Universidade Estadual Paulista “Júlio De Mesquita Filho"
Faculdade De Ciências de Bauru
Departamento De Computação
Bacharelado Em Sistemas De Informação

Guilherme Cardoso Fuentes

LightLow: Aplicativo simulador de consumo energético residencial

Bauru
2023

Guilherme Cardoso Fuentes

LightLow: Aplicativo simulador de consumo energético residencial

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação, como parte dos requisitos necessários à obtenção do título de Bacharel em Sistemas de Informação.
Orientador: Prof. Dr. Kleber Rocha de Oliveira

Bauru
2023

F954l

Fuentes, Guilherme Cardoso
LightLow : aplicativo simulador de consumo energético residencial / Guilherme Cardoso Fuentes. -- Bauru, 2022
70 p.

Trabalho de conclusão de curso (Bacharelado - Sistemas de Informação) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências, Bauru
Orientador: Prof. Dr. Kleber Rocha de Oliveira

1. Aplicativos. 2. Computação. 3. Engenharia de Software. 4. Sociedade. 5. Tecnologia. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Ciências, Bauru. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Bauru
2023

Guilherme Cardoso Fuentes

LightLow: Aplicativo simulador de consumo energético residencial

Trabalho de Conclusão de Curso do Curso de Sistemas de Informação da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof. Dr. Kleber Rocha de Oliveira

Orientador
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Engenharia e Ciências
Departamento de Engenharia de Energia

Profª Dra. Márcia A. Zanoli Meira e Silva

Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Prof. Dr. José Remo Ferreira Brega

Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Bauru, _____ de _____ de _____.

Dedico este trabalho à minha família e minha esposa que sempre me apoiaram.

Agradecimentos

Agradeço primeiramente minha família por apoiar meus estudos e me desejarem sucesso. Ensinaram a importância de estudar e trabalhar para conseguir os objetivos na vida através de respeito e educação.

Agradeço também a minha esposa Luana, na qual me inspiro, por desde sempre acreditar que conseguiria passar as adversidades e alcançar meus objetivos.

Agradeço a todos os professores do curso de Sistemas de Informação e de demais cursos, que sempre estiveram dispostos a passar o conhecimento e educar.

Agradeço também aos professores Dr. José Remo Ferreira Brega e a Dra. Márcia Aparecida Zanoli Meira e Silva pela paciência e persistência ao acreditar no meu esforço.

Por fim, agradeço meu orientador pela paciência, perspectivas e experiências que enriqueceram o trabalho.

“Tudo é energia e isso é tudo o que há.”

Albert Einstein

RESUMO

Atualmente, a energia elétrica é essencial para qualquer habitante do planeta e seu uso tem aumentado constantemente. Em paralelo, o seu consumo cresce a cada dia em residências, onde cada vez mais, as famílias adquirem itens que aumentam o seu consumo. Porém, o consumo de energia elétrica possui um custo financeiro que nem sempre é percebido pelos moradores que a utilizam. Além disso, quando percebido, é difícil para o morador levantar quais itens em sua residência, o mesmo, deve utilizar com menos frequência para economizar em sua conta de luz. Sendo assim, este aplicativo proporciona uma solução para enfrentar os problemas citados anteriormente, ao simular aproximadamente o consumo energético da residência, o valor de sua conta esperado para o próximo mês e o item que mais consome energia atualmente em sua residência.

Palavras-chave: Engenharia de Software. Gestão de Energia. LightLow. Aplicativo Móvel.

ABSTRACT

Currently, electrical energy is essential for any inhabitant of the planet and its use has constantly increased. In parallel, its consumption grows every day in homes, where more and more families acquire items that increase their consumption. However, the consumption of electricity has a financial cost that is not always perceived by the residents who use it. In addition, when noticed, it is difficult for the resident to survey which items in their residence, or even, they should use less frequently to save on their electricity bill. Therefore, this application provides a solution to address the aforementioned problems by roughly simulating your home's energy consumption, your expected bill value for the next month, and the item that currently consumes the most energy in your home.

Keywords: Software Engineering. Energy Management. LightLow. Apps.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE IMAGENS

LISTA DE ABREVIATURAS E SIGLAS

ANEEL	Agência Nacional de Energia Elétrica
API	*Application Programming Interface*
BBC	*British Broadcasting Corporation*
CPFL	Companhia Paulista de Força e Luz
DVCS	*Distributed Version control system*
ER	Entidade-Relacionamento
GW	Gigawatts
ICMS	Imposto sobre Circulação de Mercadorias e Serviços
ID	Identificação Digital
JSON	JavaScript *Object Notation*
JWT	JSON *Web Token*
KW	Quilowatt
kWh	Quilowatt-hora
MVC	*Model-View-Controller*
MW	Megawatt
RN	*React Native*
SQL	*Structured Query Language*
SGBD	Sistema de Gerenciamento de Banco de Dados
TE	Tarifa de Energia
TUSD	Tarifa de Uso do Sistema de Distribuição
TW	Terawatt
TWh	Terawatt-hora
URL	*Uniform Resource Locator*
VCS	*Version control system*
W	Watt

SUMÁRIO

1 INTRODUÇÃO

A energia elétrica é a segunda maior fonte energética consumida no Brasil, atrás apenas do petróleo e derivados. O crescimento do consumo de energia elétrica está associado a um movimento global que busca a qualidade de vida e saúde e o desenvolvimento da sociedade. No período entre 2000 e 2019, esse consumo cresceu a uma taxa média geométrica anual de 2,8% no Brasil (Abrahão; Souza, 2020). Apesar de pouco, o período demonstra que o consumo dos brasileiros está em uma crescente com o passar dos anos.

O Brasil oferta em torno de 2,3% da energia global, sendo um total de 636,4 TWh, da qual uma parcela de 35,0 TWh é importada . Sua matriz energética é diversificada e composta principalmente por matriz hidrelétrica (55,3%), gás natural (13,2%) e eólica (11,0%) (Epe, 2022). É possível perceber que o país é dependente de uma única matriz em mais de 50%, no caso, a hidráulica, ou seja, qualquer impacto sobre essa matriz energética afeta o país inteiro na oferta e no custo para o consumidor final. Por ser uma fonte de energia dependente de recursos naturais, as mudanças climáticas e o aquecimento global implicam mais maleabilidade nesse valor e aumenta o risco de os brasileiros passarem por crises energéticas frequentes.

A partir de 2015, a Agência Nacional de Energia Elétrica (ANEEL) começou a adotar o sistema de bandeiras tarifárias na conta de luz, onde elas são os indicadores que mostram o quanto está custando a geração de energia elétrica no país, variando da bandeira verde onde não há acréscimo até a bandeira vermelha II, onde há um alto acréscimo (Santos; Figer, 2022). Essa foi uma forma que o governo brasileiro encontrou para o consumidor reagir ao preço e diminuir a demanda energética ao adaptar-se. Por depender do recurso hídrico, as crises hídricas estão amplamente relacionadas às crises energéticas e elas são frequentes, sendo a última no ano 2021.

Diante desse cenário, o consumidor é obrigado a reduzir o seu consumo ou gerar a própria energia elétrica, como, por exemplo, a energia fotovoltaica, conhecida popularmente como energia solar, e mesmo assim, em ambos os casos, é procurada a economia de energia. Ao se deparar com esse problema, o morador residencial procura primeiramente identificar certos equipamentos em sua casa que

consomem alta quantidade de energia elétrica para poder reduzir o seu uso e conseguir economizar, mas nem sempre consegue diminuir com eficiência a sua conta de luz no final do mês. O morador possui apenas dados totais em relação ao seu consumo energético, como por exemplo, o valor de kWh por mês. Caso queira ser mais assertivo, certamente, ele irá buscar dados específicos de cada equipamento, como a potência em *watts* e o período em que o mesmo está ligado, para assim, poder ver quais itens estão gastando mais energia em sua casa.

Existem dois meios para esse morador listar e calcular, o primeiro seria o manual, com caneta e papel, mas logo ele irá perceber que calcular item por item poderá levar um grande tempo e o segundo seria buscar uma ferramenta tecnológica que facilitaria para ele os cálculos, mas logo notaria que as ferramentas tecnológicas existentes são muito complexas e pouco objetivas, levando-o novamente a um grande tempo para se ter um resultado preciso, e assim, fazendo-o perder o interesse em economizar. O morador está buscando por uma tecnologia que mostre a ele os equipamentos que mais consomem energia elétrica em sua residência, não necessariamente precisa ser preciso, pois o que o mesmo deseja é poder reduzir o uso somente daqueles equipamentos elétricos que impactam de forma significativa o valor.

Diante do problema levantado, este documento apresenta o desenvolvimento do aplicativo *LightLow*, de forma traduzida, Luz Baixa, onde o seu propósito é simular aproximadamente quantos kWh por mês o usuário está consumindo, o item que mais consome e o valor em reais da próxima conta do mês através de um breve questionário respondido pelo usuário sobre os itens que mais consomem energia elétrica em geral nas residências. Através disso, o morador é instigado a aprender mais sobre os dados de sua conta de luz de sua residência e se o mesmo está economizando corretamente.

A estrutura deste documento é segmentada da seguinte forma: no capítulo 2 é abordado os conceitos de engenharia de software, os cálculos elétricos utilizados no projeto e como é utilizado na conta de energia elétrica, a arquitetura de software utilizada, o versionamento de código e sua importância, além do entendimento e utilidade do banco de dados. No capítulo 3 são abordadas as ferramentas utilizadas divididas nos segmentos *backend* e *frontend*, além disso, aborda-se a tecnologia de versionamento de código, como o *Git* e *Typescript* para tipagens de linguagem dinâmica. No capítulo 4 é mostrado todo o desenvolvimento deste trabalho, como o

levantamento dos requisitos funcionais e não funcionais, a análise dos mesmos, a modelagem da estrutura do *software* e o processo de construção do aplicativo que é exemplificado através das telas do aplicativo e das APIs. No capítulo 5 encontra-se a conclusão do projeto, onde é definido se o aplicativo atingiu o resultado esperado e conseguiu resolver o problema. No capítulo 6, ficam sugestões de melhorias futuras para o desenvolvedor e os clientes do produto.

1.1 Objetivos

1.1.1 Objetivo geral

Criar um aplicativo de uso facilitado para consumidores residenciais de energia elétrica que, a partir de um questionário, obtém os dados dos equipamentos que mais consomem energia na residência desse morador e simula os valores aproximados de sua próxima conta de luz para que o mesmo possa economizar corretamente.

1.1.2 Objetivos específicos

O presente trabalho possui como objetivos específicos:

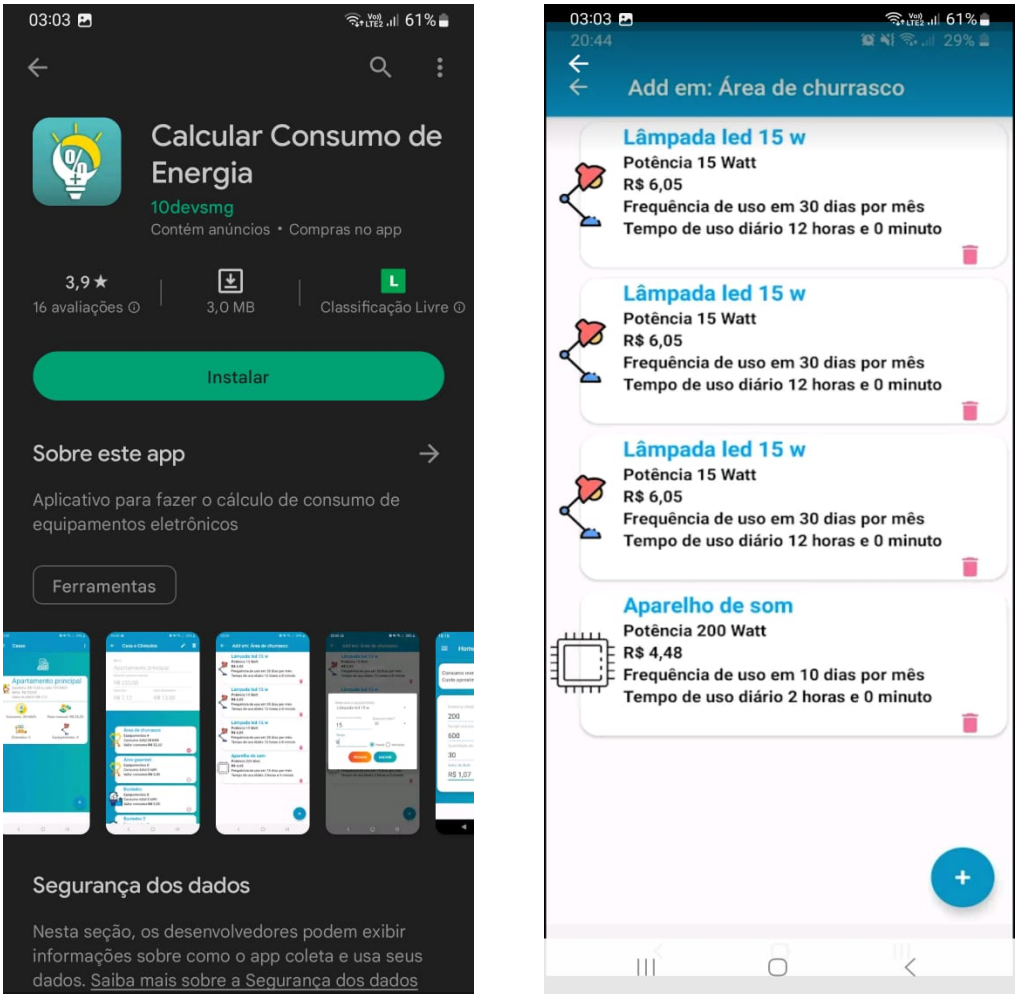
- Desenvolver um aplicativo em *React Native* para ambas plataformas *Android* e *iOS*;
- Desenvolver uma *Application Programming Interface* (API) que comunique o aplicativo com o banco de dados;
- Utilizar técnicas e ferramentas modernas para o desenvolvimento do aplicativo através de tecnologia de ponta utilizada no mercado de trabalho que complementa o conhecimento adquirido no curso;
- Manter uma interface amigável, simples e educativa ao usuário para fácil entendimento de um software com regras complexas;
- Promover o conhecimento sobre consumo energético e economia de energia; e
- Promover assertividade nos itens que mais consomem energia para que o usuário realmente tenha o valor de sua conta de luz reduzido.

1.2 Aplicativos semelhantes

1.2.1 Aplicativo Calcular Consumo de Energia

É um aplicativo que insere item por item como exemplificado na Imagem 1. Uma vez que o usuário cadastre todos os equipamentos elétricos manualmente e crie categorias de acordo com os cômodos de sua residência, o aplicativo retorna um cálculo para sua conta de energia, além de instruir dicas para economizar. Com isso, torna- se trabalhoso para o usuário realizar as ações manualmente.

Imagem 1 - Aplicativo Calcular Consumo de Energia



Fonte: Google Play (2022a)

1.2.2 Aplicativo Calculadora de Energia

É um aplicativo que serve para calcular a energia que cada equipamento elétrico está consumindo na residência do usuário, como é possível verificar na Imagem 2.

Imagem 2 - Aplicativo Calculadora de Energia

Fonte: Google Play (2022b)

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção irá abordar sobre conceitos da computação e como se relacionam diante do projeto, assim como das ferramentas utilizadas e ainda será abordado alguns conhecimentos matemáticos com cálculos relacionados a elétrica.

2.1 Engenharia de software

Segundo Pressman e Maxim (2016), a Engenharia de Software abrange um processo, um conjunto de métodos e ferramentas que possibilitam aos profissionais desenvolverem software de altíssima qualidade, conforme demonstrado na Figura 1.

Figura 1 - Camadas da engenharia de software

Fonte: Pressman e Maxim (2016, pág 16)

Se o projeto conter requisitos sólidos é possível chegar a um nível de qualidade que entregue um software de sucesso, ou seja, a qualidade do projeto está intrinsecamente ligada aos requisitos que o mesmo abrange e atende. Esta seria a primeira camada, o Foco na qualidade. Logo em seguida, constitui-se o pilar para o controle e gerenciamento do trabalho na camada de Processo. A terceira camada, a de Métodos, é a que oferece subsídios técnicos para as tarefas que são desenvolvidas durante o processo e ciclo de desenvolvimento do software. E por último, na quarta camada, tem-se as Ferramentas que fornecem automação e repetição de forma organizada para as atividades do projeto. Para Pressman e Maxim (2016), esses elementos são essenciais para que o projeto de software tenha êxito e sucesso. São essas camadas que esses autores citam em seus livros e são

difundidas para inúmeros projetos no mercado que obtiveram o resultado esperado.

2.1.1 Engenharia de requisitos

Engenharia de requisitos é o processo na engenharia de software que se inicia durante a comunicação e continua na modelagem. O processo de engenharia de requisitos visa garantir que o sistema gerado por esses requisitos atenda adequadamente às necessidades e satisfaça as expectativas dos clientes, segundo Pressman e Maxim (2016):

> A engenharia de requisitos fornece um mecanismo adequado para entender o que o cliente deseja, analisar as necessidades, avaliar a exequibilidade, negociar uma solução razoável, especificar a solução de maneira não ambígua, validar a especificação e administrar os requisitos à medida que eles são transformados num sistema em operação.

Com isso, através da engenharia de requisitos, espera-se que o sistema atenda ao propósito inicial e satisfaça os objetivos esperados pelo cliente. Quanto melhor executado o processo de desenvolvimento maiores serão as chances de se ter um produto final com sucesso através da qualidade que impactou em reduzir o retrabalho e apoiou um desenvolvimento linear e objetivo.

2.2 Cálculos elétricos

Esta seção irá abordar os cálculos e dados específicos que são utilizados no projeto para entregar os resultados aos seus usuários.

2.2.1 Watts

Um *watt* é utilizado como unidade de medida de potência que converte energia elétrica em outra energia por tempo (Casarin, 2022). Por exemplo, watts de energia elétrica é usado para medir quantos deles precisam para acender a lâmpada e gerar luz. Qualquer equipamento elétrico usa W como medida de sua potência e assim é possível saber o quanto de potência ele precisa para funcionar. De acordo Casarin (2022), corresponde abaixo os múltiplos de watts são:

- Quilowatt (KW): corresponde a mil watts.
- Megawatt (MW): corresponde a um milhão de watts.
- Gigawatt (GW): corresponde a um bilhão de watts.

- Terawatt (TW): corresponde a um trilhão de watts.

2.2.2 A conta de energia elétrica

A conta de energia no Brasil é calculada a partir de inúmeras variáveis, indo desde a companhia elétrica que fornece a energia, por exemplo a CPFL, até os impostos cobrados de cada estado como ICMS. Existem também, como citado anteriormente, as bandeiras tarifárias do governo brasileiro e inúmeros outros fatores que impactam a conta final. Este trabalho não detalha em todas as variáveis possíveis para o resultado final para o consumidor, mas utiliza-se dos cálculos principais para chegar ao resultado aproximado. Porém, ainda é necessário entender como funciona o cálculo da conta de energia elétrica que chega até o morador.

O valor da conta é calculado através do quanto o morador consumiu em sua residência de kWh por mês, ou seja, todos os equipamentos elétricos possuem uma potência em W que somados geram o total de kWh por mês e esse valor é multiplicado pela tarifa de sua companhia elétrica que a fornece, além dos impostos e taxas governamentais. Para calcular quantos kWh por mês cada item da casa consome é feito o seguinte cálculo:

$$kWh/mês = \frac{Potência\ em\ W \times horas\ em\ funcionamento \times\ dias\ do\ mês\ em\ uso}{1000}$$

A partir do total kWh por mês gerados na residência, o valor do kWh é multiplicado por duas tarifas que variam de acordo com a companhia elétrica, essas são as TE e a TUSD (Aneel, 2022) e a soma dessas duas multiplicações resulta em seu valor em reais. Posteriormente, é aplicado impostos e encargos do governo que não são detalhados neste trabalho. Entretanto, para obter valores mais próximos dos reais uma média de 20% de impostos é adicionada ao valor total. No final desse cálculo, resta o valor da conta de luz mensal do residente. Com isso, sabe-se como é calculada a conta de energia elétrica do consumidor e os dados necessários para realizar esse cálculo.

2.2.3 Levantamento dos equipamentos elétricos

Sabendo como calcular os equipamentos elétricos e o custo final da conta de luz, é necessário levantar os itens que mais consomem energia nas residências em geral no Brasil. Através da tabela de consumo médio de energia dos equipamentos

elétricos da BBC de Almudena do Cabo (Bbc, 2022), demonstrada na Figura 2, é possível identificar os itens mais relevantes e o total de seu consumo.

Figura 2 - Consumo médio de energia dos equipamentos

Aparelho	Dias de uso	Uso por dia	Consumo médio
Ar condicionado split 12.000 BTU	30	8 horas	193,8
Chuveiro elétrico 5500W	30	32 min.	88
Fogão elétrico cooktop	30	1 horas	68,6
Geladeira 2 portas frost free	30	24 horas	56,9
Lavadora de louças	30	40 min.	30,9
TV em cores 42″ (LED)	30	5 horas	30,5
Ventilador	30	8 horas	17,5
Computador	30	8 horas	15,1
Forno elétrico	30	1 hora	15
Lâmpada incandescente 100W	30	5 horas	15
Secadora de roupa	8	1 horas	14,9
Forno microondas 25L	30	20 min.	14
Aspirador de pó	30	20 min.	7,2
Secador de cabelo 1000W	30	10 min.	5,2
Notebook	30	8 horas	4,8
Torradeira	30	10 min.	4
Lâmpada fluorescente 23W	30	5 horas	3,5
Ferro elétrico a seco 1050W	12	1 hora	2,4
Modem de internet	30	8 horas	1,92
Lavadora de roupas	12	1 hora	1,76
Liquidificador	15	15 min.	0,80
Impressora	30	1 hora	0,45

Fonte: BBC (2022)

Com isso, chegamos ao levantamento total de 16 itens que mais consomem energia elétrica e estão presentes com maior frequência nas residências. Além

disso, como cada equipamento tem sua potência única de consumo e deseja-se um valor aproximado, então é selecionada a potência média em W dos equipamentos mais comuns das residências no Brasil, como ilustrado na Tabela 1.

Tabela 1 - Potência média dos aparelhos elétricos (kWh)

Aparelho Potência	**(W)**
Aparelho de som	200
Aquecedor de ambiente	1.500
Aspirador de pó	1.000
Aquecedor central de água	5.000
Balcão frigorífico	900
Batedeira	450
Boiler 40 litros	900
Boiler 80 litros	1.200
Cafeteira	300
Computador	350
Condicionador de ar	1.600
Chuveiro elétrico	5.000
Enceradeira	350
Exaustor	300
Ferro elétrico Comum	750
Ferro elétrico Regulável	1.500
Forno elétrico	5.000

Forno de micro-ondas	1.300
Freezer acima de 200 litros	150
Freezer até 200 litros	120
Freezer balcão	140
Fritadeira	1.200
Grill	1.200
Impressora jato de tinta	50
Impressora laser	400
Liquidificador	400
Máquina de lavar louça	2.700
Máquina de lavar roupa	1.500
Motor 3 cv/hp	2.200
Motor 4 cv/hp	2.960
Motor 5 cv/hp	3.700
Motor 7,5 cv/hp	5.550
Comum	200
Refrigerador Duplex ou freezer	350
Secador de cabelo	1.300
Secadora de roupa	3.500
Televisor	200
Torneira elétrica	3.500

Ventilador	100

Fonte: Cooperluz (2022)

Para facilitar o preenchimento do questionário, esses itens foram divididos em categorias. A partir dos dados da Tabela 1 o consumo médio de cada um desses itens são fornecidos à seguir:

COZINHA

- *Cooktop* elétrico (1000 W)
- Forno elétrico (4500 W)
- Geladeira (50,8 W)
- Micro-ondas (2000 W)

LAVANDERIA

- Máquina de lavar roupas (1000 W)
- Secadora de roupas (3500 W)

BANHEIRO

- Chuveiro elétrico (5500 W)
- Secador de cabelo (1000 W)
- Torneira elétrica (2500 W)

ELETRODOMÉSTICOS

- Ar condicionado (1400 W)
- Aspirador de pó (600 W)
- Ferro elétrico (1000 W)
- Ventilador (100 W)

ELETRÔNICOS

- Computador (60 W)
- Televisor (120 W)
- Videogame (20 W)

2.3 Arquitetura de software

Consiste em entender como os softwares se comunicam. São construídos e interagem entre si por meio de um conjunto de regras, decisões e designs do sistema. Sendo assim, torna-se mais evidente a divisão e responsabilidade de cada componente do sistema em visão macro geral. Isso é altamente necessário para que o software desenvolvido possua uma ótima qualidade de entrega.

2.3.1 Arquitetura MVC

A arquitetura MVC é composta de três módulos: *model, view* e *controller*. Essa divisão existe para facilitar o desenvolvimento e manutenção do software, pois com módulos isolados um dos outros, suas responsabilidades são mais facilmente geridas, facilitando o desenvolvimento e a correção quando necessário. Os módulos são definidos como:

- *Model* - Sua responsabilidade é gerenciar e controlar a forma como os dados se comportam por meio das funções, lógica e regras de negócios estabelecidas.
- *View* - Representa a visualização dos dados, é a parte que o usuário apenas vê, geralmente apresentando os dados do *model.*
- *Controller* - Onde é realizada a manipulação do *model* ou dos dados por meio de interações que podem vir de outras requisições ou através da view.

É um padrão muito utilizado atualmente no mercado de trabalho sendo a arquitetura utilizada na maioria dos *frameworks* atuais, sejam eles de *backend* ou *frontend*, porque possuem a capacidade de proporcionar facilidades, segurança e eficiência além de criar uma visão macro do projeto.

2.3.2 *Framework*

O termo é muito genérico, porém, uma das formas mais simples de definí-lo é a de uma coleção de classes abstratas, objetos e padrões dedicados a resolver determinados problemas de uma arquitetura flexível e extensível. A principal utilidade de seu uso é a facilidade que entrega aos usuários que a utilizam. Por meio

dele, é possível padronizar o processo do desenvolvimento do trabalho, além de organizar o projeto, atraindo muitos usuários para o seu uso.

Ele é caracterizado por uma grande base de código concreto que o usuário não tem influência, sendo essa a sua estrutura base. Acima dessa estrutura base, o usuário tem flexibilidade sobre o código, quase que um controle total. Dessa forma, o *framework* entrega funcionalidades semi prontas e configuradas para que o desenvolvedor construa em cima delas da forma que almeja, de forma escalada e organizada.

2.4 Versionamento de código

O versionamento de código é essencial para qualquer projeto hoje em dia, pois possibilita através do controle de versões (VCS) uma forma de acompanhar e gerenciar as alterações do seu código. Conforme a evolução do ambiente de desenvolvimento, várias formas de controle de versões foram surgindo, mas a mais utilizada e principal atualmente é o Git (Konnorate et al., 2019), um DVCS (sistema de controle de versão distribuído). Ao utilizar o Git, é possível primeiramente, ganhar produtividade entre os desenvolvedores do projeto, pois podem atuar no mesmo código, na mesma versão e ao mesmo tempo. Posteriormente, há outras inúmeras vantagens, sendo as principais, poder retroceder um código para versão anterior (*rollback)*, trabalhar em linhas do tempo diferentes com as *branches*, salvar as mudanças do projeto até aquele ponto em um repositório remoto através de *commits, push* e baixar as mesmas em qualquer lugar através do *pull*. Sua versatilidade e objetividade faz com que hoje não exista praticamente nenhum projeto de tecnologia sem utilizar algum versionamento de código.

2.4.1 *Github*

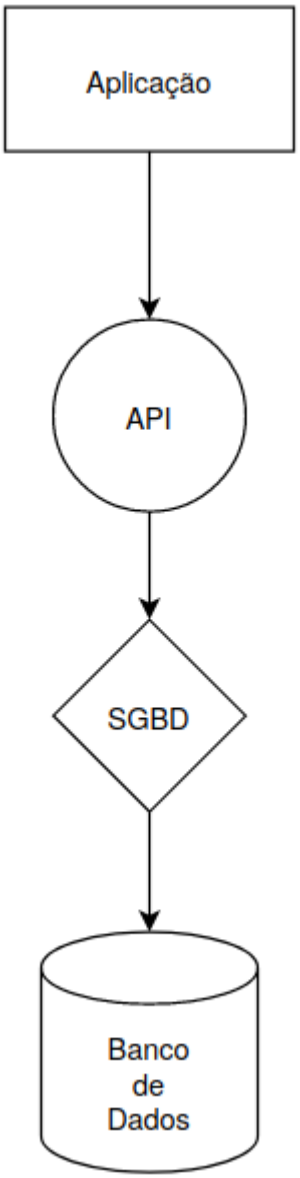
A plataforma do *Github* surgiu como uma rede social para códigos e projetos e ainda de forma gratuita, sendo possível criar repositórios remotos onde pode-se salvar projetos versionados com o Git e acessá-los de qualquer lugar pela internet, além de socializarem com outras pessoas que estão desenvolvendo outros projetos. Apesar de seu uso ser na maioria de desenvolvedores, não especificamente é feita para tal objetivo. Vale ressaltar que qualquer pessoa pode usufruir do versionamento, segundo Andrei (2019):

> O GitHub é uma ótima plataforma que mudou o método de trabalho dos desenvolvedores. Mas qualquer pessoa que deseja gerenciar seu projeto com eficiência e trabalhar com outros colaboradores também pode usar o GitHub.

2.5 Banco de dados

Como citado por Korth e Silberschatz (1994), um banco de dados “é uma coleção de dados inter-relacionados, representando informações sobre um domínio específico“. Com isso, dados agrupados que têm relação e tratam do mesmo assunto pode ser considerado um banco de dados. Exemplificando, desde uma simples lista de mercado até um sistema complexo como uma lista telefônica. Neste contexto, há também um Sistema de Gerenciamento de Banco de Dados (SGBD), um software gerencial que possui ferramentas capazes de manipular as informações do banco de dados e interagir com os usuários. Alguns exemplos são: Sqlite, Oracle, SQL Server, PostgreSQL, Cassandra, MySQL, MongoDB, e Firebase. É mostrada a partir da Figura 3, o relacionamento de uma aplicação com o SGBD.

Figura 3 - Fluxo de uma aplicação, onde o banco de dados é gerenciado pelo SGBD



Fonte: Elaborado pelo autor

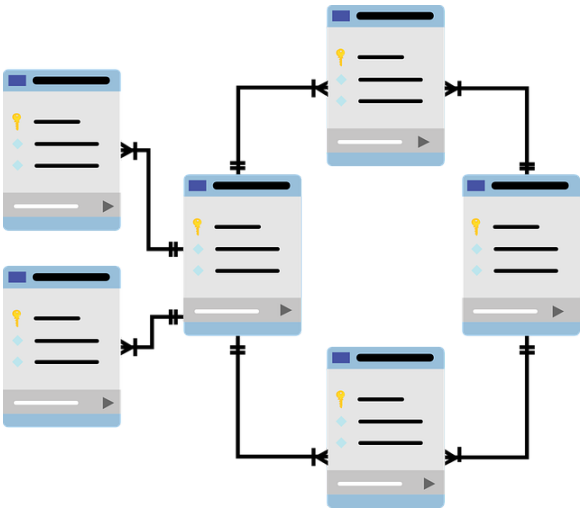
2.5.1 Banco de dados relacional

De acordo com a Oracle (2022), é um tipo de banco de dados que armazena e fornece pontos de acesso a pontos de dados relacionados entre si. Bancos de dados relacionais são baseados no modelo relacional, uma maneira intuitiva e direta de representar dados em tabelas. Em um banco de dados relacional, cada linha na tabela é um registro com uma ID exclusiva chamada chave. As colunas da tabela contém atributos dos dados e cada registro geralmente tem um valor para cada atributo, facilitando o estabelecimento das relações entre os pontos de dados.

Sua estrutura de dados lógicos são: tabelas de dados, exibições e índices que são separadas das estruturas de armazenamento físico. Essa separação significa que os administradores de banco de dados podem gerenciar o armazenamento de dados físicos sem afetar o acesso a esses dados como uma estrutura lógica. Por exemplo, a renomeação de um arquivo de banco de dados não renomeia as tabelas armazenadas nele. A distinção entre lógico e físico também se aplica às operações do banco de dados. São ações claramente definidas que permitem aos aplicativos manipular os dados e as estruturas do banco de dados. As operações lógicas permitem que um aplicativo especifique o conteúdo necessário e as operações físicas determinam como esses dados devem ser acessados e, em seguida, executa a tarefa. Para garantir que os dados sejam sempre precisos e acessíveis, os bancos de dados relacionais seguem determinadas regras de integridade. Por exemplo, uma regra de integridade pode especificar que linhas duplicadas não são permitidas em uma tabela para eliminar o potencial de informações errôneas que entram no banco de dados.

Os mais comuns usados são o SQLite, PostgreSQL, MySQL, SQL Server. Neste projeto, foi escolhido o SQLite, pois sua configuração é relativamente fácil e rápida (bem otimizado) comparada aos outros, além de ser compacto e pequeno. A Figura 4 mostra um banco relacional:

Figura 4 – Banco de dados relacional

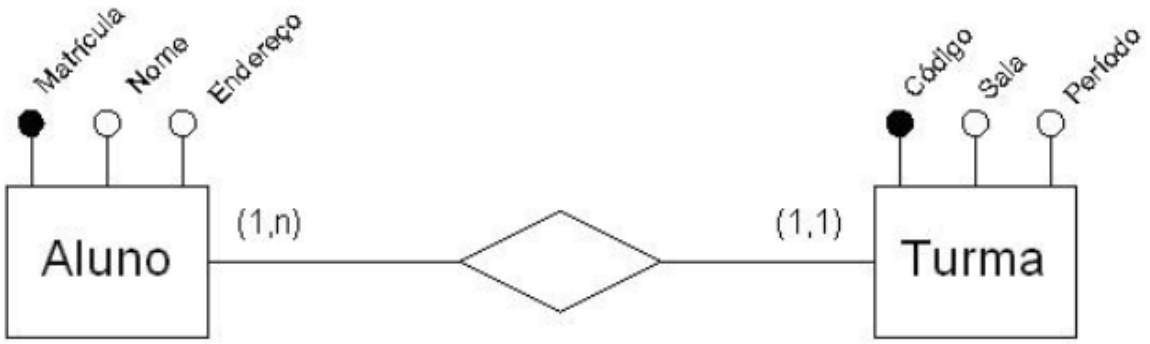


Fonte: Programadores Brasil (2020)

2.5.2 Projeto de banco de dados

Como Rezende (2006) cita, “O projeto de banco de dados se dá em duas fases: Modelagem conceitual; Projeto lógico.”. O Modelo Conceitual é uma representação do banco de dados sem se preocupar qual ferramenta será realmente usada como SGBD, o que permite desenhar todo o banco de dados de uma forma universal para decisão posterior do SGBD. Para isso, é frequentemente utilizado a abordagem entidade-relacionamento (ER), onde o modelo é representado graficamente através do diagrama entidade relacionamento (DER) representado na Figura 5.

Figura 5 - Mapeamento do modelo de classes



Fonte: Ricardo Rezende (2020)

Como se pode observar, é informado quais dados serão armazenados para cada entidade. Por exemplo, para cada aluno será armazenada sua matrícula, seu nome e endereço. Já o modelo lógico descreve o banco de dados no nível do SGBD e é totalmente dependente de qual SGBD será utilizado, que pode ser relacional, orientado a objetos, hierárquico, entre outros.

3 FERRAMENTAS

Nesta seção serão descritas as ferramentas de desenvolvimento de softwares utilizadas na construção do aplicativo.

3.1 Git

Como mencionado anteriormente, a ferramenta Git é o controle de versão mais utilizado em ambientes profissionais e é usado no projeto para facilitar a manutenção do código fonte e seu rápido acesso ao mesmo, permitindo acompanhar o projeto com total controle, gerenciá-lo e visualizar o histórico das mudanças.

3.1.1 *Github*

Igualmente mencionado, essa ferramenta permite que o código do aplicativo esteja armazenado em uma plataforma gratuita e pode ser acessível de qualquer lugar. Os repositórios do projeto estão nos seguintes links:

Aplicação onde está o código do aplicativo:

- https://github.com/guifuentes8/tcc

Aplicação onde está a API:

- https://github.com/guifuentes8/tcc-api

3.2 *Backend*

Esta seção aborda todas as ferramentas que são dedicadas ao desenvolvimento por trás da aplicação, na qual não tem contato com o usuário diretamente de forma visual, mas trabalha em toda a conexão e registro das informações para exibir no aplicativo.

3.2.1 Banco de dados SQLite

Este é o banco de dados utilizado no projeto. Atualmente se encontra na versão 3 e foi escolhido por ser de fácil configuração, otimizado e leve. Totalmente desenvolvido a partir da linguagem C que implementa uma *engine database* SQL. Lançado há 22 anos, em 17 de agosto de 2000, já apresenta uma solidez robusta no mercado, além de não pertencer a nenhuma empresa específica e ser público (Souza, 2020). Ele trabalha na forma padrão de um banco relacional, realizando relacionamentos entre dados, índices e tabelas.

3.2.2 Node.js

O Node.js é um *runtime* que surgiu através da linguagem de programação Javascript, sendo essa, apenas destinada ao ambiente do navegador de internet. Com a evolução, a linguagem Javascript avançou em diferentes segmentos e no *backend*, foi como Node.js. Como cita Pessôa (2022):

> Ele dá muito certo com os servidores de arquitetura "*single threaded*", isto significa que todos os pedidos para o servidor são executados no mesmo tópico - em vez de serem gerados em processos separados. Um dos grandes diferenciais da parceria Node.JS e Javascript é o bom desempenho no uso de APIs, já que o Javascript faz bastante uso de APIs assíncronas.

Sua escolha foi realizada, por ser escrito na linguagem Javascript e o autor deste projeto já possuir conhecimento na mesma, facilitando o desenvolvimento. Além disso, ele possui outras características que fazem sentido para o aplicativo, citadas pela Pessôa (2022):

> Multiplataforma: permite criar desde aplicativos desktop, aplicativos móveis e até sites *SaaS*;
>
> Multi-paradigma: é possível programar em diferentes paradigmas, como: Orientado a Objetos, funcional, imperativo e dirigido à eventos;
>
> *Open Source*: é uma plataforma de código aberto, isso significa que você pode ter acesso ao código fonte do Node.JS e realizar suas próprias customizações ou mesmo contribuir para a comunidade de forma direta;
>
> Escalável: Node.JS foi criado para construir aplicações web escaláveis, como podemos ver na sua documentação oficial.

O Node.js foi utilizado para todo o controle dos dados do *backend*, sendo o responsável pela comunicação com o banco de dados, o recebimento de requisições do *frontend* e o retorno das respostas que o cliente requisitou, como por exemplo, os cálculos elétricos.

3.2.3 Knex.js

Knex.js é um SQL *Query Builder* para Node.js, ou seja, é um construtor que permite se conectar com o banco de dados e gerenciar todas as instruções SQL de forma programática e independente de banco de dados. Sua escolha foi necessária, pois comparada a escrever instruções SQL à mão, usar o Knex.js ajuda a escrever um código SQL relacional mais legível e a gerar declarações mais seguras. Por exemplo, para fazer uma inserção ou deleção no banco de dados, deve-se descrever a *query* usando o knex.js como *query builder*, como mostra a Figura 6 e executar a *migration* na função *up,* que registra em um histórico essa mudança no caso de precisar reverter. Caso precise reverter, utiliza-se o comando para a *migration down* e a operação é revertida. Isso garante maior estabilidade, organização e gerenciabilidade do banco de dados.

Figura 6 - Criando uma migration com knex.js

```
exports.up = (knex) ⇒
  knex.schema.createTable("category", (table) ⇒ {
    table.increments("id");
    table.text("name").notNullable();
    table.timestamp("created_at").default(knex.fn.now());
    table.timestamp("updated_at").default(knex.fn.now());
  });

exports.down = (knex) ⇒ knex.schema.dropTable("category");
```

Fonte: Elaborado pelo autor

3.3 *Frontend*

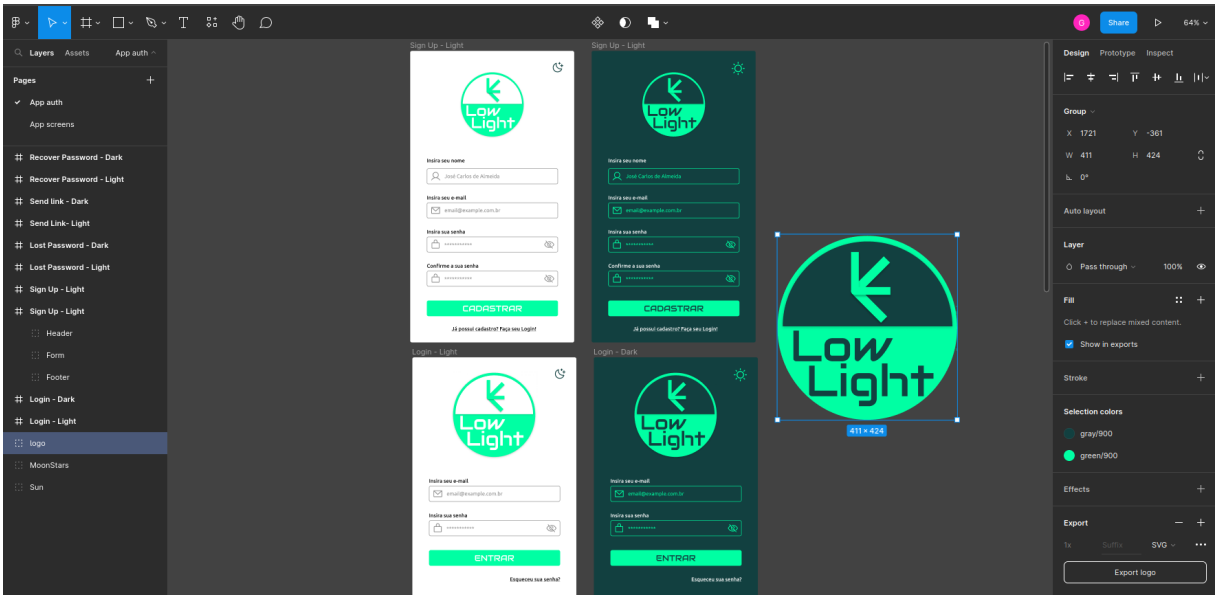
Será abordado nesta seção, todas as ferramentas utilizadas no desenvolvimento do aplicativo que são exibidas e interagem com o usuário de forma

visual.

3.3.1 Figma

Figma é uma ferramenta online para desenvolver o protótipo visual de seu software, tal como as interfaces do usuário, podendo ser para a internet ou para dispositivos móveis. Neste projeto, foram criadas as telas do aplicativo, toda a paleta de cores, os espaçamentos entre telas, o tamanho das fontes, entre todos os outros itens que estão relacionados com o design de tela do aplicativo, como mostra a Figura 7. O protótipo ṕde ser acessado através do link: https://www.figma.com/file/rSBgyR7EnkuRyCN43hVn9D/Design-System-TCC?node-id=212%3A490&t=vitGvdWxIrjIk2Qt-1

Figura 7- Protótipo no Figma para o projeto do aplicativo LightLow



Fonte: Elaborado pelo autor

3.3.2 React Native

React Native (também conhecido como RN) é uma estrutura de aplicativo móvel popular, baseada na linguagem JavaScript, que permite criar aplicativos móveis renderizados nativamente para iOS e Android. A estrutura permite criar um aplicativo para várias plataformas usando a mesma base de código.

O React Native foi lançado em 2015 pelo Facebook como um projeto de

código aberto. Em apenas alguns anos, tornou-se uma das principais soluções adotadas no desenvolvimento móvel, atualmente é utilizada por aplicativos mundialmente famosos como o Discord, Tesla, Instagram, Facebook, entre outros (Cunha, 2022b).

O React Native foi construído com base no React que é uma biblioteca JavaScript muito popular para desenvolvimento de interfaces na web. Esse foi um dos motivos para o uso da ferramenta neste projeto, uma vez que, o autor deste projeto possui conhecimento em JavaScript e React, se adaptando facilmente a nova tecnologia.

3.3.3 NativeBase

É uma biblioteca de código aberto e um *framework* desenvolvido com base no RN que disponibiliza rapidamente componentes visuais, tornando a construção do aplicativo mais ágil, rápida e consistente no design. Ela padroniza cores, tamanhos, fontes, e componentes que são reutilizados nesse projeto, sem precisar necessariamente criar algo novo. Além disso, disponibiliza mais flexibilidade ao se criar as telas, pois os componentes apresentam mais poder de modificação, algo que evita repetição e acelera o desenvolvimento do aplicativo.

3.3.4 Expo

O Expo é uma ferramenta utilizada no desenvolvimento mobile com React Native que permite o fácil acesso às APIs nativas do dispositivo sem precisar instalar qualquer dependência ou alterar código nativo (Cunha, 2022a). Pela facilidade de acesso, agiliza o desenvolvimento de recursos nativos do celular, como acesso a câmera, *storage* ou demais serviços específicos de dispositivos móveis.

Além dessas facilidades, permite executar o aplicativo em um emulador Android ou iOS e até mesmo pelo próprio celular físico. Isso acontece porque o Expo cria uma ponte de comunicação entre sua aplicação, executada em máquina local, e o emulador físico ou virtual através de seu aplicativo disponível nas lojas de aplicativo de cada dispositivo. Por essas enormes vantagens, o projeto de aplicativo foi criado utilizando essa ferramenta.

3.4 Typescript

Tanto no *backend* quanto no *frontend,* o código TypeScript é utilizado somente em ambiente de desenvolvimento e adicionando tipagem estática ao JavaScript, que, originalmente, possui tipagem dinâmica. Dessa forma, as funções e variáveis conseguem assumir tipos diferentes ao longo do tempo de execução. Com isso, é possível garantir a integridade de que os dados passados pelas funções e variáveis serão aqueles mesmo esperados a ser recebidos.

Um exemplo é mostrado na Figura 8. Inicialmente a intenção do programador será passar o valor a ser alterado através de um parâmetro da função chamado *value* e garantir que o *value* será sempre do tipo *string*. Utilizando o Typescript, isso é garantido, e caso qualquer outro tipo de dado é passado no lugar de *string*, o código apresentará um erro e forçará a manter a consistência. Neste projeto o código Typescript foi implementado apenas no código *frontend*.

Figura 8 - Demonstração da utilização do Typescript no código

```
function handleChangeSelectValue(value: string) {
  setService(value)
}
```

Fonte: Elaborado pelo autor

4 DESENVOLVIMENTO

Nesta seção será descrito todo o processo de desenvolvimento, iniciando do levantamento dos requisitos até o desenvolvimento final do aplicativo. O processo de desenvolvimento utiliza-se de várias ferramentas e técnicas de engenharia de software para entregar um aplicativo final com qualidade.

O projeto foi desenvolvido com o pensamento de haver um jeito rápido de coletar as informações de consumo energético de uma residência, não se preocupando com extrema precisão de dados na entrega final, podendo ser valores aproximados, para que o morador residencial possa encontrar rapidamente o item que está mais impactando o valor de sua conta de luz.

A primeira ação foi levantar os itens que mais impactavam este valor e, na sequência, determinar a tecnologia que resolveria o problema e se o resultado atenderia o consumidor final. Após isso, o processo foi detalhado desde a prototipação das telas, seguida da modelagem do banco de dados e do desenvolvimento do código, onde as etapas participaram de um fluxo de trabalho organizado.

4.1 Levantamento de requisitos

Durante o processo de fundamentação teórica deste trabalho, foram conceituadas ideias que o autor adquiriu durante o curso sobre requisitos funcionais e requisitos não funcionais. Como cita o artigo Devmedia (2022):

> Os requisitos funcionais referem-se sobre o que o sistema deve fazer, ou seja, suas funções e informações. Os requisitos não funcionais referem-se aos critérios que qualificam os requisitos funcionais. Esses critérios podem ser de qualidade para o software, ou seja, os requisitos de performance, usabilidade, confiabilidade, robustez, etc. Ou então, os critérios podem ser quanto a qualidade para o processo de software, ou seja, requisitos de entrega, implementação, etc.

4.1.1 Requisitos funcionais

Diante do entendimento citado na seção anterior, foram levantados os seguintes requisitos funcionais do aplicativo:

- **Cadastro e autenticação dos usuários;**

- **Possuir um questionário para cadastro dos equipamentos elétricos do usuário:**
  - O questionário deve conter elementos visuais simples e objetivos para o usuário inserir os seus dados;
  - Não deve conter um extenso número de perguntas para não ser cansativo ao usuário;
  - Deve haver a opção de pular a questão, caso não possua o item;
  - Devem ser sequenciais, para responder somente uma única vez;
  - O usuário poderá reutilizar o questionário para editar algum item posteriormente;
  - Após respondido o questionário, ao se autenticar novamente, já ser apresentada a tela de *dashboard*; e
  - O primeiro item do questionário deve ser obrigatório, não pode pular ou ser valor 0 nas respostas.

- **Visualizar os resultados em uma tela central de *dashboard* onde facilmente consegue os dados que almeja de imediato:**
  - Deverá ser exibido a quantidade calculada de kWh por mês;
  - Deverá ser exibido o item que mais consome e quanto consome do total da residência;
  - Deverá ser exibido o valor em reais estimado da próxima conta de luz; e
  - Deverá ter a foto e nome do usuário e ao clicar na foto de perfil, ir para a tela de edição de dados do usuário.

- **Possuir uma tela que lista os itens por categorias havendo possibilidade de edição:**
    - Devem ser criados componentes de *cards* para facilitar a visualização;
    - Cada *card* deve possibilitar a visualização da porcentagem que cada item consome perante ao total da residência;
    - Ao clicar em um *card*, deve abrir o questionário novamente para poder editá-lo; e
    - Após salvar a edição, deve retornar a tela de listagem com os dados atualizados.

- **Possuir uma tela onde é possível selecionar as categorias:**
    - Ao selecionar uma categoria, trazer os itens pertencentes a esta categoria em forma de lista horizontal;
    - Ao selecionar uma categoria, trazer as porcentagens que comparam ela ao total da residência, tanto no seu consumo quanto no seu tempo total de uso, e também o item que mais consome na categoria; e
    - Ao clicar em qualquer item, uma nova tela deve trazer o resumo detalhado do item que será apresentado.

- **Possuir uma tela de resumo detalhado de cada item:**
    - Passar os dados com uma visualização fácil de horas totais do item no mês, a energia em kWh por mês que consome e o custo individual esperado para esse item; e
    - Será possível alterar o item nessa tela, selecionando outro no lugar.

- **Possuir uma tela de ranking e uma visualização geral de todos os usuários:**
    - Essa tela deve conter uma seleção dos 10 usuários que mais e menos consomem, mais economizaram e possuem o maior e o menor tempo de uso.

- **Interface intuitiva, simples e objetiva para promover o uso do aplicativo; e**

- **Gráficos circulares, ícones de fácil identificação e conceitos repetitivos para o usuário se adaptar rapidamente ao uso do aplicativo.**

4.1.2 Requisitos não funcionais

Uma vez levantado os funcionais, posteriormente foram levantados os seguintes requisitos não funcionais:

- O aplicativo deve no mínimo executar em um emulador Android;
- O aplicativo deve ser implementado com React Native para dispositivos móveis através do Expo e Node;
- O aplicativo deve possuir um aprendizado rápido;
- O aplicativo deve chegar a valores aproximados de cálculo;
- Ser acessível a qualquer pessoa;
- Deve instruir ajuda em tópicos mais avançados ou que possam surgir dúvidas;
- Deve utilizar as tecnologias mais atuais no mercado;
- A programação deve ser no estilo de programação funcional;
- A interface deve possuir uma resposta clara sobre carregamentos e tempo de resposta;
- O questionário não deve prender o usuário com regras de validação. Se qualquer regra não pertencer ao escopo de validação, o item não será aplicado ao cálculo final; e
- Os equipamentos elétricos devem ter imagens acompanhadas de nome para ser totalmente compreendido pelo usuário;

4.2 Análise e modelagem dos requisitos

O objetivo geral deste aplicativo é coletar os dados do usuário de forma rápida e eficiente e apresentá-los de uma melhor forma utilizando gráficos e outros componentes de fácil entendimento e visualização. Com isso, os requisitos relacionados aos dados coletados do usuário fazem parte da interação e usabilidade do aplicativo, onde o usuário rapidamente aprenderá como está a sua conta de luz e quais itens estão impactando diretamente na sua economia. Posteriormente, o morador poderá editar o uso de cada item, vendo o resultado esperado do simulador

em tempo real, e terá mais clareza ao adotar corretamente o tempo de uso de seus aparelhos a partir deste momento.

A tela de *dashboard* responde diretamente ao usuário o que ele deseja saber de imediato, quais são os valores de sua próxima conta estimada e o item mais problemático em sua casa, seja por consumir muito ou por estar muito tempo ligado. Isso já retorna um *feedback* imediato da maioria dos requisitos para o usuário, mas caso queira se aprofundar, pode simular e encontrar itens secundários nos quais pode alterar os dados e obter simulações mais precisas e expectativas de economia reais. Durante esse processo, o próprio aplicativo já passa um ensinamento indiretamente sobre elétrica, uma vez que os dados modificados são todos dados elétricos. O consumidor conseguirá assimilar o custo com o valor em reais e entenderá parcialmente como funciona o cálculo geral de sua conta.

Sendo assim, a Tabela 2, os casos de usos levantados dos requisitos

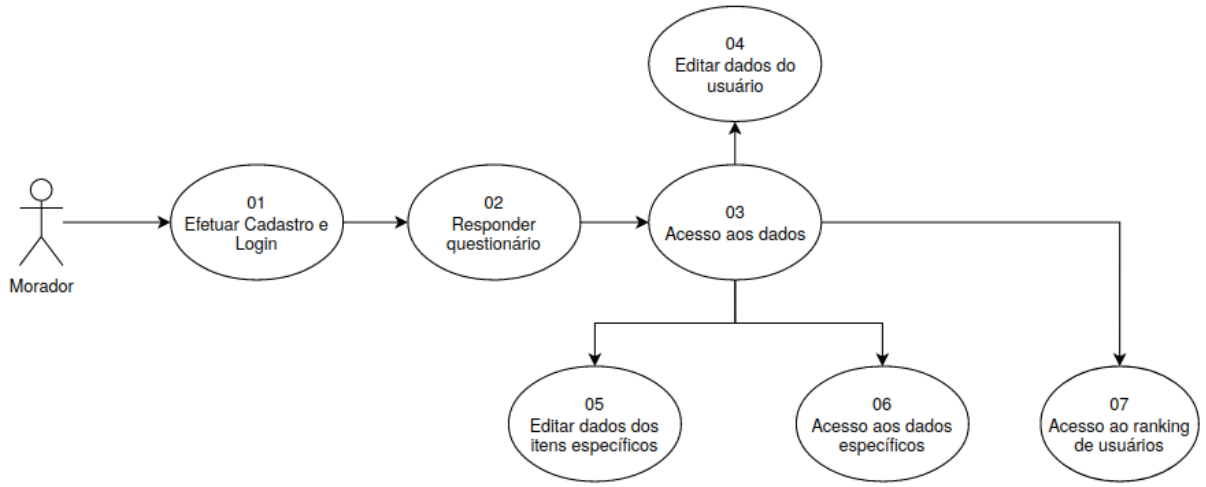
Tabela 2 - Casos de uso levantados

Casos de Uso	Nome Casos de Uso	Descrição Casos de Uso
01	Efetuar Cadastro e Login	Cadastro e autenticação dos usuários no aplicativo permitindo a coleta de seus dados e o retorno das informações calculadas.
02	Responder questionário	Usuários deverão responder o questionário para terem acesso aos seus dados.
03	Acesso aos dados	Usuários poderão consultar seus dados na tela inicial de *dashboard*.
04	Editar dados do usuário	Usuários poderão editar seus dados pessoais, como foto de perfil, nome e senha.
05	Editar dados dos itens específicos	Usuários poderão editar os dados dos itens respondidos anteriormente no questionário.
06	Acesso aos dados específicos	Usuários terão acesso a dados específicos de cada categoria e item, para melhor detalhamento do conteúdo.
07	Acesso ao ranking de usuários	Usuários poderão ver quais outros usuários estão nos dez primeiros lugares em certas especialidades.

Fonte: Elaborado pelo autor

Utilizando esses casos de uso simplificados como referência, elaborou-se o diagrama de casos de uso apresentado na Figura 9.

Figura 9 - Diagrama de casos de uso levantados

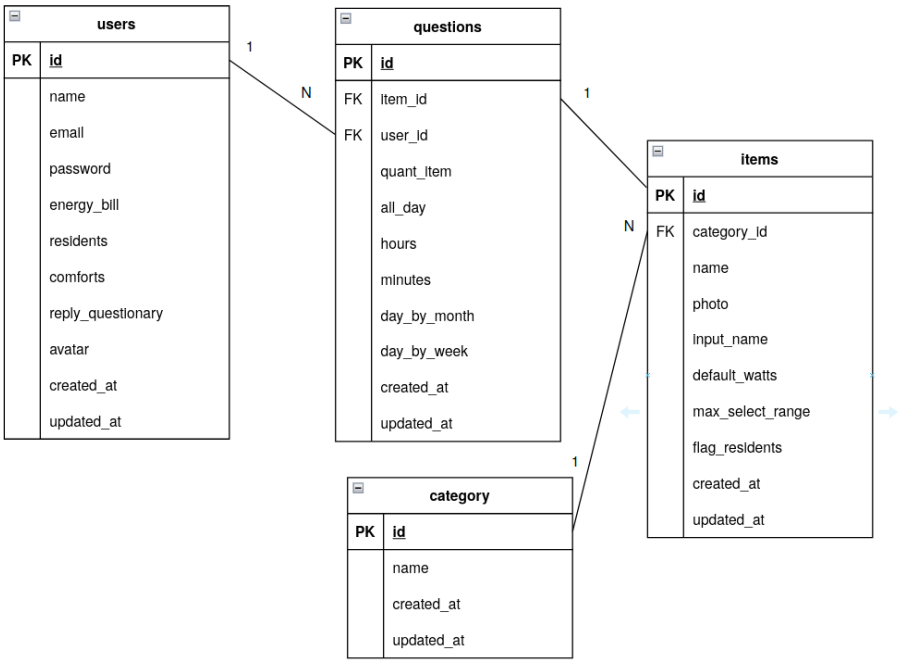


Fonte: Elaborado pelo autor

4.3 Construção da representação do banco de dados

Com os requisitos definidos nas seções anteriores foi possível criar a estrutura objetiva para o banco de dados, satisfazendo as necessidades para as funcionalidades apresentadas na Figura 10.

Figura 10 - Diagrama entidade relacionamento do banco de dados



Fonte: Elaborado pelo autor

Como todos requisitos para a aplicação são centralizados nas respostas do questionário, os relacionamentos mais fortes são entre a tabela *questions* com as restantes, na qual todos os dados do questionário são armazenados. A tabela de *category* existe apenas para listar as categorias que cada item pertence para gerar um questionário dinâmico, consumido pelo cliente e entregue pelo servidor. A tabela *items*, além de fornecer os itens que deverão ser questionados pelo formulário, ainda possui o valor padrão de potência de cada item, valor esse utilizado no cálculo final. E por fim, a tabela *users* que contém toda a autenticação e dados do usuário. Nesta tabela é atribuída a foto do usuário e as três primeiras questões pertencentes ao usuário, como o valor de sua última conta de energia, a quantidade de moradores e os cômodos de sua casa.

4.4 Processo de desenvolvimento

O processo de desenvolvimento considerou em duas bases de código, uma para o *frontend* e outra para o *backend*. O projeto foi iniciado pelo *frontend* ao desenvolver as telas do aplicativo. A partir deste tópico as telas do aplicativo serão apresentadas e suas funcionalidades explicadas.

4.4.1 Desenvolvimento do aplicativo

A tela de Login, apresentada na Figura 11, é a primeira tela que o usuário acessa na aplicação. Após preencher um formulário com validação é possível realizar a autenticação, caso possua, senão pode optar em criar uma nova ao acessar a opção sublinhada.

Figura 11 - Tela de Login

Fonte: Elaborado pelo autor

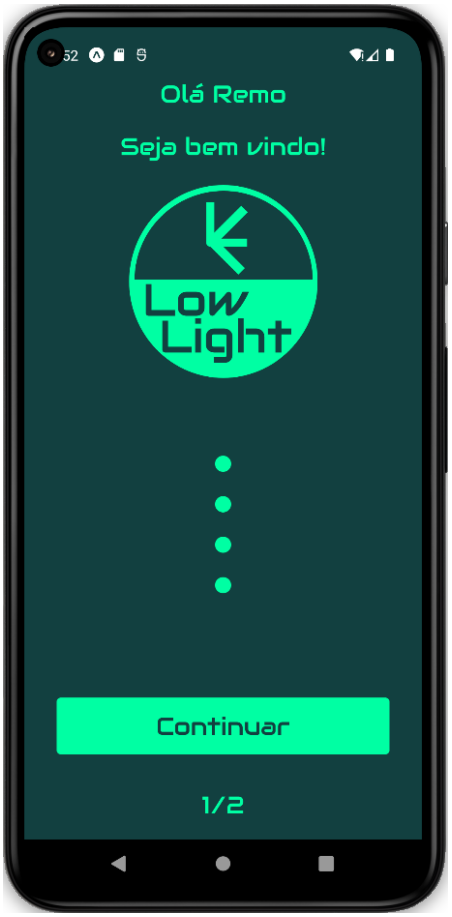
A tela apresentada na Figura 12 corresponde à tela de Cadastro. Nela é possível criar uma conta inserindo os dados no formulário e ao pressionar o botão Cadastrar, ocorre a validação. Caso haja algum problema, uma mensagem negativa aparecerá na tela, senão o usuário será cadastrado e fará o login automaticamente. É possível retornar para a tela de Login, se necessário, através da opção sublinhada.

Figura 12 - Tela de Cadastro

Fonte: Elaborado pelo autor

A tela da Figura 13 é uma tela simples de boas vindas ao aplicativo que introduz a simplicidade da interface de usuário, sendo bem intuitiva e instigando ao usuário continuar.

Figura 13 - Tela de Boas vindas



Fonte: Elaborado pelo autor

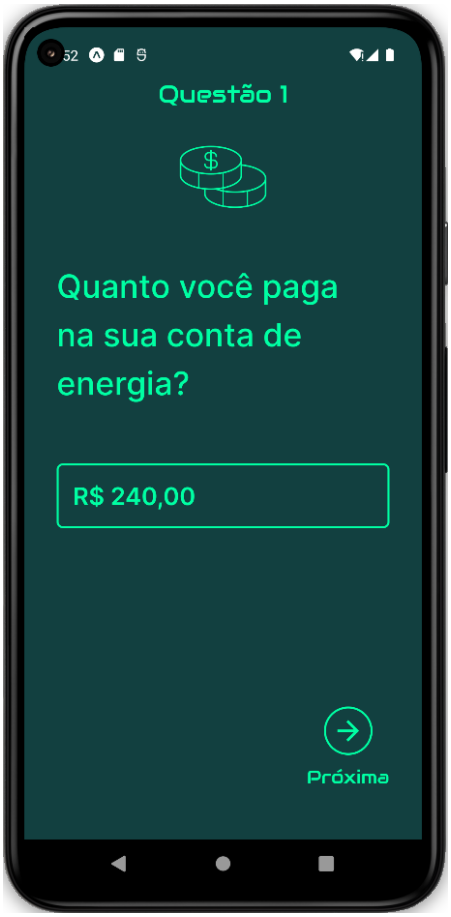
A Figura 14 é a continuação da tela de Boas vindas. Ela explica ao usuário o próximo passo que irá encontrar, sendo um questionário. Além disso, ela também deixa o usuário ciente de que está aceitando os termos de uso do aplicativo.

Figura 14 - Tela de Boas vindas parte 2

Fonte: Elaborado pelo autor

A Figura 15 é a tela da primeira questão, onde irá questionar o valor da última conta de luz que o usuário pagou. Não precisa ser exato ao real, apenas um valor aproximado.

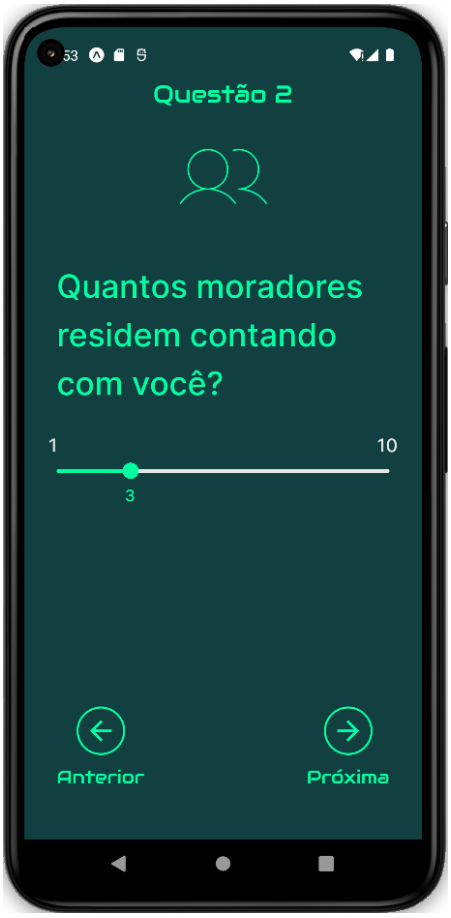
Figura 15 - Questão 1



Fonte: Elaborado pelo autor

A Figura 16 é a questão de quantos moradores moram nessa casa e residem com o usuário. O valor dessa questão é importante pois afeta os cálculos dos itens com a *flag_residents*. Alguns itens, dependendo do número de pessoas residindo, podem ter o seu consumo dobrado, enquanto outros não fazem diferença, tendo o seu consumo aumentado. Por exemplo, caso existam 3 moradores e 1 chuveiro. Nesse caso, o chuveiro é um item que impactam no seu resultado final, multiplicando o seu consumo em 3 vezes. Outros itens como a geladeira não têm impacto significativo causado pelo número de moradores.

Figura 16 - Questão 2



Fonte: Elaborado pelo autor

A Figura 17 finaliza esse breve questionário perguntando ao usuário quantos cômodos sua casa possui. Este dado é importante no cálculo inicial, ou seja, pois baseia-se nos dados obtidos pelos gráficos da PROCEL (2022) em relação a duração das lâmpadas no período mais usado (entre 17:00 e 23:00) pelos moradores brasileiros, cerca de 6 horas diárias de consumo alto e relevante. O cálculo é realizado multiplicando as horas diárias de consumo pela quantidade de cômodos na sua casa, subtraindo a quantidade de residentes, adicionando a potência média de lâmpadas (10W) e multiplicando por 30 dias (1 mês). Assim, é possível chegar a um cálculo médio de lâmpadas ativas nesse período, iniciando o cálculo final com esse valor mínimo.

Figura 17 - Questão 3

Fonte: Elaborado pelo autor

A Figura 18 inicia o questionário de cada item, sendo a primeira opção da categoria COZINHA, onde o item é a GELADEIRA. Por padrão, a geladeira já vem com quantidade mínima de 1 item, 24 horas ligado e sem opção para pular. Isso faz com que pelo menos 1 tenha sido inserido pelo usuário e os cálculos não sejam zerados.

Figura 18 - Primeira questão do questionário de equipamentos elétricos

Fonte: Elaborado pelo autor

A Figura 19 demonstra que o próximo item inicia-se com a quantia zerada. Ao clicar no botão Hora / Min, ele fica selecionado e abre novos campos para selecionar a quantidade de horas e minutos que o aparelho fica em funcionamento no dia e a frequência usada na semana ou mês. Através de componentes de *radiobuttons*, o usuário consegue preencher rapidamente, sem precisar digitar, apenas arrastar o botão.

Figura 19 - Segunda questão do questionário de equipamentos elétricos

Fonte: Elaborado pelo autor

A Figura 20 mostra que a opção mensal foi selecionada como a frequência que este aparelho é usado. Para itens com pouca frequência, o usuário pode usar a frequência mensal, considerando o uso em poucos dias do mês, entretanto, é possível selecionar o mês todo, indo de 0 a 30 dias.

Figura 20 - Frequência de uso mensal

Fonte: Elaborado pelo autor

Na Figura 21 o usuário pode selecionar a frequência semanal, quantidade de dias da semana em que o item é usado, seja uma vez por semana até a semana toda. Esse valor é convertido no cálculo para mês ao ser multiplicado 4 vezes, representando as 4 semanas que compõem um mês completo aproximadamente.

É possível também verificar um botão Pular e um que direciona para a Próxima questão. Quando este botão é selecionado, os dados daquela questão não são inseridos no cálculo, mas podem ser modificados posteriormente ao se editar um item específico. Ao clicar em Próxima, o questionário segue o fluxo dos 16 itens cadastrados e quando finalizar, aparecerá a tela de *dashboard*.

Figura 21 - Frequência de uso semanal

Fonte: Elaborado pelo autor

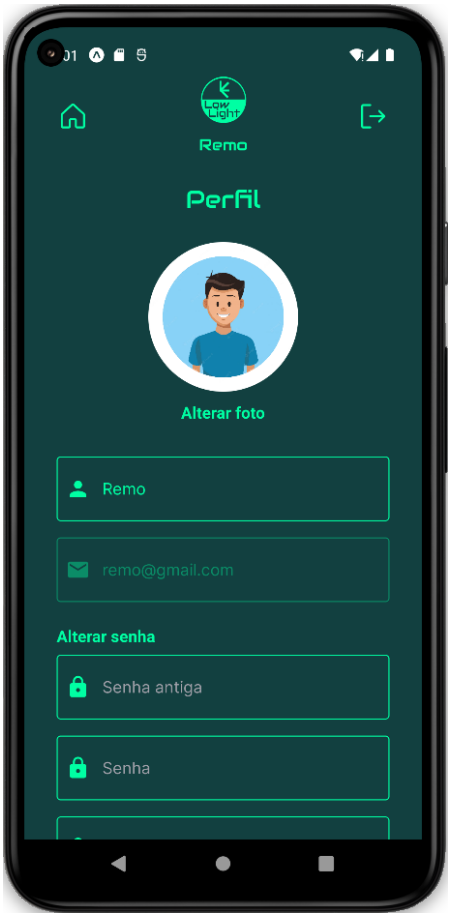
Uma vez respondido o questionário, o usuário entrará na tela de *dashboard*, que mostra seu painel de controle, podendo ser vista na Figura 22. Ela contém os dados essenciais coletados do formulário, além do avatar no lado superior esquerdo, que ao clicar é possível alterar o perfil. No centro superior, o nome do usuário com o logo do projeto LightLow. No centro, há 3 gráficos indicando os dados calculados ao usuário. No canto superior direito, um botão que ao ser clicado, faz o *logout* do usuário. Na parte inferior, um menu de barras com 3 ícones, levando para as respectivas telas: Economize, Meu Consumo, Ranking.

Figura 22 - Tela de Dashboard

Fonte: Elaborado pelo autor

Na Figura 23 encontra-se a tela de Editar perfil do usuário. Nesta tela, pode-se alterar a foto, o nome e a senha, após as validações e ao clicar no botão atualizar. Somente o e-mail se mantém como um dado fixo e imutável.

Figura 23 - Tela de Perfil de usuário



Fonte: Elaborado pelo autor

A tela de Economize pode ser visualizada na Figura 24. Ela é composta por uma listagem de *cards* dividindo os itens em suas categorias e mostrando quanto cada item está consumindo referente ao total da residência. Ao clicar em algum *card*, abre-se o mesmo formulário do questionário, para que o item selecionado seja editado.

Figura 24 - Tela de Economize energia

Fonte: Elaborado pelo autor

Na tela de Meu consumo, demonstrada na Figura 25, é possível ver o consumo detalhado por categoria, filtrando a escolhida. Ela preenche as barras de progresso com seus valores e lista os seus itens, clicáveis. Ao clicar em um item, a tela de detalhes do consumo (Figura 26) é exibida e mais informações específicas de cada item são mostradas ao usuário. É possível também filtrar os itens da categoria selecionada.

Figura 25 - Tela de Meu Consumo

Fonte: Elaborado pelo autor

Figura 26 - Tela de Consumo detalhado

Fonte: Elaborado pelo autor

E por fim, na Figura 27 é possível ver a tela de Ranking. Essa tem como objetivo filtrar dados em geral de todos os usuários da plataforma, promovendo até uma pequena gamificação para ver quem está economizando ou gastando demais.

Figura 27 - Tela de Ranking

Fonte: Elaborado pelo autor

4.4.2 Desenvolvimento da API

Após instalar todas as dependências a API foi desenvolvida utilizando um projeto Node em conjunto com o Knex.js para gerenciar o banco de dados. Primeiramente, foram criadas as APIs de autenticação e validação dos dados do usuário nas rotas do *user*. Em seguida, foram criptografadas as senhas com Hash para que tivessem segurança no banco de dados. É utilizado um *token* JWT que comunica essas informações de autenticação para o *backend* e com isso é possível fazer as chamadas para as APIs. Além disso, é esse *token* que permanece no cliente com prazo de até 1 dia para que o mesmo seja desconectado da aplicação de forma automática. Para renovar, precisaria autenticar novamente.

Posteriormente, foram criadas as rotas das categorias, itens e questões. As tabelas foram criadas por migrações de subida de tabela e logo após, foram

inseridos os dados através de *seeds* pelo Knex.js, assim preenchendo os dados de forma dinâmica.

E por fim, as regras de negócio foram criadas nas *Controllers*, como por exemplo, o cálculo total de kWh por mês dos usuários ou até as imagens dos itens são servidas em uma URL local, sendo assim possível retornar os resultados que cada tela do aplicativo exigia ao se integrar com o *backend*.

5 CONCLUSÃO

O LightLow atingiu seu objetivo de ser um aplicativo que simula os dados energéticos residenciais e entrega dados que favorecem o direcionamento correto para a economia de energia de seus usuários. A ideia foi ser um *software* fácil, intuitivo, bem amigável e que pudesse identificar rapidamente o problema energético da residência de um morador brasileiro. Os dados destacados ao consumidor final podem ser analisados para traçar um melhor direcionamento ao se tentar economizar energia elétrica em sua residência.

É um produto inovador ao consumidor brasileiro, destacando-se, ainda, a realização de testes, demonstrando que o aplicativo funciona como o esperado, possuindo responsabilidades definidas a partir da tela de *Login*. Com o foco de ser altamente acessível para inúmeras faixas etárias, a decisão de escolher ser um *software mobile* permitiu que o produto alcance a maioria de usuários do Brasil. Além disso, o LightLow entregou maior conhecimento no consumo energético aos seus usuários, através de dicas e simulações.

Em relação ao seu desenvolvimento, tecnologia e conhecimento, o aplicativo envolve inúmeras teorias e práticas fornecidas ao longo do curso de Sistemas de Informação. O principal desafio foi a tecnologia utilizada, uma vez que havia uma barreira técnica para desenvolver um aplicativo usando como tecnologia o RN. Mesmo assim, o aplicativo utiliza as tecnologias modernas de desenvolvimento do mercado de trabalho, facilitando a sua continuação para desenvolvimento de novas *features*.

Além disso, os aplicativos semelhantes apresentados anteriormente possuem falhas que o LightLow consegue suprimir e demonstrar um melhor uso, como por exemplo, fácil visualização dos dados e *interface* intuitiva, não permite que o usuário realize muitas ações manuais repetitivas, agiliza o processo de economia de energia com poucos passos.

Conclui-se, portanto, que o aplicativo LightLow conseguiu conciliar as tecnologias modernas com técnicas aprendidas durante o curso e apresenta hoje uma solução aos desafios atuais dos brasileiros perante o custo energético e a economia de energia. Essa ação vem de encontro com a previsibilidade de que seu

uso se torna cada vez mais frequente pela tendência de aumento do consumo de energia e economia do mesmo.

6 TRABALHOS FUTUROS

Visando a melhoria do aplicativo é interessante deixar os cálculos mais precisos, coletando informações de geolocalização dos usuários e assim aplicar as tarifas corretas da sua localização e não uma média geral sobre todas. Outra forma de melhorar a aplicação, é testá-la em dispositivos iOS e realizar a publicação em ambas as plataformas, para que todos os usuários de dispositivos possam utilizá-la.

Posteriormente, recursos já presentes no aplicativo podem ser explorados em maior escala, como o *ranking* entre os usuários ou até adicionar mais itens importantes, como carregadores de celular.

REFERÊNCIAS

ABRAHÃO, Karla Cristina de Freitas Jorge; SOUZA, Roberta Gonçalves Vieira de. Estimativa da evolução do uso final de energia elétrica no setor residencial do Brasil por região geográfica. [S. l.], 13 abr. 2020. Disponível em: https://www.scielo.br/j/ac/a/MC5DNWHS46jH6hCKKtCzFCc/?lang=pt&format=pdf. Acesso em: 16 jan. 2023.

ANDREI, L. O que é GitHub e Como Usá-lo. [S. l.], 27 jan. 2023. Disponível em: https://www.hostinger.com.br/tutoriais/o-que-github. Acesso em: 28 jan. 2023.

BBC. O que mais consome energia na sua casa e como economizar. [S. l.], 9 nov. 2022. Disponível em: https://www.bbc.com/portuguese/geral-63562836. Acesso em: 17 jan. 2023.

CASARIN, R. O que é um watt? Conheça as unidades de potência elétrica. [S. l.], 2022. Disponível em: https://www.portalsolar.com.br/noticias/tecnologia/curiosidades/o-que-e-um-watt-conheca-as-unidades-de-potencia-eletrica. Acesso em: 16 jan. 2023.

COOPERLUZ. Tabela de Consumo (kWh). [S. l.], 2022. Disponível em: https://www.cooperluz.com.br/tabela-de-consumo#:~:text=Como%20calcular%20o%20consumo%20de,Dia. Acesso em: 13 jan. 2023.

CUNHA, A. Como instalar e configurar o Expo do React Native. [S. l.], 2022a. Disponível em: https://www.alura.com.br/artigos/como-instalar-configurar-expo-do-react-native. Acesso em: 18 jan. 2023.

CUNHA, A. React Native: o que é e tudo sobre o Framework. [S. l.], 2022b. Disponível em: https://www.alura.com.br/artigos/react-native. Acesso em: 18 jan. 2023.

DEVMEDIA. Introdução a Requisitos de Software. [S. l.], 2013. Disponível em: https://www.devmedia.com.br/introducao-a-requisitos-de-software/29580. Acesso em: 18 jan. 2023.

ELETROBRAS. [S. l.], 2022. Disponível em: https://app.powerbi.com/view?r=eyJrIjoiZTQxMDg3M2EtMzM2My00NDk5LWIzMzctY2EwYzQxZjBkNDU5IiwidCI6IjhhMGZmYjU0LTk3MTYtNGE5My05MTU4LTllM2E3MjA2ZjE4ZSJ9. Acesso em: 18 jan. 2023.

EPE. [S. l.], 2022. Disponível em: https://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-160/topico-168/Fact%20Sheet%20-%20Anu%C3%A1rio%20Estat%C3%ADstico%20de%20Energia%20El%C3%A9trica%202022.pdf. Acesso em: 18 jan. 2023.

PLAY, GOOGLE, 2022a. Disponível em: https://play.google.com/store/apps/details?id=a10devsmg.calculo_energia. Acesso em: 18 jan. 2023.

PLAY, GOOGLE, 2022b. Disponível em: https://play.google.com/store/apps/details?id=com.kwhpro&hl=pt_BR&gl=US. Acesso em: 18 jan. 2023.

KONNORATE et al. A importância do controle de versões no desenvolvimento de software. [S. l.], Novembro 2019. Disponível em: http://raam.alcidesmaya.edu.br/index.php/SGTE/article/view/9/9. Acesso em: 19 jan. 2023.

KORTH, H. F.; SILBERSCHATZ, A. Sistemas de Bancos de Dados. 2a. edição revisada.
ed. [S.l.]: Makron Books, 1994.

ORACLE. O que é um banco de dados relacional (RDBMS)?. [S. l.], 2022. Disponível em: http://raam.alcidesmaya.edu.br/index.php/SGTE/article/view/9/9. Acesso em: 19 jan. 2023.

PESSÔA, Camila. Node.JS: definição, características, vantagens e usos possíveis. [S. l.], 8 mar. 2022. Disponível em: https://www.alura.com.br/artigos/node-js-definicao-caracteristicas-vantagens-usos. Acesso em: 18 jan. 2023.

PRESSMAN, R. S.; MAXIM, B. R. Engenharia de Software Uma Abordagem Profissional.
8. ed. [S.l.]: AMGH, 2016. 968 p

PROGRAMADORES, BRASIL. Como funciona um Banco de Dados SQL Relacional ?. [S. l.], 23 fev. 2020. Disponível em: https://programadoresbrasil.com.br/2020/02/como-funciona-um-banco-de-dados-sql-relacional/. Acesso em: 16 jan. 2023.

REZENDE, R. Conceitos Fundamentais de Banco de Dados. [S. l.], 2006. Disponível em: https://www.devmedia.com.br/conceitos-fundamentais-de-banco-de-dados/1649. Acesso em: 17 jan. 2023.

SANTOS, FIGER. Transição enérgica: Brasil dribla a crise hídrica, mas permanece em alerta em relação a conta de luz. [S. l.], 3 maio. 2022. Disponível em: https://portal.fgv.br/artigos/transicao-energica-brasil-dribla-crise-hidrica-mas-permanece-alerta-relacao-conta-luz. Acesso em: 16 jan. 2023.

SOUZA, Ivan de. O que é SQLite, por que ele é usado, e o que o diferencia do MySQL?. [S. l.], 24 nov. 2020. Disponível em: https://rockcontent.com/br/blog/sqlite/. Acesso em: 17 jan. 2023.

TARIFA Residencial. [S. l.], 2022. Disponível em: https://app.powerbi.com/view?r=eyJrIjoiOTY0NWQzOGItMmQ3ZS00MWUzLTllNmMtNTA5NTYxODdhYTkzIiwidCI6IjQwZDZmOWI4LWVjYTctNDZhMi05MmQ0LWVhNGU5YzAxNzBlMSIsImMiOjR9. Acesso em: 16 jan. 2023.

TOTVS. In: TypeScript: Saiba mais sobre ele e suas funcionalidades. [S. l.], 10 mar. 2020. Disponível em: https://www.totvs.com/blog/developers/typescript/. Acesso em: 18 jan. 2023.