requisitos do sistema, foram identificados os casos listados a seguir e representados na Figura 13.

Figura 13 – Diagrama de casos de uso.

Fonte: Autor

- UC1 - Fazer cadastro: O usuário pode criar uma nova conta no aplicativo. O sistema solicita as informações necessárias para o cadastramento, como e-mail e senha. Após o preenchimento correto dos dados, um e-mail de confirmação é enviado. Ao clicar no link enviado no e-mail, a conta é criada e o usuário pode fazer login.
- UC2 - Fazer login: O usuário realiza o processo de autenticação no aplicativo, fornecendo suas credenciais. Esse caso de uso envolve a verificação de token com o serviço de autenticação do Firebase Authentication.
- UC3 - Fazer download de documentos: O usuário pode fazer download de documentos, como histórico síntese, atestado de matrícula e currículo do curso. Os documentos são disponibilizados para download no aplicativo por meio de links que redirecionam para a página de download correspondente.
- UC4 - Visualizar disciplinas matriculadas: O usuário pode visualizar as disciplinas matriculadas, incluindo nesses dados o nome, código turma e professor(es) da disciplina. Essas informações são obtidas do banco de dados e exibidas no aplicativo.

- UC5 - Visualizar grade de horários: O usuário pode visualizar a grade de horários completa por dia da semana. Além disso, na página inicial são exibidas as próximas três disciplinas da grade de horários do aluno a partir da data e horário atuais. Esses dados também são coletados do banco de dados.
- UC6 - Receber notificações: O usuário pode receber notificações sobre o horário das aulas e eventos do calendário. Essas notificações são enviadas pelo aplicativo para informar o usuário sobre atualizações relevantes.
- UC7 - Visualizar cardápio do Restaurante Universitário: O usuário pode acessar o cardápio do Restaurante Universitário selecionando o campus desejado. As informações do cardápio são obtidas do banco de dados.
- UC8 - Visualizar notas cadastradas: O usuário pode visualizar todas as notas cadastradas por disciplina no Moodle. Essas informações são obtidas do webservice do Moodle.
- UC9 - Visualizar notícias da UFSC: O usuário pode visualizar as notícias mais recentes do portal de notícias da UFSC. As informações são obtidas do RSS do portal de notícias.
- UC10 - Visualizar eventos do calendário: O usuário pode visualizar os eventos cadastrados no calendário do Moodle. Essas informações são obtidas por meio de requisição no webservice do Moodle.

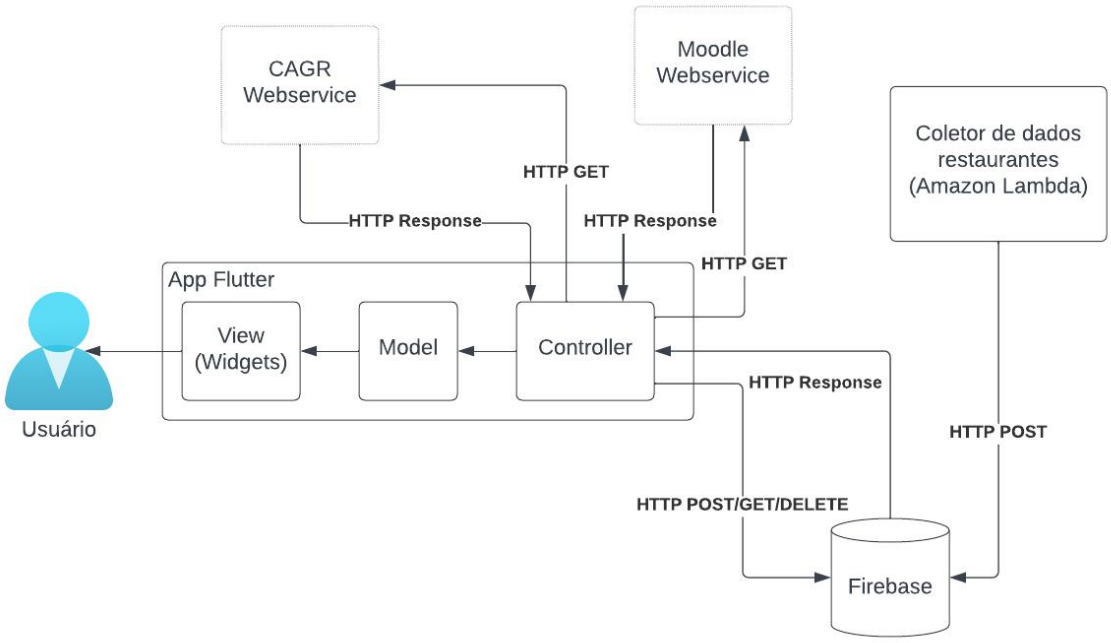
4.5 DIAGRAMA DE ARQUITETURA

O diagrama de arquitetura representa a estrutura e organização dos componentes do sistema como um todo, fornecendo uma visão geral de como se relacionam e trabalham juntos. Sendo assim, foram identificadas os principais componentes, exibidos na Figura 14 e listados a seguir.

Aplicativo Flutter: Este é o componente responsável pela interface com o usuário. Ele utiliza o padrão de arquitetura Model-View-Controller (MVC) para separar as responsabilidades e melhorar a organização do código. O Model representa os dados das disciplinas, cardápios dos restaurantes e outras informações relevantes, que são armazenados no Firebase. O View consiste nos widgets que compõem a interface do usuário, permitindo que os usuários interajam com o aplicativo. O Controller contém as funções responsáveis pela lógica de negócio, como autenticação, acesso aos dados do CAGR e Moodle, e atualização da interface com os dados obtidos.

Firebase: É o componente do banco de dados na nuvem. Ele armazena as informações das contas de usuário, bem como os dados das disciplinas, cardápios dos restaurantes e outras informações relevantes para o aplicativo.

Figura 14 – Diagrama de arquitetura.



Fonte: Autor

Amazon Lambda: Utilizado para hospedar e executar um script em Python responsável por coletar os dados dos menus dos restaurantes diariamente. Os dados coletados são salvos no Firebase para que possam ser acessados pelos usuários posteriormente.

Webservice do CAGR: O aplicativo precisa acessar o webservice do CAGR para obter os dados das disciplinas.

Webservice do Moodle: O aplicativo acessa o Webservice do Moodle para obter as informações das notas e os eventos do calendário do aluno.

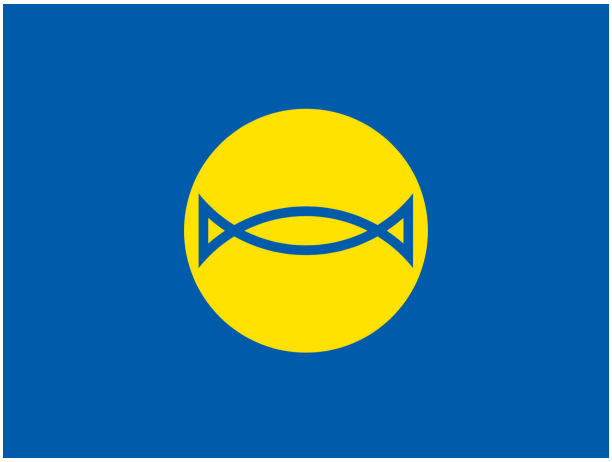
5 DESENVOLVIMENTO DA APLICAÇÃO

Neste capítulo, são apresentados os detalhes do desenvolvimento do aplicativo proposto, abordando diferentes aspectos relacionados à sua implementação.

5.1 APLICATIVO MÓVEL

O nome do aplicativo foi escolhido para ser Gradufsc, uma junção entre a palavra graduação e a sigla UFSC, remetendo ao público que se beneficiará da aplicação: estudantes de gradução da UFSC. A identidade visual possui a mesma tonalidade da cor azul da bandeira da UFSC (Figura 15).

Figura 15 – Bandeira da UFSC.



Fonte: www.ufsc.br

A estrutura do código foi feita utilizando o padrão arquitetural MVC (Model-View-Controller). Sendo assim, três pastas foram criadas, uma para o modelo (model) onde foram definidas as classes de usuário e restaurante, uma para as funções que realizam as requisições para os webservices externos (controller) e outra para a criação dos widgets responsáveis pela interface com o usuário (view). A Figura 16 mostra a estrutura dos arquivos.

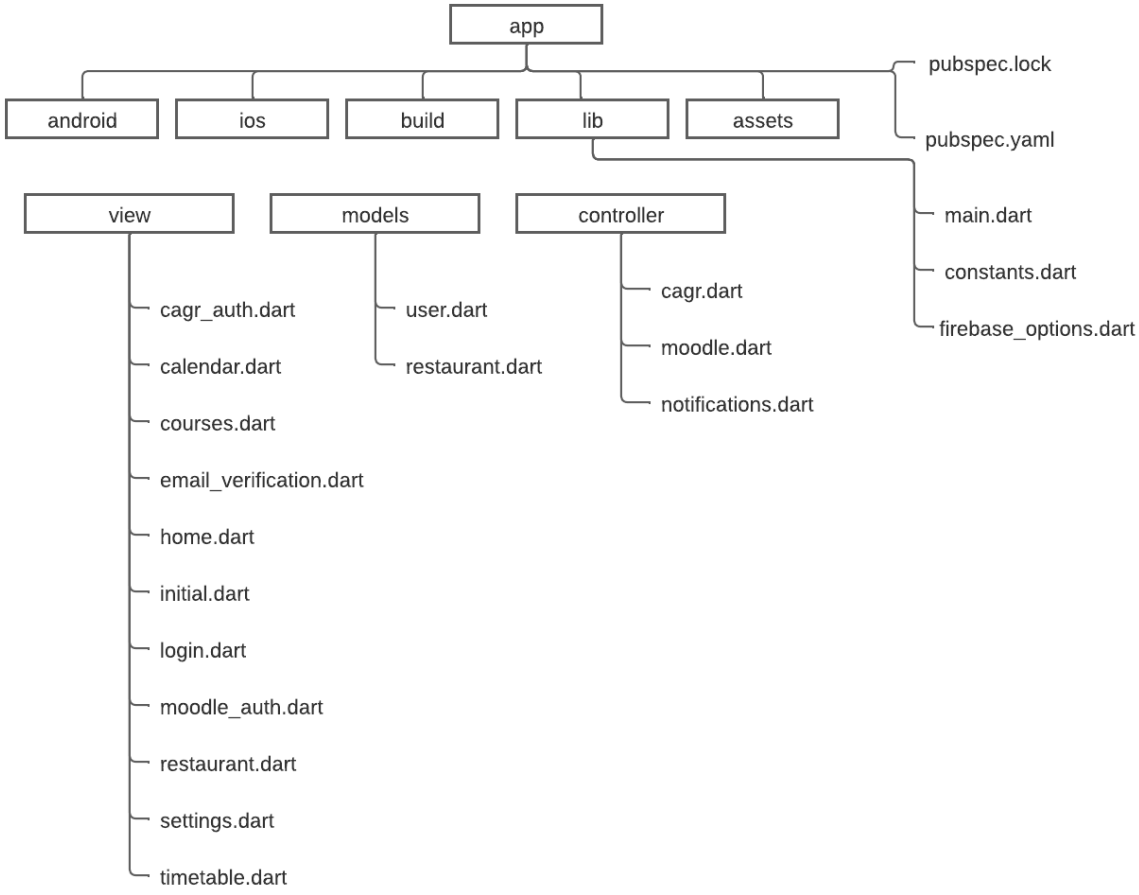
A pasta “app” corresponde à pasta raiz onde todos os arquivos do framework Flutter são definidos.

A pasta “android” contém arquivos relacionados à configuração e construção do aplicativo para o sistema operacional Android, assim como a pasta “”ios para o sistema operacional iOS.

A pasta “lib” é onde se encontra o código-fonte principal do aplicativo.

A pasta “assets” é onde ficam armazenados arquivos de imagens utilizados pelo aplicativo, que nesse caso é somente a imagem do logotipo do aplicativo.

Figura 16 – Estrutura de pastas e arquivos



Fonte: Autor

O arquivo “pubspec.yaml” especifica as dependências do projeto e outros metadados, como nome, versão e descrição.

O arquivo “pubspec.lock” é gerado automaticamente e registra as versões exatas das dependências do projeto.

Uma das vantagens do desenvolvimento do aplicativo utilizando o *framework* Flutter é a ampla variedade de *widgets* prontos disponíveis, que são elementos de interface que compõem a estrutura visual do aplicativo, agilizando o desenvolvimento e garantindo uma experiência consistente em diferentes dispositivos.

Além disso, o Flutter também oferece uma biblioteca integrada de ícones, tornando mais simples a adição de ícones às interfaces do aplicativo. Com uma ampla seleção de ícones disponíveis, não é necessário buscar ou criar manualmente cada ícone necessário, o que poupa tempo e esforço no processo de desenvolvimento.

5.1.1 Página de Login/Cadastro

O cadastro e autenticação são feitos utilizando o serviço de autenticação do Firebase, o Authentication, onde é possível utilizar diferentes métodos de autenticação, mas que por enquanto inclui apenas autenticação por e-mail e senha. Esse método permite que os usuários criem suas contas fornecendo um endereço de e-mail e uma senha, que são armazenados de forma segura pelo serviço.

A conexão entre o aplicativo e o Firebase é feita por meio do arquivo de configuração fornecido pelo Firebase Console, que é definida para dispositivos Android conforme o Código 1.

Código 1 – Configuração do Firebase

```
static const FirebaseOptions android = FirebaseOptions(
    apiKey: 'my_api_key',
    appId: 'my_app_id',
    messagingSenderId: 'my_sender_id',
    projectId: 'gradufsc',
    storageBucket: 'gradufsc.appspot.com',
    authDomain: 'Gradufsc',
);
```

E que é utilizada pela biblioteca do Firebase para Flutter para poder fazer a conexão conforme o Código 2.

Código 2 – Inicialização do Firebase

```
await Firebase.initializeApp(
  name: "gradufsc",
  options: FirebaseOptions.currentPlatform
);
```

Assim, em cada arquivo que for necessário verificar a autenticação do usuário, basta importar a biblioteca do Firebase Autentication e instanciar o usuário atual. Caso nenhum usuário tenha se autenticado, o resultado será nulo.

Código 3 – Instanciação de usuário autenticado

```
User? user = FirebaseAuth.instance.currentUser;
```

Para realizar a autenticação, o e-mail e a senha são coletados do formulário e enviados por meio de uma função do Firebase que verifica se as credenciais são válidas e correspondem a uma conta existente no sistema.

5.1.2 Páginas de Configurações Iniciais

Para obter os dados da grade de horários de cada aluno é necessário realizar a autenticação com o CAS, que será feita apenas uma vez, salvando os dados das disciplinas no banco

Figura 17 – Página de Login do Aplicativo Gradufsc.

Fonte: Autor

de dados na nuvem do Firebase para que o aluno possa acessar de qualquer dispositivo, apenas fazendo o login e também não precisando realizar autenticação sempre que for entrar no aplicativo. A Figura 18 mostra a tela que explica o primeiro passo para o usuário obter os dados e a necessidade da autenticação. Ao clicar no botão “Autenticar” o usuário será redirecionado para a tela de autenticação do CAS (Figura 19), e se a autenticação ocorrer corretamente, será redirecionado para o aplicativo, onde os dados serão coletados e salvos.

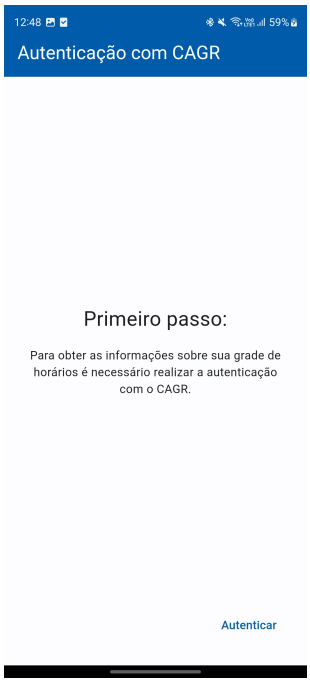
Para a autenticação com o CAS foram utilizados os parâmetros de autenticação utilizados pelos alunos Adson Leal e Angelo Leal no desenvolvimento do aplicativo Scholar (LEAL; LEAL *et al.*, 2019).

A autenticação com o CAS é feita em duas etapas. Primeiro é necessário acessar a URL de autenticação a partir de um navegador externo com uma série de parâmetros, como mostra o Código 4.

Código 4 – Chamada para autenticação do CAS

```
Uri url = Uri.parse("""
  https://sistemas.ufsc.br/oauth2.0/authorize?
  client_id=secret_id&
  client_secret=secret_client_id&
  redirect_uri=tccleal://tccleal.setic_oauth.ufsc.br&
  response_type=code&
  state=secret_state_id
""");
if (await canLaunchUrl(url)) {
  await launchUrl(url, mode: LaunchMode.externalApplication);
}
```

Figura 18 – Tela inicial para guiar autenticação.



Fonte: Autor

Após o usuário realizar a autenticação com o CAS será feito um redirecionamento para a URL que foi passada como parâmetro de redirecionamento na etapa anterior. Nos arquivos de configuração do aplicativo Android, foi realizada a configuração de forma que, ao detectar uma URL com o domínio específico, o aplicativo seja aberto. O Código 5 mostra como foi feita essa configuração, que faz com que o aplicativo seja aberto quando o navegador for redirecionado para uma URL com o host “tccleal”.

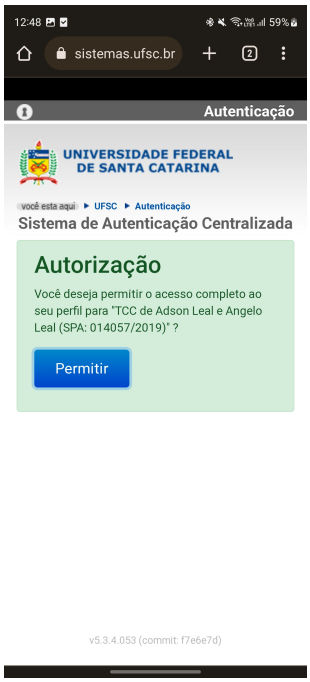
Código 5 – Configuração de redirecionamento para Android

```
<intent-filter>
  <action android:name="android.intent.action.VIEW" />
  <category android:name="android.intent.category.DEFAULT" />
  <category android:name="android.intent.category.BROWSABLE" />
  <data
    android:scheme="tccleal"
    android:host="tccleal.setic_oauth.ufsc.br" />
</intent-filter>
```

Ao abrir o aplicativo pelo redirecionamento, uma função está ouvindo o evento, e detectará essa URL, que será retornada com uma chave de acesso. Essa chave será capturada para então fazer uma nova requisição POST ao sistema CAS para então obter a chave de acesso que dá permissão de utilizar o Webservice para o usuário autenticado, conforme é feito no Código 6.

A Figura 20 representa um diagrama de sequência exibindo o passo-a-passo desde o login do usuário até o armazenamento dos dados do usuário.

Figura 19 – Tela de autenticação com CAS.



Fonte: Autor

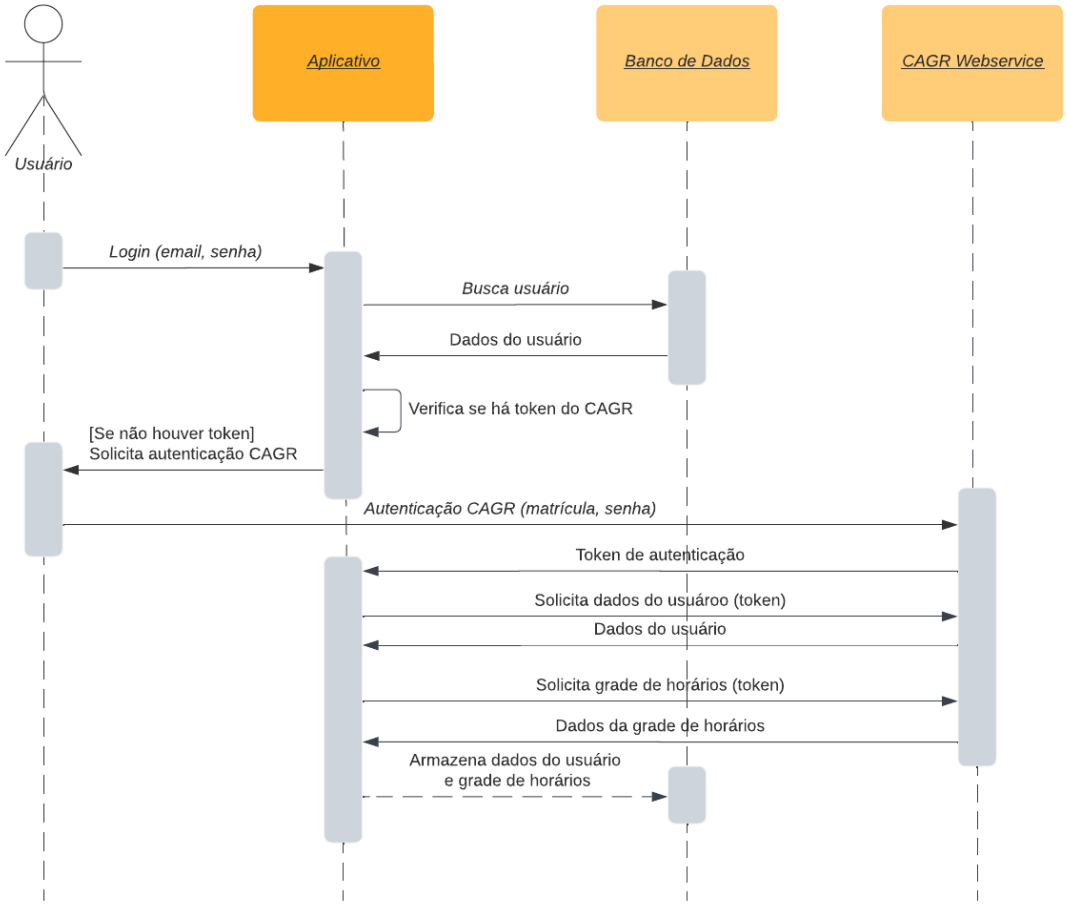
Após o login, é exibida ao usuário uma tela para configuração da chave do Moodle, onde o usuário necessita acessar a página de configurações do seu perfil do Moodle, copiar a chave e colar no campo de entrada da tela, conforme a Figura 21. Essa chave é necessária para realizar as requisições do Web Service do Moodle e obter dados relativos às disciplinas que o usuário está cursando.

Código 6 – Requisição para obter chave de acesso

```
final urlBack = await linkStream.first;
code = Uri.parse(urlBack).queryParameters['code'];

await http.post(
  Uri.parse('https://sistemas.ufsc.br/oauth2.0/accessToken'),
  headers: {
  'Content-Type': 'application/x-www-form-urlencoded'
  },
  body: {
    'grant_type': 'authorization_code',
    'client_id': 'tccleal',
    'client_secret': 'my_secret_client',
    'redirect_uri': 'tccleal://tccleal.setic_oauth.ufsc.br',
    'code': authorizationCode,
  },
);
```

Figura 20 – Diagrama de sequência de autenticação.



Fonte: Autor

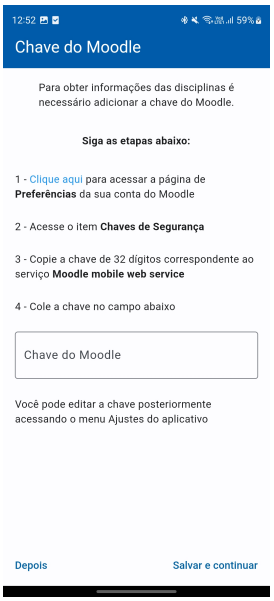
5.1.3 Página Inicial

Depois de feita as configurações iniciais, o usuário é redirecionado para a tela inicial do aplicativo (Figura 22), onde pode acessar os próximos três horários de aula, com o nome da disciplina e o local onde a aula ocorrerá. Também há um botão chamado “Ver tudo” que, ao ser pressionado, redireciona o usuário para a tela da grade completa dos horários, organizados por dia da semana (Figura 23). Além disso, são exibidas na tela inicial manchetes coletadas do RSS de notícias da UFSC, que quando clicadas redirecionam o usuário para a página da notícia completa no site da UFSC.

Na barra inferior do aplicativo há cinco botões com ícones que direcionam o usuário à tela inicial, à tela que lista as disciplinas em que o aluno está matriculado, à tela de eventos que traz os próximos eventos agendados no calendário do aluno no Moodle, à tela de cardápio do Restaurante Universitário, e à tela de configurações, respectivamente.

Para obter os dados de disciplinas do usuário é necessário realizar duas requisições para o webservice do CAGR utilizando a chave de acesso obtida na primeira etapa. A primeira

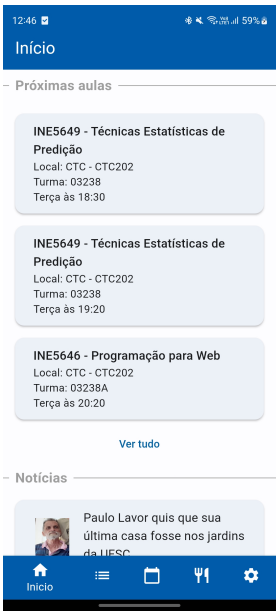
Figura 21 – Tela de configuração da chave do Moodle.



Fonte: Autor

requisição traz as informações do aluno, que possui as informações de cada disciplina em que o aluno está matriculado no semestre corrente. O Código 7 demonstra a requisição e os parâmetros utilizados.

Figura 22 – Página Inicial Aplicativo Gradufsc.



Fonte: Autor

Código 7 – Requisição para obter dados do aluno

```
String webservice = "https://ws.ufsc.br/rest/CAGRUsuarioService";
String functionName = "/getInformacaoAluno";
final response = await http.get(
    Uri.parse(
      "$webservice$functionName$?access_token=$_authCode"
    )
);
if (response.statusCode == 200) {
    final res = json.decode(response.body);
}
```

O resultado da requisição, quando bem sucedida, é um arquivo JSON, como o exemplo do Código 8:

Código 8 – Resultado da requisição de dados do aluno

```
{
  "modoInformacao": "Reduzido",
  "matricula": 22100898,
  "nome": "Francisco Camerini",
  "codigoCurso": 238,
  "nomeCurso": "SISTEMAS DE INFORMACAO (noturno)",
  "codigoCentro": "CTC",
  "codigoSituacao": 0,
  "nomeSituacao": "regular",
  "semestreIngresso": 20221,
  "disciplinas": [
    {
      "codigoDisciplina": "EGC5009",
      "codigoTurma": "03238",
      "nomeDisciplina": "Geracao de Ideias e
                         Criatividade em Informatica",
      "professores": [
        {
          "nomeProfessor": "Patricia de Sa Freire",
          "serproProfessor": 200632,
          "siapeProfessor": 1222544
        }
      ]
    },
  ...]}
```

A outra requisição feita é responsável por obter a grade de horários das disciplinas em que o usuário está matriculado. Para isso é feita uma requisição para outra função do web service do CAGR, conforme o exemplo do Código 9.

Figura 23 – Página de Grade de Horários.

Fonte: Autor

Código 9 – Requisição para obter grade de horários

```
String webservice =
  "https://ws.ufsc.br/rest/CAGRUsuarioService"
String functionName = "/getGradeHorarioAluno:
final response = await http.get(
  Uri.parse(
    "$webservice$functionName$?access_token=$_authCode"
  )
);
if (response.statusCode == 200) {
  final res = json.decode(response.body);
}
```

O resultado da requisição, quando bem sucedida, é um arquivo JSON, como o exemplo do Código 10.

Código 10 – Resultado da requisição da grade de horários

```
{
    "aluno": "Francisco Camerini",
    "matricula": 22100898,
    "curso": "SISTEMAS DE INFORMACAO (noturno)",
    "semestre": 20231,
    "horarios": [
        {
            "codigoDisciplina": "INE5649",
```

```
            "codigoTurma": "03238",
            "localizacaoCentro": "CTC",
            "localizacaoEspacoFisico": "CTC210",
            "diaSemana": 2,
            "horario": 1830
        },
        {
            "codigoDisciplina": "INE5649",
            "codigoTurma": "03238",
            "localizacaoCentro": "CTC",
            "localizacaoEspacoFisico": "CTC210",
            "diaSemana": 2,
            "horario": 1920
        },
    ...
    ]
}
```

Esses dados são lidos pelo aplicativo e então salvos no banco de dados para que o usuário não precise realizar uma nova autenticação com o CAGR toda vez que acessar o aplicativo, ou então quando a chave de acesso do CAGR perder a validade. A vantagem disso é que o usuário obtém a informação muito mais rápido e a desvantagem é que quando houverem alterações nas disciplinas em que estiver cursando ele terá que acessar a página de Ajustes do aplicativo e apertar o botão de atualizar, para que então os dados sejam atualizados. Essa estratégia implica no usuário ter que realizar a atualização da sua grade somente quando um novo semestre começar.

5.1.4 Página de Disciplinas

A segunda opção do menu inferior é a tela de disciplinas, onde são listadas todas as disciplinas em que o usuário está matriculado e que foram coletadas do CAGR na autenticação com o CAS. Aqui são exibidos o código, nome e local das disciplinas, assim como os professores que ministram as aulas, conforme Figura 24. Ao clicar em uma disciplina, o usuário é redirecionado à tela específica da disciplina, onde são exibidos todos os horários, assim como as sala em que serão administradas. Há também uma seção mostrando as notas do usuário referentes a cada disciplina, conforme ilustrado na Figura 25.

Para obter as informações das notas é necessário fazer uma autenticação junto ao webservice do Moodle, passando o nome da função que retornará as notas, o número de identificação do usuário, e o número de identificação da disciplina, conforme o exemplo do Código 11.

Para obter o número de identificação do usuário é necessário fazer uma requisição prévia utilizando o valor “core_webservice_get_site_info” no parâmetro “wsfunction”. Essa requisição traz também o nome de todas as funções disponíveis para o usuário.

Para obter o número de identificação de cada disciplina em que o usuário está matriculado também é necessário realizar uma requisição prévia para a função

Figura 24 – Página de Disciplinas.

Fonte: Autor

"core_enrol_get_users_courses", porém dessa vez o resultado será todas as disciplinas em que o usuário está cadastrado no Moodle, desde a criação da conta, e não apenas as disciplinas do semestre atual. O Código 12 representa como é estruturado um item da lista das disciplinas retornado pela requisição

Código 11 – Requisição para obter disciplonas do aluno

```
    http.get("""
        https://moodle.ufsc.br/webservice/rest/server.php?
        wstoken=$userToken&
        moodlewsrestformat=json&
        wsfunction=core_webservice_get_site_info&
        courseid=$courseId&
        userid=$userId
    """);
```

Utilizando a informação da chave "idnumber" dos dados é possível separar apenas a informação necessária para obter os dados das disciplinas em que o usuário está matriculado naquele semestre, fazendo uma comparação com os dados que foram salvos ao obter os dados referentes ao webservice do CAGR. A verificação é feita separando os valores pelo caractere "|" e então comparando com o semestre, código da turma e código da disciplina, conforme o Código 13

Figura 25 – Página Específica da Disciplina.

Fonte: Autor

Código 12 – Resultado da requisição para obter disciplinas do Moodle

```
{
    "id": 66027,
    "shortname": "INE5401-01208A (20162)",
    "fullname": "INE5401-01208A (20162)
                - Introdução à Computação",
    "enrolledusercount": 59,
    "idnumber": "turma|CAGR|GR|208|20162|INE5401|01208A",
    "visible": 1,
    "summary": "Uso versus funcionamento ...",
    "summaryformat": 1,
    "format": "topics",
    "showgrades": true,
    "lang": "",
    "enablecompletion": false,
    "category": 2552,
    "progress": null,
    "startdate": 1470608969,
    "enddate": 0
},
```

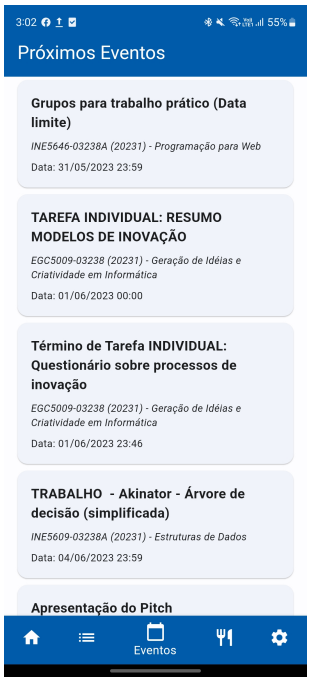
Código 13 – Verificação de identificadores de disciplinas

```
List<String> ids = data["idnumber"].split("|");
if (
    ids.length >= 6 &&
    semester == ids[4] &&
    courseCode == ids[5] &&
    classCode == ids[6]) {
        saveCourseId(data["id"]);
    }
)
```

5.1.5 Página de Eventos

Na página de eventos são listados os próximos eventos (provas, trabalhos, etc.) do calendário do Moodle do aluno, exibindo além do nome do evento, a disciplina correspondente e a data em que o evento ocorrerá, conforme mostrado na Figura 26.

Figura 26 – Página de Eventos do Moodle.



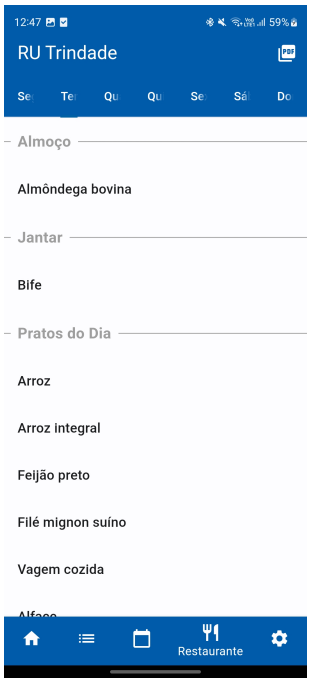
Fonte: Autor

Para obter esses dados é feita uma requisição ao webservice do Moodle com a função chamada “core_calendar_get_action_events_by_course” para cada disciplina em que o aluno está matriculado, usando os identificadores das disciplinas que foram obtidas conforme descrito na seção anterior. Esses eventos são ordenados por data, em ordem crescente.

5.1.6 Página do Restaurante Universitário

Na página do Restaurante Universitário, mostrada na Figura 27, é exibido o cardápio do dia, sendo possível arrastar a página lateralmente ou clicar no ícone referente ao dia da semana para o qual deseja ver o menu. Como os dados são coletados de documentos PDF, podem ocorrer erros na obtenção dos dados. Caso ocorra algum erro durante a obtenção do cardápio, um ícone com referência ao PDF original do menu ficará disponível no topo da página para o usuário consultar.

Figura 27 – Página do Restaurante Universitário.

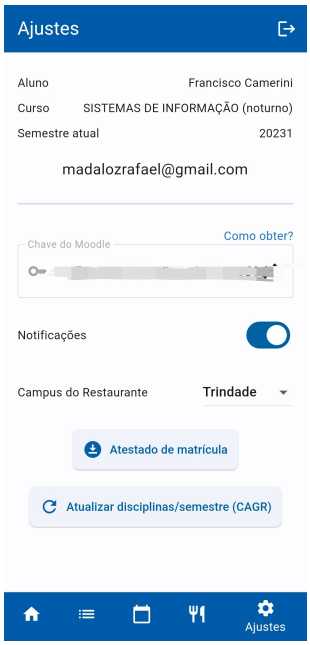


Fonte: Autor

5.1.7 Página de Ajustes

Na página de ajustes, mostrada na Figura 28, são exibidos os dados do usuário, além de um botão para que o usuário possa habilitar ou silenciar as notificações do aplicativo, que notifica com trinta minutos de antecedência cada aula. Também é possível alterar de qual campus será exibido o cardápio do restaurante, alterar a chave do Moodle no caso de perder a validade, e atualizar as disciplinas quando há troca de semestre ou cancelamento de matrícula durante o semestre. O usuário também tem acesso ao link para fazer o download do atestado de matrícula, que redireciona o usuário para o link de download do CAGR, podendo neste caso ser solicitada a autenticação do usuário para fazer o download do mesmo.

Figura 28 – Página de Ajustes.



Fonte: Autor

5.2 COLETA DE DADOS DO MENU DOS RESTAURANTES

O script desenvolvido consiste em uma ferramenta de *scraping* do menu dos restaurantes. A principal função desta ferramenta é coletar informações sobre o menu diário de cada restaurante e armazená-las no banco de dados do Firebase, de modo que essas informações possam ser fornecidas aos usuários.

A coleta dos dados dos menus dos restaurantes foi realizada utilizando a linguagem de programação Python e algumas bibliotecas específicas para acessar as URLs e ler os arquivos PDFs. O objetivo é acessar a página do Restaurante Universitário da UFSC (https://ru.ufsc.br/cardapios/) e obter os links para os seis restaurantes disponíveis nos diferentes campi da universidade:

- Campus Araranguá
- Campus Blumenau
- Campus CCA
- Campus Curitibanos
- Campus Joinville
- Campus Trindade

Para obter os menus atualizados, foi necessário acessar o link mais recente de cada página de restaurante, o qual direciona para um arquivo PDF contendo as informações desejadas. O desenvolvimento consistiu em realizar o *parsing* de cada PDF obtido.

Figura 29 – Parte do cardápio semanal do campus Trindade

RESTAURANTE UNIVERSITÁRIO - PROGRAMAÇÃO SEMANAL DO CARDÁPIO			
SEGUNDA-FEIRA	***Arroz parboilizado / Arroz integral / Feijão com vegetais***		
15-mai-23	***Carne:***	Risoto de Frango	
	Complemento:	Batata palha	
	Saladas:	Rúcula	Rabanete Ralado
	Molho de mostarda		Maçã
TERÇA-FEIRA	***Arroz parboilizado / Arroz integral / Feijão***		
16-mai-23	***Carne:***	Filé mignon suíno acebolado ao molho agridoce	
	Complemento:	Vagem refogada	
	Saladas:	Alface	Beterraba Ralada
	Vinagrete		Banana

Fonte: CARDÁPIO... (2023)

A estrutura de dados utilizada para representar as informações do menu de cada restaurante foi um formato JSON. Essa estrutura contém os detalhes sobre as refeições oferecidas. A figura 30 mostra uma parte dos dados extraídos do PDF da figura 29. A chave corresponde a data da segunda-feira da semana em que o cardápio será servido. Alguns restaurantes possuem cardápio mensal, e como o aplicativo mostra os dados por semana, essa estrutura facilita o acesso aos dados.

Após o *parsing* de cada PDF, os dados extraídos são armazenados no banco de dados Firebase. Essa escolha se deve às características de escalabilidade e facilidade de integração oferecidas pela plataforma.

O script foi implementado em Python e empacotado em um container Docker. A utilização do container Docker permitiu a criação de um ambiente isolado e consistente, garantindo que todas as bibliotecas e pacotes necessários estivessem disponíveis durante a execução das funções Lambda.

A coleta de dados foi automatizada e programada para ser executada em intervalos regulares. Para isso, o serviço foi implantado em um ambiente de execução no serviço de computação em nuvem da Amazon, conhecido como Amazon Lambda. A função Lambda foi configurada para executar duas vezes ao dia, às 7h e às 10h (horário de Brasília), dessa forma, o aplicativo atualiza as informações do menu diariamente, mantendo os dados sempre atualizados para os usuários.

Figura 30 – Dados em JSON do menu da Trindade


```
{
    "2023-05-15": {
      "friday": {
        "common": [
          "Arroz",
          "Arroz integral",
          "Feijão preto",
          "Bife",
          "Acelga",
          "Beterraba ralada",
          "Vinagrete",
          "Banana"
        ],
        "dinner": [
          "Purê de batatas"
        ],
        "lunch": [
          "Abóbora ao forno"
        ]
      },
      "monday": {
        "common": [
          "Arroz",
          "Arroz integral",
          "Feijão preto com vegetais",
          "Risoto de frango",
          "Batata palha",
          "Rúcula",
          "Rabanete ralado",
          "Molho de mostarda",
          "Maçã"
        ]
      }
    }
  }
```


Fonte: Autor

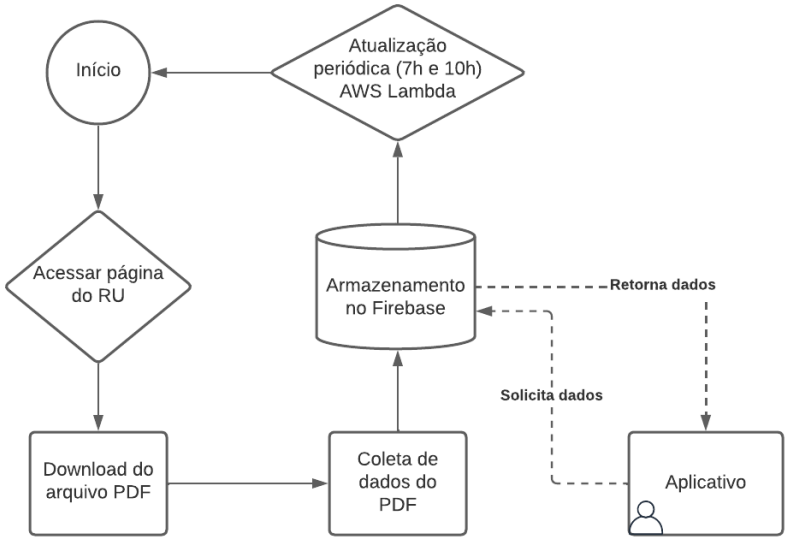
A figura 31 apresenta um exemplo de diagrama de fluxo que pode ser utilizado para ilustrar o processo de coleta de dados dos menus dos restaurantes, desde o acesso à página do Restaurante Universitário até a disponibilização dos dados no banco de dados Firebase e o consumo desses dados pelo aplicativo.

O diagrama de fluxo demonstra visualmente as etapas do processo, incluindo a extração dos links dos menus, o download dos arquivos PDF, o *parsing* dos PDFs para coleta das informações do cardápio e, por fim, o armazenamento no banco de dados Firebase. As informações coletadas serão consumidas posteriormente pelo aplicativo desenvolvido.

O banco de dados não-relacional servido pelo Firebase ficou modelado como mostra o diagrama da Figura 32, sendo que um documento é uma unidade básica de dados e é representado como um objeto JSON que contém pares chave-valor. Os documentos são organizados em coleções, que podem ser consideradas como tabelas em um banco de dados relacional. Uma coleção é um grupo lógico de documentos que têm algum tipo de relação.

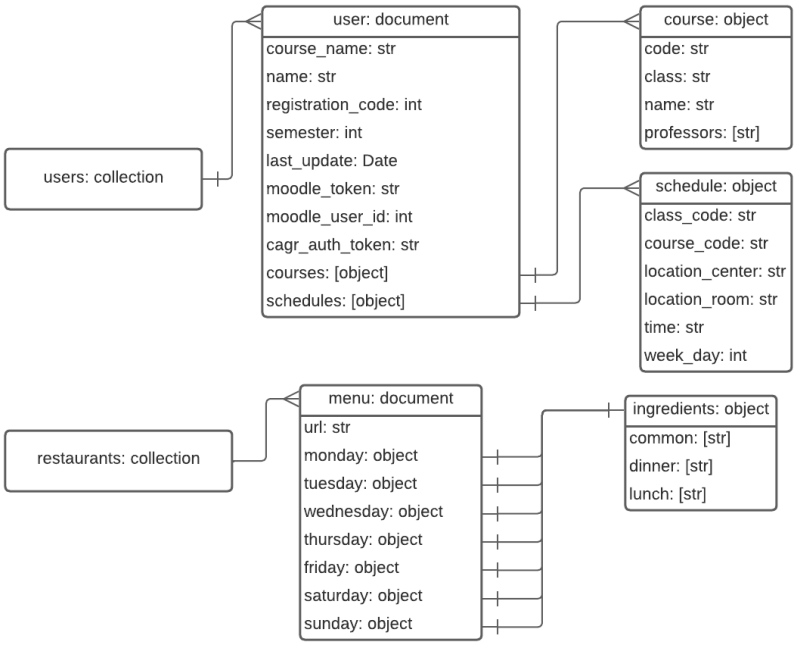
As únicas informações armazenadas são as coletadas do webservice do CAGR. Os dados referentes ao webservide do Moodle não são armazenados e são feitas novas requisições em cada novo acesso do usuário para que a informação esteja sempre atualizada.

Figura 31 – Diagrama de fluxo da coleta de dados do menu dos restaurantes



Fonte: Autor

Figura 32 – Diagrama do Banco de Dados



Fonte: Autor

6 APLICAÇÃO DA PESQUISA DE SATISFAÇÃO

Para alcançar um número significativo de usuários para a avaliação do aplicativo, foi necessário empregar uma estratégia que envolveu o envio de convites para utilização do aplicativo para usuários do Moodle que haviam acessado a plataforma nas últimas 24 horas antes do envio.

No Moodle, qualquer usuário tem acesso aos dados dos outros usuários matriculados na plataforma, sendo possível ordenar a lista por última visualização. Também, qualquer usuário possui acesso e a possibilidade de enviar mensagens para qualquer aluno, mantendo assim um canal de contato. Adicionalmente, o autor deste trabalho está cadastrado no Moodle como aluno e possui a possibilidade de enviar mensagens para qualquer outro aluno. Sendo assim, foram compilados os e-mails de 500 usuários provenientes de diferentes cursos e campi da instituição. Essa abordagem garantiu uma amostra diversificada e representativa da comunidade acadêmica.

A plataforma Mailchimp foi então utilizada para o envio de e-mails personalizados para esse grupo de usuários. O conteúdo dos e-mails foi cuidadosamente elaborado em um formato breve e conciso, com o objetivo de garantir que os destinatários pudessem ler rapidamente as informações essenciais sem perder o interesse, destacando os principais benefícios e funcionalidades do aplicativo em questão.

No sétimo dia após o lançamento da campanha, observou-se que 394 dos 500 destinatários abriram o e-mail, e desses, um total de 97 usuários instalaram o aplicativo em seus dispositivos móveis, obtendo uma taxa de conversão de 24.6%. Outros 5 usuários foram convidados a utilizar o aplicativo, sendo 3 dos cursos de Ciências da Computação, 1 de Sistemas da Informação e 1 da Engenharia Eletrônica, totalizando 102 usuários.

Além disso, com o objetivo de obter uma avaliação mais aprofundada da experiência dos usuários, foi implementado um formulário de pesquisa no aplicativo, apresentado como um pop-up. Essa estratégia permitiu a coleta de informações sobre a usabilidade, satisfação do usuário e possíveis melhorias a serem realizadas no aplicativo. Dos 97 usuários que instalaram o aplicativo, um total de 14 pessoas responderam ao formulário de pesquisa, as outras 5 respostas foram dos alunos convidados.

O questionário utilizado para coletar informações dos usuários foi o SUS, que é uma escala amplamente reconhecida e validada para medir a usabilidade de um sistema ou interface de usuário. Ele consiste em um conjunto de 10 perguntas que avaliam a facilidade de uso, eficiência e satisfação geral do usuário em relação ao sistema avaliado. Cada pergunta é respondida em uma escala de Likert de 5 pontos, variando de "discordo totalmente"a "concordo totalmente".

Dos resultados, a maioria dos usuários demonstrou interesse em utilizar o aplicativo com frequência. Eles reconheceram o valor e a utilidade do aplicativo, indicando que ele atende às suas necessidades e oferece benefícios que os incentivariam a usá-lo regularmente (Figura 33).

Foi observado que apenas alguns usuários consideraram o aplicativo mais complexo do que o necessário. Isso sugere que, em geral, a interface e as funcionalidades do aplicativo são percebidas como adequadas e não sobrecarregam os usuários com informações ou processos

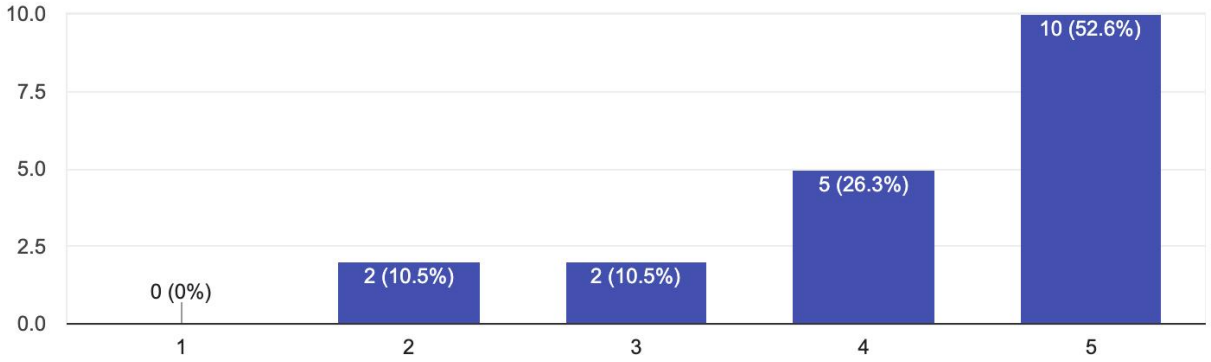
Tabela 5 – Respostas dos Alunos por Curso

Curso	Quantidade
Agronomia	1
Antropologia	1
Arquitetura	1
Ciências da Computação	3
Engenharia Elétrica	2
Engenharia Eletrônica	1
Engenharia Sanitária e Ambiental	1
Farmácia	1
Letras Português	1
Medicina	2
Química Licenciatura	1
Serviço Social	1
Sistemas de Informação	2
Zootecnia	1

Tabela 6 – Respostas dos Alunos por Campus

Curso	Quantidade
Araranguá	1
Blumenau	1
Florianópolis	17

Figura 33 – Pergunta 1: Acho que gostaria de utilizar este aplicativo com frequência.



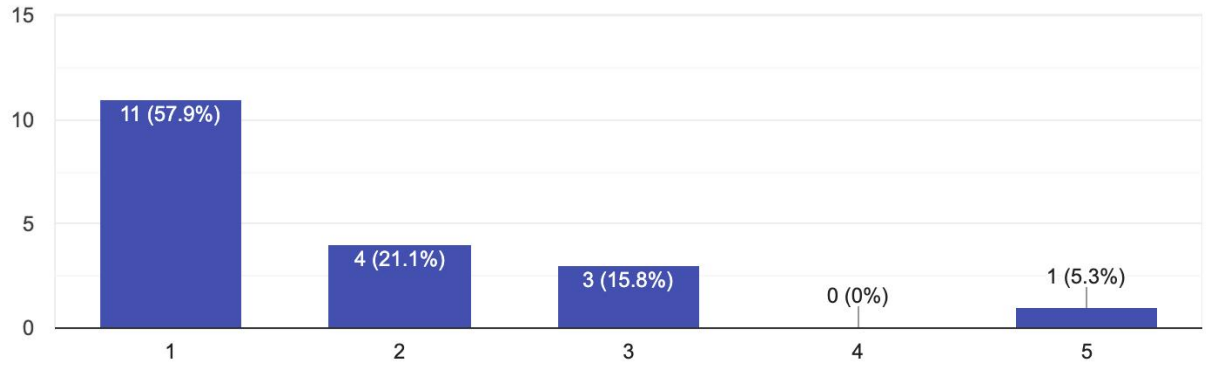
Fonte: Autor

excessivamente complexos (Figura 34).

A maioria dos usuários avaliou o aplicativo como fácil de usar. Isso indica que a interface do aplicativo é intuitiva e amigável, facilitando a navegação e a interação dos usuários com as diferentes funcionalidades disponíveis (Figura 35).

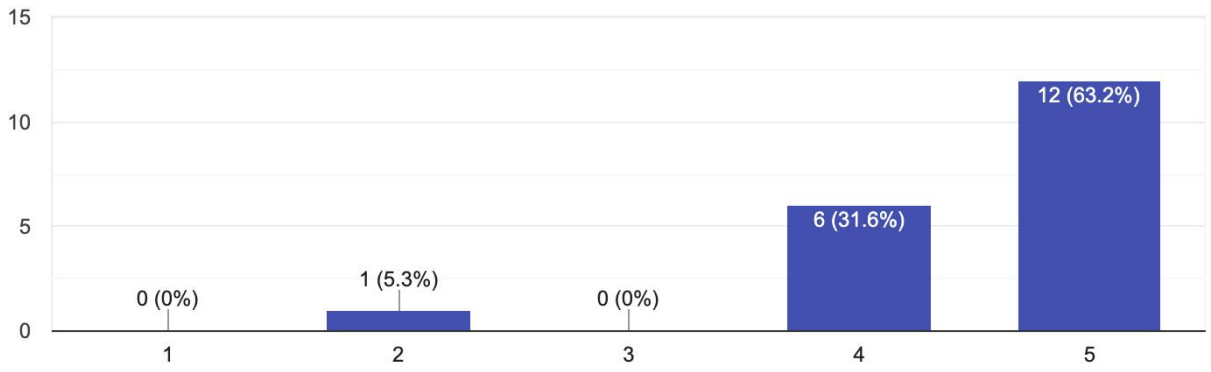
Apenas alguns usuários sentiram que precisariam de suporte adicional para utilizar o aplicativo. Essa observação sugere que, em geral, os usuários se sentem confiantes e capazes de explorar e utilizar as funcionalidades do aplicativo sem assistência adicional (Figura 36).

Figura 34 – Pergunta 2: Considerei o aplicativo mais complexo do que necessário.



Fonte: Autor

Figura 35 – Pergunta 3: Achei o aplicativo fácil de usar.



Fonte: Autor

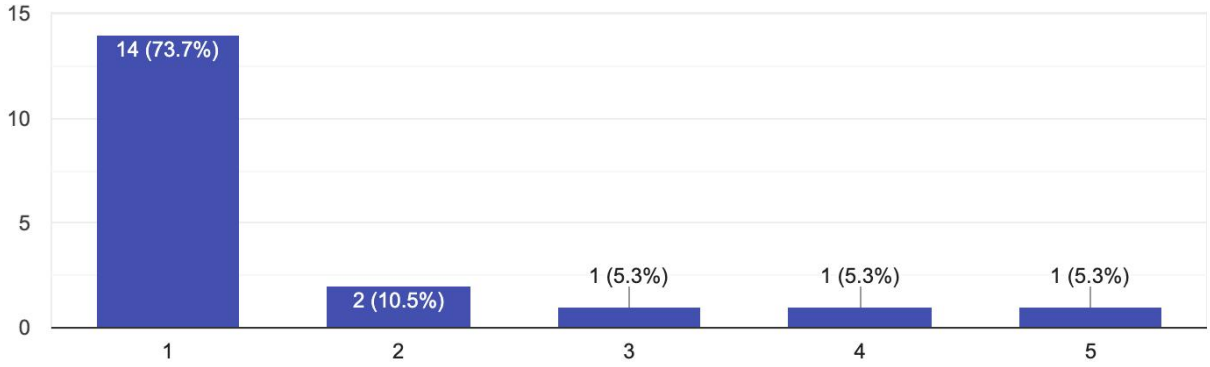
A maioria dos usuários percebeu uma integração satisfatória entre as diversas funcionalidades do aplicativo. Isso indica que as diferentes partes do aplicativo estão bem conectadas e proporcionam uma experiência coesa e consistente (Figura 37).

Alguns usuários mencionaram que encontraram algumas inconsistências no aplicativo. Essa observação sugere que existem áreas que podem ser aprimoradas em termos de coerência e padronização, a fim de proporcionar uma experiência mais consistente e livre de contradições (Figura 38).

A maioria dos usuários acredita que grande parte das pessoas seria capaz de aprender a utilizar o aplicativo rapidamente. Isso sugere que a interface e as funcionalidades do aplicativo são intuitivas o suficiente para que os usuários possam se familiarizar e começar a utilizar o aplicativo com facilidade (Figura 39).

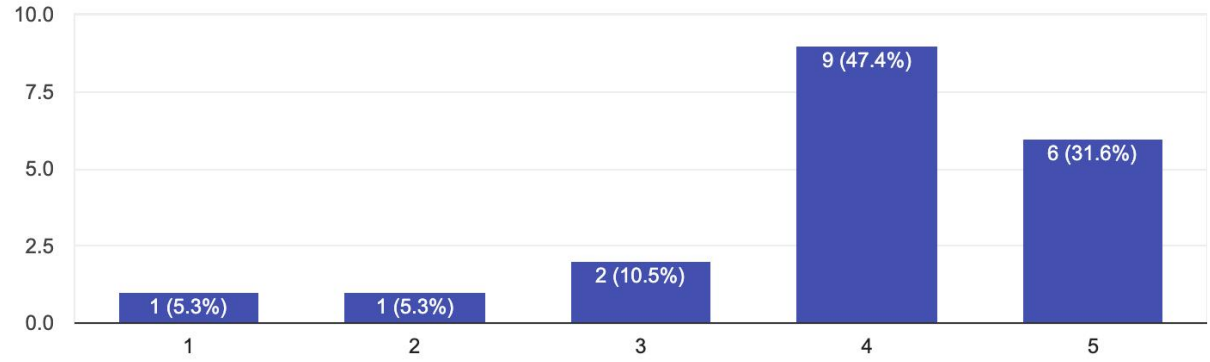
Alguns usuários consideraram o aplicativo como sendo muito complicado de utilizar. Essa percepção pode indicar a necessidade de simplificar certas funcionalidades ou fornecer orientações mais claras para ajudar os usuários a se sentirem mais confiantes ao utilizar o

Figura 36 – Pergunta 4: Acho que necessitaria de ajuda do suporte para conseguir utilizar este aplicativo.



Fonte: Autor

Figura 37 – Pergunta 5: Considerei que as várias funcionalidades deste aplicativo estavam bem integradas.



Fonte: Autor

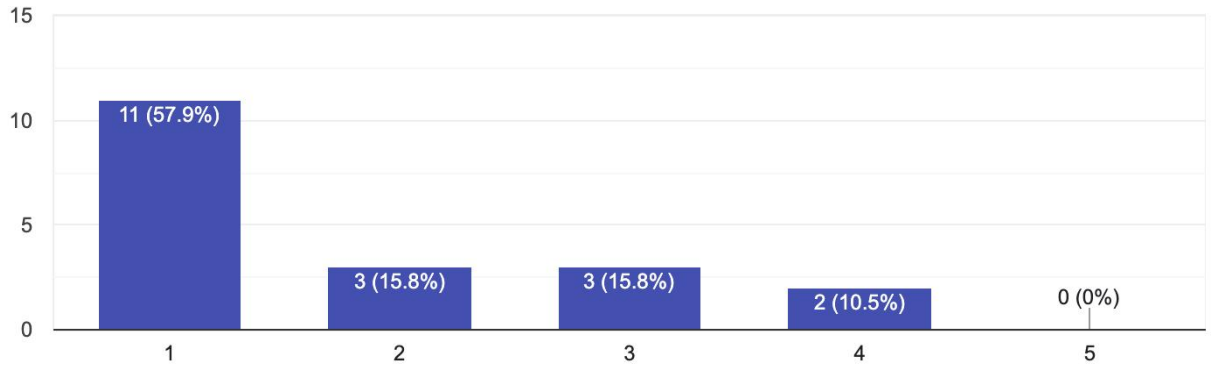
aplicativo (Figura 40).

A maioria dos usuários considerou o aplicativo confiável. Isso é um indicativo positivo, pois a confiabilidade é um aspecto crucial para a aceitação e satisfação dos usuários em relação ao aplicativo (Figura 41).

Alguns usuários mencionaram que precisaram aprender novas coisas antes de conseguir utilizar o aplicativo. Essa observação sugere que, apesar da maioria dos usuários considerarem o aplicativo fácil de usar, ainda existem aspectos que podem ser aprimorados para tornar a curva de aprendizado mais suave e reduzir a necessidade de aprender novos conceitos ou procedimentos antes de utilizar o aplicativo (Figura 42).

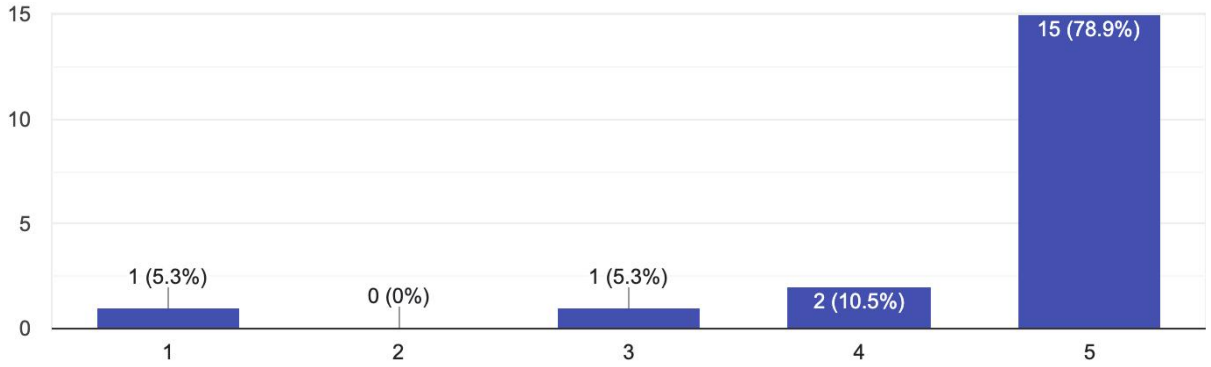
Um campo opcional para sugestões e observações foi incluído no formulário e dentre os resultados, duas observações se destacaram como pontos importantes. A primeira foi automatizar o procedimento de cadastro da chave de segurança do Moodle, visando facilitar o primeiro

Figura 38 – Pergunta 6: Achei que este aplicativo tinha muitas inconsistências.



Fonte: Autor

Figura 39 – Pergunta 7: Suponho que a maioria das pessoas aprenderia a utilizar rapidamente este aplicativo.

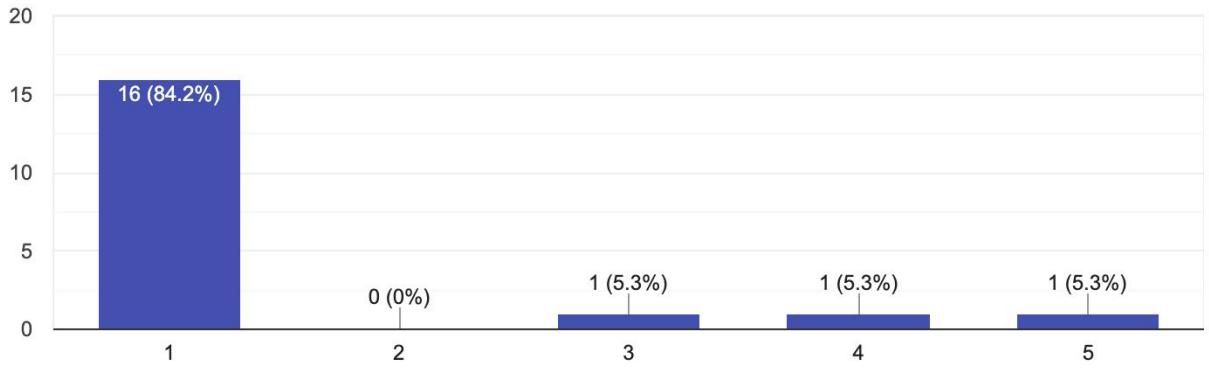


Fonte: Autor

contato dos usuários com o aplicativo. Por outro lado, um usuário relatou um problema específico relacionado ao carregamento da página de login, ressaltando a necessidade de investigar e solucionar eventuais falhas de funcionamento para garantir uma melhor experiência aos usuários.

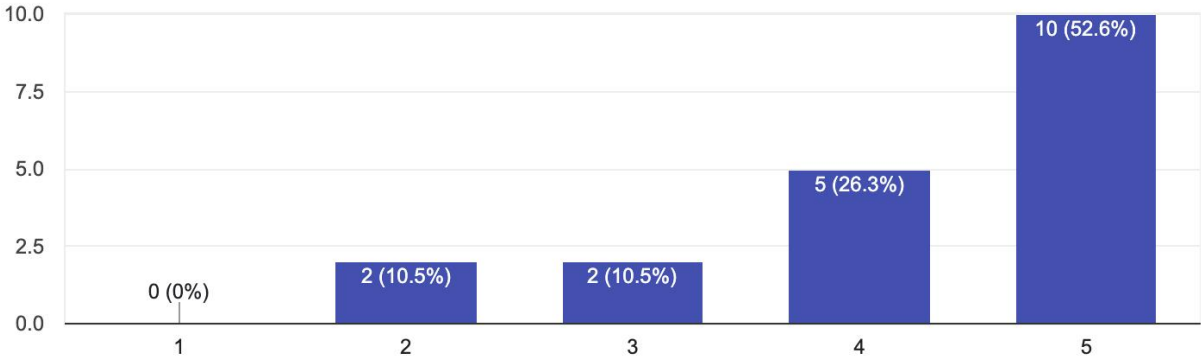
Após aplicar o SUS às respostas dos usuários, a pontuação total obtida foi de 84.34, que segundo as diretrizes, indica um nível excelente de satisfação e usabilidade do aplicativo. De acordo com estudos anteriores, uma pontuação acima de 74 é considerada excelente, o que confirma a qualidade e eficácia do aplicativo avaliado. Esses resultados reforçam a ideia de que os usuários estão altamente satisfeitos com a experiência de uso, considerando-o fácil de usar, integrado e confiável.

Figura 40 – Pergunta 8: Considerei o aplicativo muito complicado de utilizar.



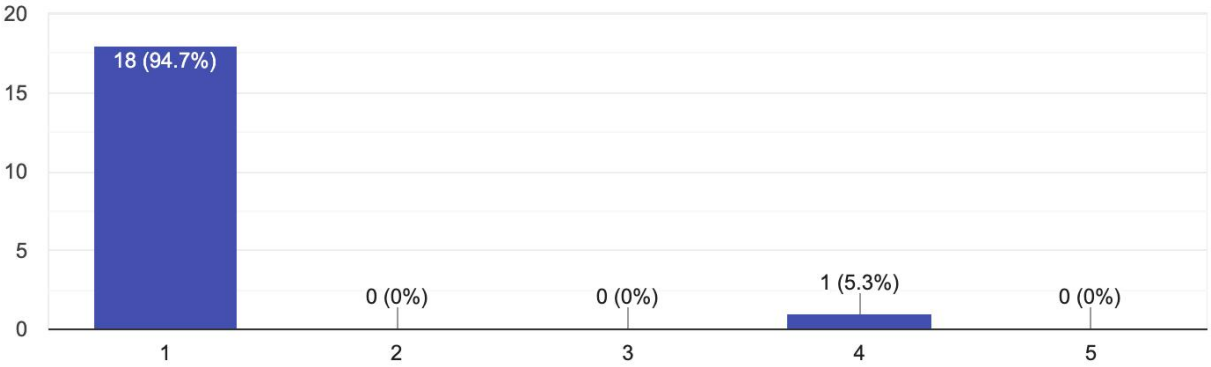
Fonte: Autor

Figura 41 – Pergunta 9: O aplicativo pareceu confiável.



Fonte: Autor

Figura 42 – Pergunta 10: Precisei aprender várias coisas novas antes de conseguir usar o aplicativo.



Fonte: Autor

7 CONCLUSÃO

Através da utilização das tecnologias modernas e eficientes, como Flutter, Firebase e Amazon Lambda, foi possível desenvolver um aplicativo que proporcionou uma experiência satisfatória aos usuários.

O uso do Flutter como framework para o desenvolvimento do aplicativo permitiu a criação de uma interface de usuário atraente e responsiva, com a vantagem de um código-fonte único para múltiplas plataformas. A linguagem de programação Dart, combinada com a flexibilidade do Flutter, proporcionou um ambiente de desenvolvimento produtivo e eficaz.

A integração com o Firebase foi fundamental para o armazenamento dos dados do aplicativo, garantindo um acesso rápido às informações, enquanto o Firebase Authentication assegurou a segurança e autenticação dos usuários.

A hospedagem do coletor de dados dos restaurantes na Amazon, através do serviço Amazon Lambda, mostrou-se uma escolha acertada em termos de disponibilidade. Essa abordagem permitiu que a aplicação fosse executada de forma programada em horários predeterminados, garantindo a coleta e atualização automática dos cardápios dos restaurantes da UFSC. A utilização do Docker para criação da imagem do Python e instalação de dependências também contribuiu para uma fácil implantação e manutenção do sistema no servidor da Amazon.

A exibição das disciplinas matriculadas, grade de horários, cardápio do Restaurante Universitário, eventos do calendário, notas cadastradas e notícias da UFSC proporcionou aos usuários acesso rápido e conveniente a informações relevantes para o seu dia a dia acadêmico.

Por fim, a pesquisa de satisfação realizada com os usuários demonstrou resultados positivos. Através do questionário SUS, foi possível avaliar a usabilidade e a satisfação geral dos usuários com o aplicativo. Os resultados obtidos indicaram uma pontuação excelente de 84.34, confirmando a usabilidade do aplicativo. A diversidade de cursos dos usuários participantes da pesquisa reforça a representatividade dos resultados e a aceitação do aplicativo em diferentes áreas acadêmicas.

Portanto, a combinação das tecnologias utilizadas, juntamente com os resultados positivos da pesquisa de satisfação, consolidam o êxito do desenvolvimento e aplicação do aplicativo proposto. O uso do Flutter, Firebase e Amazon Lambda possibilitou a criação de um aplicativo prático, eficiente e de alto desempenho, que proporcionou aos usuários uma experiência satisfatória e contribuiu para a melhoria do cotidiano acadêmico.

7.1 TRABALHOS FUTUROS

Durante o desenvolvimento deste trabalho, identificou-se que a API do Moodle possui um grande potencial de exploração, oferecendo diversas oportunidades para aprimorar o aplicativo desenvolvido e adicionar novos recursos.

Um possível aprimoramento está relacionado à obtenção automática da chave de acesso ao Moodle. Atualmente, os usuários precisam cadastrar manualmente a chave de acesso no

aplicativo. No entanto, seria interessante explorar mecanismos de autenticação do Moodle para obter a chave de acesso de forma automática, simplificando ainda mais o processo para o usuário.

Além disso, sugere-se a exploração de recursos adicionais para a exibição do conteúdo das disciplinas. No presente trabalho, foram exibidas as informações de notas e eventos das disciplinas do Moodle, sendo interessante expandir essa funcionalidade para exibir o conteúdo completo das disciplinas, como links para recursos relevantes, incluindo materiais de estudo em diversos formatos, como documentos, apresentações e PDFs. Dessa forma, os usuários terão acesso facilitado aos materiais de estudo relacionados a cada disciplina, enriquecendo a experiência de aprendizado.

Outra sugestão para trabalhos futuros é a incorporação de recursos avançados de interação com as atividades do Moodle. Por exemplo, é possível implementar funcionalidades que permitam aos usuários enviar tarefas diretamente pelo aplicativo, sem a necessidade de acessar o ambiente virtual de aprendizagem por meio de um navegador. Essa integração mais estreita entre o aplicativo e o Moodle tornará as atividades acadêmicas mais práticas e eficientes.

REFERÊNCIAS

ADETUNJI, Oluwatofunmi *et al.* Dawning of Progressive Web Applications (PWA): Edging Out the Pitfalls of Traditional Mobile Development. **American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)**, v. 68, n. 1, p. 85–99, 2020.

ALLAIN, Hugo *et al.* Improving productivity and reducing costs of mobile app development with Flutter and Backend-as-a-Service, 2020.

ALTALEB, Bashar; KHALAF, Abo *et al.* **Scalability and Economy of Amazon Lambda, EKS, and ECS**. [*S.l.*: *s.n.*], 2022.

BANGOR, Aaron; KORTUM, Philip; MILLER, James. Determining what individual SUS scores mean: Adding an adjective rating scale. **Journal of usability studies**, Citeseer, v. 4, n. 3, p. 114–123, 2009.

BEHL, Kashish; RAJ, Gaurav. Architectural pattern of progressive web and background synchronization. *In:* IEEE. 2018 International Conference on Advances in Computing and Communication Engineering (ICACCE). [*S.l.*: *s.n.*], 2018. P. 366–371.

BOULAKBECH, Marwa *et al.* Visual configuration for restful mobile web mashups. *In:* IEEE. 2017 IEEE International Conference on Web Services (ICWS). [*S.l.*: *s.n.*], 2017. P. 870–873.

BROOKE, John *et al.* SUS-A quick and dirty usability scale. **Usability evaluation in industry**, London–, v. 189, n. 194, p. 4–7, 1996.

BROWN, S.; DAVIS, M. The Impact of Click Reduction on User Satisfaction and Task Completion Time in Web Applications. **Journal of Usability Studies**, 2022.

CARDÁPIO semanal do Campus Trindade. [*S.l.*: *s.n.*], 2023. Recuperado de `https://siteru.paginas.ufsc.br/files/2023/05/12.-15-a-21.mai_.pdf`.

CHANG, YoungHyun; OH, SangYeob. A study on the development of one source multi use cross-platform based on zero coding. **Multimedia Tools and Applications**, Springer, v. 74, n. 7, p. 2219–2235, 2015.

CHATTERJEE, Nilanjan *et al.* Real-time communication application based on android using Google firebase. **Int. J. Adv. Res. Comput. Sci. Manag. Stud**, v. 6, n. 4, 2018.

CYBIS, Walter; BETIOL, Adriana Holtz; FAUST, Richard. **Ergonomia e usabilidade: conhecimentos, métodos e aplicações**. [*S.l.*]: Novatec editora, 2017.

DAHL, Ola. **Exploring End User's Perception of Flutter Mobile Apps**. [*S.l.*: *s.n.*], 2019.

DENNISS, William; BRADLEY, John. OAuth 2.0 for Native Apps. **Internet Engineering Task Force (IETF), Published Oct**, 2017.

DOGLIO, Fernando. **Pro REST API Development with Node. js**. [*S.l.*]: Apress, 2015.

FETT, Daniel; KÜSTERS, Ralf; SCHMITZ, Guido. A comprehensive formal security analysis of OAuth 2.0. *In:* PROCEEDINGS of the 2016 ACM SIGSAC Conference on Computer and Communications Security. [*S.l.*: *s.n.*], 2016. P. 1204–1215.

GAO, Yumeng. Design and Implementation of End-User Programming Tools for Web Mashups, 2019.

GLEZ-PEÑA, Daniel *et al.* Web scraping technologies in an API world. **Briefings in bioinformatics**, Oxford University Press, v. 15, n. 5, p. 788–797, 2014.

GRAMMEL, Lars; STOREY, Margaret-Anne. A survey of mashup development environments. *In:* THE smart internet. [*S.l.*]: Springer, 2010. P. 137–151.

GUTIÉRREZ-CARREÓN, Gustavo; DARADOUMIS, Thanasis; JORBA, Josep. Integrating learning services in the cloud: An approach that benefits both systems and learning. **Journal of Educational Technology & Society**, JSTOR, v. 18, n. 1, p. 145–157, 2015.

IBGE, Instituto Brasileiro de Geografia e Estatística. Pesquisa nacional por amostra de domicílios contínua. **Estatísticas sociais**, 2019.

ISO. 9241-11. Ergonomic requirements for office work with visual display terminals (VDTs) — Part II guidance on usability, 1998.

EL-KASSAS, Wafaa S *et al.* Taxonomy of cross-platform mobile applications development approaches. **Ain Shams Engineering Journal**, Elsevier, v. 8, n. 2, p. 163–190, 2017.

KAYA, Aycan; OZTURK, Reha; GUMUSSOY, Cigdem Altin. Usability measurement of mobile applications with system usability scale (SUS). *In:* INDUSTRIAL engineering in the big data era. [*S.l.*]: Springer, 2019. P. 389–400.

KHAN, Asharul Islam; AL-BADI, Ali; AL-KINDI, Mahmood. Progressive web application assessment using AHP. **Procedia Computer Science**, Elsevier, v. 155, p. 289–294, 2019.

KITCHENHAM, Barbara *et al.* Systematic literature reviews in software engineering–a systematic literature review. **Information and software technology**, Elsevier, v. 51, n. 1, p. 7–15, 2009.

LATIF, Mounaim; LAKHRISSI, Younes; ES-SBAI, Najia *et al.* Cross platform approach for mobile application development: A survey. *In:* IEEE. 2016 International Conference on Information Technology for Organizations Development (IT4OD). [*S.l.*: *s.n.*], 2016. P. 1–5.

LEAL, Angelo Manoel de Matos; LEAL, Adson Pereira *et al.* Scholar: desenvolvimento de um aplicativo móvel genérico de apoio acadêmico a estudantes em universidades. Florianópolis, SC, 2019.

LEWIS, James Jim R; SAURO, Jeff. Revisiting the Factor Structure of the System Usability Scale. **Journal of Usability Studies**, v. 12, n. 4, 2017.

LEWIS, James R; BROWN, Joshua; MAYES, Daniel K. Psychometric evaluation of the EMO and the SUS in the context of a large-sample unmoderated usability study. **International Journal of Human-Computer Interaction**, Taylor & Francis, v. 31, n. 8, p. 545–553, 2015.

MALAVOLTA, Ivano *et al.* Assessing the impact of service workers on the energy efficiency of progressive web apps. *In:* IEEE. 2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft). [*S.l.*: *s.n.*], 2017. P. 35–45.

MARTIN-LOPEZ, Alberto; SEGURA, Sergio; RUIZ-CORTÉS, Antonio. Test coverage criteria for RESTful web APIs. *In:* PROCEEDINGS of the 10th ACM SIGSOFT International Workshop on Automating TEST Case Design, Selection, and Evaluation. [*S.l.*: *s.n.*], 2019. P. 15–21.

MATTOSINHO, FJAP. Mining Product Opinions and Reviews on the Web. **Technische Universitat Dresden**, 2010.

MOL, Mayke *et al.* Dimensionality of the system usability scale among professionals using internet-based interventions for depression: a confirmatory factor analysis. **BMC psychiatry**, Springer, v. 20, p. 1–10, 2020.

OLSSON, Matilda. **A Comparison of Performance and Looks Between Flutter and Native Applications: When to prefer Flutter over native in mobile application development**. [*S.l.*: *s.n.*], 2020.

SAURKAR, Anand V; PATHARE, Kedar G; GODE, Shweta A. An overview on web scraping techniques and tools. **International Journal on Future Revolution in Computer Science & Communication Engineering**, v. 4, n. 4, p. 363–367, 2018.

SCHNEIDER, Cristiano. Moodle Mobile (M-learning Moodle) avaliação de usabilidade com o uso do SUS-System Usability Scale. Universidade Federal de Santa Maria, 2017.

SMITH, J.; JOHNSON, E. Reducing User Effort by Minimizing Clicks: A Comparative Study. **International Journal of Human-Computer Interaction**, 2021.

TRAN, Thanh. Flutter Native Performance and Expressive UI/UX, 2020.

WOODLAND, Michael; KLASS, Dan. Podcast solutions: the complete guide to podcasting. **Nueva York: Friendsoft**, 2005.

YAN, Weiwei; DENG, Shengli; ZHANG, Yin. Factors influencing the intention to use information service mashups: an empirical study of digital libraries in China. **The Electronic Library**, Emerald Group Publishing Limited, 2016.

ZHAI, Zhongyi *et al.* An end-user oriented tool suite for development of mobile applications. *In:* PROCEEDINGS of the 31st IEEE/ACM International Conference on Automated Software Engineering. [*S.l.*: *s.n.*], 2016. P. 768–773.

Apêndices

APÊNDICE A – E-MAIL DE CONVITE PARA INSTALAÇÃO DA APLICAÇÃO

Prezado(a) estudante!

Tenho o prazer de anunciar o lançamento de um aplicativo especialmente desenvolvido para facilitar a sua vida acadêmica.

Com integração ao CAGR, Moodle e Restaurantes Universitários, o Gradufsc oferece acesso rápido a informações relevantes sobre suas disciplinas, tudo em um só lugar. Baixe gratuitamente na loja de aplicativos Android. Pesquise por "Gradufsc" ou clique no link abaixo.

Abrir na Play Store

Desenvolvido como projeto de TCC por Rafael Lazzaretti Madalóz, aluno de Ciências da Computação da UFSC, sugestões e reclamações são bem-vindas. Compartilhe seu feedback por meio do formulário no link abaixo.

Responder Feedback

Experimente o Gradufsc e ajude a aprimorar a experiência acadêmica dos alunos da UFSC. Conto com sua participação para tornar o aplicativo ainda melhor.

Fonte: Autor

APÊNDICE B – REPOSITÓRIO PÚBLICO DAS APLICAÇÕES

Aplicativo móvel: https://github.com/rafaelmadaloz/gradufsc-app
Parser de restaurantes: https://github.com/rafaelmadaloz/ru-parser

APÊNDICE C – ARTIGO

Desenvolvimento de aplicativo móvel para consulta de informações acadêmicas para alunos de graduação da UFSC

Rafael L. Madalóz[1]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-370 – Florianópolis – SC – Brazil

rafael.madaloz@grad.ufsc.br

Abstract. *This article presents a mobile application development project aimed at facilitating access to information regarding academic activities and routines for undergraduate students at the Federal University of Santa Catarina (UFSC). Currently, there are scattered systems within the university that are not adapted for mobile devices, thereby limiting accessibility and user-friendliness. The project's objective is to address this gap by providing an intuitive and user-friendly solution that integrates with existing systems to deliver important information and an enhanced user experience. Additionally, a usability study was conducted to gather feedback and continuously improve the solution, ensuring a satisfactory user experience.*

Resumo. *Este artigo apresenta um projeto de desenvolvimento de um aplicativo móvel para facilitar o acesso a informações das atividades acadêmicas e rotinas dos alunos de graduação da Universidade Federal de Santa Catarina (UFSC). Atualmente, existem sistemas dispersos na universidade que não estão adaptados para dispositivos móveis, o que limita a acessibilidade e a usabilidade dos usuários. O objetivo do projeto é preencher essa lacuna, oferecendo uma solução intuitiva e amigável, integrada aos sistemas existentes, para fornecer informações importantes e uma experiência mais completa. Além disso, foi realizada uma pesquisa de usabilidade, fornecendo informações para melhorar continuamente a solução e garantir uma experiência satisfatória aos usuários.*

1. Introdução

Os alunos de graduação da Universidade Federal de Santa Catarina possuem uma gama de sistemas disponíveis para acessar informações sobre o dia-a-dia acadêmico.

Por exemplo, para que possam verificar seus horários de aula, é necessário entrarem e fazerem a autenticação no Sistema de Controle Acadêmico da Graduação (CAGR). Já se quiserem ver o cardápio do Restaurante Universitário (RU), precisam acessar o site do restaurante e fazer o download do arquivo PDF correspondente. Caso necessitem ver a data da próxima prova de uma determinada disciplina ou saberem a nota de uma atividade, é necessário utilizar o Moodle. Se quiserem saber as novidades que estão acontecendo, precisam encontrar o site de notícias da UFSC.

Por mais que existam vários sistemas funcionais para diversas tarefas, alguns não são responsivos e possuem uma interface antiga, ou seja, existe uma dificuldade de ler o conteúdo a partir de um dispositivo móvel, além de que esses sistemas estão dispersos em

múltiplos domínios. Portanto, uma aplicação que una as funcionalidades desses diversos sistemas com uma interface responsiva seria benéfica para os estudantes.

Portanto, o objetivo deste trabalho é desenvolver um aplicativo móvel abrangente que ofereça aos estudantes de graduação da UFSC acesso a diversas informações relevantes obtidas dos sistemas mais importantes, além de analisar a usabilidade da aplicação através de uma pesquisa com os usuários.

2. Fundamentação Teórica

Neste capítulo, são apresentados os fundamentos teóricos que sustentam o desenvolvimento do aplicativo proposto. Esses conceitos são cruciais para compreensão da abordagem utilizada e as tecnologias adotadas para a implementação do projeto.

2.1. Aplicativo Híbrido

Os aplicativos híbridos se tornaram populares por oferecerem uma experiência offline, ou seja, o aplicativo pode funcionar mesmo sem conexão com a internet, deixando de ser restrito apenas à web e se tornando um aplicativo real. Esses aplicativos são executados em um webview existente dentro de um aplicativo nativo, o que permite o acesso a recursos nativos, como sensores e notificações. Essa ponte entre a parte nativa e o webview é o que proporciona a funcionalidade híbrida [Adetunji et al. 2020].

No projeto em questão, a abordagem híbrida foi utilizada com o framework Flutter, que usa a linguagem de programação Dart e pode ser compilado para várias plataformas, como Android, iOS, Windows, Mac, Linux, Google Fuchsia e Web. O Flutter oferece um ambiente de desenvolvimento produtivo, permitindo criar aplicativos a partir de um único código-base. Isso economiza tempo e esforço, pois é necessário desenvolver e manter apenas um aplicativo em vez de vários para cada plataforma. Além disso, o Flutter oferece uma experiência de usuário nativa, utilizando widgets personalizados que se assemelham aos elementos de interface de cada plataforma, garantindo uma aparência autêntica de aplicativo nativo.

2.2. Flutter

Flutter é um Kit de Desenvolvimento de Software (SDK) moderno e ágil, que possui uma estrutura de interface móvel para a construção de aplicativos híbridos lançados em 2018 pelo Google e é um SDK de código aberto, o que significa que a comunidade pode ajudar a desenvolver e melhorar a estrutura.

O objetivo do Google com a criação do Flutter é fornecer uma estrutura capaz de melhorar a velocidade de desenvolvimento e oferecer aplicativos híbridos com desempenho e uma interface gráfica semelhantes aos aplicativos nativos [Olsson 2020]. Além disso, o Flutter tem uma linguagem de renderização e codificação diferente de outras estruturas, que geralmente utilizam JavaScript, [Dahl 2019]. No caso do Flutter, a linguagem de programação usada é Dart.

Este framework foi escolhido para o desenvolvimento pela suas vantagens, que são a velocidade de desenvolvimento, pois ele utiliza uma linguagem de programação única e um conjunto de widgets personalizáveis que facilitam a criação de interfaces de usuário atraentes e responsivas.

2.3. Firebase

Firebase é uma estrutura útil para a construção de aplicativos portáteis e da web para quem requer um banco de dados em tempo real, o que implica que quando um usuário atualiza um registro no banco de dados, a atualização deve ser transmitida a cada usuário instantaneamente. Ele fornece uma plataforma básica e unificada para muitos aplicativos, juntamente com uma série de outros recursos do Google incluídos no serviço. O Firebase lida com a maior parte do trabalho do lado do servidor quando se trata de desenvolvimento de aplicativos [Chatterjee et al. 2018].

Os recursos utilizados dessa plataforma são as seguintes:

- Autenticação: Permite que apenas usuários autorizados acessarem o aplicativo. Ele oferece suporte à autenticação por meio de senhas, e-mail ou nome de usuário, número de telefone, ou através de contas associadas a outros serviços.
- Banco de dados em tempo real: É um banco de dados em nuvem e não precisa de consultas baseadas em SQL para armazenar e buscar dados. É altamente confiável, portanto, mesmo se a conexão for perdida, os dados serão mantidos. Os dados são armazenados como Notação de Objetos JavaScript (JSON) e sincronizados continuamente para cada usuário associado.

O Firebase foi utilizado pois possuí uma solução completa e integrada para o armazenamento de dados, possuindo facilidade na integração com o desenvolvimento de aplicativos móveis pois fornece uma API simples e intuitiva que permite a comunicação direta entre o aplicativo e o banco de dados, tornando o processo de armazenamento e recuperação de dados rápido e eficiente.

2.4. Amazon Lambda

O Amazon Lambda é um serviço de computação sem servidor fornecido pela Amazon Web Services (AWS). Ele permite que os desenvolvedores executem código de forma escalável e eficiente, sem a necessidade de provisionar e gerenciar servidores. O Lambda é especialmente útil para aplicações que exigem respostas rápidas a eventos, como a execução de funções em resposta a gatilhos específicos. O sistema é baseado em funções, e uma função é um código escrito pelo programador. Um evento aciona a função e o trecho de código será executado toda vez que o gatilho for acionado [Altaleb et al. 2022].

Esse serviço foi escolhido como parte do projeto para agendar e executar o programa responsável pela coleta de dados dos restaurantes universitários. Sua utilização se baseia na necessidade de manter os menus atualizados diariamente de forma automatizada.

3. Proposta de Desenvolvimento

Neste capítulo é apresentado a proposta de desenvolvimento do aplicativo móvel, que tem como objetivo atender às necessidades identificadas com base na vivência do autor como aluno da UFSC. A proposta visa proporcionar aos usuários uma ferramenta eficiente e acessível para acesso a informações acadêmicas e facilitar o gerenciamento de suas atividades relacionadas à universidade. Ao compreender a proposta de desenvolvimento e a arquitetura do sistema, será possível visualizar a abordagem adotada para a implementação do aplicativo móvel.

3.1. Requisitos Funcionais

O levantamento dos requisitos mencionados a seguir foi feito com base na vivência e nas necessidades identificadas pelo autor desse trabalho como aluno da UFSC, além de uma análise dos recursos disponíveis pelos webservices do Moodle e do CAGR.

- RF1: O aplicativo móvel deve exibir as três próximas disciplinas matriculadas do usuário na página inicial, informando o nome da disciplina, o local e o horário;
- RF2: O aplicativo móvel deve exibir a grade de horários completa separado por dia da semana, exibindo o nome da disciplina e o horário;
- RF3: O aplicativo móvel deve exibir uma lista com todas as disciplinas matriculadas, informando o nome, código e o nome dos professos que ministram as aulas;
- RF4: O aplicativo móvel deve exibir o cardápio do Restaurante Universitário, separado por dia da semana, do campus selecionado pelo usuário;
- RF5: O aplicativo móvel deve exibir um botão para o usuário acessar o PDF do cardápio do Restaurante Universitário do campus selecionado;
- RF6: O aplicativo móvel deve exibir os eventos cadastrados do calendário do usuário no Moodle, exibindo a informação do título, nome da disciplina e o horário do evento;
- RF7: O aplicativo móvel deve exibir todas as notas cadastradas nas disciplina no Moodle;
- RF8: O aplicativo móvel deve exibir as manchetes mais recentes do portal de notícias da UFSC;
- RF9: O aplicativo móvel deve permitir o download do histórico síntese do aluno;
- RF10: O aplicativo móvel deve permitir o download do atestado de matrícula do aluno;
- RF11: O aplicativo móvel deve permitir o download do currículo do curso do aluno;
- RF12: O aplicativo móvel deve notificar o usuário sobre o horário das aulas, enviando uma mensagem do aplicativo com o nome da disciplina, o local e o horário;
- RF13: O aplicativo móvel deve notificar o usuário sobre o horário dos eventos do calendário, enviando uma mensagem no aplicativo com o título do evento, e horário de entrega.

3.2. Arquitetura

O diagrama de arquitetura representa a estrutura e organização dos componentes do sistema como um todo, fornecendo uma visão geral de como se relacionam e trabalham juntos. Sendo assim, foram identificadas os principais componentes, exibidos na Figura 1 e listados a seguir.

Aplicativo Flutter: Este é o componente responsável pela interface com o usuário. Ele utiliza o padrão de arquitetura Model-View-Controller (MVC) para separar as responsabilidades e melhorar a organização do código. O Model representa os dados das disciplinas, cardápios dos restaurantes e outras informações relevantes, que são armazenados no Firebase. O View consiste nos widgets que compõem a interface do usuário, permitindo que os usuários interajam com o aplicativo. O Controller contém as funções responsáveis pela lógica de negócio, como autenticação, acesso aos dados do CAGR e Moodle, e atualização da interface com os dados obtidos.

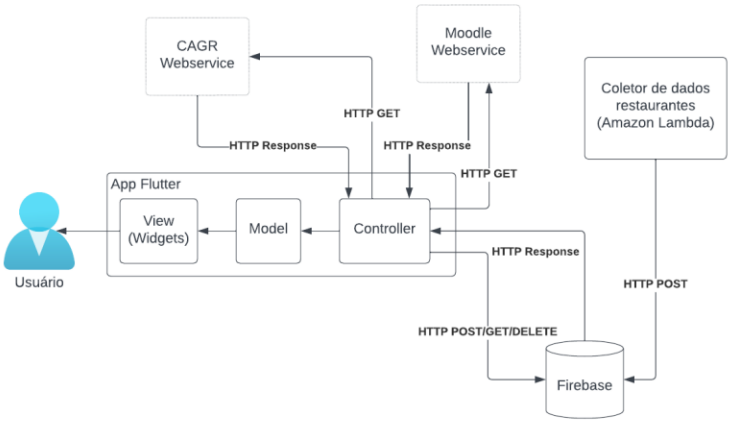


Figura 1. Diagrama de arquitetura.

Firebase: É o componente do banco de dados na nuvem. Ele armazena as informações das contas de usuário, bem como os dados das disciplinas, cardápios dos restaurantes e outras informações relevantes para o aplicativo.

Amazon Lambda: Utilizado para hospedar e executar um script em Python responsável por coletar os dados dos menus dos restaurantes diariamente. Os dados coletados são salvos no Firebase para que possam ser acessados pelos usuários posteriormente.

Webservice do CAGR: O aplicativo precisa acessar o webservice do CAGR para obter os dados das disciplinas.

Webservice do Moodle: O aplicativo acessa o Webservice do Moodle para obter as informações das notas e os eventos do calendário do aluno.

4. Avaliação do Aplicativo

No presente capítulo é abordada a avaliação do aplicativo com foco na análise da sua usabilidade, utilizado o System Usability Scale (SUS), um questionário amplamente reconhecido e validado para medir a usabilidade de produtos. Os resultados obtidos foram analisados, fornecendo informações importantes sobre a percepção dos usuários e possibilitando a identificação de áreas passíveis de melhorias.

4.1. System Usability Scale

O System Usability Scale, conhecido como SUS, é um dos sistemas mais populares e fáceis de usar um questionário para medir o nível de usabilidade de qualquer produto [Kaya et al. 2019].

John Brooke desenvolveu o SUS em 1996. Ele contém dez questões básicas e simples sobre a usabilidade de um sistema, sendo uma ferramenta útil para entender os problemas que os usuários enfrentam enquanto usam o sistema, fornecendo uma pontuação única para usabilidade, projetada como uma medida unidimensional [Brooke et al. 1996].

Dez declarações são apresentadas ao usuário, que se relacionam a vários aspectos da usabilidade em uma escala Likert de 5 pontos, variando de forte desacordo (1) a forte acordo (5). A pontuação final para o SUS varia de 0 a 100, com pontuações mais altas indicando maior percepção de usabilidade [Mol et al. 2020].

Os dez itens da escala são os seguintes:

1. Acho que gostaria de utilizar este sistema com frequência.
2. Considerei o sistema mais complexo do que necessário.
3. Achei o sistema fácil de utilizar.
4. Acho que necessitaria de ajuda de um técnico para conseguir utilizar este sistema.
5. Considerei que as várias funcionalidades deste sistema estavam bem integradas.
6. Achei que este sistema tinha muitas inconsistências.
7. Suponho que a maioria das pessoas aprenderia a utilizar rapidamente este sistema.
8. Considerei o sistema muito complicado de utilizar.
9. O sistema pareceu confiável ao utilizar.
10. Precisei aprender várias coisas novas antes de conseguir usar o sistema.

Esta escala de dez itens tem sido referida como um padrão da indústria e é considerada um dos questionários mais amplamente usados na prática de usabilidade [Lewis and Sauro 2017].

Como se pode verificar no SUS, as perguntas ímpares têm significados positivos e as perguntas pares têm significados negativos. A pontuação das perguntas positivas é feita da seguinte forma: A pontuação do usuário é reduzida em um ponto. Por exemplo, se a pontuação do usuário for 4 para a pergunta 5, a pontuação do resultado será 3. A pontuação das perguntas negativas é feita da seguinte forma: A pontuação do usuário é subtraída de 5. Por exemplo, se a pontuação do usuário for 3 para a questão 4, então a nova pontuação será 2. Após todas as pontuações serem determinadas, a soma das pontuações é multiplicada por 2,5 para fazer o intervalo entre 0 e 100.

O SUS possui excelentes propriedades psicométricas, pois de acordo com pesquisas tem mostrado consistentemente que possui confiabilidade igual ou superior a 90% [Lewis et al. 2015]. De acordo com esses dados normativos, uma pontuação de 68 pode ser interpretado como um resultado aceitável.

4.2. Resultados

Foi utilizada uma estratégia para obter um número significativo de usuários para avaliação do aplicativo. Essa estratégia envolveu o envio de convites para usuários do Moodle que haviam acessado a plataforma nas últimas 24 horas. O Moodle permite acessar os dados dos usuários matriculados e ordená-los por última visualização. Além disso, qualquer usuário pode enviar mensagens para outros alunos, o que possibilitou o contato direto.

O convite foi enviado para 500 usuários de diferentes cursos e campi da instituição, garantindo uma amostra diversificada e representativa da comunidade acadêmica. Essa abordagem permitiu obter um feedback abrangente sobre o aplicativo, envolvendo usuários com diferentes perfis e necessidades. Desses, 97 usuários que instalaram o aplicativo, um total de 19 pessoas responderam ao formulário de pesquisa, obtendo uma pontuação média para cada afirmativa do questionário conforma a Tabela 1.

4.3. Análise

Dos resultados obtidos na pesquisa de satisfação, foi observado que a maioria dos usuários demonstrou interesse em utilizar o aplicativo com frequência, reconhecendo seu valor e utilidade. Além disso, a maioria dos usuários considerou o aplicativo fácil de usar e

Tabela 1. Resultados da pesquisa de satisfação

Pergunta	Média
Acho que gostaria de utilizar este aplicativo com frequência	4,21
Considerei o aplicativo mais complexo do que necessário	1,73
Achei o aplicativo fácil de usar	4,52
Acho que necessitaria de ajuda do suporte para conseguir utilizar este aplicativo	1,57
Considerei que as várias funcionalidades deste aplicativo estavam bem integradas	3,95
Achei que este aplicativo tinha muitas inconsistências	1,78
Suponho que a maioria das pessoas aprenderiam a utilizar rapidamente este aplicativo	4,58
Considerei o aplicativo muito complicado de utilizar	1,31
O aplicativo pareceu confiável	4,21
Precisei aprender várias coisas novas antes de conseguir usar o aplicativo	1,15

sentiu-se confiante em explorar suas funcionalidades sem a necessidade de suporte adicional.

Os resultados também indicaram uma percepção positiva em relação à integração das diversas funcionalidades do aplicativo, sugerindo uma experiência coesa e consistente. No entanto, alguns usuários mencionaram a presença de inconsistências, indicando áreas que podem ser aprimoradas em termos de coerência e padronização. Além disso, houve menção de que o aplicativo pode ser percebido como complicado por alguns usuários, o que pode sugerir a necessidade de simplificação ou orientações mais claras.

No geral, os resultados da pesquisa demonstraram uma recepção positiva do aplicativo pelos usuários, com a maioria reconhecendo sua utilidade e facilidade de uso.

Após aplicar o cálculo para obtenção da nota pelo SUS às respostas dos usuários, a pontuação total obtida foi de 84.34, que segundo as diretrizes, indica um nível excelente de satisfação e usabilidade do aplicativo. De acordo com estudos anteriores, uma pontuação acima de 74 é considerada excelente.

5. Conclusão

Através da utilização das tecnologias modernas e eficientes, como Flutter, Firebase e Amazon Lambda, o desenvolvimento do aplicativo foi bem-sucedido, proporcionando uma experiência satisfatória aos usuários. O Flutter permitiu criar uma interface de usuário atraente e responsiva, enquanto o Firebase garantiu o armazenamento e acesso rápido aos dados do aplicativo, incluindo a autenticação de usuários. A integração com o Amazon Lambda facilitou a coleta automatizada dos cardápios dos restaurantes da UFSC.

As funcionalidades do aplicativo, como a exibição de disciplinas matriculadas, grade de horários, cardápio do Restaurante Universitário, eventos do calendário, notas e notícias da UFSC, proporcionaram acesso rápido a informações relevantes para os usuários. Além disso, a pesquisa de satisfação com os usuários, utilizando o questionário SUS, revelou uma alta pontuação de usabilidade e satisfação geral.

A combinação dessas tecnologias, juntamente com os resultados positivos da pesquisa de satisfação, demonstra o sucesso do desenvolvimento e aplicação do aplicativo proposto. O uso do Flutter, Firebase e Amazon Lambda resultou em um aplicativo prático e eficiente, melhorando a experiência acadêmica dos usuários.

Referências

Adetunji, O., Ajaegbu, C., Otuneme, N., Omotosho, O. J., et al. (2020). Dawning of progressive web applications (pwa): Edging out the pitfalls of traditional mobile development. *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)*, 68(1):85–99.

Altaleb, B., Khalaf, A., et al. (2022). Scalability and economy of amazon lambda, eks, and ecs.

Brooke, J. et al. (1996). Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7.

Chatterjee, N., Chakraborty, S., Decosta, A., and Nath, A. (2018). Real-time communication application based on android using google firebase. *Int. J. Adv. Res. Comput. Sci. Manag. Stud*, 6(4).

Dahl, O. (2019). Exploring end user's perception of flutter mobile apps.

Kaya, A., Ozturk, R., and Gumussoy, C. A. (2019). Usability measurement of mobile applications with system usability scale (sus). In *Industrial engineering in the big data era*, pages 389–400. Springer.

Lewis, J. J. R. and Sauro, J. (2017). Revisiting the factor structure of the system usability scale. *Journal of Usability Studies*, 12(4).

Lewis, J. R., Brown, J., and Mayes, D. K. (2015). Psychometric evaluation of the emo and the sus in the context of a large-sample unmoderated usability study. *International Journal of Human-Computer Interaction*, 31(8):545–553.

Mol, M., van Schaik, A., Dozeman, E., Ruwaard, J., Vis, C., Ebert, D. D., Etzelmueller, A., Mathiasen, K., Moles, B., Mora, T., et al. (2020). Dimensionality of the system usability scale among professionals using internet-based interventions for depression: a confirmatory factor analysis. *BMC psychiatry*, 20:1–10.

Olsson, M. (2020). A comparison of performance and looks between flutter and native applications: When to prefer flutter over native in mobile application development.