Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Departamento de Computação e Sistemas

Tiza App - Um aplicativo mobile para organização de horários individuais

Luis Guilherme Godim da Fonseca

TRABALHO DE CONCLUSÃO DE CURSO

ORIENTAÇÃO:
George Henrique Godim da Fonseca

Setembro, 2025
João Monlevade–MG

Luis Guilherme Godim da Fonseca

Tiza App - Um aplicativo mobile para organização de horários individuais

Orientador: George Henrique Godim da Fonseca

Monografia apresentada ao curso de Sistemas de Informação do Instituto de Ciências Exatas e Aplicadas, da Universidade Federal de Ouro Preto, como requisito parcial para aprovação na Disciplina "Trabalho de Conclusão de Curso II".

Universidade Federal de Ouro Preto

João Monlevade

Setembro de 2025

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE COMPUTACAO E SISTEMAS

FOLHA DE APROVAÇÃO

Luís Guilherme Godim da Fonseca

Tiza App - Um aplicativo mobile para organização de horários individuais

Monografia apresentada ao Curso de Sistemas de Informação da Universidade Federal
de Ouro Preto como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação

Aprovada em 04 de setembro de 2025

Membros da banca

Dr. George Henrique Godim da Fonseca - Orientador(a) - Universidade Federal de Ouro Preto
Dr. Carlos Henrique Gomes Ferreira - Universidade Federal de Ouro Preto
Dra. Helen de Cássia Sousa da Costa Lima - Universidade Federal de Ouro Preto

Dr. George Henrique Godim da Fonseca, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 11/09/2025

Documento assinado eletronicamente por **George Henrique Godim da Fonseca**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 15/04/2026, às 14:43, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **1089713** e o código CRC **546B1915**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: (31)3808-0819 - www.ufop.br

Agradecimentos

Ao meu irmão e orientador, Dr. George Henrique Godim da Fonseca, deixo meu agradecimento mais sincero. Agradeço os bons conselhos, muitas vezes acompanhados de uma xícara de café, o apoio constante e a paciência nas revisões, que deram direção a este trabalho e sustentaram cada etapa do caminho.

Aos meus pais, agradeço por acreditarem em mim, pelo incentivo contínuo e por me darem o alicerce que tornou possível chegar até aqui.

À Fernanda, minha companheira, por sempre ter acreditado em mim, pela companhia em todos os momentos — bons e ruins — e pelo seu apoio incondicional. Deixo aqui todo o meu amor e gratidão.

Aos amigos, pela presença leal e pelas boas memórias, pela escuta e pelas palavras certas quando foi preciso, meu muito obrigado.

Registro ainda meu reconhecimento ao professor Dr. Carlos Henrique Gomes, cujas orientações ao longo da graduação contribuíram muito para minha formação. Estendo este agradecimento, inclusive, aos professores que, com dedicação e seriedade, compartilharam conhecimento e abriram caminhos de aprendizado.

Por fim, a todas as pessoas que, de algum modo, ajudaram compartilhando seu tempo, críticas, incentivos ou simples disponibilidade, deixo minha gratidão. Cada gesto somou para que este trabalho se tornasse possível.

"Se vi mais longe, foi por estar sobre ombros de gigantes."

— Isaac Newton.

Adaptação de um trecho de uma carta de Newton para Robert Hooke, 5 de fevereiro de 1676, baseado numa metáfora atribuída a Bernardo de Chartres.

Resumo

Este trabalho apresenta o desenvolvimento de um aplicativo móvel multiplataforma, em *React Native* com *Expo*, voltado à organização do horário acadêmico no nível do estudante. O objetivo é permitir que o usuário visualize as ofertas institucionais publicadas no *Tiza*, componha um único horário pessoal a partir dessas ofertas e gerencie os respectivos eventos em uma interface *calendar-first* com visões de semana, dia, três dias e agenda. Metodologicamente, a solução integra-se ao *Tiza* como fonte oficial, consumindo os horários publicados e persistindo dados no *Firebase* (*Firestore*) e em cache local. O aplicativo implementa autenticação de usuário compatível com o ecossistema *Tiza*, ingestão e normalização de dados institucionais, montagem do cronograma por seleção de aulas/turmas, verificação de conflitos por sobreposição temporal e visualizações adequadas ao acompanhamento cotidiano; em modo offline, permite leitura a partir do cache e fila remoções para sincronização posterior. O resultado é um aplicativo funcional que consolida o fluxo de "oferta institucional" para "uso individual" em dispositivos móveis, complementando o ecossistema ao aproximar a visualização do modelo de calendário já presente no dia a dia do aluno.

Palavras-chave: programação de horários. Tiza. aplicativo móvel. React Native. Firebase. calendário acadêmico.

Abstract

This work presents the development of a cross-platform mobile application, built with React Native and Expo, aimed at helping students organize their personal class schedules. The goal is to let users browse institutional timetables published in Tiza, assemble a single personal schedule from those offerings, and manage events through a calendar-first interface featuring week, day, three-day, and agenda views. Methodologically, the applicative integrates with Tiza as the authoritative source, consuming published schedules and persisting data in Firebase (Firestore) and local cache. It implements user authentication aligned with the Tiza ecosystem, data ingestion and normalization, timetable building by selecting classes, time-overlap conflict checking, and views tailored to daily use; in offline mode, it reads from cache and queues deletions for later sync. The result is a functional app that bridges "institutional offering" to "individual use" on mobile, complementing the ecosystem with familiar calendar-based interactions.

Keywords: timetabling. Tiza. mobile app. React Native. Firebase. academic calendar.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

RN React Native

IA inteligência Artificial

SDK Software Development Kit

VS Code Visual Studio Code

IDE Integrated Development Environment

PDF Portable Document Format

XLSX Excel Spreadsheet XML

XML Extensible Markup Language

CSV Comma-Separated Values

HTML HyperText Markup Language

JSON JavaScript Object Notation

POC Proof of Concept

UI User Interface

UX User Experience

Sumário

1 Introdução

Planejar horários acadêmicos é uma tarefa central para gestores, diretores e coordenadores de instituições de ensino. A cada período letivo, é preciso conciliar disponibilidade de docentes, alocação de salas, regras de intervalos, limites de carga horária, turmas e prioridades institucionais, mantendo aderência a normas internas e externas. À medida que o número de disciplinas, turmas e recursos cresce, o esforço de coordenação aumenta e a complexidade combinatória torna o processo sensível a erros e retrabalhos (FONSECA et al., 2017).

O problema de elaboração de horários acadêmicos tem sido amplamente estudado nas áreas de inteligência Artificial (IA) e pesquisa operacional. Há um corpo consistente de métodos — desde formulações exatas (programação inteira e por restrições) até heurísticas e meta-heurísticas — capazes de tratar múltiplas regras e preferências simultaneamente, produzindo cronogramas mais adequados ao contexto institucional (FONSECA et al., 2017). Na prática, há soluções consolidadas que vêm sendo amplamente adotadas por instituições de ensino. Entre elas, destacam-se o *aSc TimeTables*, o *Fet* e o *Tiza*. Neste trabalho, o *Tiza* será a referência central e a base para integração.

Apesar dos avanços na geração automática de horários, o planejamento individual do estudante ainda apresenta desafios significativos, como comparar rapidamente alternativas de montagem, conciliar objetivos concorrentes — evitar sobreposições, equilibrar a carga semanal, compatibilizar obrigações externas e atender a preferências de horários e docentes. Esse processo decisório, de caráter recorrente e multiobjetivo, depende fortemente da qualidade da visualização dos horários e do suporte à análise de cenários. No entanto, embora existam sistemas eficazes para a construção do quadro institucional, poucos oferecem um aplicativo móvel orientado ao estudante, com interface adequada ao acompanhamento e à organização cotidiana do próprio cronograma. Essa lacuna limita o alcance dos horários no nível individual, sobretudo em um contexto em que o acesso via dispositivos móveis é predominante (STATCOUNTER, 2025)(TAYLOR, 2025).

Nesta monografia, o *Tiza* é adotado como sistema de referência. Trata-se de uma plataforma web que automatiza a criação de horários a partir de dados e preferências informados pelos usuários, permitindo atribuir prioridades a restrições e exportar resultados em diferentes formatos (TIZA, 2025). Apesar do suporte ao fluxo institucional, o *Tiza* não oferece, até o momento, um aplicativo móvel voltado ao estudante, o que abre espaço para soluções complementares focadas na montagem e no acompanhamento individual do cronograma.

Posto isso, este trabalho propõe um aplicativo móvel multiplataforma (Android/iOS)

que consome os dados publicados pelo *Tiza* e oferece ao estudante recursos de visualização, montagem e edição do próprio horário em uma interface baseada em calendário. A proposta busca aproximar os resultados institucionais do uso cotidiano, reduzindo o esforço manual e tornando mais direto o processo de organização pelo aluno.

1.1 Objetivos

O objetivo deste trabalho é desenvolver um aplicativo móvel, para Android e iOS, que permita ao estudante montar e acompanhar seus horários de aula a partir das ofertas institucionais publicadas no *Tiza*.

1.1.1 Objetivos específicos

- Permitir a visualização dos horários públicos disponibilizados pelo *Tiza*.
- Exibir informações de disciplinas, como docentes e salas, para apoiar a tomada de decisão.
- Permitir ao aluno criar o próprio horário individualizado a partir das ofertas institucionais.
- Editar o horário individualizado criado através da adição e remoção de eventos.
- Publicar o aplicativo nas lojas oficiais.

1.2 Justificativa

Para o estudante, montar o próprio horário se assemelha a resolver um quebra-cabeças: é preciso evitar sobreposições, distribuir a carga semanal e conciliar compromissos externos. Surgem então questões práticas: quais dificuldades os alunos encontram ao compor seus horários a partir das ofertas institucionais? A tomada de notas e a comparação manual de alternativas são viáveis e eficientes? De que modo essa complexidade afeta a organização da rotina acadêmica e a qualidade das escolhas?

Este trabalho parte dessas indagações e propõe uma resposta focada no nível individual do aluno. Após a geração institucional das ofertas pelo *Tiza*, permanece a necessidade de organizar o cotidiano de forma simples. Apresenta-se, portanto, um aplicativo móvel em React Native, voltado à montagem e à gestão do horário a partir das ofertas institucionais, com visualizações orientadas ao acompanhamento diário. O objetivo é tornar o processo de organização mais direto, reduzir retrabalho e apoiar decisões informadas pelo estudante.

1.3 Organização do trabalho

A estrutura do trabalho é dividida em cinco capítulos. O Capítulo 2 aborda os trabalhos relacionados, fornecendo suas considerações pertinentes ao tema. O Capítulo 3 detalha a metodologia empregada, as tecnologias utilizadas, os requisitos do sistema e uma visão geral do mesmo. No Capítulo 4 são apresentados os resultados, incluindo as telas implementadas. Por fim, o Capítulo 5 engloba as considerações finais do trabalho.

2 Revisão bibliográfica

Dispositivos móveis assumiram papel central nas atividades cotidianas, inclusive no contexto educacional, impulsionados por portabilidade, disponibilidade contínua e maturidade de interface (BUDIU, 2015). Em termos de adoção, a participação global de mercado é de 58,39% para *mobile*, 40,04% para *desktop* e 1,57% para *tablet* (Statcounter, jul./2025) (STATCOUNTER, 2025). Projeções indicam que o número de usuários de smartphones no mundo deve alcançar 7,948 milhões até 2028 (TAYLOR, 2025).

Nesse cenário, aplicações que apoiam a organização do percurso acadêmico têm ganhado destaque, especialmente aquelas que permitem construir e consultar horários a partir das ofertas institucionais, utilizando visualizações já familiares ao estudante. Nesse contexto, esta subseção apresenta sistemas amplamente empregados na elaboração e disponibilização de horários, como *aSc TimeTables*, *FET* e *Tiza*, além de aplicações voltadas especificamente ao uso discente, ressaltando características funcionais e padrões de interface relevantes para este trabalho.

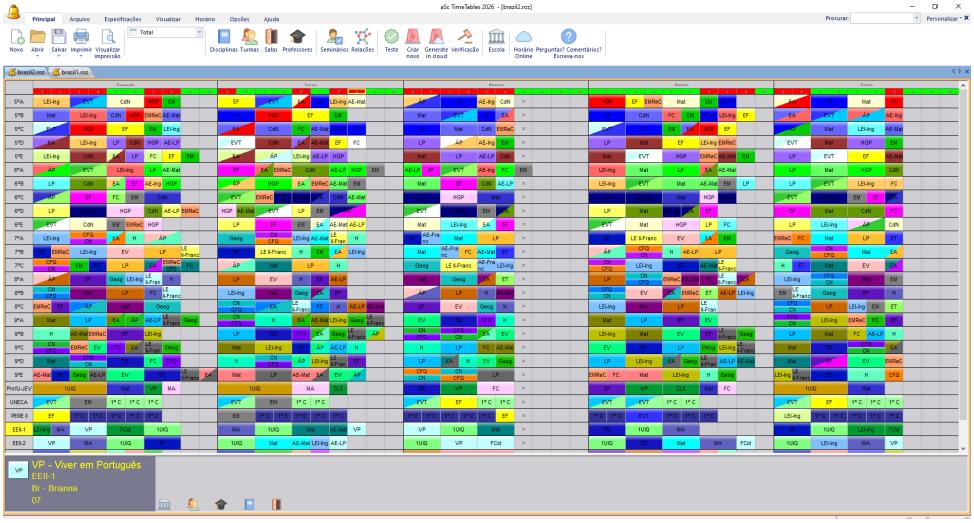
2.1 aSc TimeTables

O *aSc TimeTables* é uma solução consolidada para construção e manutenção de quadros de horários escolares, combinando geração automática com edição manual assistida. A ferramenta oferece verificação rápida de conflitos, assistentes de entrada de dados e ampla personalização visual (cores, fontes, logotipos e layout), além de exportação e publicação em formatos como Excel e Portable Document Format (PDF). Segundo o fornecedor, o sistema reúne cerca de 30 anos de evolução, está presente em 173 países, atende mais de 150.000 escolas e contabiliza mais de 12,5 milhões de horários gerados (ASC, 2025a).

O ecossistema do *aSc TimeTables* inclui módulos e serviços complementares, como o *aSc Substitutions* para gestão colaborativa de substituições, publicação online e acesso móvel aos horários, importação de dados e recursos voltados a cenários com múltiplos prédios e otimização de deslocamentos. A solução também dispõe de edições comerciais com diferentes níveis de suporte e capacidades (por exemplo, opções com consultoria especializada e funcionalidades orientadas a contextos universitários), além de versões online e para computadores interoperáveis (ASC, 2025a).

A Figura 1 apresenta um exemplo de quadro de horários gerado pela aplicação. A interface organiza as informações de forma clara, evidencia conflitos e utiliza cores para diferenciar disciplinas e turmas. O usuário pode rearranjar eventos manualmente para acomodar preferências e ajustes pontuais.

Figura 1 – Aplicação aSc Timetables



Fonte: (ASC, 2025b)

Integrado ao *aSc TimeTables*, o *EduPage* funciona como uma camada em nuvem para gerenciamento e publicação escolar. Ele centraliza a disponibilização dos horários e seu uso no dia a dia pela comunidade acadêmica, oferecendo acesso via web e aplicativo móvel a diferentes perfis como docentes, discentes e responsáveis (ASC, 2025a).

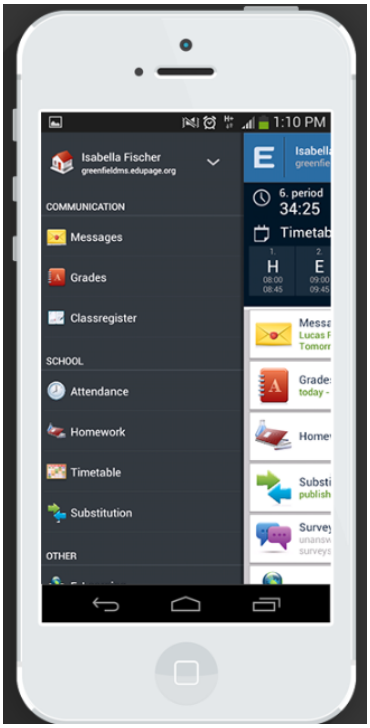
Para os professores, o *EduPage* permite registrar notas, tarefas e frequência durante as aulas, eliminando retrabalho e duplicação manual de dados. As informações são transferidas automaticamente para o sistema escolar. A plataforma dá suporte à preparação e avaliação de apresentações, testes e trabalhos de casa; o docente pode criar provas no próprio aplicativo e atribuí-las aos alunos em aulas interativas ou como atividade domiciliar, com os resultados exibidos automaticamente após a conclusão. O ambiente inclui recursos de *e-learning* e materiais interativos voltados ao acompanhamento pedagógico (ASC, 2025a).

Do ponto de vista de alunos e responsáveis, o *EduPage* reúne em um só lugar o acesso aos horários, aos conteúdos abordados, às tarefas e avaliações, possibilitando responder atividades diretamente pelo smartphone. Os pais podem acompanhar informações como notas, assiduidade, tarefas e testes, além de comunicados e publicações da escola (incluindo registros visuais, como fotos). Na Figura 2 observa-se um exemplo da interface do aplicativo (ASC, 2025a).

2.2 Tiza

O *Tiza* é uma plataforma web baseada em IA para geração automática de horários escolares e universitários a partir de dados e restrições informados pelos usuários. O solucionador considera simultaneamente múltiplos requisitos, permite atribuição de prioridades

Figura 2 – Aplicação EduPage



Fonte: (ASC, 2025a)

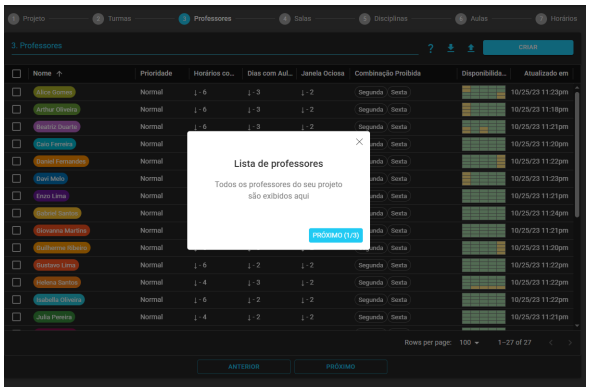
e retorna, em poucos instantes, alternativas compatíveis de quadro. Uma versão preliminar do sistema Proof of Concept (POC) foi desenvovida e apresentada sobre o nome *Optables* em (PAIVA, 2021), tendo evoluído desde então.

A interface organiza o preenchimento em seções bem delineadas — Projeto, Turmas, Salas, Disciplinas, Aulas e Horários — com balões informativos que orientam passo a passo a inserção dos dados como pode ser visto na Figura 3a. O sistema, também, facilita a carga inicial por meio de importação de planilhas Excel Spreadsheet XML (XLSX) e também pela área de transferência, permitindo mapear colunas e revisar registros antes da gravação como pode ser observado na Figura 3b. Essa estrutura favorece a criação de projetos distintos, como semestres ou anos letivos diferentes, e o reuso de cadastros (TIZA, 2025).

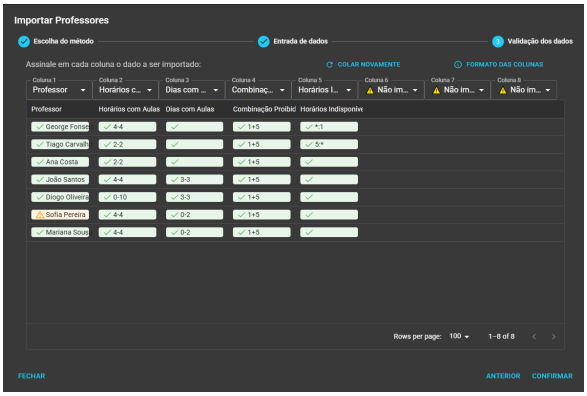
No momento da geração, o *Tiza* exibe uma visão abrangente do quadro de horários acompanhada de um relatório com as restrições não atendidas e seus respectivos pesos. O usuário pode iterar sobre dados e parâmetros, refinar prioridades e gerar novos horários. Para difusão e integração, os resultados podem ser exportados em formatos usuais, como PDF e XLSX. Quando a conta do usuário dispõe do plano Premium, é oferecida ainda a opção de publicar o horário para consumo por aplicativos móveis, conforme ilustrado na Figura 4. Um exemplo de visualização de horários gerados é apresentado na Figura 5.

Do ponto de vista de acesso e capacidades, a plataforma oferece três faixas de plano.

Figura 3 – Interface do Tiza no momento de hidratação dos dados



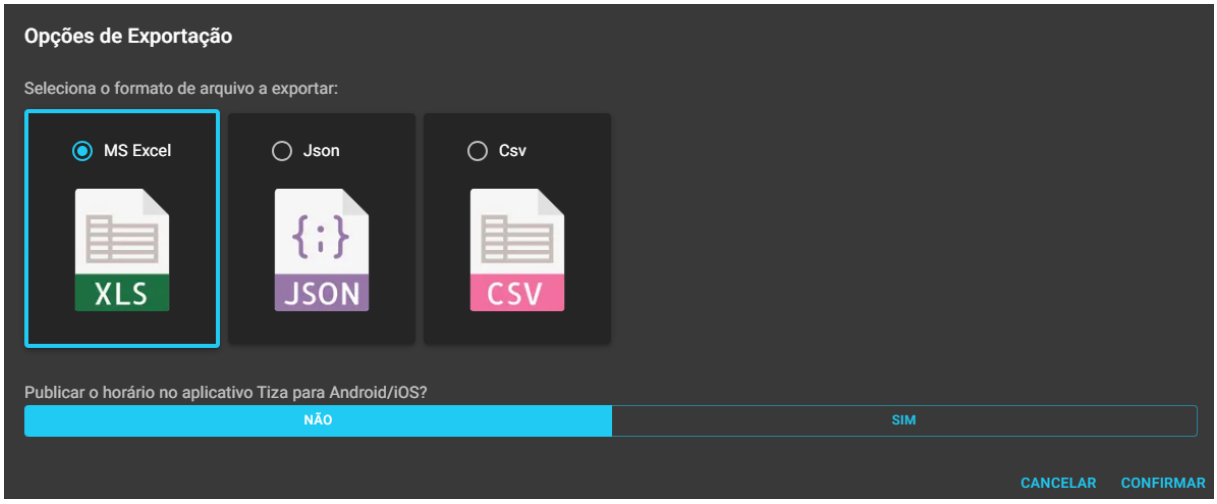
(a) Exemplificação das dicas de uso do Tiza por trecho de preenchimento.



(b) Exibição dos dados após usuário realizar a importação via Planilha XLSX dos dados.

Fonte: (TIZA, 2025)

Figura 4 – Exemplificação da possibilidade de disponibilização dos horários para dispositivos mobile.



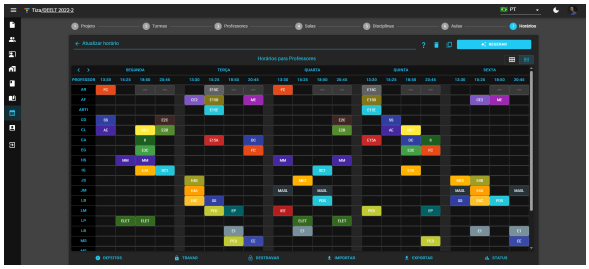
Fonte: (TIZA, 2025)

O plano Gratuito contempla o uso essencial com número reduzido de projetos, limite de aulas por projeto e tempo de execução curto. O plano Básico amplia esses limites, habilita formatos de exportação como XLSX e dá acesso a restrições avançadas e priorização. O plano Premium remove os principais limites operacionais, como quantidade de projetos e de aulas, estende o tempo de execução e habilita exportações adicionais em XLSX, JavaScript Object Notation (JSON) e Comma-Separated Values (CSV), além de permitir colaboração e a funcionalidade de publicar horários no aplicativo móvel. Este último ponto é determinante para o presente trabalho: o aplicativo proposto somente consegue consumir dados do *Tiza* quando a instituição publica seus horários por meio dessa funcionalidade, disponível exclusivamente no plano Premium.

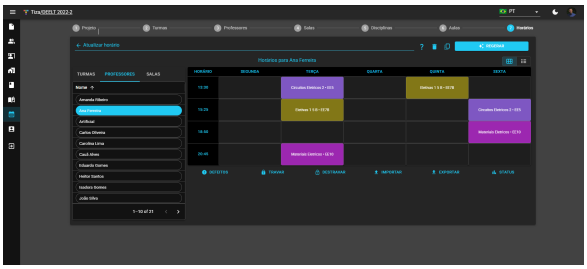
No contexto desta monografia, o *Tiza* é a fonte institucional dos dados: os horários

publicados pela plataforma alimentam o aplicativo proposto, que organiza o uso no nível do estudante, concentrando-se na montagem e no acompanhamento do horário pessoal.

Figura 5 – Interface dos horários gerados pelo Tiza segmentadas em dois tipos de visualização



(a) Tipo de visualização por horários compactos.



(b) Tipo de visualização por horários individuais segmentado por Professores.

Fonte: (TIZA, 2025)

2.3 FET

O *FET* (Free Timetabling Software) é um software livre, mantido por Liviu Lalescu, voltado à geração automática de horários em escolas, faculdades e universidades. O projeto é distribuído sob a licença GNU Affero General Public License, versão 3 ou posterior, o que permite uso, estudo e modificação do código (LALESCU, 2025).

A ferramenta é multiplataforma, com suporte a Windows, macOS e Linux, desenvolvida sobre o framework Qt. Oferece modos de geração automática, além de operações semiautomáticas e manuais. O modelo de dados é flexível: entrada em Extensible Markup Language (XML), importação e exportação em CSV e geração de saídas em HyperText Markup Language (HTML), XML e CSV, o que facilita o intercâmbio com outras aplicações e planilhas (LALESCU, 2025).

O *FET* adota um conjunto amplo de restrições de tempo e espaço com pesos configuráveis, permitindo modelar desde disponibilidade de professores e alunos até limites diários e semanais, intervalos, sequências e sobreposições máximas. Há suporte a estruturas com conjuntos e subgrupos de alunos, múltiplos professores ou alunos por atividade e cenários com várias salas e edifícios, com preferências e limites de trocas por dia ou por semana (LALESCU, 2025).

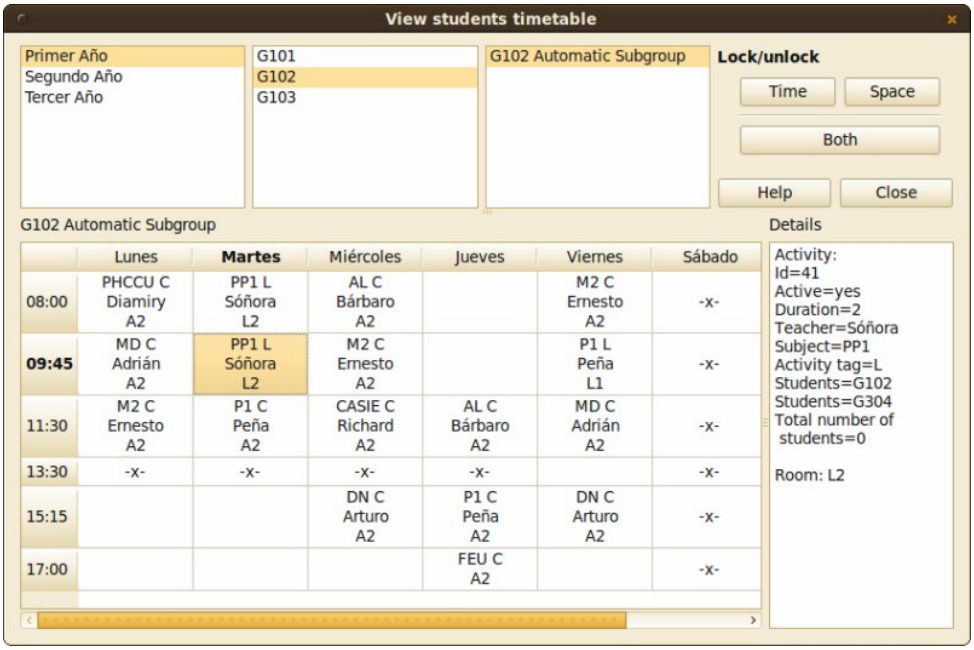
Quanto ao desempenho, a solução típica de cronogramas pode levar de cinco a vinte minutos em casos complexos, podendo variar conforme o tamanho do problema e o conjunto de restrições adotado (LALESCU, 2025).

Importante salientar que, apesar desses recursos, o *FET* não dispõe de aplicativo móvel integrado nem de canal nativo de publicação para dispositivos móveis. A difusão dos

horários para a comunidade acadêmica ocorre, em geral, por meio dos arquivos exportados ou de páginas web externas.

Na Figura 6 é possível visualizar um modelo de horários completo disponibilizado pelo *FET*.

Figura 6 – Aplicação FET



Fonte: (LALESCU, 2025)

2.4 Docendo

O *Docendo* é um software para programação escolar, planejamento de recursos e gestão de substituições, concebido em Aalborg, Dinamarca, e utilizado por escolas em diferentes países. A aplicação combina montagem de horários por arrastar e soltar com contagem automática de horas, o que reduz o uso de planilhas e facilita ajustes durante o ano letivo. Segundo o fornecedor, mais de 10.000 administradores e professores empregam a ferramenta no dia a dia (DOCENDO, 2025).

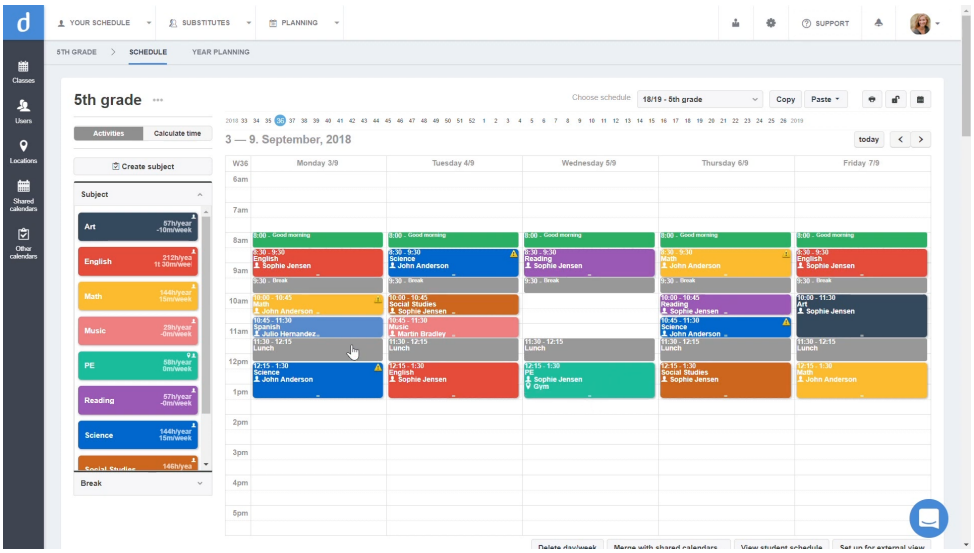
A proposta é manter o quadro flexível: alterações na grade semanal podem ser feitas rapidamente e replicadas para outras semanas quando necessário. O sistema destaca ocupações de professores e salas no momento do arraste, evitando dupla alocação, e mantém uma contagem em tempo real das horas e minutos por turma ou docente. Há uma visão contínua do calendário de cada professor e substituto, com atualização em tempo real e integração com calendários externos (iCal, Outlook, Google Calendar e Apple Calendar). Entre os recursos, estão planejamento do ano, visão geral por equipe, atribuição de horas, agendamento e reserva de salas, além de notas e planos semanais (DOCENDO, 2025).

A gestão de substituições contempla registro de ausências, administração de bancos de substitutos e comunicação por telas informativas, com notas específicas para docentes e substitutos. Para estudantes e responsáveis, o *Docendo* oferece publicação de horários e acesso móvel, favorecendo acompanhamento e comunicação da rotina escolar (DOCENDO, 2025).

Na Figura 7 observa-se um exemplo de grade gerada pelo Docendo. As disciplinas são diferenciadas por cores e os blocos de aula podem ser rearranjados por arrastar e soltar, permitindo ajustes de posição e, quando necessário, de duração. A composição evidencia que o quadro não é rígido, admitindo alterações manuais conforme a demanda da escola.

No contexto desta monografia, o *Docendo* é apresentado como exemplo de solução institucional que combina montagem dinâmica de horários, contagem automática e colaboração entre equipe escolar, compondo o quadro comparativo de ferramentas da área.

Figura 7 – Aplicativo Docendo



Fonte: (DOCENDO, 2025)

2.5 Smart Timetable

O *Smart Timetable* é um aplicativo voltado ao acompanhamento do calendário acadêmico pelo aluno, com foco em organizar aulas e tarefas ao longo da semana. A proposta é simplificar o dia a dia: o usuário monta um ou mais cronogramas pessoais e consulta rapidamente o que tem pela frente, sem depender de planilhas ou anotações dispersas (TIMETABLE, 2025).

Entre os recursos, destacam-se notificações para lembrar o início das aulas e prazos de tarefas, além de um widget com cronômetro de atividades. O planejador de horários é flexível: permite múltiplas programações e ciclos de rotação (por exemplo, 1, 2, 3 ou 4

semanas), visualizações semanal e quinzenal e um calendário de turnos quando necessário. O aplicativo também oferece um gerenciador de tarefas com anexos (fotos, vídeos, áudios e documentos) e um planejador específico para trabalhos de casa, facilitando centralizar materiais e orientações em um só lugar (TIMETABLE, 2025).

Há ainda opções para compartilhar o quadro de horários, o que pode ser útil para comunicação com colegas ou responsáveis. No contexto desta monografia, o *Smart Timetable* é tomado como exemplo de solução orientada ao uso individual do estudante, com ênfase em visualizações de calendário, lembretes e organização cotidiana, servindo como referência de interface e fluxo para comparação com outras ferramentas analisadas (TIMETABLE, 2025).

Nesse trabalho, o *Smart Timetable* é tomado como referência de design pelas interfaces claras e pelo uso consistente de calendário e cores, inspirando decisões de UI/UX do aplicativo proposto. Na Figura 8, apresenta-se um modelo de horários completo disponibilizado pelo próprio *Smart Timetable*.

Figura 8 – Aplicativo Smart Timetable

Fonte: (TIMETABLE, 2025)

2.6 Optables View

O *Optables View* foi um trabalho desenvolvido por (DIAS, 2023), que apresenta um aplicativo móvel voltado à criação e *simulação* de horários individuais para estudantes, motivado pelas dificuldades recorrentes na composição e no gerenciamento de quadros de

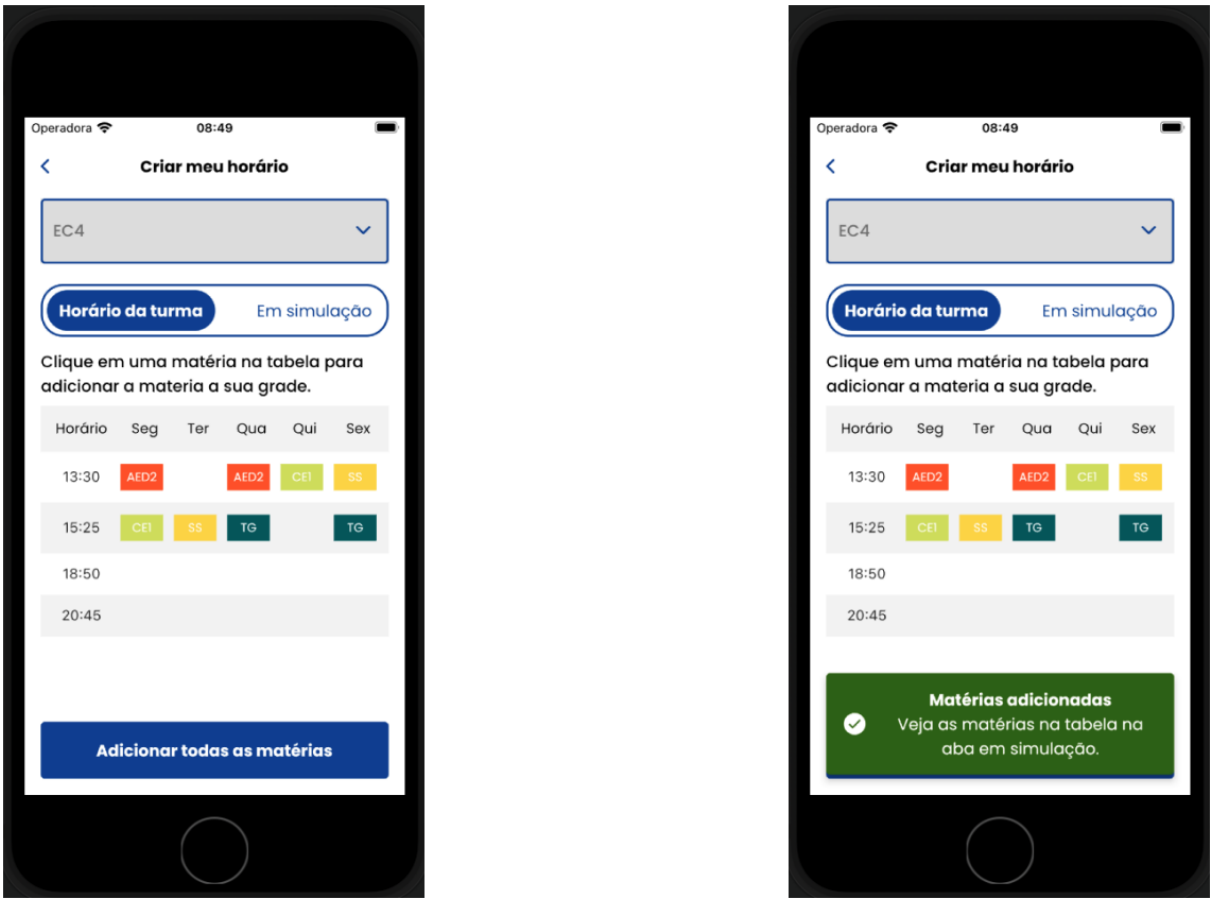
aula. A solução foi desenvolvida para Android e iOS, obtendo os dados institucionais a partir do sistema Optables, posteriormente reestruturado como *Tiza* e armazenando-os no Firebase, contemplando, além da organização do horário, o registro de notas e faltas pelo próprio aluno.

Na arquitetura proposta, a aplicação web Optables gera e exporta os horários; esses dados são publicados e consumidos pelo aplicativo móvel e armazenados em um banco de dados NoSQL, o Firebase. O aplicativo cliente, implementado em Flutter, acessa esses conjuntos para permitir simulações, emitir alertas de conflito e gerenciar as disciplinas cadastradas pelo usuário.

Entre os requisitos funcionais destacados no estudo estão: visualizar os horários fornecidos pela instituição; criar horários personalizados; editar disciplinas, com inclusão e remoção, e excluir horários; visualizar e editar informações de disciplinas, incluindo notas e faltas; alternar entre diferentes horários publicados; e autenticar por meio de conta Google.

Em relação ao presente trabalho, destacam-se diferenças fundamentais. Do ponto de vista tecnológico, a solução aqui proposta adota um conjunto tecnológico baseado em React Native, TypeScript e Expo, enquanto o estudo de (DIAS, 2023) emprega Flutter. Quanto ao escopo de uso, este projeto concentra-se na montagem e na gestão do horário a partir das ofertas publicadas, sem simulação generalizada nem acompanhamento de notas e faltas, ao passo que o trabalho de (DIAS, 2023) inclui ambos. Por fim, a experiência de acompanhamento diário aqui proposta organiza-se por uma abordagem centrada em calendário, oferecendo, além da visão semanal para construção, as visões de dia, três dias e agenda para consulta cotidiana, enquanto o aplicativo de (DIAS, 2023) privilegia operações de simulação e gerenciamento acadêmico com um desenho distinto de interface e navegação. Na Figura 9 podemos ver um conjunto de exemplos da aplicação criada.

Figura 9 – Aplicativo Optables View



Fonte: (DIAS, 2023)

3 Metodologia

Neste capítulo são apresentadas as abordagens adotadas para o desenvolvimento do aplicativo, incluindo a arquitetura do sistema, os requisitos atendidos, os casos de uso, o banco de dados utilizado e a prototipação de alta fidelidade. O objetivo é descrever, de forma objetiva, as decisões que sustentam o cumprimento dos objetivos definidos.

3.1 Arquitetura do sistema

O aplicativo proposto é um cliente móvel multiplataforma (Android/iOS) desenvolvido em React Native (RN) com *Expo*, cujo propósito é permitir ao estudante montar e acompanhar seu próprio horário a partir das ofertas institucionais geradas no *Tiza*. Nesse ecossistema, o *Tiza* é responsável pela geração dos quadros sob restrições e pela publicação dos horários; o aplicativo consome esses dados para organizar o uso no nível individual do aluno. Esta versão não contempla simulação de cenários: o foco recai na montagem do horário e na visualização em formato de calendário para o acompanhamento cotidiano.

A arquitetura lógica organiza-se em três camadas complementares, conforme ilustrado na Figura 10.

- Geração institucional (*Tiza*): criação de horários sob múltiplas restrições e disponibilização dos horários para consumo externo;
- Camada de dados (Firebase): armazenamento dos horários institucionais e dos artefatos operacionais (recursos, aulas e hirários por instituição/arquivo), além do espaço do usuário para persistir o horário montado;
- Cliente móvel (RN/Expo): autentica o usuário, acessa aos horários publicados, permite a seleção de turmas e registra o horário escolhido, oferecendo visões de calendário para consulta e gestão.

Figura 10 – Visão geral da arquitetura de sistema

3.2 Tecnologias utilizadas

Nesta parte são apresentadas as ferramentas e recursos utilizados ao longo das etapas de planejamento e desenvolvimento deste projeto.

3.2.1 React Native

React Native (RN) é uma biblioteca de código aberto, mantido pela Meta, para criar aplicativos móveis nativos em Android e iOS usando JavaScript ou TypeScript. A interface segue o modelo do React e é renderizada por componentes nativos de cada plataforma, o que favorece desempenho e integração com as Interfaces (API)s do dispositivo. O ciclo de desenvolvimento é ágil graças ao recurso de atualização rápida, que permite ver mudanças quase em tempo real (META, 2025).

Entre os benefícios estão o reaproveitamento de grande parte do código entre Android e iOS, a facilidade de acessar funcionalidades do aparelho por meio de módulos nativos e a disponibilidade de um ecossistema amplo de bibliotecas do React. Neste projeto, o uso do Expo simplifica a configuração, a execução em dispositivos e emuladores e o acesso a utilitários prontos (como mídia e notificações), encurtando o caminho entre a implementação e os testes (EXPO, 2025b).

Na prática, a combinação de React Native e Expo atende aos requisitos deste trabalho: entrega a interface baseada em calendário com boa responsividade, integra-se

ao Firebase para autenticação e persistência, e permite evoluir o aplicativo mantendo um único código multiplataforma.

3.2.2 Firebase

Firebase, plataforma do Google para desenvolvimento de aplicativos móveis e web, reúne serviços integrados que cobrem autenticação, banco de dados, hospedagem e mensageria, entre outros. Na prática, funciona como um back-end como serviço, reduzindo a necessidade de infraestrutura própria e acelerando do protótipo à produção (FIREBASE, 2025a).

Neste projeto, dois componentes são centrais. O primeiro é o Firebase Authentication, que viabiliza o acesso por e-mail/senha e conta Google, simplificando o controle de sessão no cliente móvel e a associação de dados ao usuário autenticado. O segundo é o Cloud Firestore, banco NoSQL gerenciado que armazena tanto as soluções de horários publicadas pelo Tiza quanto o horário montado pelo estudante, permitindo leituras e gravações rápidas a partir do aplicativo. Regras de segurança do Firestore restringem a leitura e a escrita do espaço do usuário ao próprio titular, enquanto coleções públicas permanecem acessíveis para consulta (FIREBASE, 2025b).

A escolha pelo Firebase atende a requisitos deste trabalho: integra-se bem ao ecossistema RN, oferece *Software Development Kit (SDK)s* estáveis para autenticação e persistência, opera em tempo real quando necessário e reduz o esforço operacional, mantendo o foco no desenvolvimento das funcionalidades de montagem e visualização do horário.

3.2.3 Expo e Expo Go

Expo é um conjunto de ferramentas e bibliotecas que simplifica o desenvolvimento de aplicativos em RN, oferecendo um fluxo gerenciado para criar, executar e depurar apps sem configuração nativa inicial. No contexto deste projeto, o Expo reduz a fricção de setup (Android/iOS), padroniza o ambiente de desenvolvimento e expõe utilitários prontos para uso (acesso a câmera, mídia, arquivos, entre outros), o que acelera a implementação e os testes das telas de calendário e das integrações com o back-end de dados (EXPO, 2025b).

O Expo Go, por sua vez, é o aplicativo complementar utilizado durante o desenvolvimento. Ele carrega o *bundle* JavaScript do projeto diretamente no dispositivo físico (ou emulador) por meio do servidor de desenvolvimento, permitindo visualização imediata das mudanças com *Fast Refresh*, acesso ao menu de depuração e teste rápido de navegação e estados. Esta abordagem favorece ciclos curtos de feedback e ensaios em hardware real, sem necessidade de compilar binários nativos a cada iteração (EXPO, 2025a).

Na prática, a combinação Expo + Expo Go viabiliza o desenvolvimento ágil

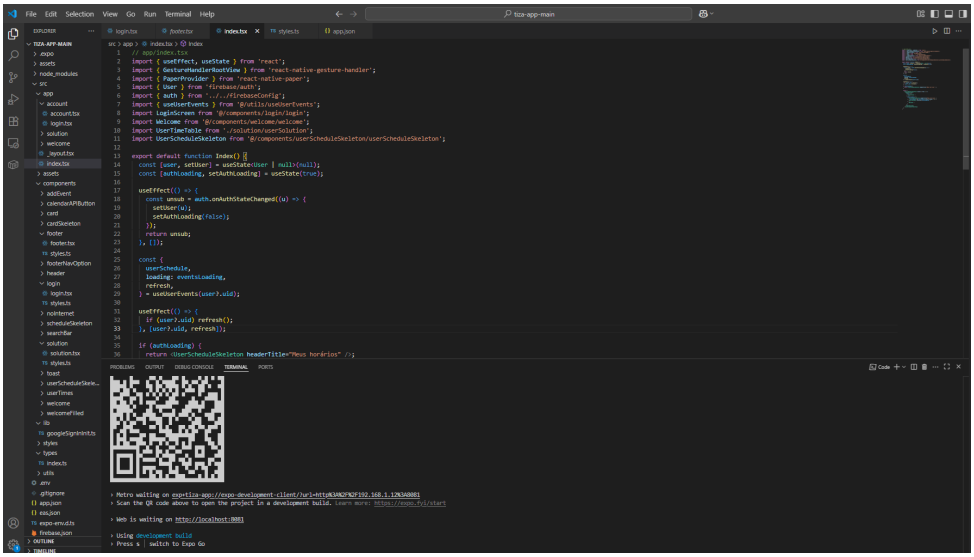


Figura 11 – Representação do VS Code

do cliente móvel: o projeto é executado em ambiente uniforme, a inspeção das telas e comportamentos ocorre em tempo quase real, e a equipe mantém o foco nas funcionalidades de montagem e visualização do horário em vez de tarefas operacionais de infraestrutura.

3.2.4 Ambiente de desenvolvimento: Visual Studio Code

O Visual Studio Code (VS Code) é um editor leve e multiplataforma que reúne recursos de edição, depuração e controle de versão em uma única interface, com suporte nativo a JavaScript/TypeScript e um ecossistema amplo de extensões. Na Figura 11 é apresentada uma representação de código no ambiente de desenvolvimento do VS Code.

No projeto, o VS Code foi utilizado como Integrated Development Environment (IDE) principal para o desenvolvimento do aplicativo em RN, viabilizando produtividade por meio de recursos como IntelliSense, formatação e linting integrados, terminal embutido para execução do servidor de desenvolvimento e integração com Git para versionamento contínuo. A disponibilidade de extensões específicas para RN, Expo e Firebase simplifica tarefas recorrentes (como inspeção de logs, execução em dispositivos e gerenciamento de configurações), reduzindo o tempo entre codificação, teste e ajuste fino das interfaces (MICROSOFT, 2025).

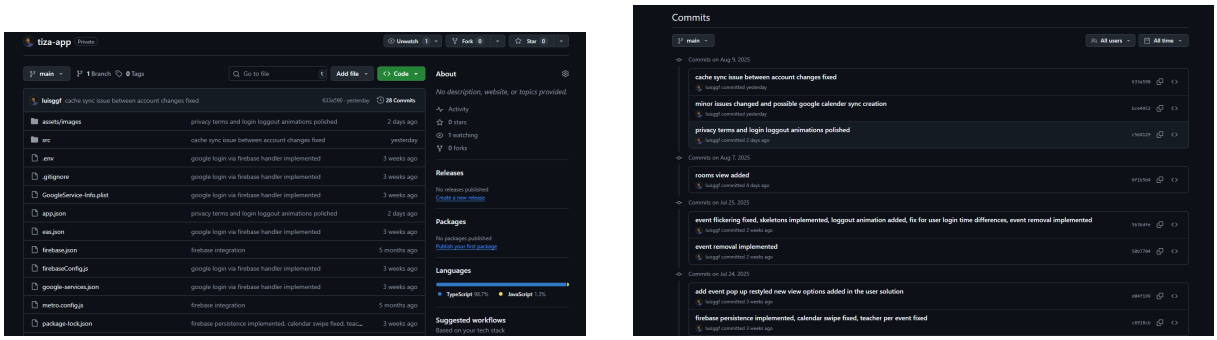
3.2.5 Git e GitHub

Para controle de versão e colaboração, o projeto adota o *Git* como sistema distribuído de versionamento e o *GitHub* como repositório remoto. O Git registra cada alteração no código como um conjunto de *commits*, possibilitando recuperar estados anteriores, comparar diferenças e evoluir o software em linhas paralelas por meio de *branches*. Essa

abordagem facilita o trabalho incremental, reduz riscos de perda de histórico e incentiva ciclos curtos de integração.

No GitHub concentram-se o repositório principal, o registro de atividades e os recursos de colaboração. A organização em *branches* permite isolar desenvolvimento de funcionalidades e correções antes da integração ao ramo principal, enquanto *pull requests* viabilizam revisão de código e discussão de mudanças. O uso de README e convenções de mensagem de commit melhora a rastreabilidade e a compreensão do projeto ao longo do tempo, beneficiando manutenção e futuras extensões (GITHUB, 2025). A Figura 12 ilustra a interface do repositório no GitHub, destacando elementos comuns do fluxo de trabalho (lista de *commits*, *branches* e *pull requests*), utilizados durante o desenvolvimento deste aplicativo.

Figura 12 – Interface do GitHub — repositório e histórico de *commits*



(a) Visão geral do repositório (b) Histórico de *commits* (versionamento)

Fonte: (GITHUB, 2025)

3.2.6 Prototipação e design: Figma

O Figma é uma plataforma baseada em navegador voltada ao design colaborativo de interfaces e à criação de protótipos navegáveis. Ele permite que várias pessoas trabalhem simultaneamente no mesmo arquivo, com histórico de versões e comentários em tempo real, o que acelera a validação visual e a tomada de decisões de layout (FIGMA, 2025).

Entre os recursos mais usados no projeto estão componentes e variantes, que padronizam elementos recorrentes (botões, cards, cabeçalhos) e seus estados; Auto Layout e grids, que ajudam na consistência de espaçamentos e alinhamentos; além de protótipos que encadeiam fluxos próximos da experiência final do aplicativo. A inspeção de propriedades e a exportação de ativos em formatos vetoriais ou bitmap facilitam a passagem do design para a implementação em React Native (FIGMA, 2025).

No contexto deste trabalho, o Figma foi empregado para produzir protótipos de alta fidelidade das telas baseadas em calendário, definir paleta de cores, tipografia e ícones, e consolidar um guia de estilos que orienta a construção das interfaces. Esses

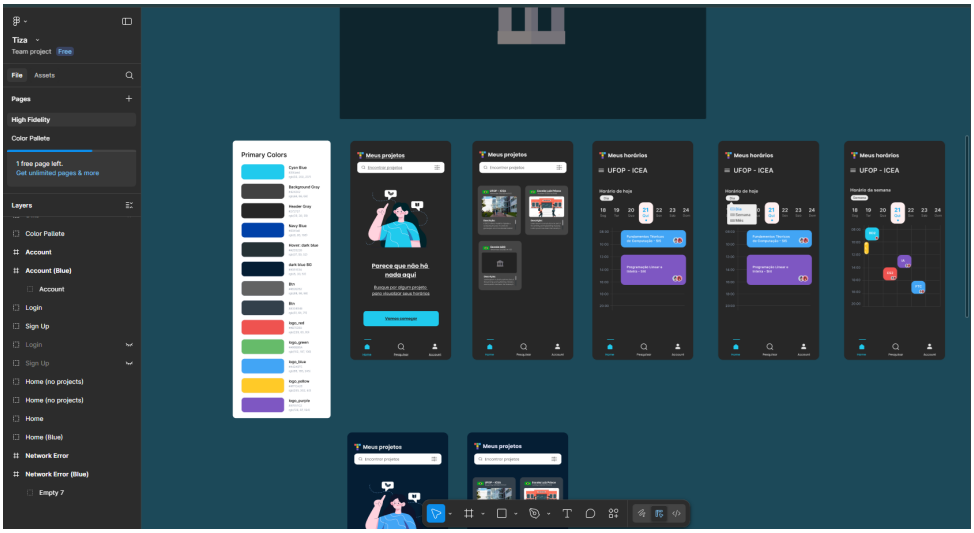


Figura 13 – Guias de estilo do projeto.

artefatos serviram como referência única durante o desenvolvimento, reduzindo retrabalho e garantindo uniformidade visual entre as diferentes visões do calendário e as telas auxiliares (FIGMA, 2025). No projeto, além dos protótipos por fluxo e experimentações de estilos, foi elaborada uma diretriz de estilo própria, como evidenciado na Figura 13.

3.3 Requisitos do sistema

Esta seção apresenta os requisitos que orientam o desenvolvimento do aplicativo, organizados em dois grupos: funcionais e não funcionais. Os requisitos funcionais descrevem as capacidades que o sistema deve oferecer ao usuário, isto é, o conjunto de ações e fluxos necessários para montar e gerenciar o horário a partir das ofertas institucionais do *Tiza*. Já os requisitos não funcionais especificam propriedades de qualidade e restrições que influenciam a experiência e a operação do sistema, como usabilidade, desempenho, segurança e disponibilidade (SOMMERVILLE, 2016).

3.3.1 Requisitos funcionais

A seguir estão os requisitos funcionais que o sistema deve cumprir para atender, de maneira eficaz, as suas funcionalidades principais e satisfazer as necessidades do usuário:

1. Autenticar o usuário (Google ou e-mail/senha previamente cadastrados no *Tiza*); em caso de cadastro, redirecionar via link externo para a plataforma.
2. Carregar e listar horários públicos provenientes do *Tiza*; permitir busca textual e ordenações simples (por nome ou data de criação).
3. Exibir a visão semanal do horário escolhido e permitir a montagem do horário do estudante por seleção de aulas/turmas.

4. Armazenar o horário montado no espaço do usuário (Firebase) e no cache local.
5. Exibir o calendário do usuário com visões de semana, e — após o horário estar montado — visões de dia, 3 dias e agenda.
6. Permitir edição do horário (adição e remoção de eventos); abrir detalhe do evento (docentes, salas e demais metadados).
7. Apresentar a tela de conta (foto/e-mail quando Google), link para Política de Privacidade e opção de sair.

3.3.2 Requisitos não funcionais

Nesta seção, discutiremos os requisitos não funcionais que definem qualidades esperadas do sistema:

- Portabilidade: execução em Android e iOS, com instalação pelas lojas oficiais.
- Desempenho: renderização fluida das visões de calendário e respostas rápidas nas operações de montagem e edição.
- Confiabilidade: persistência do horário do usuário em nuvem e cache local; verificação consistente de conflitos por sobreposição temporal.
- Segurança e privacidade: autenticação obrigatória e acesso aos dados do usuário restrito ao próprio usuário.
- Operação offline: leitura do horário a partir do cache; sincronização posterior quando a conexão for restabelecida; adições dependem de conectividade para consulta aos horários.
- Integração: aderência aos formatos e estruturas de dados publicados pelo Tiza, garantindo consumo estável dos horários institucionais.
- Manutenibilidade: código organizado e versionado, com mensagens de erro claras e estrutura preparada para evoluções incrementais.

3.4 Banco de dados e modelos de dados

O aplicativo utiliza o *Cloud Firestore* como repositório principal de dados. Trata-se de um banco NoSQL hospedado na nuvem, acessível por meio de SDKs nativos para Android e iOS, que organiza informações em documentos e coleções e mantém sincronização em tempo quase real entre cliente e servidor (FIREBASE, 2025b). Neste projeto, o Firestore cumpre dois papéis complementares: atua como camada de publicação dos

horários institucionais gerados no *Tiza* e persiste o horário montado e as preferências do usuário no espaço do próprio usuário.

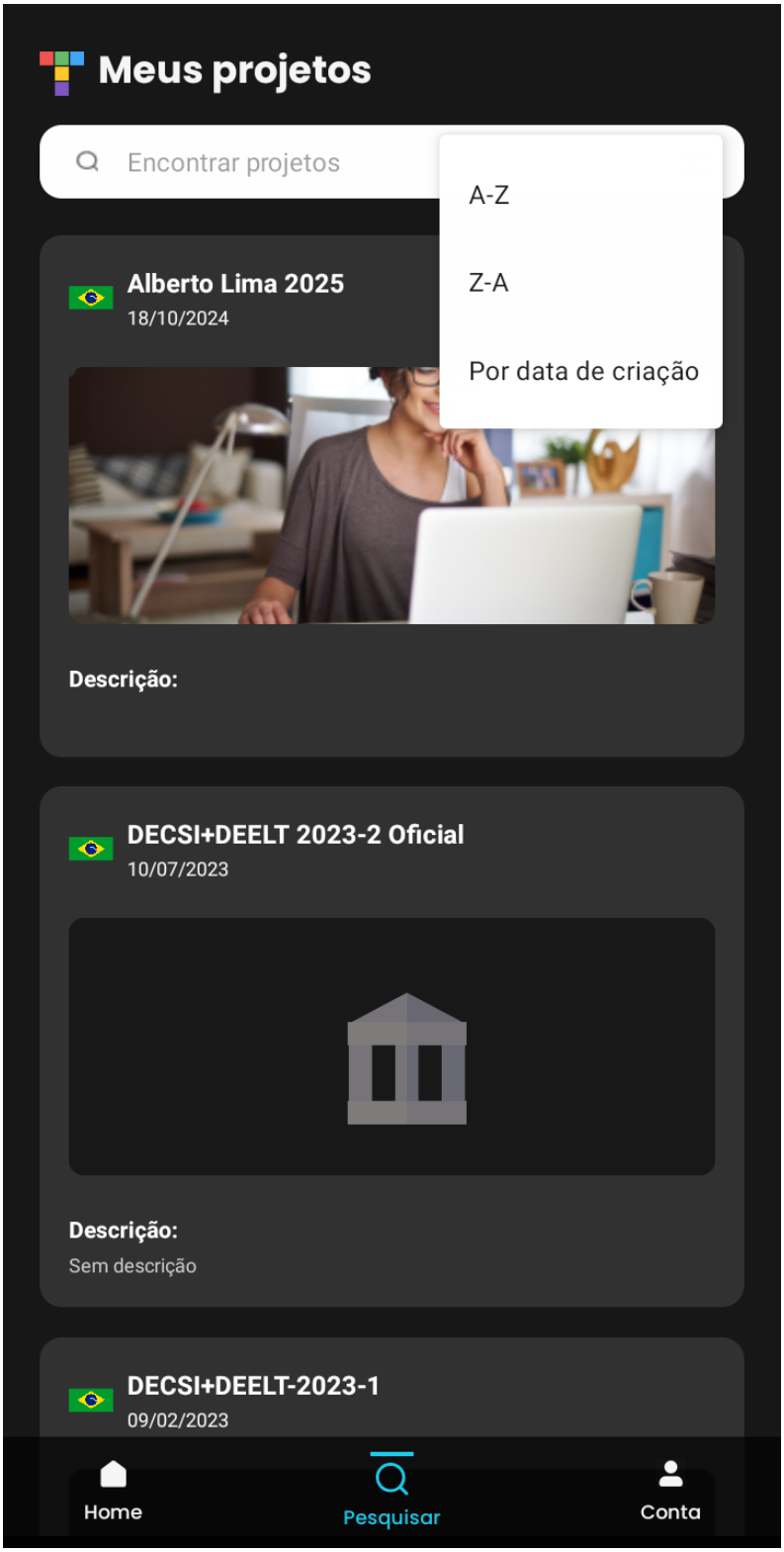


Figura 14 – Coleção `publicSolutions` — catálogo de horários publicados

Camada institucional As ofertas oficiais são catalogadas na coleção `publicSolutions`. Cada documento funciona como apontador e armazena, entre outros campos, `filename`

e `url`. O campo `filename` identifica o projeto publicado e a `url` indica o caminho para os dados detalhados da instituição. Ao acessar esse caminho, encontram-se as coleções `resources`, `lessons` e `solutions`. Em `resources` ficam docentes, salas, abreviações e cores; em `lessons` estão as aulas e turmas com seus identificadores, cores, professores e salas; em `solutions` aparecem os *assignments* que relacionam aulas e recursos aos respectivos *timeslots*, base para a montagem do calendário no aplicativo. Os registros de modelagem e a árvore das coleções podem ser vistos nas Figura 14 e Figura 16.

Camada do usuário O calendário pessoal do estudante é salvo no Firestore em documentos do próprio usuário, incluindo o nome do horário e a lista de eventos (*title*, *start*/*end*, *color*, *rooms*, *teachers*). Esse mesmo conjunto é replicado em cache local para leitura offline. A organização das coleções e documentos do usuário é ilustrada na Figura 15.

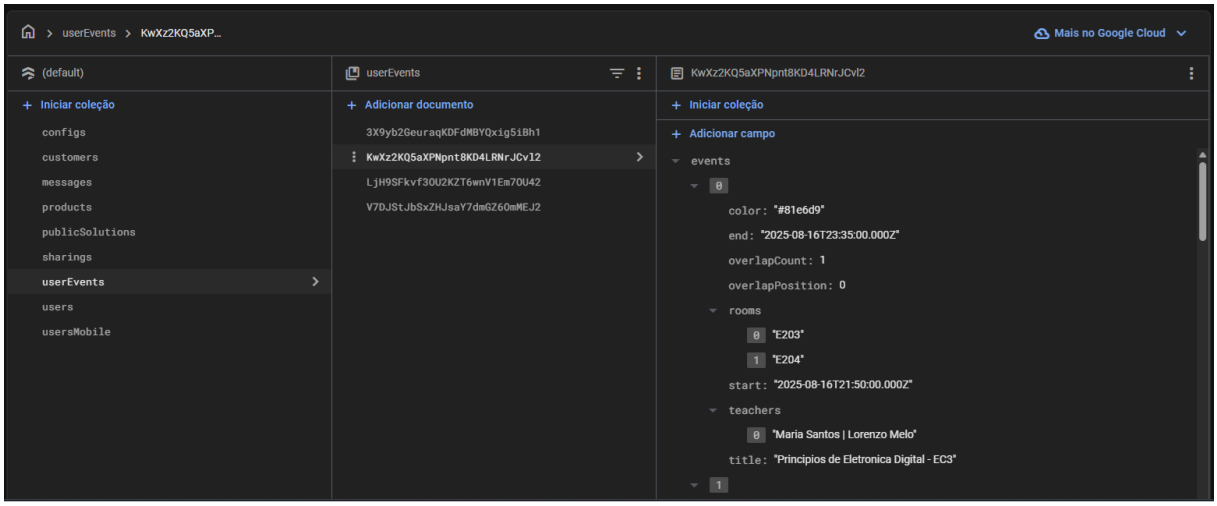


Figura 15 – Espaço do usuário: horário montado e eventos

Para consolidar os elementos do modelo, a Tabela 1 apresenta as principais entidades e campos utilizados no projeto. O catálogo de horários públicos reúne documentos que apontam para os dados de cada instituição. Os modelos institucionais incluem `Lessons`, que descreve aulas e turmas; `Resources`, que reúne docentes, salas e demais recursos; e `Solutions` com seus `Assignments`, que definem a alocação de aulas a recursos e intervalos de tempo.

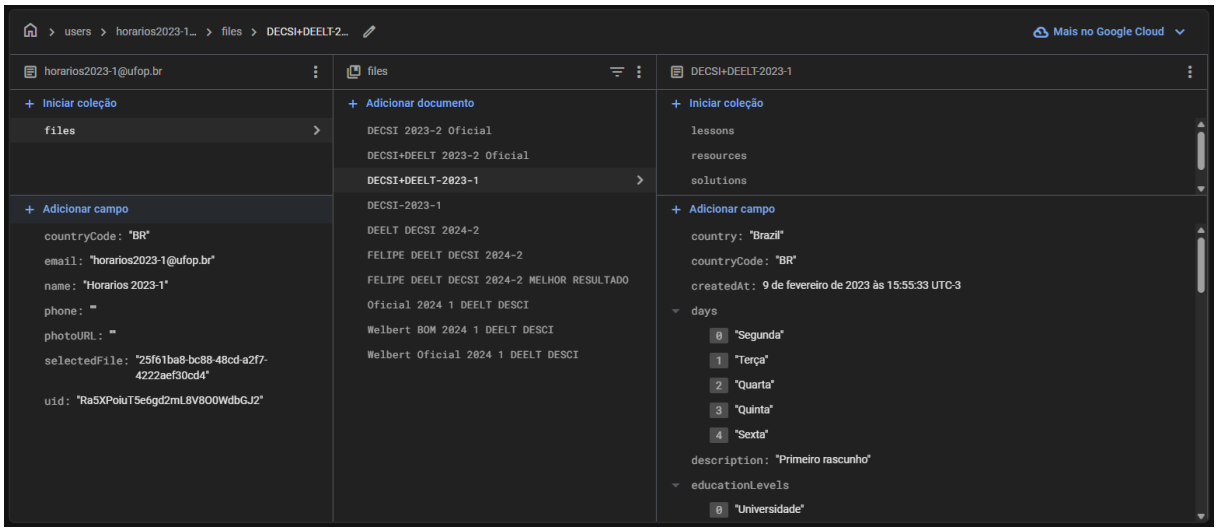


Figura 16 – Estrutura institucional: `resources`, `lessons`, `solutions`

Tabela 1 – Entidades e atributos do modelo de dados

Entidade	Atributo	Tipo	Descrição
Event	title	string	Título da disciplina.
Event	start	Date	Início da disciplina (data e hora).
Event	end	Date	Fim da disciplina (data e hora).
Event	color	string	Cor da disciplina.
Event	rooms	string[]?	Salas associadas à disciplina. (opcional)
Event	teachers	string[]	Docentes associados à disciplina.
BuiltSchedule	name	string	Nome do horário montado pelo usuário.
BuiltSchedule	events	Event[]	Lista de eventos que compõem o horário do usuário.
Class	id	string	Identificador único da disciplina.
Class	name	string	Nome da turma.
Class	color	string	Cor da turma.
Class	rooms	string[]?	Salas vinculadas à turma (opcional).
Class	createdAt	Date	Data de criação do registro.
Class	classes	string[]	Lista de turmas relacionados.
Class	teachers	string[]	Lista de docentes associados.
Assignment	lessonRef	string	Referência da aula (ligação com `Class/Lesson`).
Assignment	resources	Resources[]	Recursos (ex.: docentes, salas) vinculados à aula.
Assignment	timeslots	string[]	Intervalos de tempo da aula (usados para montar os eventos).
Resources	id	string	Identificador único do recurso.
Resources	name	string	Nome do recurso (ex.: docente, sala).
Resources	short	string	Abreviação do recurso.
Resources	color	string	Cor associada ao recurso.
Resources	photoURL	string	URL de foto/ícone do recurso.
Resources	createdAt	Date	Data de criação do recurso.
Resources	updatedAt	Date	Data da última atualização do recurso.
Solution	id	string	Identificador único do horário publicado.
Solution	name	string	Nome do horário/arquivo institucional.
Solution	creator	string	Criador/autor do horário.
Solution	status	string	Estado do horário (ex.: publicada, rascunho).
Solution	createdAt	Date	Data de criação do horário.
Solution	updatedAt	Date	Data da última atualização do horário.
Solution	assignments	Assignment[]	Lista de associações aula–recursos–tempos.

3.5 Casos de uso

O sistema é modelado em torno de um único ator principal, o *Usuário*, responsável por interagir com os fluxos centrais da aplicação. Esses fluxos incluem a autenticação e o encerramento de sessão; a consulta a horários públicos do *Tiza*; a montagem do horário pessoal a partir desses horários; a edição do calendário por meio da adição e remoção de eventos; a visualização em diferentes perspectivas (semana, dia, três dias e agenda); a exibição detalhada dos eventos; e o acesso à área de conta com a política de privacidade. Além disso, foram contemplados cenários de operação offline, com leitura a partir de cache local e sincronização posterior das alterações. A Figura 17 sintetiza esses comportamentos no diagrama de Casos de Uso desenvolvido com a ferramenta Astah (Change Vision, Inc., 2025).

As descrições completas dos casos de uso — abrangendo objetivos, pré-condições, cenários principais e extensões — encontram-se organizadas no Apêndice A. Esse material funciona como referência central para a implementação, os testes e a validação dos requisitos funcionais definidos neste projeto.

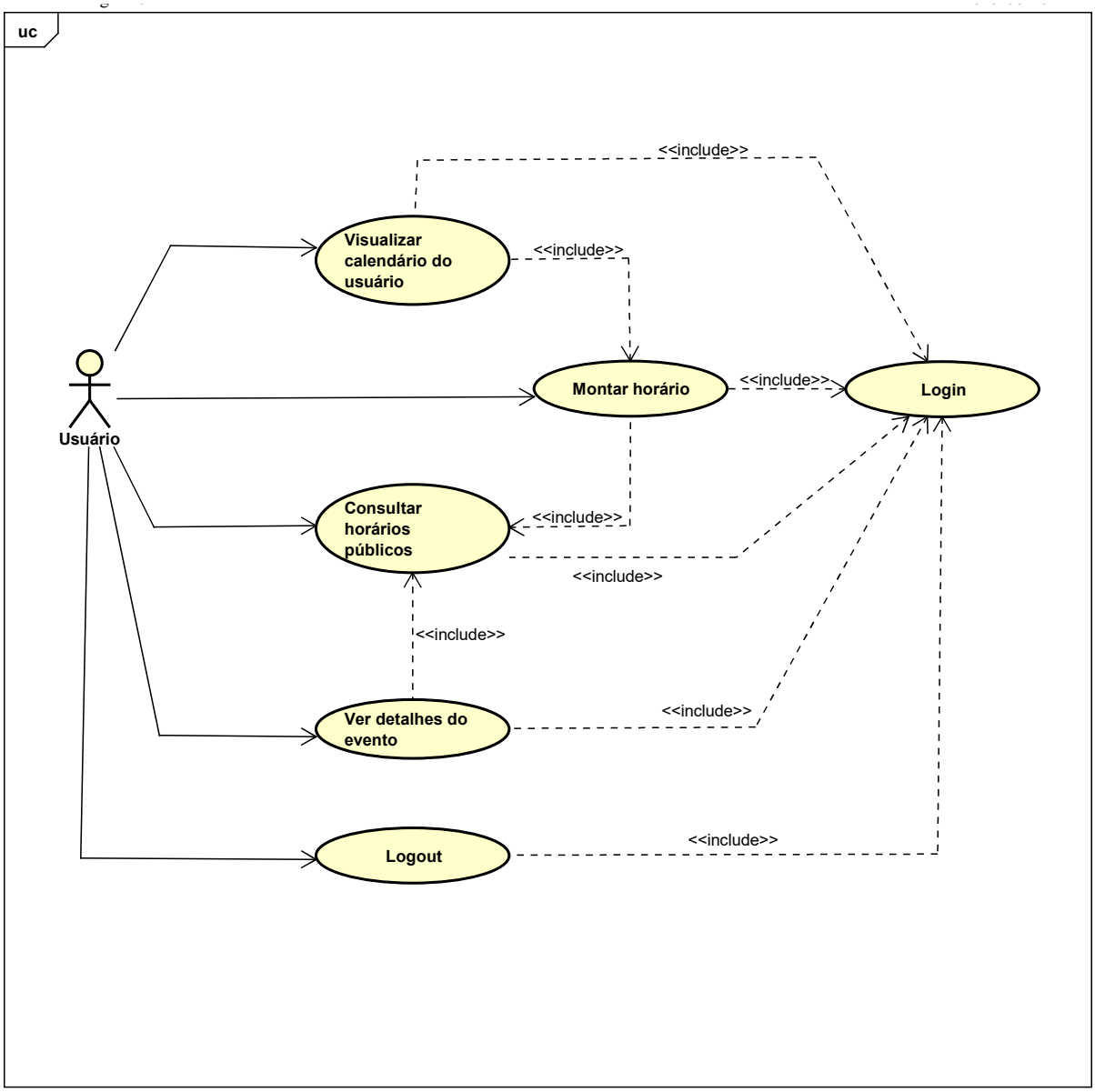


Figura 17 – Diagrama de casos de uso

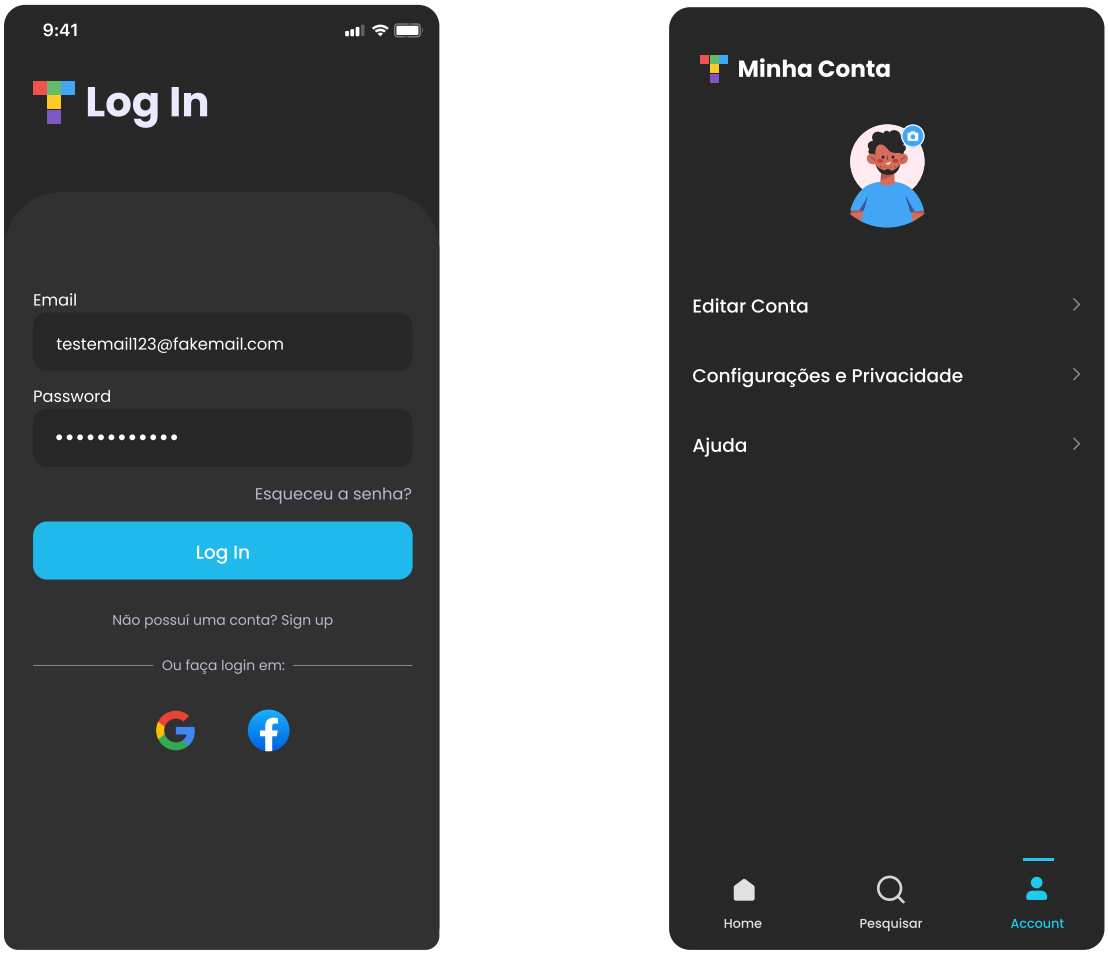
3.6 Protótipos de alta fidelidade

Protótipos de alta fidelidade são representações interativas e detalhadas que se aproximam bastante do produto final, tanto no aspecto visual User Interface (UI) quanto na experiência de uso User Experience (UX). Esse tipo de protótipo utiliza elementos gráficos refinados, fluxos de navegação completos e interações próximas da realidade, permitindo que usuários e partes interessadas testem o sistema como se estivessem utilizando a versão definitiva. Assim, são especialmente úteis em etapas avançadas de desenvolvimento, quando é necessário validar usabilidade, apresentar o sistema a stakeholders ou orientar a implementação técnica (Nielsen Norman Group, 2023; Interaction Design Foundation, 2024). Conforme discutido na subseção 3.2.6, a construção desses protótipos foi realizada com o apoio do Figma, que possibilitou a criação colaborativa das interfaces e a definição

do fluxo de interação do aplicativo.

As telas de autenticação e de gerenciamento de conta estão reunidas na Figura 18, onde se observa o fluxo de entrada e os elementos básicos do perfil. As telas iniciais pós-autenticação, que orientam o primeiro acesso e apresentam o catálogo de horários públicos, são mostradas na Figura 19. As diferentes perspectivas de calendário, no caso dia e semana, podem ser vistas na Figura 20, evidenciando a organização dos eventos e a alternância entre modos de visualização. Por fim, a paleta cromática empregada no aplicativo está apresentada na Figura 21, servindo como referência de consistência visual ao longo das telas.

Figura 18 – Protótipos das telas referentes à autenticação e à conta do usuário

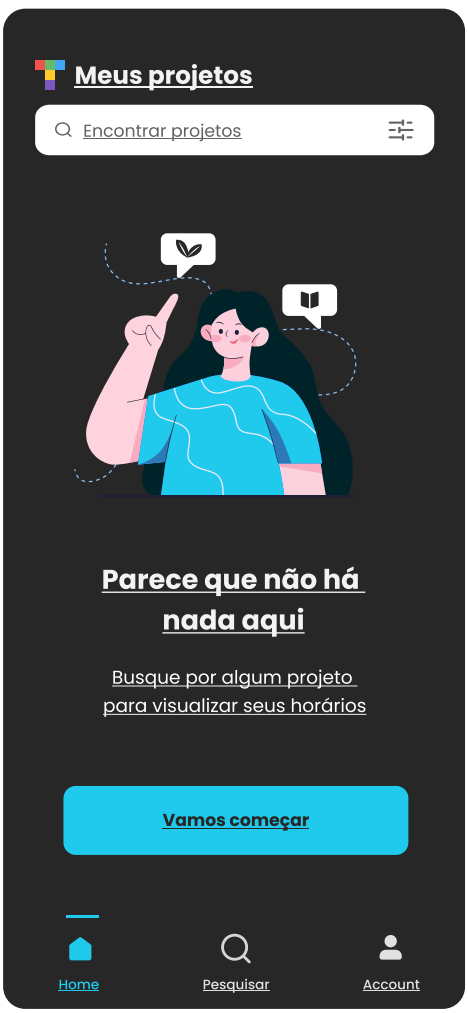


(a) Protótipo de tela inicial de autenticação

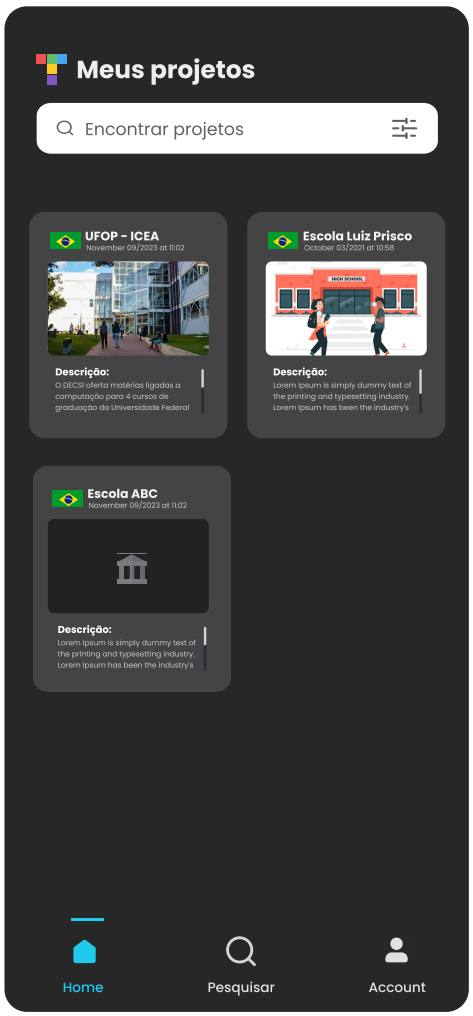
(b) Protótipo de gerenciamento de conta do usuário

Fonte: (FIGMA, 2025)

Figura 19 – Protótipos das telas iniciais pós-autenticação



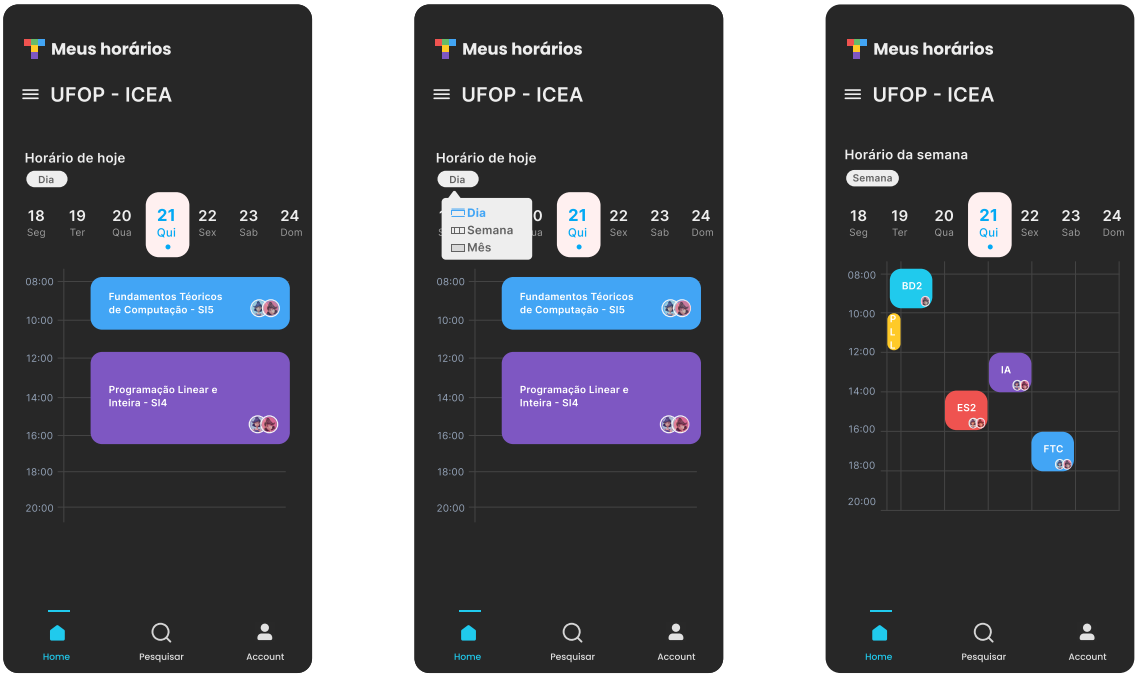
(a) Protótipo de tela inicial pós-login instruindo o usuário



(b) Protótipo de tela contendo horários públicos do Tiza

Fonte: (FIGMA, 2025).

Figura 20 – Protótipos das telas referentes à visualização dos horários



(a) Visualização do horário por dia

(b) Comutador entre modos de visualização

(c) Visualização semanal do calendário

Fonte: (FIGMA, 2025)

Figura 21 – Protótipo referente à paleta de cores do aplicativo

Fonte: (FIGMA, 2025)

4 Resultados

Este capítulo apresenta o aplicativo desenvolvido por meio de capturas de tela acompanhadas de descrições dos principais fluxos de uso. A intenção é demonstrar de que forma os requisitos do sistema foram traduzidos em interface e interações, abrangendo desde o primeiro acesso do estudante até o acompanhamento contínuo de sua rotina de horários.

4.1 Montagem de horário do usuário

Esta seção descreve o fluxo de montagem do horário pessoal a partir dos horários públicos disponibilizados pelo *Tiza*. São apresentadas a tela inicial do usuário, a listagem de horários públicos com suas opções de filtragem, a visualização semanal do horário público com seleção de turma e, por fim, o processo de adição de eventos a partir do detalhe de cada aula.

4.1.1 Tela inicial do usuário

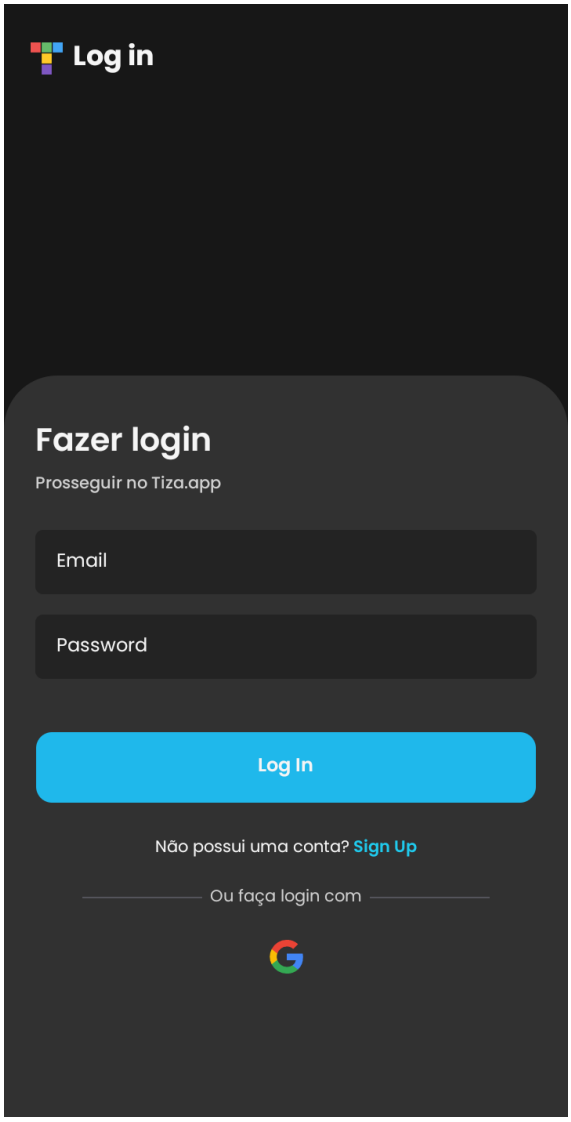
O processo de acesso ao sistema tem início na autenticação do usuário por e-mail e senha vinculados ao Tiza ou por integração via Google. Considerando isso, a Figura 22 ilustra a tela de *login*, bem como a opção alternativa de autenticação via Google. Alguns casos de exceções possíveis de serem encontrados nessa tela são tratadas de forma explícita por meio de mensagens. Como demonstra a Figura 24, credenciais inválidas geram um *toast*[1] informativo de erro, enquanto o cancelamento da autenticação pelo Google também é devidamente sinalizado ao usuário.

Ao acessar a aba *Conta* no rodapé, o usuário é levado à tela de gestão de conta, onde visualiza e-mail e, quando disponível, a foto de perfil, além das ações *Encerrar sessão* e *Política de Privacidade*. A Figura 23a ilustra essa interface. Ao escolher *Sair da conta*, a aplicação finaliza a autenticação e limpa o cache local do dispositivo. Em *Política de Privacidade*, o app abre o documento oficial do Tiza no navegador do dispositivo e disponibiliza a opção *Solicitar exclusão*, que compõe automaticamente uma mensagem de e-mail com destinatário e assunto predefinidos, pronta para envio. A Figura 23b apresenta essa etapa.

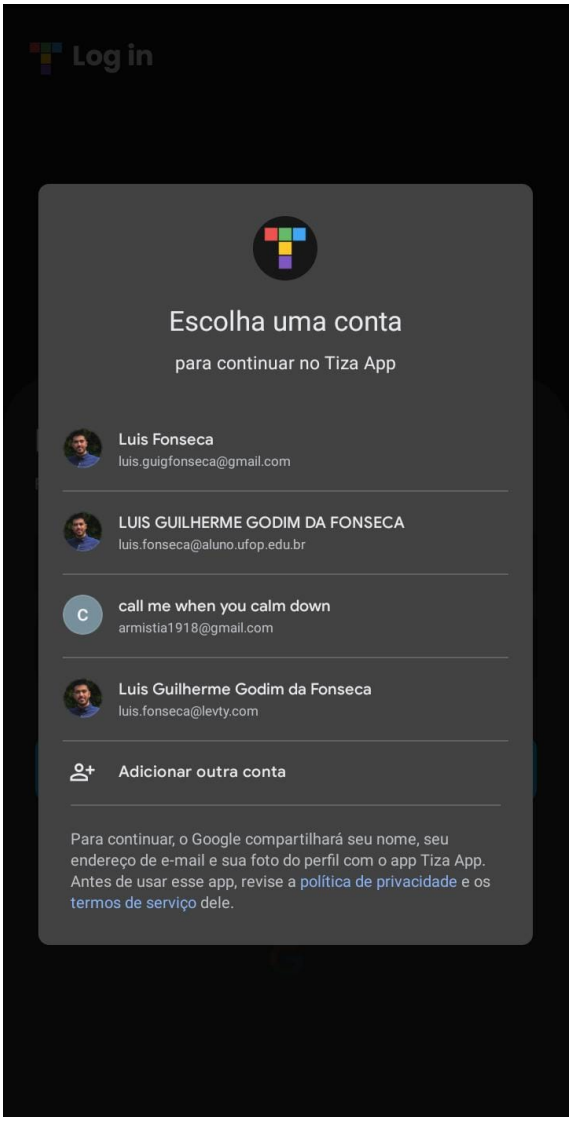
Após o processo de autenticação, o estudante encontra a área do próprio calendário. Quando ainda não existem eventos, o aplicativo apresenta um estado de boas-vindas

[1] Toast é uma mensagem temporária exibida sobre a interface, geralmente na parte inferior da tela, para informar o usuário de forma breve e não intrusiva.

Figura 22 – Tela de Login.



(a) Tela de Login.



(b) Popup de Login via Google.

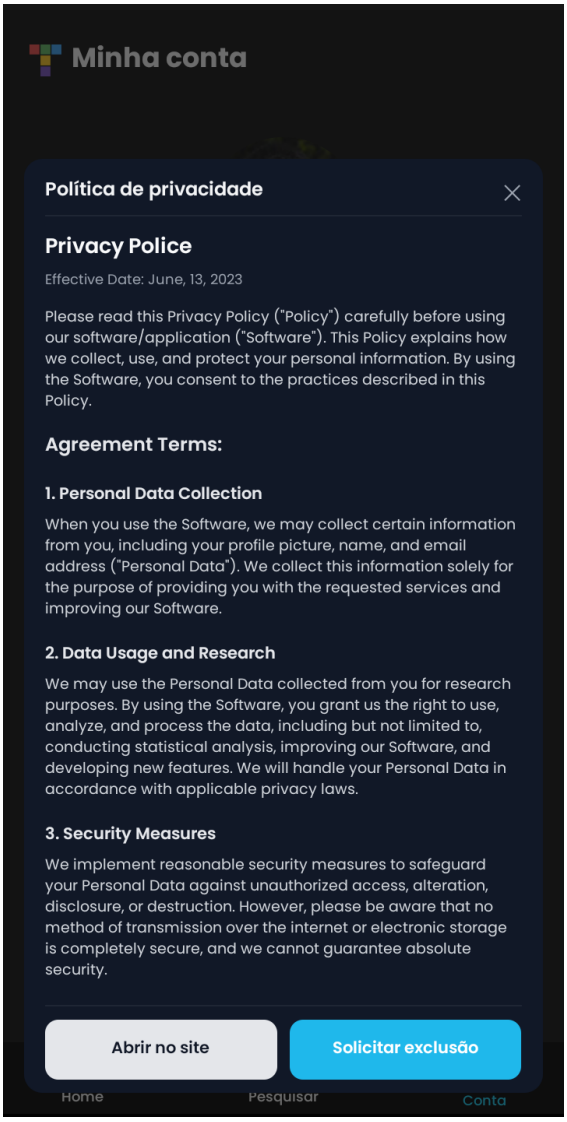
com orientação para consultar horários públicos como pode ser observado na Figura 25a. Entretanto, quando o usuário já possui um horário montado, a visão semanal é carregada com os horários escolhidos previamente. A Figura 25 ilustra esses dois cenários.

4.1.2 Listagem de horários públicos e filtragens

A consulta aos horários públicos é o ponto de partida para montar o calendário pessoal. A Figura 26 apresenta a listagem em formato de blocos, contendo nome do projeto, descrição e imagem cadastrada no Tiza. Além disso, o usuário dispõe de recursos de busca textual por nome e de ordenação por data de criação ou ordem alfabética. Esse acesso pode ser feito tanto pela tela de boas-vindas, discutida anteriormente, quanto pelo atalho na opção “Pesquisa” localizada no rodapé de navegação do aplicativo.

Figura 23 – Tela de conta do usuário e Política de Privacidade.

(a) Tela de conta do usuário.

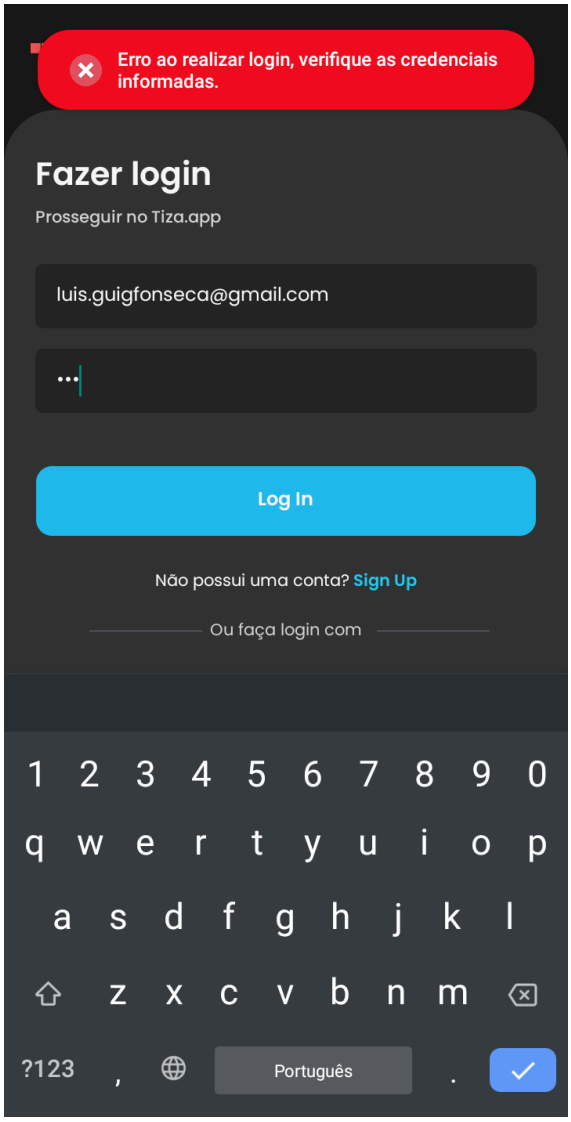


(b) Política de Privacidade exibida a partir dessa tela.

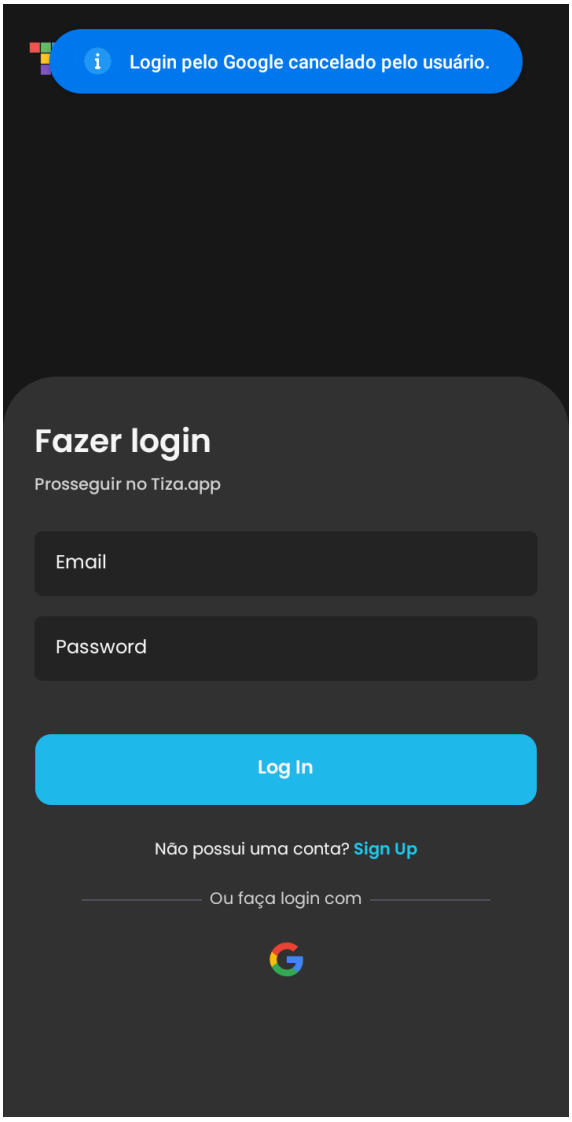
Vale destacar que o aplicativo obtém esses dados diretamente do Tiza, o que torna a interface dependente de conexão com a internet. Em caso de indisponibilidade de rede, o sistema sinaliza a falha e oferece a opção de tentar novamente, conforme ilustrado na Figura 26c. Pelo mesmo motivo, enquanto as informações são carregadas, são exibidos *skeletons*[2], que indicam o andamento do processo e contribuem para manter a navegação contínua, como mostra a Figura 32c.

[2] Skeleton é uma estrutura visual de carregamento, em formato de blocos ou linhas cinzas, que antecipa a disposição dos elementos da interface enquanto os dados reais ainda estão sendo carregados.

Figura 24 – Tela de Login e suas exceções.



(a) Toast exibido na tela de Login quando credenciais são inválidas.



(b) Toast exibido na tela de Login quando o usuário cancela o Login pelo Google.

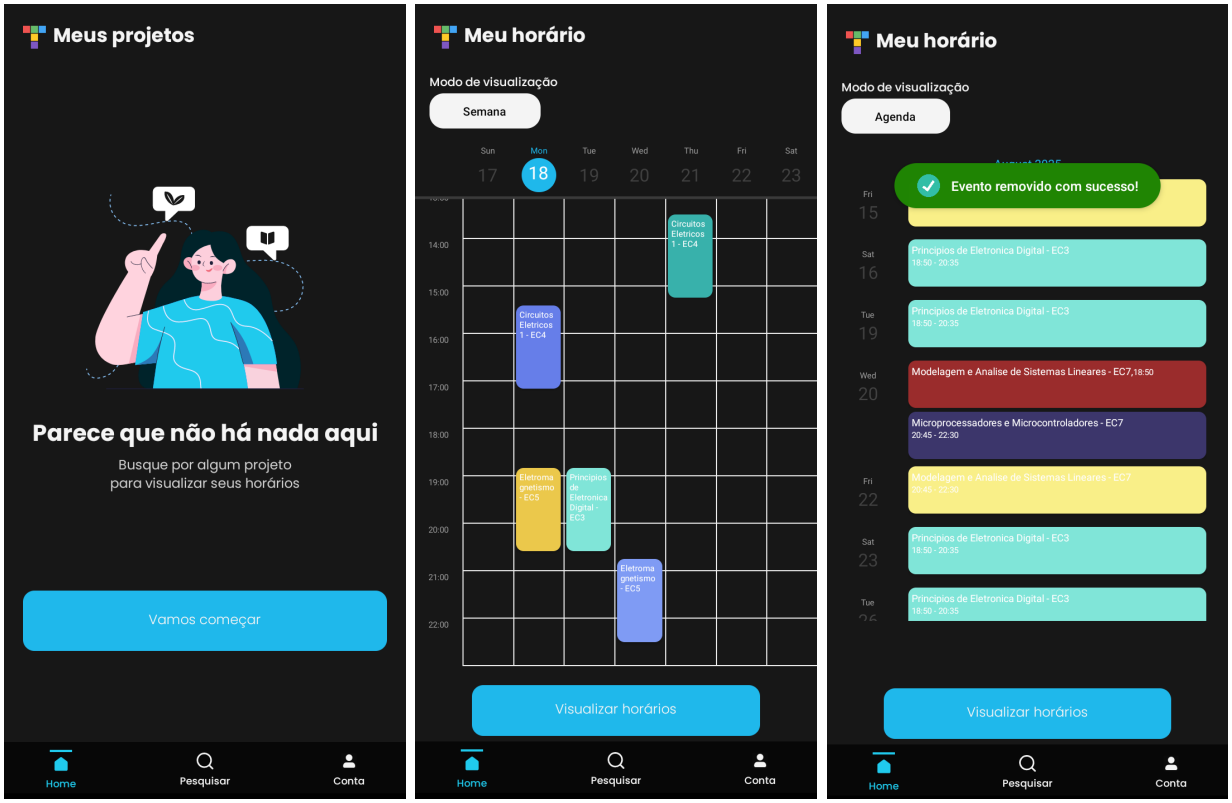
4.1.3 Visualização do horário público e seleção de turma

Ao selecionar um horário público, o aplicativo apresenta a grade semanal correspondente, permitindo que o estudante escolha a turma cujas aulas deseja incluir. Essa visão exibe os *slots*[3] de aula disponíveis e serve de base para a montagem do horário pessoal. Além disso, para suavizar o tempo de carregamento dos horários da instituição escolhida, é exibido um *skeleton* conforme pode ser visto na Figura 32a.

Seleção de turma: A escolha da turma é feita por meio de um menu dedicado. Esse seletor concentra as turmas publicadas para o horário em questão e, ao mudar a opção, a grade semanal é atualizada para refletir apenas as aulas da turma escolhida.

[3] Slot no contexto desse trabalho é um intervalo de tempo definido no calendário acadêmico usado para alocar uma aula.

Figura 25 – Telas de horários do usuário.



(a) Usuário novo.

(b) Usuário com horário cadastrado.

(c) Toast exibido ao remover um horário.

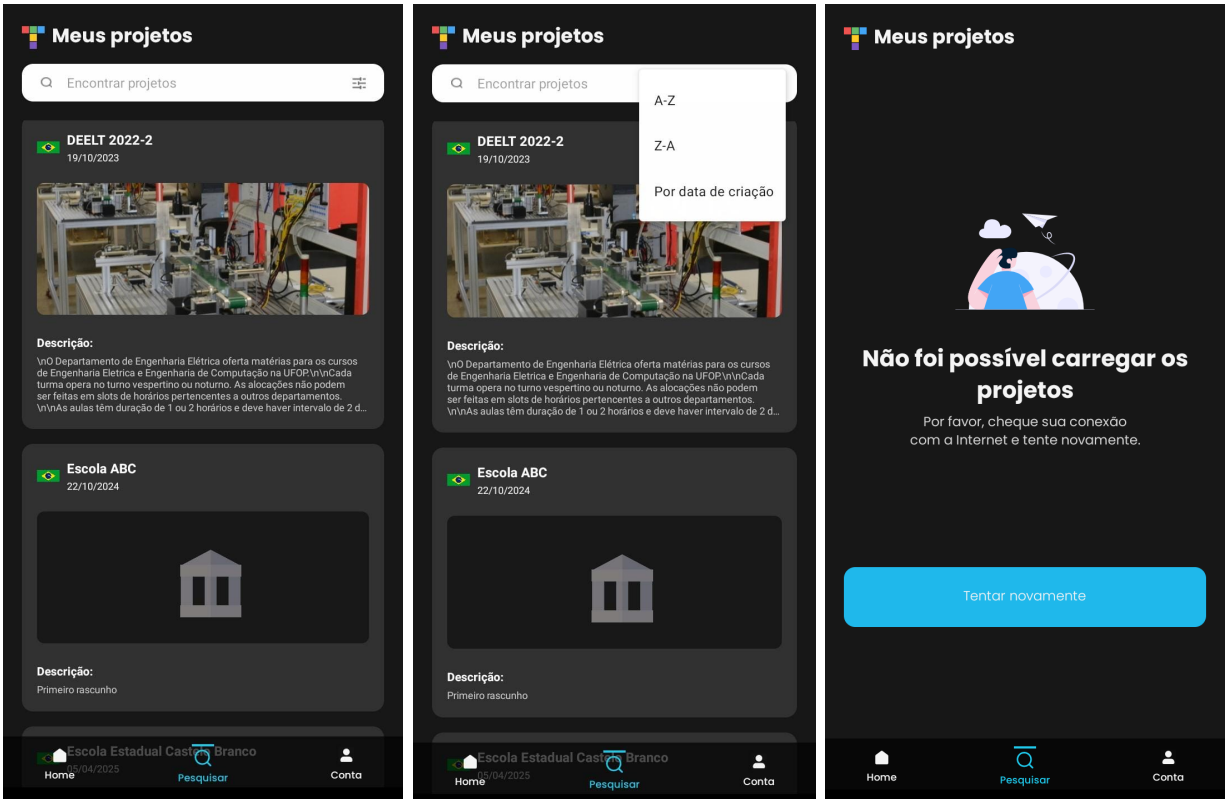
A Figura 27c exemplifica o menu de seleção e o efeito imediato sobre a listagem de aulas.

Conjunto de aulas disponíveis: Depois de selecionar a turma, o estudante visualiza as aulas associadas a ela ao longo da semana, com identificação clara de nome, professores quando informados e salas quando informadas. A Figura 27a ilustra essa organização, facilitando a decisão sobre o que incluir no horário pessoal.

Detecção de sobreposições.: Antes de confirmar uma inclusão, o sistema verifica conflitos com o que já foi adicionado ao horário do estudante. Há conflito quando o intervalo de início e término da aula que se deseja incluir coincide, no todo ou em parte, com o intervalo de qualquer evento já presente no mesmo dia. A verificação considera apenas o tempo; não leva em conta professor ou sala. Quando existe sobreposição, os eventos já adicionados são exibidos com opacidade para evidenciar a ocupação, como mostrado na Figura 27b, e a inclusão é bloqueada.

Feedback de erro na inclusão com conflito: Quando a tentativa de adicionar uma aula conflita com o horário pessoal, o aplicativo informa o problema por meio de uma mensagem breve na tela. Esse aviso explica que o período escolhido já está ocupado

Figura 26 – Tela de horários públicos disponibilizados no Tiza.



(a) Tela de horários públicos.

(b) Opções de filtragem dos horários públicos.

(c) Usuário sem conexão com a internet.

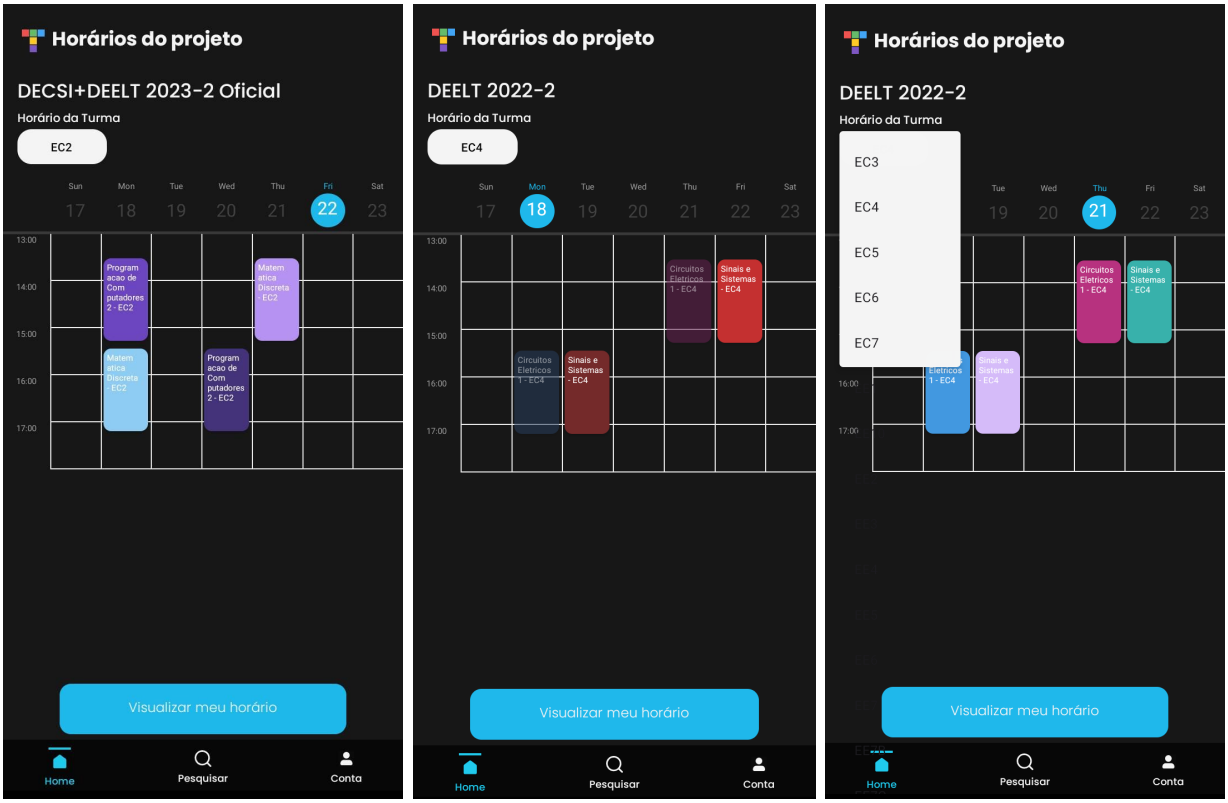
e orienta o estudante a revisar a seleção. A Figura 29b mostra o *toast* apresentado nesse cenário.

4.1.4 Adição a partir do detalhe do evento

Ao selecionar um *slot*, o aplicativo apresenta uma janela com os metadados do evento — título com identificação da turma, sala e docentes (quando informados), além do horário de início, término e indicação de recorrência. Nessa mesma janela é exibido o comando de ação para inclusão no horário pessoal. A Figura 28a ilustra esse *popup*.

Antes de confirmar a inclusão por meio do botão “Adicionar”, o sistema realiza a validação de possíveis conflitos por sobreposição temporal, conforme discutido anteriormente. Na ausência de conflito, o evento é efetivamente incorporado ao horário. Como resultado prático, ocorre a persistência dos dados tanto em cache quanto no Firestore, acompanhada da exibição de uma mensagem de sucesso, conforme ilustrado na Figura 29a.

Figura 27 – Tela do horário público escolhido.



(a) Aulas disponíveis para turma.

(b) Exemplo de eventos com conflitos de horário.

(c) Seleção de turma.

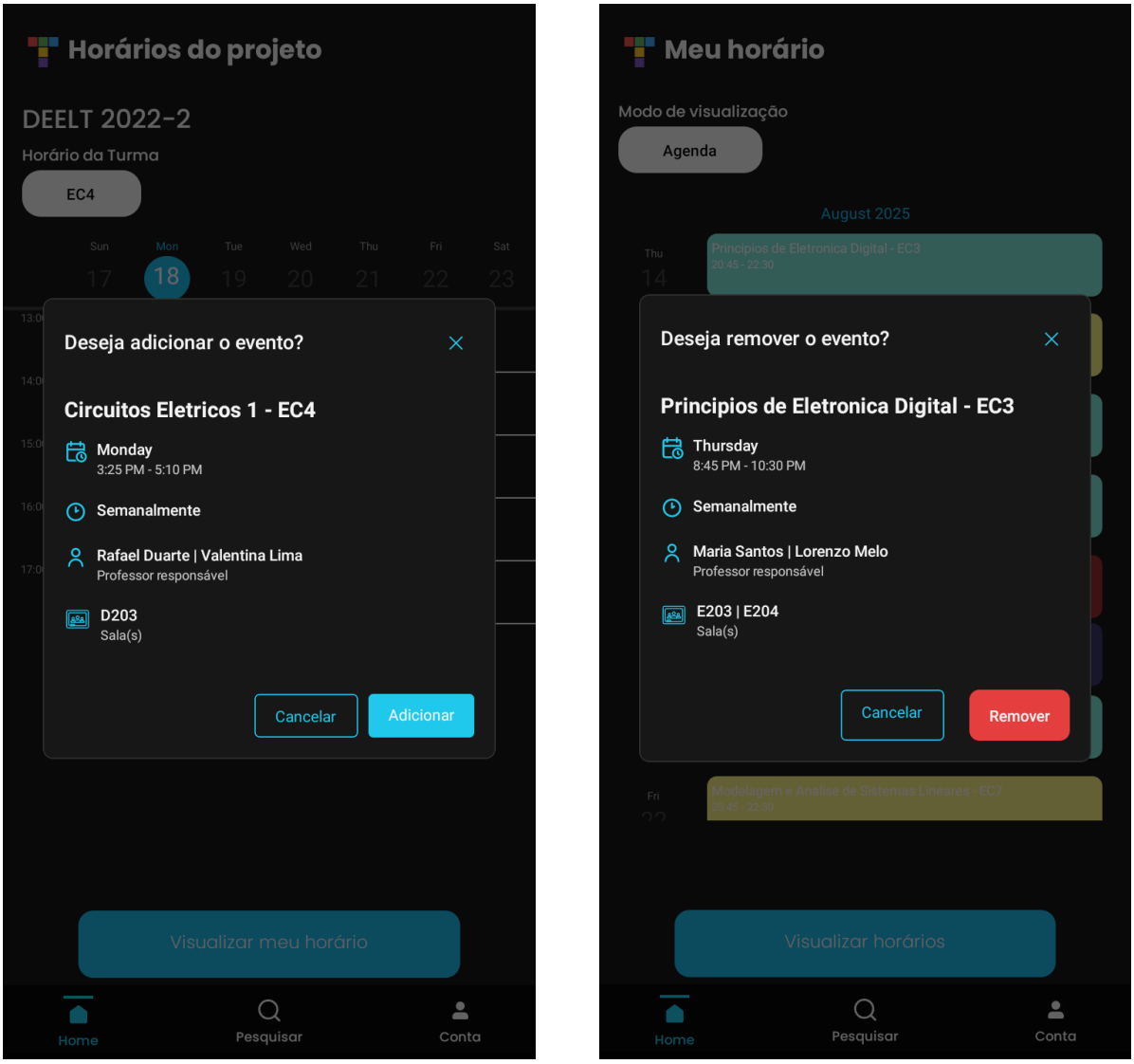
4.2 Horário pessoal do usuário

Esta seção descreve como o estudante acompanha seu calendário após a montagem do horário, destacando as visões disponíveis, a forma de seleção entre elas, o carregamento com *skeletons* e o fluxo de remoção de eventos a partir do detalhe.

Visões de calendário e seleção. O acompanhamento inicia na visão **Semana**, adotada como padrão para dar uma leitura ampla da distribuição das aulas. O usuário pode alternar para **Dia**, **3 Dias** ou **Agenda** por meio de um menu expansível posicionado na barra superior da tela. Cada modo atende a uma necessidade específica: *Semana* favorece o planejamento geral do período letivo, *3 Dias* reduz a densidade visual mantendo um horizonte útil, *Dia* foca a rotina imediata e *Agenda* lista as ocorrências em ordem temporal de forma similar à diária. As variações *Semana* e *3 Dias* estão ilustradas nas Figura 30a e Figura 30b, enquanto *Dia* e *Agenda* são mostradas nas Figura 31a e Figura 31b. Durante a abertura do calendário do usuário, um *skeleton* indica o carregamento e mantém a interface responsiva até a leitura do estado local ou da nuvem, conforme pode ser visto na Figura 32b.

Entrada no contexto de construção. Ao acionar o botão **Visualizar horários**, o

Figura 28 – Popup de adição ou remoção de horário.



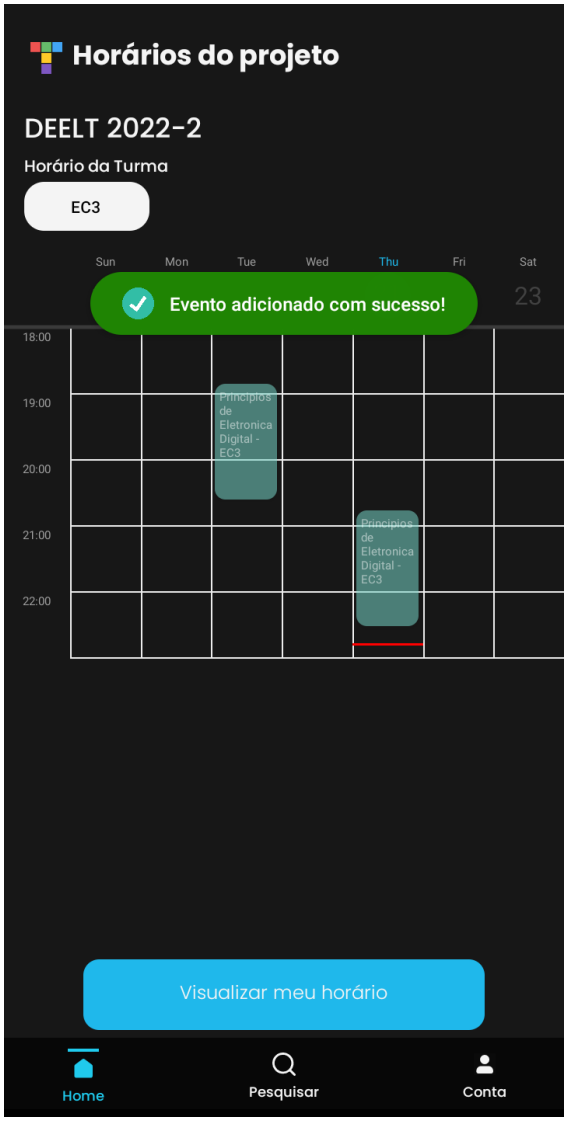
(a) Adição de horário.

(b) Remoção de horário.

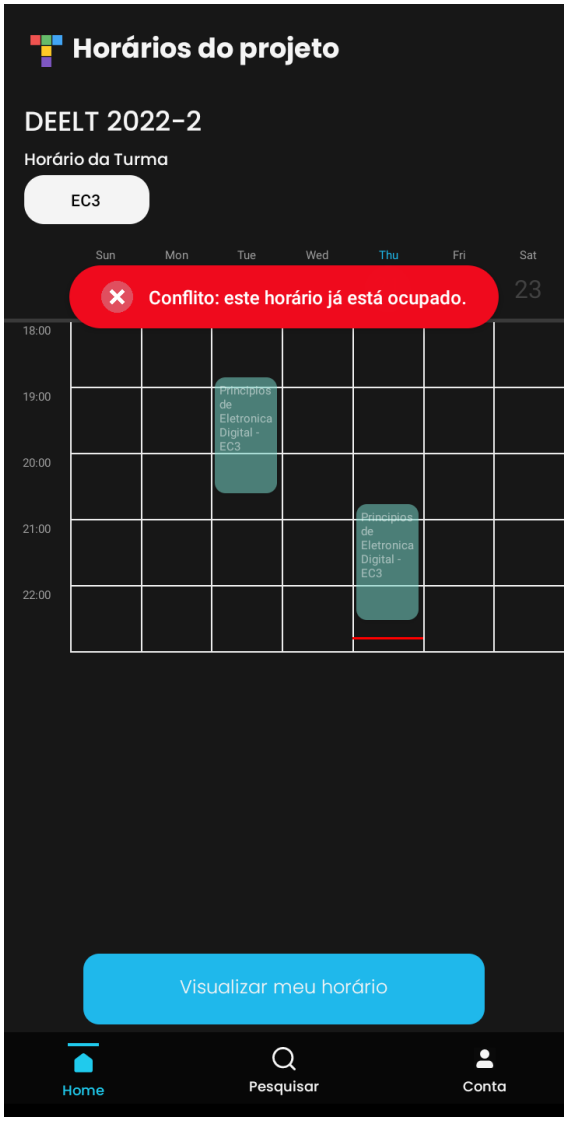
aplicativo retorna ao último *horário público* acessado, preservando o contexto de escolha de turma e a navegação prévia. Esse comportamento evita regressos desnecessários à lista geral de horários e acelera o ciclo de adição de novas aulas ao calendário pessoal.

Detalhe do evento e remoção. O toque em uma aula abre uma janela com as informações do evento de forma análoga a vista na subseção 4.1.4. Nessa janela, entretanto, como o evento pertence ao horário do usuário, é oferecida a ação **Remover**, conforme a Figura 28b. Após a confirmação, o sistema apresenta um retorno imediato por meio de um *toast* de confirmação, como mostrado na Figura 25c, e atualiza o calendário. A remoção é refletida no *Firestore* e no cache local, mantendo o estado consistente nas próximas aberturas do aplicativo.

Figura 29 – Toasts exibidos na tela de adição de horários.



(a) Toast de sucesso na adição de horário.



(b) Toast de erro ao tentar adicionar aula com conflito de horário.

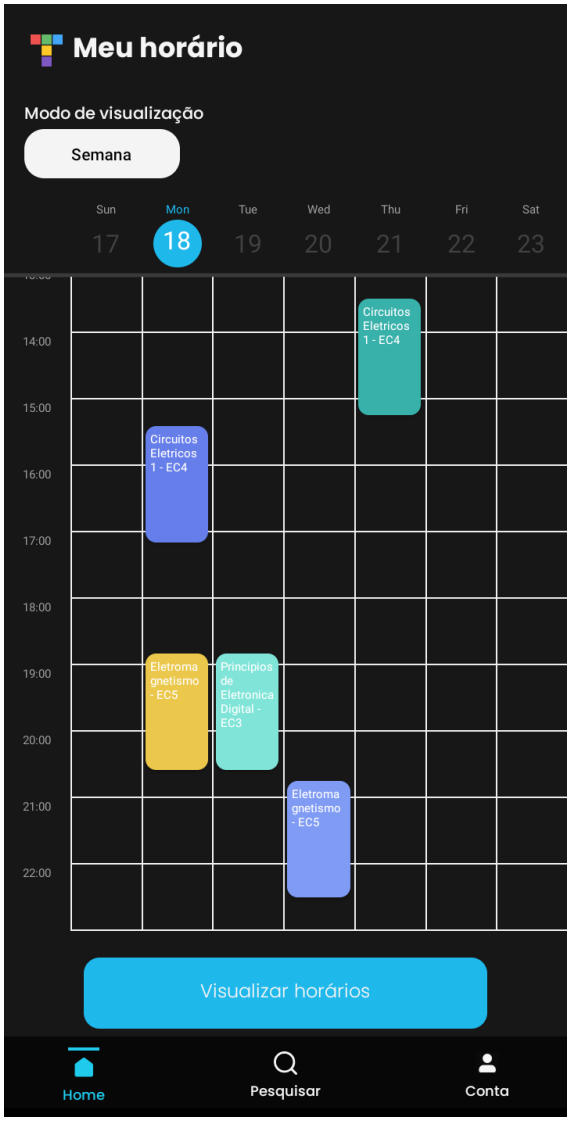
4.3 Persistência, cache e operação offline

Esta seção descreve como o aplicativo persiste o horário do usuário, mantém uma cópia local para acesso rápido e sincroniza alterações com a nuvem quando a conectividade é restabelecida.

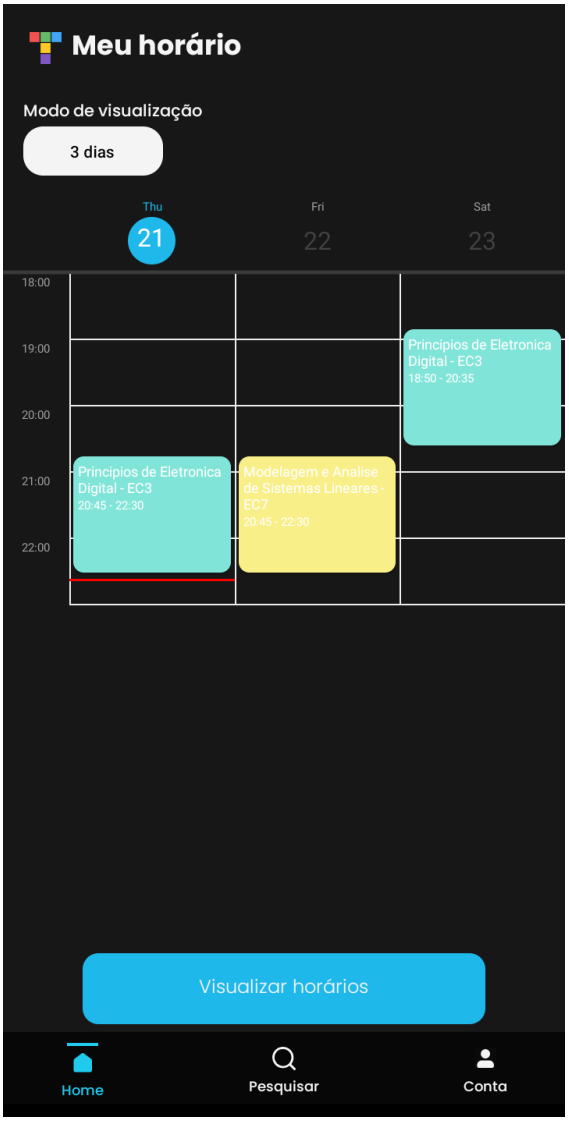
Armazenamento principal e cache local. Como discutido na subseção 3.2.2, o *Cloud Firestore*, além de ser utilizado para buscar os horários públicos do Tiza, guarda oficialmente o horário do usuário. No dispositivo, esse mesmo estado é mantido em paralelo no *AsyncStorage*[4], o que permite abrir o calendário pessoal mesmo quando o usuário não possuí conexão ativa com a internet.

[4] Armazenamento local assíncrono (chave–valor) no dispositivo, adequado para uso offline.

Figura 30 – Variações de telas de usuário com horário montado.



(a) Visualização de horários definida como 'Semana'.



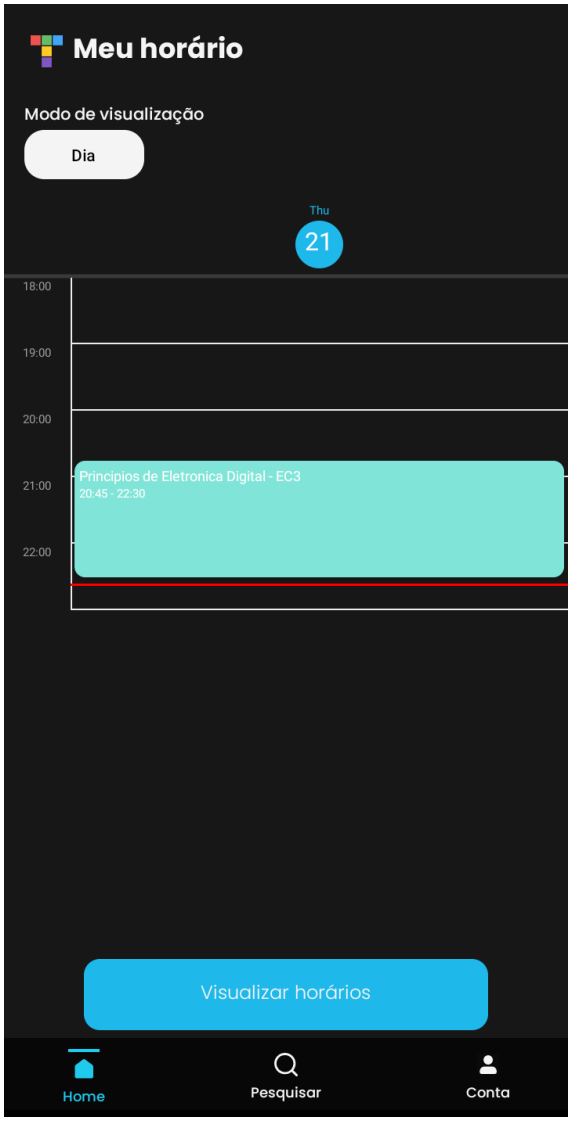
(b) Visualização de horários definida como '3 Dias'.

Escritas e leituras. Ao adicionar ou remover um evento, o aplicativo atualiza o cache local e emite a solicitação correspondente ao *Firestore*. Na leitura, o aplicativo prioriza o cache para exibir o calendário sem atraso e, quando há conectividade, revalida o conteúdo com o servidor, atualizando a cópia local caso existam diferenças.

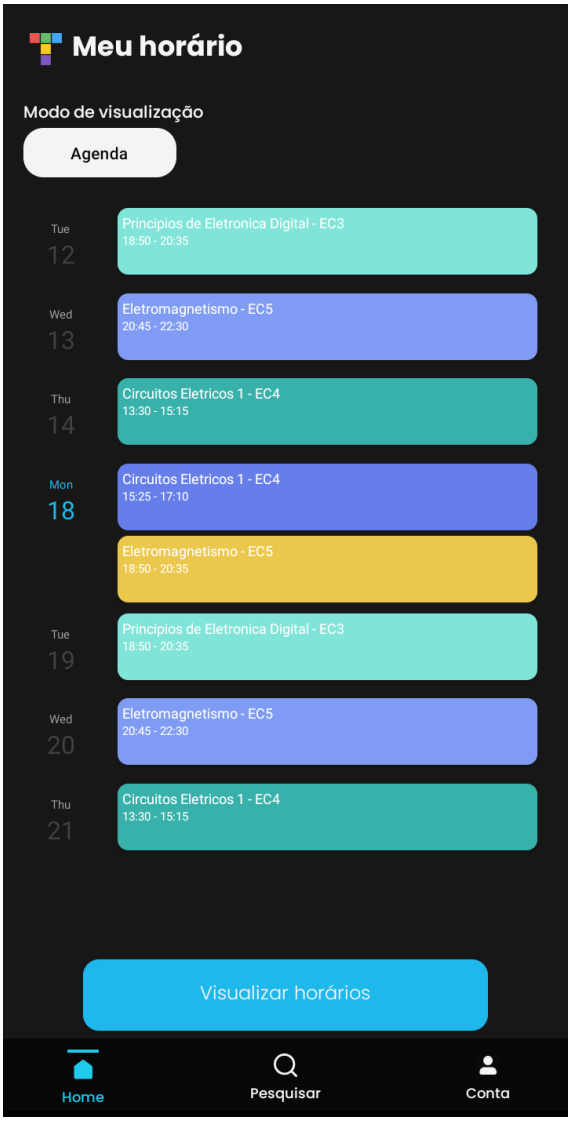
Uso sem internet. Alterações feitas sem conexão são registradas em uma fila local de operações identificadas por evento e ação (adição ou remoção). Remoções iniciadas a partir do calendário do usuário entram sempre nessa fila. Vale ressaltar que as inclusões também podem ser enfileiradas, desde que a tela de um horário público já esteja carregada no dispositivo. Caso contrário, o usuário precisa acessar novamente a lista de horários públicos, o que exige internet.

Sincronização e consistência. Ao detectar conectividade, o aplicativo processa a fila

Figura 31 – Variações de telas de usuário com horário montado - 2.



(a) Visualização de horários definida como 'Dia'.



(b) Visualização de horários definida como 'Agenda'.

em ordem, atualiza o *Firestore* e reconcilia o estado com o servidor.

Sessão e limpeza. Enquanto a sessão estiver ativa, o usuário pode reabrir o aplicativo e visualizar o calendário a partir do cache local. Quando faz *logout*, o armazenamento em memória *cache* e as alterações pendentes são removidos, e no próximo *login* o aplicativo recarrega o calendário do *Firestore* para recompô-lo.

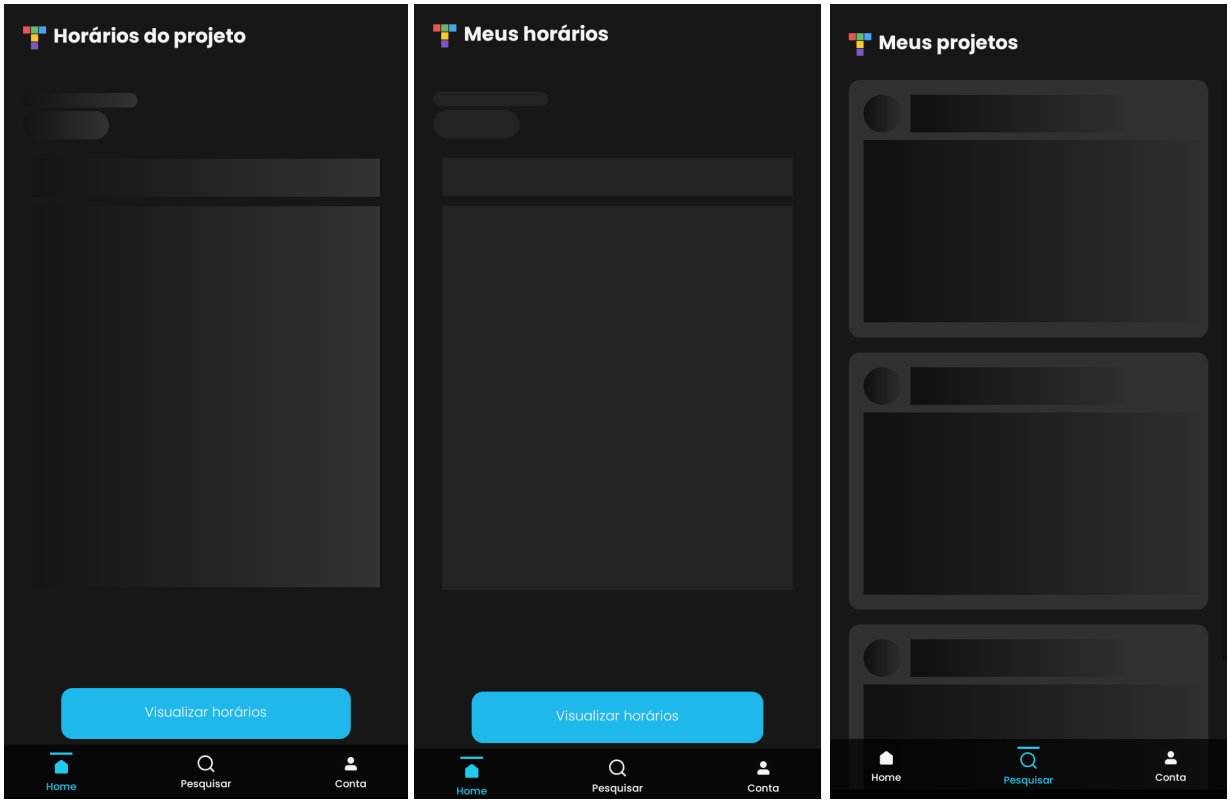
4.4 Publicação na Google Play

Como etapa final deste capítulo, registra-se que o aplicativo foi **publicado em ambiente de produção** na Google Play Store, cumprindo o objetivo específico de disponibilização em lojas oficiais. O pacote foi construído no modo *release*, assinado

e submetido por meio do Google Play Console, passando pelas verificações padrão da plataforma e pelo processo de revisão. Após aprovação, a versão ficou disponível ao público-alvo definida na configuração de distribuição[5].

A publicação contempla as telas e fluxos apresentados neste capítulo, incluindo autenticação, consulta de *horários públicos*, montagem e acompanhamento do *horário pessoal*, bem como a seção de *Conta*. A partir desta entrega, novas atualizações podem ser liberadas seguindo o mesmo canal de distribuição, garantindo evolução incremental sem alterar o escopo funcional aqui demonstrado.

Figura 32 – Telas de carregamento utilizando Skeletons.



(a) Tela de carregamento do horário público escolhido.

(b) Tela de carregamento do horário montado do usuário.

(c) Tela de carregamento dos horários públicos do Tiza.

[5] https://play.google.com/store/apps/details?id=com.anonymous.tizaapp

5 Conclusão

O projeto implementou um aplicativo móvel, em *React Native* com *Expo*, para apoiar estudantes na organização do horário acadêmico a partir dos horários institucionais publicados no *Tiza*. O fluxo de uso consiste, basicamente, em o aluno acessar os horários disponibilizados pela instituição, selecionar as turmas de interesse e obter um calendário acadêmico personalizado.

No que tange às funcionalidades centrais, a aplicação contempla autenticação do usuário, mecanismos de busca e ordenação dos horários públicos, seleção de aulas por turma e geração de um calendário individual com verificação automática de sobreposições. A checagem de conflito impede inclusões inconsistentes e a remoção pontual de eventos viabiliza ajustes ao longo do período letivo. Esses recursos, apresentados nas telas do Capítulo 4, materializam os requisitos funcionais definidos na Introdução e orientam o uso cotidiano pelo estudante.

A integração com o *Tiza* mantém o aplicativo aderente ao que é publicado pela instituição, favorecendo a atualização do conteúdo consultado e a coerência entre a visão institucional e o calendário do aluno. Para continuidade de uso em cenários de conectividade limitada, o aplicativo preserva o acesso ao horário pessoal no dispositivo e sincroniza alterações quando a comunicação com o servidor é restabelecida, garantindo consistência entre os estados local e remoto.

A experiência de navegação organiza o acompanhamento do calendário em visões complementares — Semana, Dia, Três Dias e Agenda — de modo a contemplar consultas rápidas e verificações mais detalhadas. Retornos visuais para ações bem-sucedidas ou inválidas, juntamente com o detalhamento de eventos por *popup*, contribuem para que o usuário compreenda cada etapa e mantenha o calendário consistente com suas escolhas. Por fim, a disponibilização em produção na Google Play confirma a viabilidade técnica e funcional da solução e completa o ciclo de entrega.

Em síntese, os objetivos propostos foram atendidos: o estudante consegue consultar os horários publicados no *Tiza*, montar e ajustar o próprio calendário com checagem de conflitos e utilizar o aplicativo em produção. Como próximos passos, destacam-se iniciativas que podem ampliar o alcance e a utilidade do sistema:

- **Segmentação de acesso por código institucional.** Permitir que a instituição forneça ao aluno um código de acesso que libera a visualização de seus horários públicos no aplicativo. Após informar o código, o aluno passa a ver apenas os horários daquela instituição. O acesso pode ter validade definida e ser revogado pela

instituição quando necessário.

- **Integração com Google Agenda.** Permitir a sincronização do horário pessoal diretamente com o Google Calendar, possibilitando a criação e atualização automática de eventos mediante consentimento do usuário.
- **Publicação na App Store.** Ampliar a distribuição para iOS, garantindo paridade funcional com Android e conformidade com as diretrizes de revisão e privacidade da Apple. A entrega inclui testes via TestFlight e disponibilização em produção.
- **Edição de metadados do horário pelo usuário.** Habilitar ajustes locais de informações como sala e docentes em eventos já adicionados ao horário pessoal, preservando os dados institucionais originais.
- **Notificações e lembretes.** Ativar avisos de início de aula com antecedência configurável (por exemplo, 5, 10 ou 30 minutos). Permitir habilitar por disciplina e escolher som ou modo silencioso. Alterações no horário do aluno, nesse caso, devem refletir automaticamente nos lembretes.
- **Acompanhamento de faltas.** Possibilitar o lançamento manual de faltas por disciplina e a visualização de um resumo por semana e por mês, com indicação do percentual de presença e metas configuráveis.

Referências

ASC. *ASC EduPage App*. 2025. Disponível em: <https://mobile.edupage.org/>. Citado 3 vezes nas páginas 16, 17 e 18.

ASC. *ASC EduPage App*. 2025. Disponível em: <https://www.asctimetables.com/a/verso-de-teste/>. Citado na página 17.

BUDIU, R. *Mobile user experience: Limitations and strengths*. 2015. Disponível em: <https://www.nngroup.com/articles/mobile-ux/>. Citado na página 16.

Change Vision, Inc. *Astah UML Tool*. 2025. <https://astah.net/pt/>. Accessed: 2025-08-20. Citado na página 37.

DIAS, H. B. *Optables App — Um Aplicativo Móvel para Organização de Horários Escolares Individuais*. João Monlevade, MG: [s.n.], 2023. Monografia de Conclusão de Curso. Disponível em: <https://www.monografias.ufop.br/handle/35400000/7068>. Citado 3 vezes nas páginas 23, 24 e 25.

DOCENDO. *Docendo*. 2025. Disponível em: <https://docendo.co>. Citado 2 vezes nas páginas 21 e 22.

EXPO. *Expo Go — Documentação*. [S.l.], 2025. Disponível em: <https://docs.expo.dev/get-started/expo-go/>. Citado na página 28.

EXPO. *Expo — Documentação*. [S.l.], 2025. Disponível em: <https://docs.expo.dev/>. Citado 2 vezes nas páginas 27 e 28.

FIGMA. *The Collaborative Interface Design Tool*. 2025. Disponível em: <https://www.figma.com/>. Citado 6 vezes nas páginas 30, 31, 39, 40, 41 e 42.

FIREBASE, G. *Documentação do Firebase*. 2025. Disponível em: <https://firebase.google.com/docs?hl=pt>. Citado na página 28.

FIREBASE, G. *Firestore | Firebase*. 2025. Disponível em: <https://firebase.google.com/docs/firestore?hl=pt-br>. Citado 2 vezes nas páginas 28 e 32.

FONSECA, G. H. et al. Integer programming techniques for educational timetabling. *European Journal of Operational Research*, Elsevier BV, v. 262, n. 1, p. 28–39, 2017. Disponível em: <https://doi.org/10.1016/j.ejor.2017.03.020>. Citado na página 13.

GITHUB. *GitHub features: The right tools for the job*. 2025. Disponível em: <https://github.com/features>. Citado na página 30.

Interaction Design Foundation. *High-Fidelity Prototypes – definition and examples*. 2024. Acesso em: 21 ago. 2025. Disponível em: <https://www.interaction-design.org/literature/topics/high-fidelity-prototypes>. Citado na página 38.

LALESCU, L. *Fet free timetabling software::*. 2025. Disponível em: <https://lalescu.ro/liviu/fet/>. Citado 2 vezes nas páginas 20 e 21.

META. *React Native — Documentação Oficial*. [S.l.], 2025. Disponível em: <https://reactnative.dev/>. Citado na página 27.

MICROSOFT. *Documentation for Visual Studio Code*. 2025. Disponível em: <https://code.visualstudio.com/docs>. Citado na página 29.

Nielsen Norman Group. *Low-fidelity vs. high-fidelity prototyping*. 2023. Acesso em: 21 ago. 2025. Disponível em: <https://www.nngroup.com/articles/ux-prototype-hi-lo-fidelity/>. Citado na página 38.

PAIVA, V. N. *Optables - um sistema web para a programação de horários educacionais*. 2021. Monografia (Graduação) - Instituto de Ciências Exatas e Aplicadas, Universidade Federal de Ouro Preto. Disponível em: <http://www.monografias.ufop.br/handle/35400000/5214>. Citado na página 18.

SOMMERVILLE, I. *Software Engineering*. 10. ed. Harlow: Pearson, 2016. ISBN 978-0133943030. Disponível em: <https://www.oreilly.com/library/view/software-engineering-10th/9780133943030/>. Citado na página 31.

STATCOUNTER. *Desktop vs mobile VS tablet market share worldwide*. 2025. Disponível em: <https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet>. Citado 2 vezes nas páginas 13 e 16.

TAYLOR, P. *Mobile network subscriptions worldwide 2028*. 2025. Disponível em: <https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/>. Citado 2 vezes nas páginas 13 e 16.

TIMETABLE, S. *Class schedule and homework planner for college, School and University*. 2025. Disponível em: <https://smart-timetable.app/>. Citado 2 vezes nas páginas 22 e 23.

TIZA. *Tiza - Horários Escolares*. 2025. Disponível em: <https://tiza.app/pt>. Citado 4 vezes nas páginas 13, 18, 19 e 20.

Apêndices

APÊNDICE A – Casos de Uso

A.1 Login

Caso de Uso: Login

Objetivo: Autenticar o estudante por meio de e-mail e senha válidos no Tiza.
Ator: Usuário
Pré-condições:

1. Aplicativo instalado no dispositivo.
2. Conexão com a internet.
3. Conta previamente cadastrada no Tiza.

Cenário Principal:

1. O usuário abre o aplicativo.
2. O sistema exibe a tela de autenticação.
3. O usuário informa e-mail e senha.
4. O sistema valida as credenciais junto ao serviço de autenticação.
5. Em caso de validação bem-sucedida, o usuário é direcionado à tela principal.

Extensões:

- *Sem internet:* o sistema exibe mensagem de erro e oferece tentar novamente.
- *Credenciais inválidas:* o sistema informa o erro e permite nova tentativa.
- *Criar conta:* ao acionar a opção de cadastro, o sistema abre a página do Tiza para criação de conta. Ao retornar, o usuário pode efetuar o login.

A.2 Logout

Caso de Uso: Logout

Objetivo: Encerrar a sessão do usuário no aplicativo.
Ator: Usuário
Pré-condições:

1. Usuário autenticado.

Cenário Principal:

1. O usuário navega até a área de conta.
2. O usuário escolhe a opção de sair.
3. O sistema encerra a sessão e retorna à tela de autenticação.

Extensões:

- *Cancelamento:* caso o usuário desista, o sistema mantém a sessão ativa.

A.3 Consultar horários públicos

Caso de Uso: Consulta de horários públicos

Objetivo: Permitir a busca e a navegação por horários publicados no Tiza.
Ator: Usuário
Pré-condições:

1. Usuário autenticado.
2. Conexão com a internet.
3. A instituição deve ter publicado horários no Tiza.

Cenário Principal:

1. O usuário acessa a área de pesquisa de horários.
2. O sistema carrega o catálogo de horários públicos.
3. O usuário pode ordenar por nome ou por data de criação.
4. O usuário pode realizar busca textual por nome.
5. O usuário pode selecionar um horário para visualização semanal segmentada por turma.

Extensões:

- *Sem internet ou falha no carregamento:* o sistema apresenta um componente de fallback com a opção de tentar novamente.
- *Sem resultados:* nenhum horário público é exibido.

A.4 Montar horário a partir de horário público

Caso de Uso: Montar horário

Objetivo: Construir e ajustar o horário pessoal do usuário a partir de um horário institucional publicado.
Ator: Usuário
Pré-condições:

1. Usuário autenticado.
2. Conexão com a internet.
3. Horário público selecionado e carregado.

Cenário Principal:

1. O usuário navega pela visão semanal do horário publicado.
2. O usuário seleciona aulas individuais que deseja cursar.
3. Para cada seleção sem conflito, o sistema adiciona o evento ao horário pessoal.
4. O usuário pode remover eventos previamente adicionados.
5. O sistema atualiza a visualização semanal indicando os eventos já adicionados.
6. O horário pessoal é persistido no espaço individual do usuário.

Extensões:

- *Conflito de horário:* ao detectar sobreposição temporal, o sistema bloqueia a adição e sinaliza o conflito. As aulas já adicionadas podem ser mostradas com opacidade para indicar ocupação.
- *Perda de conexão durante a montagem:* se a tela de adição já estiver carregada, solicitações podem ser enfileiradas para sincronização posterior; caso contrário, o sistema orienta a retomar a operação quando houver conectividade.

A.5 Visualizar calendário do usuário

Caso de Uso: Visualização do calendário

Objetivo: Exibir o horário pessoal do usuário em diferentes visões para acompanhamento diário.

Ator: Usuário
Pré-condições:

1. Usuário autenticado.
2. Horário pessoal previamente montado, persistido em nuvem ou cache local.

Cenário Principal:

1. O usuário acessa a tela de calendário.
2. O sistema apresenta a visão semanal.
3. O usuário alterna entre as visões de dia, três dias e agenda conforme a necessidade.
4. O usuário pode navegar por datas e abrir eventos para ver detalhes.

Extensões:

- *Sem eventos:* o sistema exibe estado vazio e oferece atalho para a área de pesquisa de horários.
- *Sem internet:* o sistema carrega o horário a partir do cache local quando disponível.

A.6 Ver detalhe do evento

Caso de Uso: Detalhe do evento

Objetivo: Exibir informações completas de um evento e oferecer a ação contextual de adicionar ao horário pessoal ou remover dele.
Ator: Usuário
Pré-condições:

1. Usuário autenticado.
2. Presença de um dos contextos:
    a) Evento oriundo de um horário público carregado; ou
    b) Evento pertencente ao horário pessoal do usuário.

Cenário Principal:

1. O usuário abre a tela de detalhes a partir da visão semanal de um horário público ou do próprio calendário.

2. O sistema exibe título com turma, docentes quando informados, salas quando informadas, início e fim, cor e informação de recorrência quando disponível.

3. Se o evento ainda não fizer parte do horário pessoal, o sistema oferece a ação *Adicionar*. Ao confirmar:

    a) O sistema verifica conflitos de horário.

    b) Se não houver conflito, o evento é adicionado ao horário pessoal e a visualização é atualizada.

    c) Se houver conflito, o sistema informa o problema e bloqueia a adição.

4. Se o evento já fizer parte do horário pessoal, o sistema oferece a ação *Remover*. Ao confirmar:

    a) O sistema remove o evento do horário pessoal e atualiza a visualização.

Extensões:

- *Sem internet:* a visualização permanece disponível. Solicitações de remoção são enfileiradas para sincronização posterior. Solicitações de adição exigem conectividade, salvo quando a tela de adição já estiver carregada, caso em que podem ser enfileiradas.
- *Conflito na adição:* o sistema destaca a colisão temporal e impede a inclusão.