INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
CAMPUS CAJAZEIRAS
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

MANAGER RU

CICERO RUDAN DE LUCENA NASCIMENTO

Cajazeiras
2021

CICERO RUDAN DE LUCENA NASCIMENTO

MANAGER RU

Trabalho de Conclusão de Curso apresentado junto ao Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia da Paraíba - Campus Cajazeiras, como requisito à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador

Prof. Me. Francisco Paulo de Freitas Neto.

Cajazeiras
2021

IFPB
Campus Cajazeiras
Coordenação de Biblioteca
Biblioteca Prof. Ribamar da Silva
Catalogação na fonte: Daniel Andrade CRB-15/593

N244m

Nascimento, Cicero Rudan de Lucena

Manager RU / Cicero Rudan de Lucena Nascimentos; orientador Francisco Paulo de Freitas Neto.- 2021.
53 f.: il.

Orientador: Francisco Paulo de Freitas Neto.
TCC (Tecnólogo em Análise e Desenvolvimento de Sistemas.) – Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, Cajazeiras, 2021.

1. Softwares 2. Restaurantes universitários 3. Ionic 4. API 5. Rest I. Título

004.4(0.067)

SERVIÇO PÚBLICO FEDERAL
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
COORDENAÇÃO DA UNIDADE ACADÊMICA DA ÁREA DE INFORMÁTICA - CAMPUS CAJAZEIRAS

Às **14:00** horas do dia **04** do mês de **fevereiro** do ano de **2021,** via Google Meet, compareceu para defesa pública do **Trabalho de Conclusão de Curso**, requisito obrigatório para a obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas, o(a) aluno(a) **CICERO RUDAN DE LUCENA NASCIMENTO**, matrícula **201512010340**, tendo como Título do Trabalho **MANAGER RU**. Constituíram a Banca Examinadora os professores **FRANCISCO PAULO DE FREITAS NETO** (orientador), **DIOGO DANTAS MOREIRA** (examinador) e **PAULO EWERTON GOMES FRAGOSO** (examinador).

Após a apresentação e as observações dos membros da Banca Examinadora, ficou definido que o trabalho foi considerado **APROVADO** com nota **80,** com a condição de que o (a) aluno (a) entregue, no prazo máximo de 30 dias, a versão final do trabalho, via processo eletrônico à coordenação de curso. A versão deve conter a ficha catalográfica e atender às sugestões feitas pelos membros da banca. O código fonte desenvolvido no trabalho (caso haja) deve ser enviado para o e-mail da coordenação do curso (cads.cz@ifpb.edu.br).

Cajazeiras-PB, 4 de fevereiro de 2021.

Documento assinado eletronicamente por:

- **Cicero Rudan de Lucena Nascimento, ALUNO (201512010340) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 03/03/2021 09:53:19.
- **Diogo Dantas Moreira, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 08/02/2021 09:50:21.
- **Paulo Ewerton Gomes Fragoso, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 05/02/2021 15:59:39.
- **Francisco Paulo de Freitas Neto, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 04/02/2021 15:35:05.

Este documento foi emitido pelo SUAP em 04/02/2021. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 154825
Código de Autenticação: 165540a34c

Dedico este trabalho aos meus pais (José Elias e Maria Socorro), que sempre me apoiaram, me incentivaram e sempre estiveram ao meu lado dedicando sempre amor e carinho, e que apesar de todas as dificuldades nunca mediram esforços para que meus irmãos e eu pudéssemos estudar.

AGRADECIMENTOS

Agradeço primeiramente a Deus por mais um trabalho realizado e a toda minha família que sempre me apoio e incentivou minha carreira. Agradeço também ao meu orientador Paulo Freitas por todo apoio e dedicação na construção desse projeto.

RESUMO

Milhares de alunos da rede publica de ensino podem contar com os restaurantes universitários, esses por sua vez são custeados com recursos públicos federais e são destinados para alunos em vulnerabilidade sociais. Manter a organização e controle dos alunos pode se tornar uma tarefa árdua sem as ferramentas adequadas. Desperdício de alimentos, monitoramento de faltas e manejo no acesso dos comensais são as principais dificuldades apresentadas para gerenciar estes estabelecimentos de forma eficaz. Nesse documento apresento uma ferramenta denominada "*Manager RU*", que tem como objetivo disponibilizar funcionalidades que solucionem esses problemas. O Manager RU dispõe de recursos como *checkin* de alunos, solicitação de refeições, cancelamento de refeições, geração de relatório e varias outras funcionalidades que serão abordadas mais adiante. Com essa plataforma espera-se ganhar agilidade nas filas dos refeitórios, evitar o desperdício de alimentos e facilitar na tomada de decisões por parte do setor responsável, como a exemplo ela pode auxiliar na hora de conceder ou remover auxílios de alimentação para os alunos ou informar a quantidade necessária de refeições para um determinado dia.

Palavras-chave: Tecnologia, Refeitório. Desperdício. Angular. Ionic. API. REST.

ABSTRACT

Thousands of students from the public school system can count on university restaurants, which in turn are funded by federal public resources and are destined for students in social vulnerability. Keeping students organized and in control can become a chore without the right tools. Waste of food, monitoring of absences and handling the access of diners are the main difficulties presented to manage these establishments effectively. In this document, I present a tool called "Manager RU", which aims to provide features that solve these problems. Manager RU has resources such as student checkin, meal request, meal cancellation, report generation and several other features that will be covered later. With these platforms, it is expected to gain agility in the rows of cafeterias, avoid the waste of food and facilitate decision making by the responsible sector, as, for example, it can help when granting or removing food aid to students or informing the required amount of meals for a given day.

Keywords: Technology, Refectory. Waste. Angular. Ionic. API. REST.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ADS	Análise e Desenvolvimento de Sistemas
API	*Application Programming Interface*
CSS	*Cascading Style Sheets*
DSDM	*Dynamic Systems Development Methodolog*
FDD	*Feature Driven Development*
HTML	*HyperText Markup Language*
IFPB	Instituto Federal da Paraíba(IFPB)
QR Code	*Quick Response Code*
RF	Requisito Funcional
REST	*REpresentational State Transfer*
RNF	Requisito Não Funcional
RU	Restaurante Universitário
SUAP	Sistema Unificado de Administração Publica
TCC	Trabalho de Conclusão do Curso
UC	Caso de Uso
UFRN	Universidade Federal do Rio Grande do Norte
URL	*Uniform Resource Locator*
UML	*Unified Modeling Language*
XP	*Extreme Programming*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, diversas instituições federais de ensino superior possuem restaurantes universitários (RU) que geralmente são custeados com recursos públicos federais (CARVALHO. et al., 2015). Os restaurantes são destinados aos alunos de baixa renda, com o objetivo de evitar o trancamento de matrícula e o abandono dos cursos, garantindo assim um maior conforto para o discente e consequentemente sua permanência no curso.

Porém, manter a organização desses espaços pode ser uma tarefa difícil pra instituições que possuem uma grande quantidade de alunos e que podem servir milhares de refeições mensalmente.

Algumas instituições fazem a identificação do aluno por meio do Registro Geral (RG) juntamente com algum documento que comprove o vínculo com a mesma, outras fazem o uso de carteirinhas específicas para o refeitório, e em alguns casos, nas instituições que dispõem de mais recursos, pode-se haver o controle de acesso realizado de forma mais sofisticada, como o uso de biometria, por exemplo.

Entretanto, em muitos casos ainda utilizam-se técnicas manuais, como o uso de listas impressas com os nomes dos alunos. Essa, por sua vez, é muito ineficiente e requer um grande esforço para administrá-la, além de influenciar negativamente na agilidade do atendimento.

Além de ser necessário fazer o controle dos alunos nos refeitórios, também é necessário atender as solicitações de alunos que necessitam fazer refeições em períodos que não tenham acesso, e também sempre ter uma estimativa diária dos alunos que farão o uso do RU. Esse controle é fundamental para garantir a eficiência do benefício.

Segundo pesquisas realizadas por Santos (2016), os restaurantes universitários têm um índice de desperdício acima do aceitável. Um estudo realizado por VARELA (2015 apud SANTOS, 2016, p. 25) afirma que na Universidade Federal do Rio Grande do Norte(UFRN), no intervalo de sessenta e um dias, concluiu que eram diariamente desperdiçados em média 142,55 quilos de comida, contabilizando apenas o almoço, onde foram realizadas 136.531 refeições.

1.1 MOTIVAÇÂO

Manter um bom controle sobre esses restaurantes acaba se tornando um grande desafio para essas instituições. Como citado, algumas possuem meios tecnológicos sofisticados para realizarem essas tarefas, porém, muitas delas ainda utilizam técnicas manuais, como a utilização de listas impressas para realizar o controle dos comensais.

A utilização desses meios primitivos dificulta a agilidade e eficiência no controle desses usuários. A identificação por nome realizando buscas manualmente, torna-se desgastante tanto para quem faz esse controle, quanto para os usuários envolvidos, que acabam desperdiçando parte do seu tempo, além de ser suscetível a erros.

Outro problema está relacionado ao acesso esporádico ao restaurante. Em alguns momentos, o setor responsável por conceder acesso temporário pode receber uma grande quantidade de requerimentos, ocasionando assim tumulto no local. Os alunos também podem encontrar dificuldade em justificar suas faltas com antecedência e isso pode gerar desperdício de alimentos.

Esses problemas citados foram observados no Instituto Federal da Paraíba (IFPB) campus Cajazeiras e se tornaram a motivação da construção da ferramenta. Atualmente, o controle do refeitório é feito com base em uma lista fixa, que contém o nome de todos os bolsistas do programa auxílio alimentação e outras listas adicionais que são enviadas ao refeitório quase diariamente, com os nomes dos alunos que não são bolsistas, mas necessitam utilizar o RU de forma temporária.

Com base em pesquisas realizadas durante a fase de levantamentos de requisitos do sistema, foi notado que os alunos não justificavam suas faltas quando deixavam de usar o RU em um período que tinham alguma refeição agendada, e eventualmente com um grande número de faltas, o índice de desperdício poderia ser elevado.

1.2 OBJETIVOS

1.2.1 Objetivo Gerais

Auxiliar o IFPB campus Cajazeiras a gerenciar o restaurante universitário

1.2.2 Objetivos Específicos

- Diminuir o tempo de espera na fila do RU.

- Evitar aglomerações de alunos no setor da CAEST (alunos que necessitam de auxilio temporário)
- Conscientizar os alunos sobre o desperdício de alimentos notificando sobre suas faltas.

1.3 ATIVIDADES

Para o desenvolvimento da ferramenta e elaboração deste documento, algumas atividades foram realizadas a fim de explorar o domínio da aplicação e manter um registro formal de como o sistema foi desenvolvido. São elas:

A1 - levantamento de requisitos: é uma das partes mais importantes do desenvolvimento de *software*, através dessa etapa passa-se a entender melhor o que o cliente realmente deseja que seja desenvolvido. Através de algumas reuniões com o professor orientador, os funcionários do refeitórios e a assistente social, além de uma pesquisa que foi feita com os alunos do campus, foi elaborado o documento de requisitos com as funcionalidades necessárias para alcançar os objetivos propostos. A pesquisa consistia em um formulário impresso onde os alunos informavam os principais problemas com o RU, assim como podiam sugerir melhorias.
A2 - definição dos casos de uso: os casos de uso descrevem como os usuário farão uso das funcionalidades do sistema e como essas funcionalidades interagem entre si. Em notações mais modernas faz-se o uso da Unified Modeling Language(UML) para representar os elementos e seus relacionamentos presentes em um sistema (RODRIGUES, 2014).
A3 - prototipagem: os protótipos servem como modelo visual para a construção das telas da aplicação, são mais rápidos e econômicos de serem construídos e podem ser utilizados para uma primeira experiência do usuário com o sistema.
A4 - definição da arquitetura do sistema: com base na problemática e nas tecnologias adotadas para a sua resolução, uma arquitetura foi definida para prover funcionalidades de forma eficiente.
A5 - construção do modelo entidade relacionamento: é um modelo conceitual, ele descreve as entidades que fazem parte do domínio da aplicação assim como seus atributos e relacionamentos.
A6 - estudo das tecnologias utilizadas: nessa etapa foram estudadas as tecnologias que foram adotadas para o desenvolvimento do sistema. Essa atividade foi realizada durante todo o trabalho.
A7: criação da lista de requisitos: criação, ordenação e separação das funcionalidades que foram desenvolvidas.
A8: desenvolvimento do modulo administrativo WEB: desenvolvimento das funcio-

nalidades voltadas para os assistentes sociais/nutricionistas.
A9: desenvolvimento da plataforma web (módulo refeitório): desenvolvimento das funcionalidades voltadas para os funcionários do refeitório.
A10: desenvolvimento do aplicativo: desenvolvimento do aplicativo que serão utilizados pelos alunos.
A11: testes e validação: todas as funcionalidades desenvolvidas na atividade 8 (A8), atividade 9 (A9), e atividade 10 (A10) foram devidamente testadas.
A12: elaboração do documento: com base na aplicação desenvolvida foi escrito este documento de monografia para a defesa do TCC II.

1.4 ESTRUTURA E ORGANIZAÇÃO DO DOCUMENTO

Esse documento está organizado em 4 capítulos, facilitando assim a compreensão da problemática, solução e atividades realizadas para alcançar os objetivos. No capítulo 2 encontra-se a fundamentação teórica, onde é detalhado a metodologia de desenvolvimento ágil *scrum* e as tecnologias dos *frameworks angular* e *ionic*, além de detalhes sobre o *Quick Response Code* (QR Code), que foi utilizado no desenvolvimento do sistema. No capítulo 3 é detalhado o sistema *Manager RU*, especificando as atividades realizadas para a construção da aplicação. No capítulo 4 são discutidas as considerações finais, informando os objetivos alcançados, as dificuldades encontradas e os próximos passos almejados para este trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Nesse capitulo são abordados algumas tecnologias e processos utilizados durante o desenvolvimento da aplicação *Manager RU*. É descrito o funcionamento e características que justificam suas escolhas na aplicação do projeto

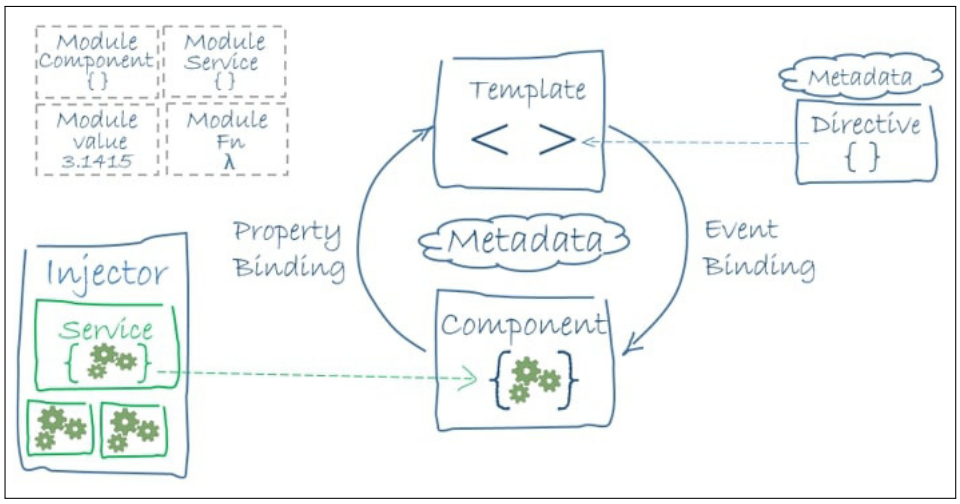
2.1 *FRAMEWORKS*

2.1.1 *Angular*

Atualmente, mantido pela *Google*, o *Angular* é uma plataforma *open source* que utiliza *HTML, CSS*, e *Javascript* para a construção das interfaces de uma aplicação web. Ele adota o conceito de *Single-Page Applications*, isso significa que se trata de uma aplicação web que busca fornecer ao usuário uma experiência similar a de uma aplicação *desktop* (ALEXANDRE, 2018). Na figura 1 são exibidos os elementos do angular, destacando-se os seguintes elementos e funcionalidades

- *Two-way data binding*: o modelo é atualizado sempre que a visão é atualizada, o mesmo acontece com a visão que se atualiza quando o modelo sofre alterações. O próprio Angular se encarrega de manter esse sincronismo de dados.
- Filtros: também denominados *pipes*, são funções para a manipulação de um valor que será exibido. Podendo ser utilizado para valores monetários, datas, números ou textos simples. Ele também permite criar filtros customizáveis.
- Componentes: uma aplicação angular se baseia no uso de componentes, e eles podem encapsular regras da interface, comportamentos e até mesmo encapsular outros componentes.
- Serviços: é onde fica todas as regras de negócio da aplicação. É nos serviços que pode ser feita a comunicação com uma *Application Programming Interface* (API).
- Módulos: ajudam a organizar a aplicação agrupando componentes, serviços e outros elementos.
- Injeção de dependência: é um padrão utilizado para obter-se um menor acoplamento entre nossas classes. Ou seja, ao invés de permitir que o seu componente de software manuseie dependências, você explicitamente passará as dependências que ele necessita. Todas as regras de negócios e tarefas mais complexas são delegadas aos serviços.

Figura 1 – Arquitetura do Angular



Fonte: https://blog.geekhunter.com.br/um-overview-sobre-o-framework-angular/

2.1.2 *IONIC*

Criado no final de 2013 o *ionic* é um *framework open source* gratuito voltado para a criação de aplicações híbridas para dispositivos móveis e *desktop* de forma rápida e de fácil desenvolvimento. Uma de suas vantagens é que ele objetiva-se em criar apps utilizando os recursos mais novos do *HTML, CSS e javascript*. Assim, é possível obter uma gama de componentes pré prontos de alta qualidade e desempenho.

O *ionic* permite o desenvolvimento multiplataforma, a comunicação com os dispositivos móveis acontece através do Córdova, e esse é o responsável por injetar o código *HTML/CSS/JavaScript* na *WebView* do dispositivo, com isso, podem ser criados apps para plataformas diferentes utilizando o mesmo código.

2.2 *QUICK RESPONDE CODE (QR CODE)*

Criado em 1994 pela *Denso-Wave*, uma empresa do grupo *Toyota* o *Quick Response Code* é utilizado atualmente para acesso rápido a sites, textos e números através de um código em 2D que pode ser lido pela maioria dos *smartphones* e *webcams*. Ele é um código visual e isso simplifica sua leitura, podendo estar na forma digital ou impressa. Seu principal objetivo é levar o usuário diretamente a uma página especifica de forma rápida sem a necessidade de inserção de URLs (Uniform Resource Locator) por parte do consumidor. Bastante utilizado atualmente em etiquetas, cardápios e principalmente nas estratégias de marketing, além de várias outras utilidades.

O QR Code possui uma margem que determina o início e o seu fim, nos cantos da imagem existem uma espécie de caixa menor, essas por sua vez determinam o formato do código e também mostram a sua função. Assim, o aplicativo que fizer a leitura poderá identificar se o *QR Code* está no formato de letras ou números e como poderá ser decodificado. Podemos assim codificar um site, rede social um número de telefone, entre outros.

2.3 PROCESSO DE DESENVOLVIMENTO

2.3.1 *SCRUM*

O *scrum* permite fazer a gestão e planejamento de projetos de forma ágil. Ele define papéis e processos que devem ser seguidos para alcançar esse objetivo. Utilizando o *scrum* o *software* é desenvolvido de forma interativa e incremental. As interações são denominadas *sprints*, e os incrementos são as novas funcionalidades realizadas em cada *sprint*. O *scrum* define os seguintes papéis:

- *Product Owner*: é o dono do produto, ele que conhece bem o negócio e transmiti esse conhecimento em forma de requisitos para a equipe, é de sua responsabilidade priorizar os requisitos e solicitar mudanças caso necessário. Deve ser a fonte de consulta para tirar as dúvidas do time de desenvolvimento.
- *Scrum Master*: é o responsável por garantir que os princípios, práticas e valores do *scrum* sejam aplicadas de forma correta e contínua pelos *stakeholders*. Também é responsável por facilitar o trabalho da equipe removendo barreiras que elas venham a enfrentar. Ele deve organizar e facilitar as reuniões e se manter sempre atualizado em relação ao status do projeto.
- Time de Desenvolvimento: normalmente composta entre 6 a 9 pessoas e deve ser auto-organizada. Independente do cargo específico devem trabalhar todos juntos para finalizar a *sprint*. A mesma deve ter autonomia para tomar decisões de como o projeto será desenvolvido e montar a próxima *sprint* de acordo com os requisitos priorizados pelo *product owner*.

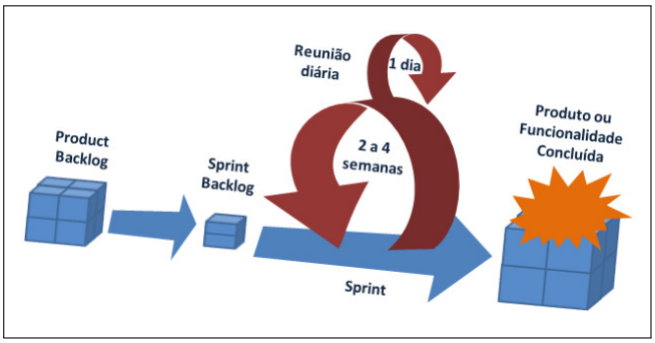
O *scrum* também define os seguintes processos:

- *Product Backlog*: é uma lista contendo todos os requisitos do projeto definidos pelo *product owner*. O mesmo deve ser ordenado de acordo com os requisitos mais importantes que agregam maior valor ao cliente.

- *Daily Scrum*: são reuniões diárias onde cada membro da equipe responde a três perguntas básicas feitas pelo *scrum master*: o que fez ontem? o que vai fazer hoje? a algum impedimento que possa atrapalhar a meta do *sprint*: normalmente essas reuniões são realizadas sempre no mesmo horário e duram cerca de 15 min. O objetivo dessas reuniões é que todos fiquem atualizados do progresso da sprint.
- *Sprint Planning*: é o planejamento da *sprint*, nessa reunião o *product owner*, o *scrum master* e a equipe de desenvolvimento irão construir a *sprint backlog*
- *Sprint Backlog*: é uma lista com todos os requisitos que serão contemplados durante uma *sprint*, esses requisitos são retirados do *product backlog*, cada *sprint backlog* deve ser formada pelos requisitos prioritários do *product backlog*.
- *Sprint Review*: é uma reunião de apresentação de tudo que foi feito na *sprint*, todos os envolvidos no projeto fazem parte dessa reunião e as novas funcionalidades serão avaliadas.
- *Sprint Retrospective*: essa reunião ocorre depois da *sprint review*, o objetivo dessa reunião é avaliar o processo de trabalho durante a *sprint* e caso necessário mudanças deverão ocorrer no processo.

A figura 2 possui um resumo do ciclo de desenvolvimento ágil de *software* utilizando o *scrum*. Através dela podemos observar como o processo é iniciado e os ciclos subsequentes que devem ser seguidos. Inicialmente temos o *Product Backlog* que representa todo o escopo do projeto e será desenvolvido por partes a cada novo clico.

Figura 2 – Ciclo de desenvolvimento ágil com o scrum



Fonte: http://www.mindmaster.com.br/scrum/

2.3.2 Utilização

Durante o desenvolvimento do sistema foram utilizados os principais conceitos presentes no *scrum*. Na etapa inicial foi construído o *product backlog* e em seguida montadas as *sprints backlogs*. A cada semana as funcionalidades selecionadas eram avaliadas e validadas caso estivessem de acordo com o planejado, as mais importantes eram tidas como prioridade e colocadas nas *sprints* iniciais.

Devido ao tamanho da equipe, que era composta apenas pelo discente e professor orientador, bem como pela dinâmica do Trabalho de Conclusão de Curso (TCC), que ocorre juntamente com outras disciplinas, não foi possível seguir todos os processos do *Scrum* como as reuniões diárias e a criação de um *Burndown chart* (gráfico que assegura que os *Sprints* estão sendo finalizados dentro do prazo determinado). Foram utilizados somente os conceitos de *Product backlog*, *sprints* (semanais) e *Sprint Review* no desenvolvimento deste trabalho.

3 FERRAMENTA

Neste capítulo serão abordados todos os processos realizados para a construção do *Manager RU*, detalhando o levantamento de requisitos, elaboração dos casos de uso e criação dos protótipos na parte de análise. Também será mostrada a representação da arquitetura e o modelo Modelo E-R do banco na subsecção de projeto. Por último na subseção de implementação abordo as principais funcionalidades implementadas no sistema, descrevendo as telas e o seu funcionamento

3.1 ANÁLISE

3.1.1 Requisitos

Todo sistema tem seu início apenas após o levantamento de requisitos, é uma fase fundamental no desenvolvimento de *software*. Para a construção de uma aplicação bem sucedida é de extrema importância conhecer o domínio da aplicação, assim como coletar todas as funcionalidades que o cliente deseja. Requisitos mal formulados e que gerem inconsistências podem levar o projeto ao fracasso ou elevar o custo do projeto devido às mudanças que podem ocorrer.

Sommerville (2011) classifica os requisitos de sistema de software como funcionais, não funcionais e como requisitos de domínio:

Requisitos funcionais (RF) definem as funcionalidades do sistema como deve reagir em condições específicas e como se comportar em determinadas situações. Podem ainda declarar o que o sistema não deve fazer.

Requisitos não funcionais (RNF) são restrições sobre serviços ou funções oferecidas pelo sistema. Dentre elas, destacam-se restrições de tempo, sobre o processo de desenvolvimento e de padrões. A descrição das restrições complementa a definição de requisitos (PAULA FILHO, 2000).

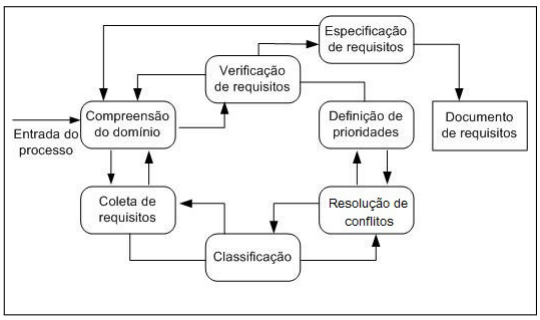
Requisitos de domínio são restrições originárias do domínio da aplicação do sistema e refletem características do mesmo. Podem ser requisitos funcionais ou não funcionais.

Sommerville (2011) propõe ainda, um processo genérico de levantamento e análise que contém as seguintes atividades:

- **Compreensão do domínio:** Os analistas buscam meios para aprofundarem seus conhecimentos sobre o domínio da aplicação. A coleta de requisitos é uma ótima técnica para garantir esse objetivo.
- **Coleta de requisitos:** nesta etapa os requisitos são levantados através de interações com o cliente e outros *stakeholders*.
- **Classificação:** os requisitos são estruturados e organizados em grupos coerentes;
- **Resolução de conflitos:** atividade voltada para solucionar conflitos entre requisitos, isso acontece pois cada *stakeholder* pode ter uma visão diferente de cada requisito.
- **Definição das prioridades:** os requisitos são priorizados de acordo com as necessidades dos clientes. geralmente as funcionalidades que agregam maior valor são desenvolvidas primeiro.
- **Verificação de requisitos:** é observado se os requisitos estão de acordo com as necessidades do usuário, os requisitos devem estar completos e consistentes.

A figura 3 descreve as etapas para definir os requisitos de um sistema. Tendo o objetivo de na etapa final ter uma lista com todas as funcionalidades bem definidas e não conflitantes.

Figura 3 – Processo de levantamento e análise de requisitos



Fonte: (SOMMERVILLE, 2013)

Para o levantamento dos requisitos do sistema, foram coletadas informações através de reuniões com os principais *stakeholders* envolvidos. Foram realizadas entrevistas com os alunos, funcionários do refeitório e assistentes sociais do IFPB campus

Cajazeiras e os seguintes requisitos foram levantados para o desenvolvimento do sistema. Uma descrição mais detalhada desses requisitos, bem como a classificação destes, é apresentada no Apêndice A deste documento.

REQUISITOS FUNCIONAIS (RF)

- **RF01:** gerenciar Alunos.
- **RF02:** conceder auxílio permanente.
- **RF03:** conceder auxílio temporário.
- **RF04:** fazer *check-in*.
- **RF05:** listar comensais.
- **RF06:** relatório Refeições.
- **RF07:** acessar carteirinha.
- **RF08:** imprimir carteirinha.
- **RF09:** cancelar refeição.
- **RF10:** exibir notificação de status do pedido.
- **RF11:** autenticar usuário.
- **RF12:** cadastrar faltas.
- **RF13:** cadastrar funcionários .
- **RF14:** Configurar Períodos .
- **RF15:** Configurar horários .
- **RF16:** Ranking Faltas .

REQUISITOS NÃO FUNCIONAIS (RNF)

- **RNF01:** Multiplataforma.

3.1.2 Casos de uso

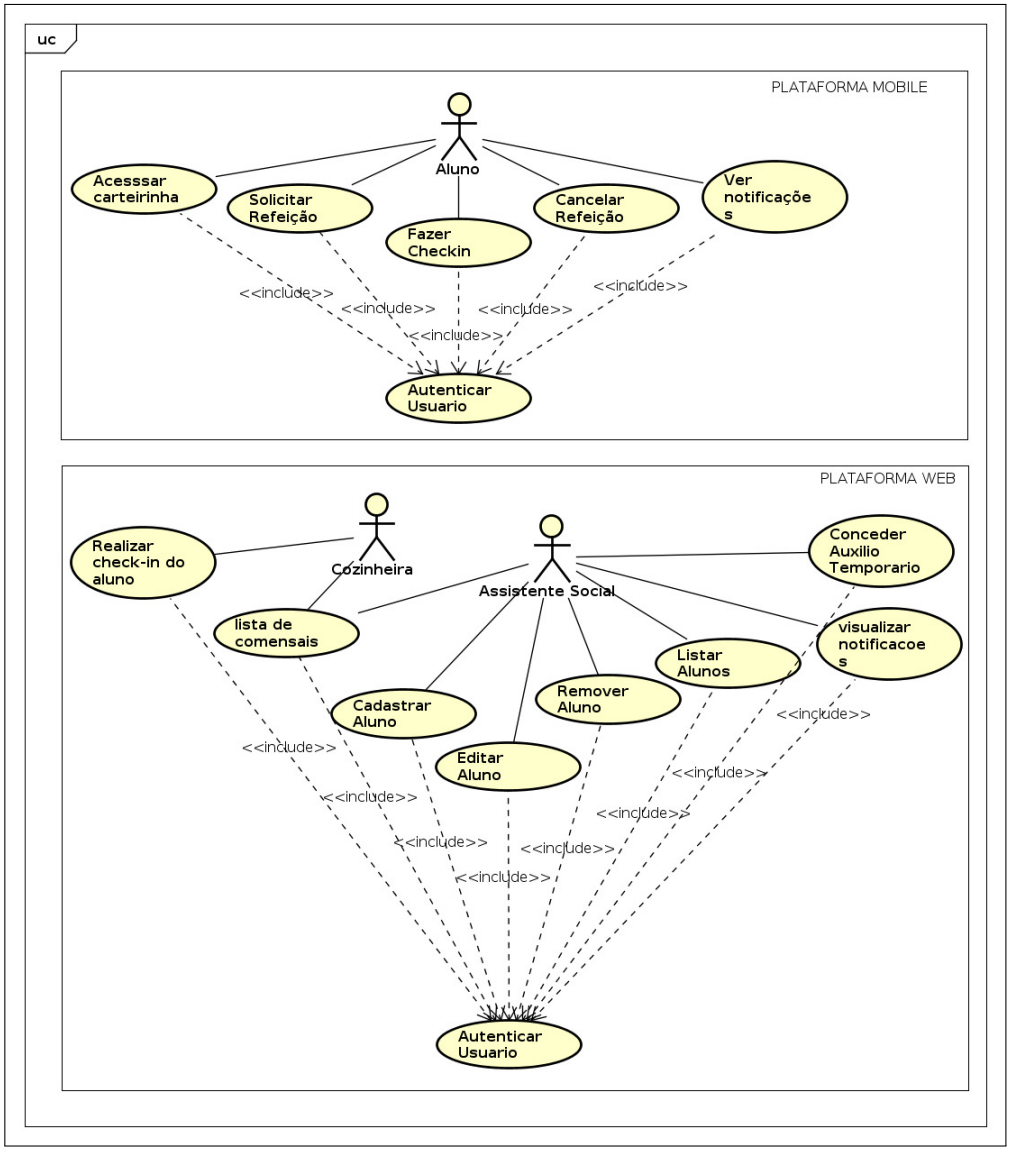
Um caso de uso (UC) tem o objetivo descrever como uma funcionalidade será utilizada no sistema. Existem algumas regras para a criação de casos uso, pois diferentes pessoas podem utilizá-lo para diferentes finalidades. Programadores podem fazer seu uso para desenvolver as funcionalidades e/ou o cliente vai validar esse caso de uso de acordo com o que foi descrito. “Casos de uso fornecem uma abordagem para os desenvolvedores chegarem a uma compreensão comum com os usuários finais e especialistas do domínio, acerca da funcionalidade a ser provida pelo sistema”. (BOOCH. et al., 2006).

As especificações de caso de uso são narrativas em texto que normalmente descrevem os requisitos funcionais do sistema. Podem ser representadas graficamente os casos de uso por meio de um diagrama de casos de uso. Neste diagrama, um caso de uso é representado por uma elipse contendo o nome do caso de uso no seu interior, e os atores são representados por bonecos palito. A ferramenta *Astah* [1] foi utilizada para fazer os diagramas de caso de uso da ferramenta descrita neste trabalho.

Na figura 3 apresenta-se o diagrama de casos de uso da ferramenta que foi desenvolvida. Os casos de uso são separados em dois, visto que, serão desenvolvidos dois módulos: Administrativo (web) e *mobile*. Os principais atores são aluno, assistente social e cozinheira, que representam os funcionários do refeitório.

[1] software de diagramação *UML* Disponível em: https://astah.net/products/astah-uml/

Figura 4 – Diagrama de casos de uso do Manager RU



Fonte: Elaborado pelo autor

Os casos de uso foram descritos na forma narrativa. Como exemplo de um caso de uso, o Quadro 1 apresenta a descrição do caso de uso fazer *check-in*, que se refere ao processo de identificação do aluno por parte dos funcionários do refeitório. Para facilitar a organização do documento, os demais casos de uso encontram-se descritos no apêndice B deste documento.

Quadro 1 – caso de uso para realizar check-in no RU

Nome do Cenário	fazer *check-in*
Descrição	o aluno se identifica no sistema do refeitório através de sua carteirinha.
Ator	aluno.
Pré-condição	o aluno deve estar cadastrado no refeitório de acordo com o dia de acesso.
Fluxo Normal	1. o aluno informa seu *QR code* presente na carteirinha. 2. o sistema verifica se o aluno tem acesso ao refeitório 3. o sistema exibe uma foto e outras informações básicas sobre o aluno.
Fluxo de Exceção	E1: Acesso negado. 1- o sistema exibe a mensagem “este aluno não tem permissão para acessar o refeitório nesse horário”. o sistema fica aguardando outro aluno informar seu *QR code*.
Pós-Condição	1- o aluno tem sua presença confirmada no refeitório. 2- o sistema deverá computar as faltas de todos os alunos que não compareceram

Fonte: Elaborado pelo autor

3.1.3 Protótipos

A prototipagem na engenharia de software serve para validar uma ideia, e consequentemente, diminuir os riscos e investimentos, uma vez que eles permitem inicialmente testar os atributos e os elementos visuais do sistema. Os protótipos são econômicos e rápidos para serem desenvolvidos e permitem uma forma interessante de interação com o cliente, e assim, fornecem um *feedback* do usuário quanto a sua validação. Eles podem ser de baixa ou alta fidelidade, para os de baixa fidelidade é necessário apenas um lápis e papel ou a utilização de alguns aplicativos mais simples. Já os protótipos de alta fidelidade representam melhor a versão final do produto, esses permitem interações entre telas e algumas validações de dados, porém demandam

mais tempo para serem desenvolvidos.

A prototipação é uma técnica que visa construir um protótipo inicial do sistema proposto. É uma versão inicial do sistema de *software* usado para demonstrar conceitos, experimentar opções de projeto, conhecer mais sobre o problema e suas possíveis soluções. Usa-se, normalmente, uma metodologia de desenvolvimento rápido e interativo para que os clientes e/ou usuários possam usar o sistema o mais cedo possível (SOMMERVILLE, 2007).

Para criação dos protótipos da aplicação foi utilizada a ferramenta *moqups*[2], ela é uma aplicação web que possui um plano gratuito e outros planos pagos. A figura 5 apresenta um dos protótipos da aplicação mobile que são destinados aos alunos. Para facilitar a organização do documento, os demais protótipos do sistema encontram-se no apêndice C deste documento.

Figura 5 – Protótipo da Tela onde é exibida a carteirinha virtual do aluno

Fonte: Elaborado pelo autor

[2]Um aplicativo web simplificado que ajuda você a criar e colaborar em tempo real em wireframes, maquetes, diagramas e protótipos. Disponível em: https://moqups.com/

3.2 PROJETO

3.2.1 Representação da arquitetura

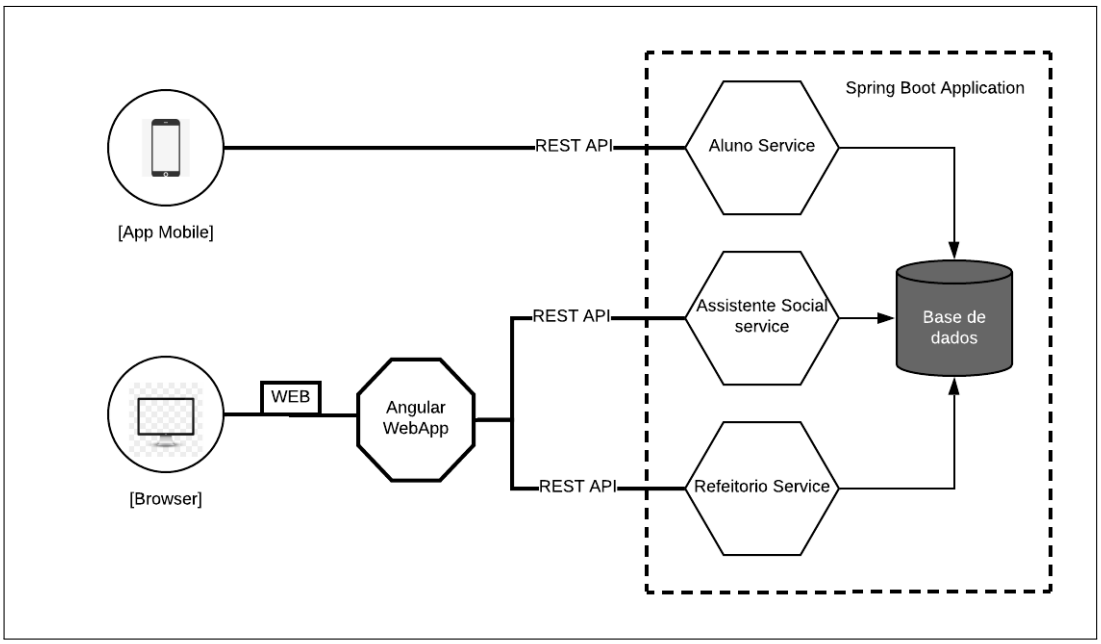
O Manager RU é um sistema que será utilizado por diferentes plataformas e usuários e suas tecnologias foram escolhidas pensando nesse requisito. Foi desenvolvida uma *API REST* monolítica para dar suporte a todas as funcionalidades em ambas as plataformas (web e mobile). O *framework Angular* [3] foi utilizado para a construção do front-end e é o responsável por fazer o consumo dessa API na plataforma web, onde os assistentes sociais, nutricionistas e funcionários do refeitório irão fazer o gerenciamentos dos comensais. Para a aplicação mobile foi utilizado o *framework Ionic* [4], no qual os estudantes podem utilizar o App para acessar o refeitório, solicitar e cancelar refeições.

Na figura 6, pode-se observar um esboço geral da arquitetura do sistema. Ela descreve como cada módulo será acessado e como as tecnologias se comunicam para realizar as funcionalidades. Essa arquitetura fornece um maior desacoplamento entre as plataformas possibilitando alterações em ambas de forma individual com interferência mínima na outra plataforma. Com isso, podemos ter o aluno solicitando recursos para usuários do seu perfil através do *smartphone*, como, por exemplo, acessar sua carteirinha virtual. E usuários com papel de assistente social poderá obter recursos da API através da plataforma web, podendo assim inscrever alunos na lista de bolsistas, a exemplo.

[3] Framework para construção da interface de aplicações usando *HTML, CSS e TypeScript*. Disponível em: https://angular.io/

[4] *Framework Open Source* gratuito sobre a licença MIT para desenvolvimento de aplicações *mobile* híbridas. Disponível em: https://ionicframework.com/

Figura 6 – Arquitetura do Sistema



Fonte: Elaborado pelo autor

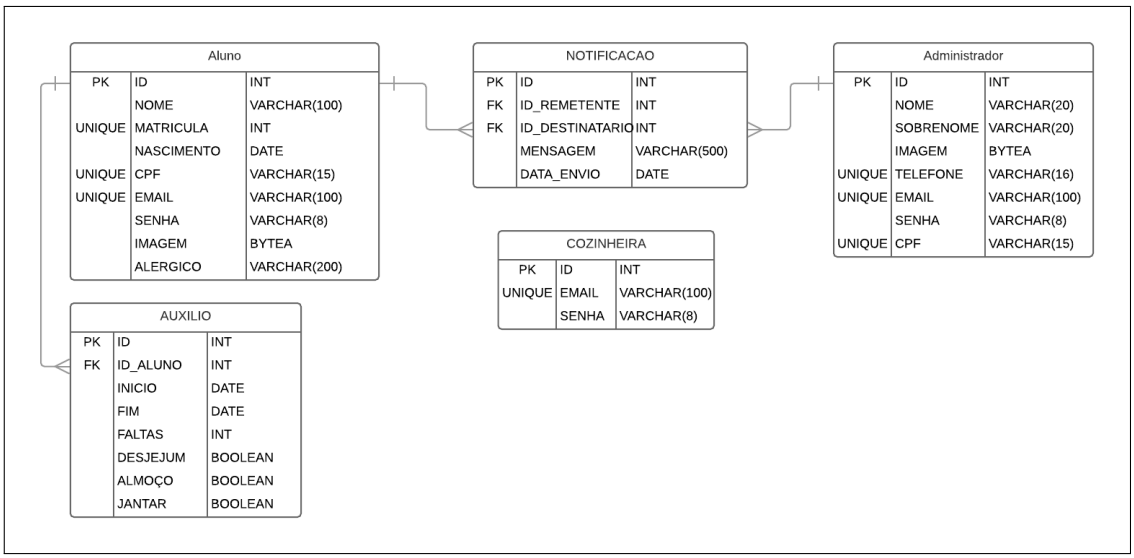
3.2.2 Modelo E-R do banco

O diagrama entidade relacionamento é a representação do modelo conceitual definido no modelo entidade relacionamento. Ele define uma linguagem comum, permitindo uma melhor comunicação entre os membros da equipe. Em notações mais modernas faz-se o uso da *Unified Modeling Language* (UML) para representar os elementos e seus relacionamentos presentes em um sistema (RODRIGUES, 2014).

A figura 7 mostra o diagrama entidade relacionamento do sistema (Manager RU), nela está presente as entidades que são representada pelos retângulos e seus atributos que ficam dentro das entidades, as linhas representam o relacionamento entre os elementos. Como sistema gerenciador de banco de dados (SGBD) foi utilizado o PostgreSQL [5]

[5]Sistema de gerenciamento de banco de dados objeto-relacional. Disponível em: https://www.postgresql.org/

Figura 7 – Modelo Entidade Relacionamento



Fonte: Elaborado pelo autor

3.3 DESCRIÇÃO DA SOLUÇÃO

Visando contribuir na solução dos problemas relatados na seção 1.1, o sistema proposto nesse documento possui funcionalidades voltadas para quatro tipos de usuários: assistentes sociais, funcionários do refeitório, nutricionistas e alunos.

O usuário com perfil de **assistente social** é o responsável por cadastrar todos os alunos que farão uso do RU. Ele pode aceitar ou recusar solicitações de refeições realizadas pelos alunos e poderá configurar um número máximo de faltas permitidas em um mês e verificar os alunos que estão acima desse limite. Também será disponível para este perfil de usuário uma lista contendo o número de comensais separados por turno, que farão uso do RU em uma determinada semana, permitindo um melhor planejamento para a aquisição de alimentos, por exemplo.

Os **funcionários do refeitório** têm acesso a lista de comensais por meio de um módulo do sistema que será implantado no setor. A autenticação dos discentes é feita neste módulo. O *software* fará uso de um leitor de *Quick Response code* (QR Code), permitindo o aluno posicionar seu QR code presente em sua carteirinha ou informar sua matrícula. O sistema irá identificar o aluno e informar se ele está autorizado a utilizar o RU naquele instante.

nutricionistas têm acesso a lista de comensais e demais relatórios presentes

no sistema, como por exemplo, uma estimativa de quantos alunos irão realizar refeições em um determinado intervalo de datas.

Por fim, os **alunos** farão uso do sistema por meio de um aplicativo. Além disso, poderão solicitar a realização de refeições em um período em que não estejam autorizados, como em aulas extras, por exemplo. Ao fazer essa solicitação uma notificação é enviada para o(a) assistente social que poderá aceitar ou rejeitar o pedido. O aluno também receberá uma notificação de acordo com a resposta do(a) assistente social.

Outra funcionalidade disponível para o discente é a de poder cancelar uma refeição agendada, assim seu nome é retirado da lista de comensais do dia em questão. Ele também poderá visualizar o número de faltas que obteve.

A ferramenta desenvolvida é gratuita e poderá ser utilizada por qualquer instituição de ensino que utilize o Sistema Unificado de Administração Pública (SUAP), visto que realiza integração com essa ferramenta. A API do SUAP fornece informações dos docentes e discentes de uma instituição, esses são dados abertos e podem ser obtidos através de requisições HTTP utilizando o padrão REST.

Para obter o aplicativo, os alunos poderão utilizar o *Google Play Store*, caso seja usuário do sistema operacional *Android*, em seguida o aluno deverá preencher suas informações de login e assim poderá obter todos os recursos disponíveis. O app foi desenvolvido para as plataformas *Android* e IOS, porém por questões de recursos, só foi possível publicar na loja *Google Play*. O código fonte do *Backend* [6], *Frontend* [7] e do *Mobile* [8] podem ser baixados no site do Github. Ambos podem ser livremente alterados para se adequar a outros restaurantes universitários ou ganhar novos recursos e melhorias.

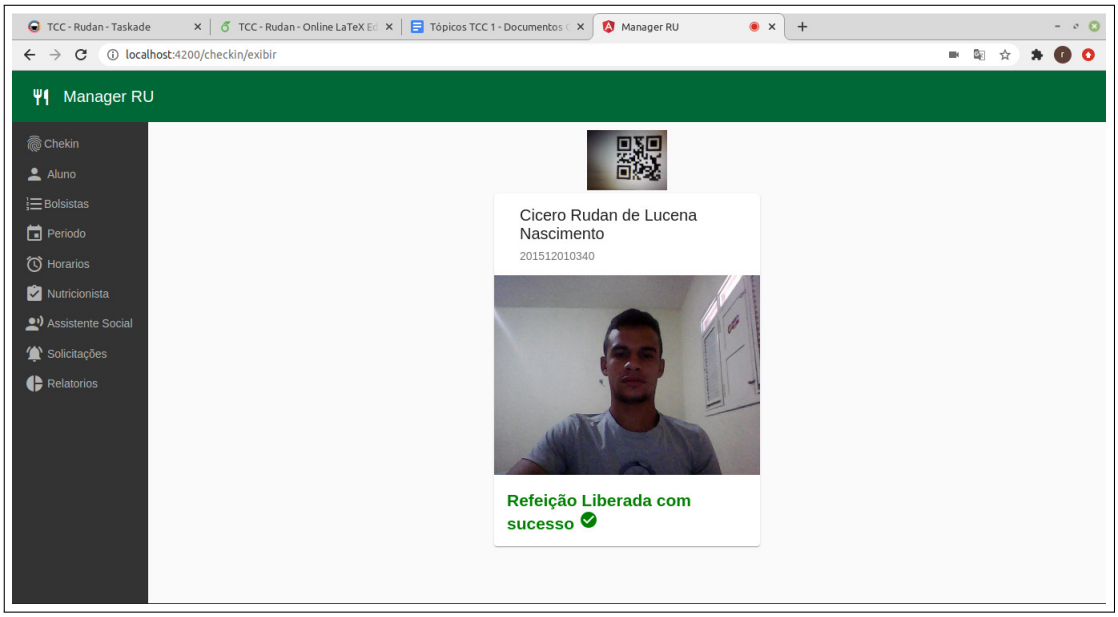
3.4 IMPLEMENTAÇÃO

Pretende-se com a implantação dessa ferramenta gerar economia para o campus em relação ao desperdício de alimentos, possibilitando um melhor planejamento por parte dos funcionários do setor. Além de permitir a gerência dos alunos de forma tecnológica e mais eficiente. O controle dos comensais poderá ser realizado através da tela de *chekin*, onde dado um QR Code informado pelo aluno através de sua carteirinha física ou virtual o sistema busca o aluno e verifica se o mesmo está autorizado a utilizar o refeitório naquele momento, como mostra a figura 8. Assim os responsáveis

[6] Backend disponível em: https://github.com/rudanlucena/ru-back
[7] Frontend disponível em: https://github.com/rudanlucena/ru-front
[8] Aplicativo Mobile disponível em: https://github.com/rudanlucena/ru-app

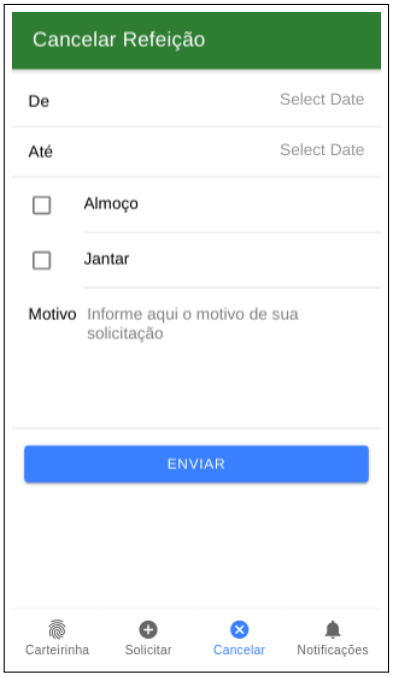
poderiam obter relatórios de alunos presentes e alunos que não confirmaram presença com refeições agendadas. Na figura 9 é mostrada o fluxo para que um aluno possa cancelar uma refeição agendada e assim evitar o desperdício de comida. Para realizar tal operação o discente deve informar a data de inicio e termino que ficará ausente e informar os turnos que não estará presente, ele também deve informar o motivo pelo qual está solicitando este cancelamento. Feito este procedimento o aluno ficará livre de faltas naquele intervalo de tempo.

Figura 8 – Tela de chekin do aluno



Fonte: Elaborado pelo autor

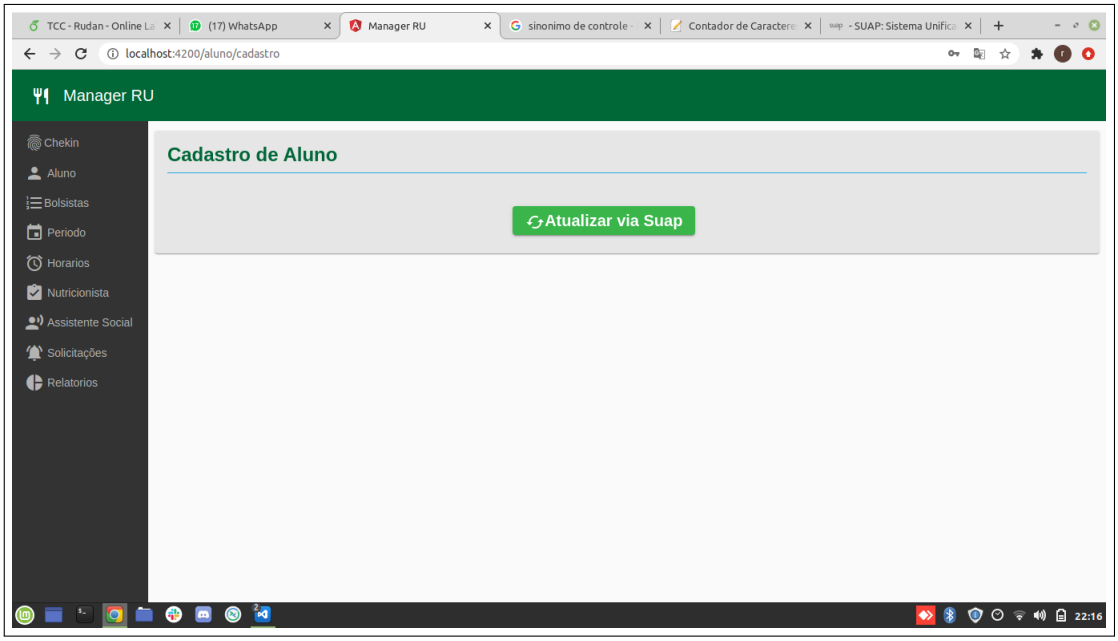
Figura 9 – Tela para cancelar uma refeição



Fonte: Elaborado pelo autor

O cadastro/atualização de alunos pode ser feita automaticamente através da API do Sistema unificado de Administração Publica (SUAP). A operação pode ser realizada necessitando apenas o administrador selecionar a opção "Atualizar via SAUP"para o sistema atualizar sua base conforme os dados fornecidos pela API. Esse processo é mostrado na figura 10.

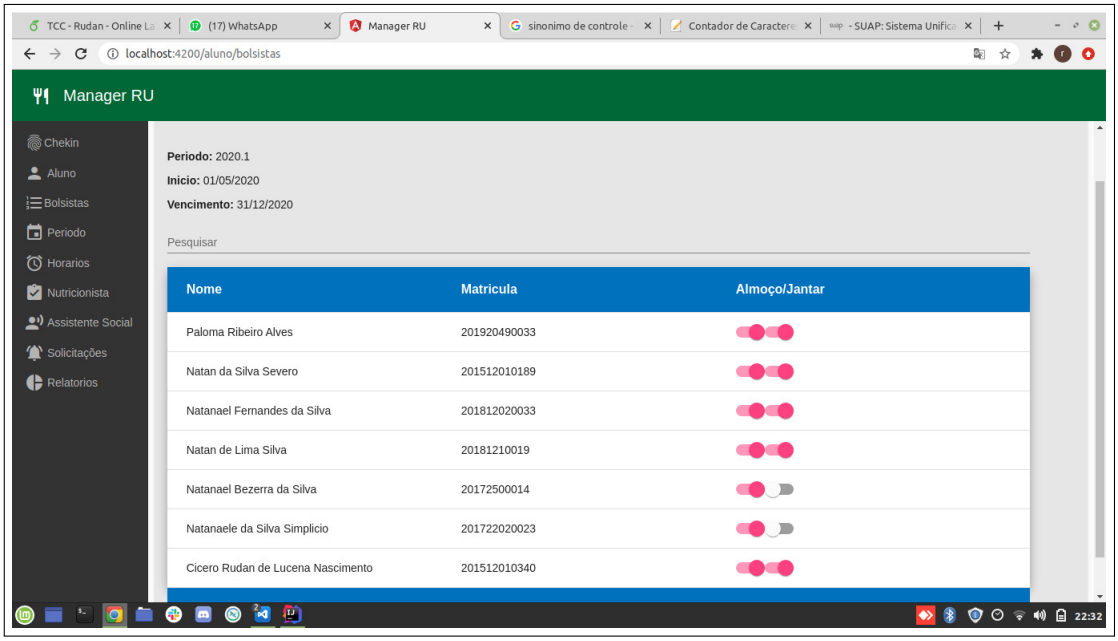
Figura 10 – Tela para sincronizar banco de dados com o SUAP



Fonte: Elaborado pelo autor

Após a sincronização a Assistente social poderá conceder o auxílio alimentação para os alunos que foram contemplados com tal benefício, no sistema ela poderá obter a lista desses usuário através do menu "alunos"e selecionar a opção "bolsistas", na figura 11. é possível observar essa lista.

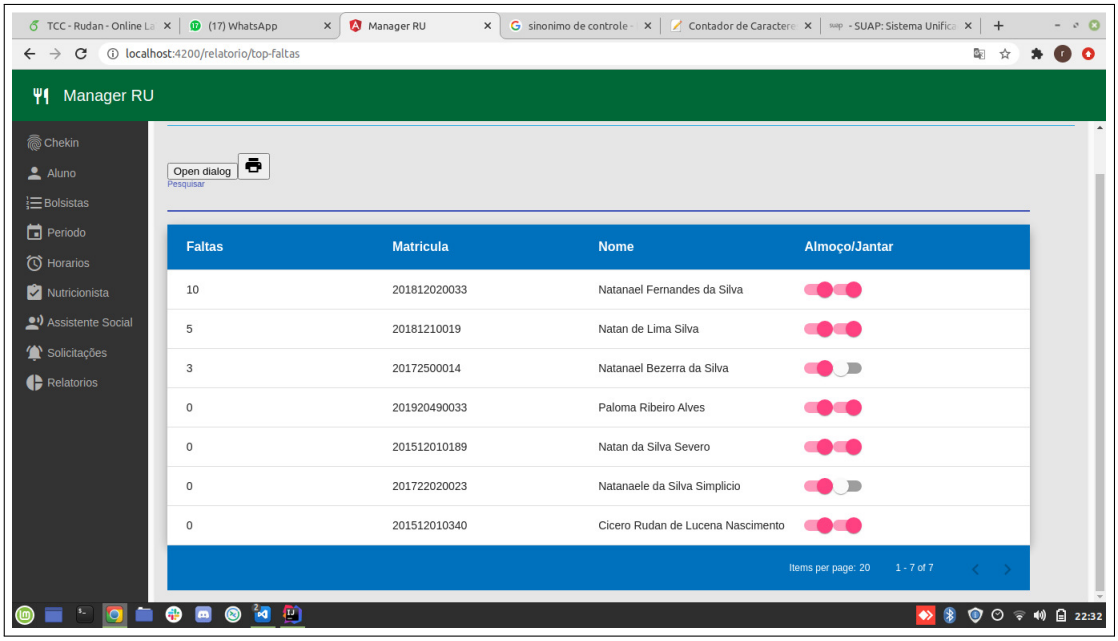
Figura 11 – Tela onde é exibida a lista de bolsistas



Fonte: Elaborado pelo autor

Outra funcionalidade disponível para a assistente social é a de ver a lista de alunos com maior número de faltas como mostra a figura 12. Caso o total de faltas seja maior que o permitido ela poderá remover os auxílios cadastrados para qualquer aluno. Para acessar a lista a assistente social deve clicar na opção "relatórios"e selecionar o item "*Ranking* Faltas".

Figura 12 – Tela onde é exibida a lista de alunos por numero de faltas



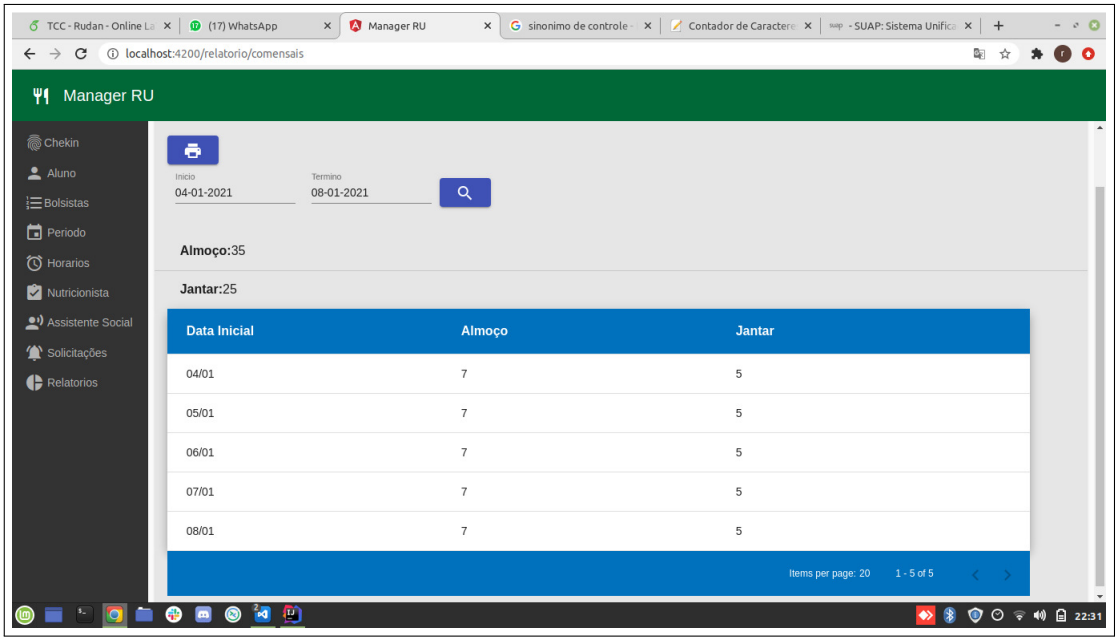
Fonte: Elaborado pelo autor

a figura 13. detalha o relatório de comensais que irão fazer o uso do restaurante em um intervalo de tempo selecionado. Este relatório é a junção dos alunos que possuem auxilio permanente juntamente com os alunos que adquiriram algum auxilio temporário e por último são removidos os alunos que cancelaram as suas refeições em algum dia, tornando o total de refeições a serem realizadas igual o número de comensais. O aplicativo pode ser baixado através da Google Play [9] e uma versão demo da plataforma web está hospedada no servidor do Heroku [10].

[9] Aplicativo disponivel na Google Play no seguinte endereço: https://play.google.com/store/apps/details?id=io.ionic.starter.ManagerRU

[10] Versão web demonstrativa disponível no seguinte endereço: https://front-ru.herokuapp.com/

Figura 13 – Tela onde é exibida a lista de comensais em um tempo determinado



Fonte: Elaborado pelo autor

3.5 TRABALHOS RELACIONADOS

3.5.1 RU UFG

Lançado em 22 de novembro de 2019 o RU UFG é um aplicativo desenvolvido pela Universidade Federal de Goias. O mesmo foi lançado para dispositivos Android e IOS e possui alguns recursos como: conferir o cardápio do dia, avaliar a refeição e ter informações sobre o saldo de créditos. Também será possível substituir a carteira por meio de um QR Code disponibilizado no app. O App utiliza o Portal UFGNet para que os alunos possam se autenticar.

4 CONCLUSÃO

Espera-se que essa ferramenta facilite o trabalho dos assistentes sociais/nutricionistas e funcionários do restaurante universitário do IFPB campus Cajazeiras, além de simplificar e agilizar a forma de acesso e requisições dos alunos. Com a utilização desse sistema também é previsto que o desperdício de alimentos providos dos restos sejam reduzidos, pois o mesmo permite os alunos cancelarem uma refeição com antecedência e também disponibiliza uma lista atualizada dos comensais que farão uso do RU naquele período. Com todos esses recursos implementados os objetivos e requisitos foram alcançados, tornando a plataforma viável para sua implementação.

4.1 TRABALHOS FUTUROS

A ferramenta desenvolvida é de código livre e gratuita, ela poderá ser modificada e utilizada por qualquer instituição de ensino que utilize o SUAP. Pretende-se construir uma plataforma onde seja possível fazer download da aplicação além de disponibilizar tutoriais de utilização e um espaço para receber *feedbacks* dos usuários em geral, permitindo assim sua constante melhoria. O objetivo é evoluir a ferramenta para que possa cada vez mais atender as diferentes necessidades e aperfeiçoar seus principais objetivos que são diminuir o desperdício de alimentos e gerenciar o acesso dos alunos nos restaurantes universitários.

REFERÊNCIAS

ALEXANDRE, A. **O que é angular?** [S.l.], 2018. Disponível em: <https://blog.algaworks.com/o-que-e-angular/>. Acesso em: 22 out. 2018.

BOOCH., G.; RUMBAUGHH., J.; JACOBSON, I. **Engenharia de Requisitos**. [S.l.], 2006.

BRASILEIRO, R. **Manifesto Ágil: o que é e qual a sua história**. [S.l.], 2017. Disponível em: <http://www.metodoagil.com/manifesto-agil/>. Acesso em: 02 jan. 2019.

CAMPOS, D. F. **Por que adotar o Angular para desenvolvimento?** [S.l.], 2017. Disponível em: <https://www.eng.com.br/artigo.cfm?id=5143&post=por-que-adotar-o-angular-para-desenvolvimento?#>. Acesso em: 22 out. 2018.

CARVALHO., M. C. M. V. D. da R.; OLIVEIRA., R. M. A. de; DANTAS, M. G. da S. **O custo dos desperdícios: um estudo de caso no restaurante universitário da Universidade Federal do Rio Grande do Norte**. [S.l.], 2015.

CRUZ, F. **Scrum e Agile em Projetos (2a. edição): guia completo**. [S.l.]: Brasport, 2018.

GENTIL, E. **Introdução ao Spring Framework**. [S.l.], 2019. Disponível em: <https://www.devmedia.com.br/introducao-ao-spring-framework/26212>. Acesso em: 15 jan. 2019.

MATOS, E.; ZABOT, D. **Aplicativos com Bootstrap e Angular: Como Desenvolver Apps Responsivos**. [S.l.]: Saraiva Educação SA, 2020.

RODRIGUES, J. **Modelo Entidade Relacionamento (MER) e Diagrama Entidade-Relacionamento (DER)**. [S.l.], 2014.

SANTOS, J. A. d. **Desperdício de alimentos em restaurantes universitários no brasil**. Universidade Federal do Rio Grande do Norte, Rio Grande do Norte, 2016.

SILVÉRIO., G. de A.; OLTRAMARI, K. **Desperdício de alimentos em Unidades de Alimentação e Nutrição brasileiras**. https://revistas.unicentro.br/index.php/ambiencia/article/viewFile/1587/2220, 2015.

SOMMERVILLE, I. F. **Engenharia de software**. [S.l.: s.n.], 2011.

____. **Engenharia de software**. [S.l.: s.n.], 2013.

UFG. **UFG lança aplicativo para Restaurante Universitário**. [S.l.], 2019. Disponível em: <https://www.ufg.br/n/122209-ufg-lanca-aplicativo-para-restaurante-universitario>.

WEISSMANN, H. **Spring Boot: simplificando o Spring**. [S.l.], 2015. Disponível em: <https://www.devmedia.com.br/spring-boot-simplificando-o-spring/31979>. Acesso em: Acesso em: 20 jan. 2019.

APÊNDICE A – REQUISITOS DO SISTEMA

REQUISITOS FUNCIONAIS:

RF01: Gerenciar alunos - permitir que o usuário com papel de assistente social sincronize os alunos com o SUAP, edite e busque alunos já cadastrados no refeitório.
prioridade: [x] essencial [ ] importante [ ] desejável

RF02: Conceder auxílio permanente - permitir que alunos contemplados no programa auxílio alimentação tenham acesso ao RU.
prioridade: [x] essencial [ ] importante [ ] desejável

RF03: Conceder auxílio temporário - permitir que alunos não participantes do programa auxílio alimentação ou para aqueles que necessitam realizar uma refeição em outro turno que não tenha permissão, recebam um acesso temporário ao refeitório.
prioridade: [x] essencial [ ] importante [ ] desejável

RF04: Fazer check-in - dado um QR code o sistema deverá identificar aquele usuário e informar se ele tem acesso ao refeitório naquele horário.
prioridade: [x] essencial [ ] importante [ ] desejável

RF05: Listar comensais - exibir todos os alunos que possuem auxílio permanente
prioridade: [ ] essencial [x] importante [ ] desejável

RF06: Relatório de refeições - permitir que usuários com papel de assistente social, nutricionistas ou funcionário do refeitório visualizem o total de pessoas que irão utilizar o refeitório durante um determinado intervalo de tempo.
prioridade: [ ] essencial [x] importante [ ] desejável

RF07: Acessar carteirinha - permitir que os alunos acessem suas carteirinhas por meio de um aplicativo.
prioridade: [x] essencial [ ] importante [ ] desejável

RF08: Imprimir carteirinha - permitir que os alunos possam imprimir suas carteirinhas por meio do aplicativo.
prioridade: [ ] essencial [x] importante [ ] desejável

RF09: Cancelar refeição - permitir que os usuários cadastrados no sistema cancelem a sua refeição em uma determinada data e turno ou durante um período e turno específicos.
prioridade: [ ] essencial [x] importante [ ] desejável

RF10 Exibir notificação - permitir que notificações referentes aos pedidos de alimentação sejam exibidos para os interessados conforme seu status muda.
prioridade: [ ] essencial [x] importante [ ] desejável

RF11: Autenticar usuário - apenas usuários autenticados podem realizar as funcionalidades do seu respectivo papel.

prioridade: [x] essencial [ ] importante [ ] desejável

RF12: Cadastrar faltas - permitir que sejam atribuídas faltas automaticamente para os alunos que não comparecerem em alguma refeição que estivesse agendada.

prioridade: [ ] essencial [x] importante [ ] desejável

RF13: Cadastrar funcionários - permitir que o administrador crie novas contas de usuários.

prioridade: [x] essencial [ ] importante [ ] desejável

RF14: Configurar Períodos - permitir que o administrador crie novos períodos, podendo assim associar os auxílios ao período registrado.

prioridade: [x] essencial [ ] importante [ ] desejável

RF15: Configurar Horários - permitir que o administrador configure os horários de abertura e fechamento do refeitório nas respectivas refeições.

prioridade: [x] essencial [ ] importante [ ] desejável

RF16: Ranking de faltas - Exibir em ordem crescente os alunos com mais faltas registradas.

prioridade: [ ] essencial [ ] importante [x] desejável

REQUISITOS NÃO FUNCIONAIS:

RNF01: Multiplataforma - o sistema deve estar disponível na plataforma web para usuários com papel de assistente social e funcionários do refeitório e disponível para usuários com papel de aluno através de uma plataforma mobile.

prioridade: [x] essencial [ ] importante [ ] desejável

APÊNDICE B – CASOS DE USO

Nome do Cenário	cadastrar aluno
Descrição	permite cadastrar novos alunos
Ator	assistente social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. o assistente social clica na opção “sincronizar alunos com SUAP” 2 - O sistema mostra para o usuário o progresso da sincronização
Fluxo de Exceção	E1: API indisponível - o sistema deve exibir uma mensagem informando que não foi possivel sincronizar os alunos com o SUAP.
Pós-condição	1- A base de dados é atualizada com a API do SUAP.

Nome do Cenário	editar aluno
Descrição	permite editar a foto do aluno cadastrado no sistema
Ator	assistente social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. o assistente social clica na opção “listar alunos” 2 - o mesmo deve clicar no botão “alterar imagem” ao lado do aluno desejado: 3 - a câmera do dispositivo é aberta 4 - o botão salvar é pressionado para finalizar a operação
Fluxo de Exceção	E1: upload negado - o sistema deve exibir uma mensagem informando que não foi possivel alterar a imagem naquele momento. o sistema retorna para o passo 3 do fluxo
Pós-condição	1- os novos dados do aluno devem ser atualizados na base de dados.

Nome do Cenário	remover aluno
Descrição	permite remover alunos cadastrados no sistema
Ator	assistente social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. 0 assistente social clica na opção “listar alunos” 2 - o mesmo deve clicar no botão “remover” ao lado do nome do aluno desejado: 3 - para confirmar a exclusão deve-se clicar no botão “remover” na janela de diálogo que irá se abrir
Fluxo de Exceção	não existe
Pós-condição	1- todos os dados do aluno devem ser removidos da base de dados.

Nome do Cenário	listar alunos
Descrição	permite exibir os alunos cadastrados no sistema
Ator	assistente social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. o assistente social clica na opção “listar alunos” 2 - todos os alunos devem ser mostrados em uma tabela com os seguintes dados: -nome -matrícula -imagem -curso
Fluxo de Exceção	E1: lista vazia - o sistema deve exibir uma mensagem informando que não existe nenhum aluno cadastrado ainda .
Pós-condição	1- os dados de todos os alunos ficaram disponíveis.

Nome do Cenário	conceder auxílio temporário
Descrição	permite um aluno receber acesso temporário ao refeitório.
Ator	assistente social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. a assistente social clica na opção “solicitações” 2. todas as solicitações são listadas, podendo cada uma ser aceite ou recusada.
Fluxo de Exceção	E1: pedido vencido - uma mensagem deve ser exibida informando que o pedido não é mais válido
Pós-condição	1 - o aluno que teve seu pedido aceite terá acesso ao refeitório durante o intervalo de tempo que solicitou. 2 - uma notificação deve ser exibida para cada aluno que tiver seu pedido aceito ou negado.

Nome do Cenário	conceder auxílio permanente
Descrição	permite um aluno receber acesso permanente ao refeitório.
Ator	assistente social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. a assistente social clica na opção “listar alunos” 2. O assistente libera o acesso para o aluno desejado.
Fluxo de Exceção	não existe
Pós-condição	1 - o aluno que teve acesso liberado terá acesso ao refeitório durante o intervalo de tempo que o auxílio durar.

Nome do Cenário	fazer check-in
Descrição	o aluno se identifica no sistema do refeitório através de sua carteirinha.
Ator	aluno
Pré-condição	o aluno deve estar cadastrado no refeitório de acordo com o dia de acesso.
Fluxo Normal	1. o aluno informa seu QR Code presente na carteirinha 2. o sistema verifica se o aluno tem acesso ao refeitório 3. o sistema exibe uma foto e outras informações básicas sobre o aluno.
Fluxo de Exceção	E1: acesso negado. 1- o sistema exibe a mensagem “este aluno não tem permissão para acessar o refeitório nesse horário” . o sistema fica aguardando outro aluno informar seu QR Code.
Pós-condição	1- o aluno tem sua presença confirmada no refeitório. 2- o sistema deverá computar as faltas de todos os alunos que não compareceram

Nome do Cenário	listar quantidade de comensais
Descrição	lista os comensais cadastrados em um intervalo de tempo.
Ator	cozinheiras e assistente social.
Pré-condição	o usuário deve estar logado com o perfil de “cozinheira” ou “assistente social”
Fluxo Normal	1. a chef/assistente social clica na opção lista de comensais 2. deve-se informar um intervalo de tempo e turno 3. o sistema exibe o total de alunos cadastrados de acordo com o intervalo e turno selecionado
Fluxo de Exceção	E1: datas inválidas 1- o sistema deve exibir uma mensagem informando que o intervalo de datas selecionado é inválido. o sistema retorna para o passo 2 do fluxo
Pós-condição	1- o total de comensais deve ser exibido de acordo com o turno selecionado.

Nome do Cenário	acessar carteirinha
Descrição	o aluno poderá acessar sua carteirinha virtual que contém informações sobre o aluno.
Ator	aluno
Pré-condição	o aluno deve estar logado no aplicativo
Fluxo Normal	1. o aluno clica na opção “carteirinha”
Fluxo de Exceção	não existe
Pós-condição	a carteirinha virtual é exibida.

Nome do Cenário	solicitar refeição
Descrição	o aluno poderá solicitar acesso ao refeitório por um determinado tempo e turno.
Ator	aluno
Pré-condição	o usuário deve estar logado com o papel de “aluno” .
Fluxo Normal	1. o aluno faz login no sistema 2. o aluno clica na opção “solicitar refeição” 3. o aluno informa um intervalo de tempo e turno que necessita da refeição.
Fluxo de Exceção	E1: intervalo de tempo inválido 1- o sistema exibe a mensagem “informe um intervalo de datas válidas” . o sistema retorna para o passo 3 do fluxo
Pós-condição	1- o pedido é enviado para o setor responsável. 2 - uma notificação é exibida informando a existência de um novo pedido.

Nome do Cenário	cancelar refeição
Descrição	o aluno poderá cancelar sua refeição por determinado tempo e turno.
Ator	aluno
Pré-condição	o usuário deve estar logado com o papel de “aluno” .
Fluxo Normal	1. o aluno faz login no sistema 2. o aluno clica na opção “cancelar refeição” 3. o aluno informa um intervalo de tempo e turno que necessita cancelar a refeição.
Fluxo de Exceção	E1: intervalo inválido 1- o sistema exibe a mensagem “informe um intervalo de datas válidos” . o sistema retorna para o passo 3 do fluxo E2: data refeição inválida: 1- o sistema exibe a mensagem “você não possui refeições registradas para esta data” . o sistema retorna para o passo 3 do fluxo
Pós-condição	1- a refeição do aluno fica cancelada de acordo com o intervalo de tempo selecionado. 2- esse usuário não deverá possuir faltas durante esse período.

Nome do Cenário	Configurar Períodos
Descrição	o usuário com perfil de assistente social deverá configurar o período letivo atual .
Ator	Assistente Social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. o usuário faz login no sistema 2. o usuário clica na opção “Configurar Períodos” 3. o usuário informa o período do ano letivo atual.
Fluxo de Exceção	E1: período inválido 1- o sistema exibe a mensagem “informe um período válido”.
Pós-condição	

Nome do Cenário	Configurar Horários
Descrição	o usuário com perfil de assistente social deverá configurar os horários de abertura e fechamento do refeitório com as respectivas refeições .
Ator	Assistente Social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. o usuário faz login no sistema 2. o usuário clica na opção “Configurar Horários” 3. o usuário informa o horário da respectiva refeição.
Fluxo de Exceção	E1: horários inválidos 1- o sistema exibe a mensagem “informe um Horário válido”.
Pós-condição	

Nome do Cenário	Ranking Faltas
Descrição	este recurso exibirá os alunos bolsistas em ordem crescente de acordo com o total de faltas registradas .
Ator	Assistente Social
Pré-condição	o usuário deve estar logado com o papel de “assistente social” .
Fluxo Normal	1. o usuário faz login no sistema 2. o usuário clica na opção “Ranking Faltas” .
Fluxo de Exceção	
Pós-condição	o sistema exibirá a lista

Nome do Cenário	autenticar usuário
Descrição	cada tipo de usuário poderá se autenticar no sistema
Ator	aluno, assistente social, cozinheiras.
Pré-condição	usuário e senha válidos.
Fluxo Normal	1. o usuário informa o login e senha 2. o usuário clica na opção “entrar”
Fluxo de Exceção	E1: dados inválidos 1- o sistema exibe a mensagem “informe um login e senha válidos” . o sistema retorna para o passo 1 do fluxo.
Pós-condição	1- o usuário é redirecionado para sua tela inicial de acordo com o perfil do usuário.

APÊNDICE C – PROTÓTIPOS

Protótipo da carteirinha virtual do aluno, dados basicos e QR Code de acesso é apresentado nessa tela, conforme mostra a figura 14

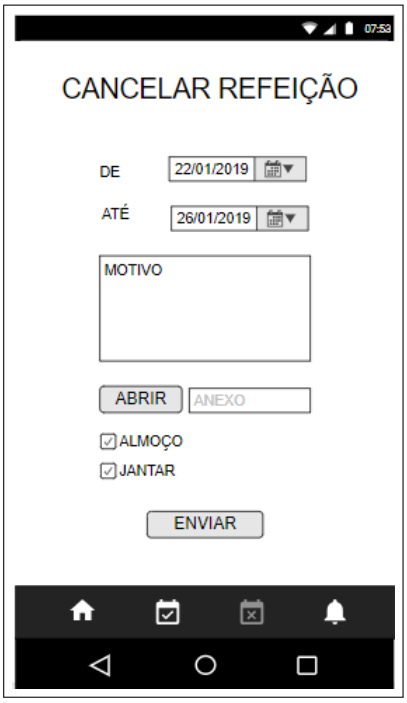
Figura 14 – Tela inicial onde é exibida a carteirinha virtual do aluno.

Solicitar refeição, nessa tela o aluno poderá solicitar uma refeição através do formulário como mostra a figura 15

Figura 15 – tela onde o aluno poderá solicitar uma refeição por um determinado tempo.

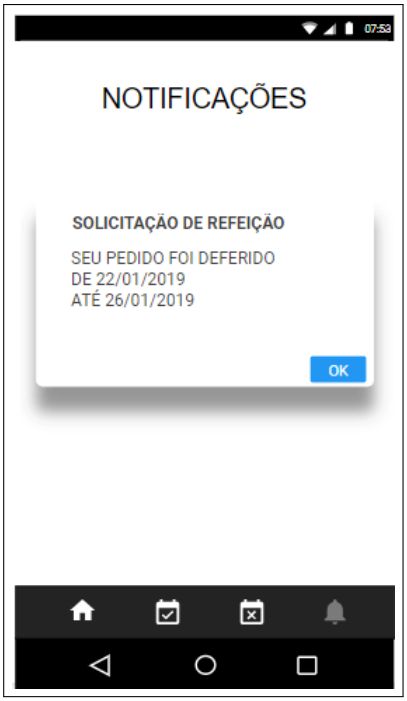
Cancelar refeição, nessa tela o aluno poderá cancelar uma refeição agendada. a figura 16 mostra as informações necessárias

Figura 16 – tela onde o aluno poderá cancelar uma refeição por um determinado tempo.



Tela de notificações, nesta tela o aluno fica informado sobre o progresso das suas solicitações.

Figura 17 – tela onde o aluno poderá visualizar as respostas das suas solicitações



Tela de Chekin, aqui onde o aluno deve informar o QR Code para ter acesso ao RU. Este modulo fica disponível para os funcionários do refeitório.

Figura 18 – tela que será exibida aos funcionários do refeitório quando o usuário faz check-in.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA
Campus Cajazeiras
Rua José Antônio da Silva, 300 - Bairro Jardim Oásis, CEP 58.900-000, Cajazeiras (PB)
CNPJ: 10.783.898/0005-07 - Telefone: (83) 3532-4100

Documento Digitalizado Restrito

Entrega do TCC II

Assunto: Entrega do TCC II
Assinado por: Rudan Lucena
Tipo do Documento: Anexo
Situação: Finalizado
Nível de Acesso: Restrito
Hipótese Legal: Informação Pessoal (Art. 31 da Lei no 12.527/2011)
Tipo do Conferência: Cópia Simples

Documento assinado eletronicamente por:

- **Cicero Rudan de Lucena Nascimento**, **ALUNO (201512010340) DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - CAJAZEIRAS**, em 03/03/2021 11:43:55.

Este documento foi armazenado no SUAP em 03/03/2021. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifpb.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 182702
Código de Autenticação: 5c90c0afab