UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO PRÓ-REITORIA DE GRADUAÇÃO CENTRO MULTIDISCIPLINAR DE PAU DOS FERROS CURSO DE BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

DHIOGO SAMUEL PINHEIRO

CONSTRUÇÃO DE UM PROTÓTIPO DE ESTACIONAMENTO AUTOMATIZADO INTEGRADO COM APLICAÇÃO MOBILE

PAU DOS FERROS

2020

DHIOGO SAMUEL PINHEIRO

CONSTRUÇÃO DE UM PROTÓTIPO DE ESTACIONAMENTO AUTOMATIZADO INTEGRADO COM APLICAÇÃO MOBILE

Trabalho de conclusão de curso apresentada a Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em Engenharia de Computação.

Orientador: Cecílio Martins de Sousa Neto, Prof. Dr.

PAU DOS FERROS

2020

P654c Pinheiro, Dhiogo Samuel.
CONSTRUÇÃO DE UM PROTÓTIPO DE ESTACIONAMENTO AUTOMATIZADO INTEGRADO COM APLICAÇÃO MOBILE / Dhiogo Samuel Pinheiro. - 2020.
27 f. : il.

Orientador: Cecilio Martins de Sousa Neto.
Monografia (graduação) - Universidade Federal Rural do Semi-árido, Curso de Engenharia de Computação, 2020.

1. Arduino. 2. Mobile. 3. Estacionamento. I. Sousa Neto, Cecilio Martins de, orient. II. Título.

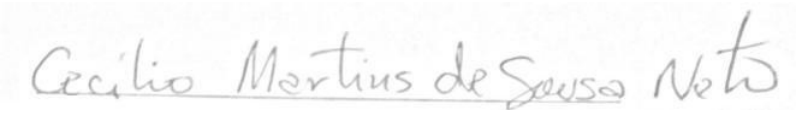
DHIOGO SAMUEL PINHEIRO

CONSTRUÇÃO DE UM PROTÓTIPO DE ESTACIONAMENTO AUTOMATIZADO INTEGRADO COM APLICAÇÃO MOBILE

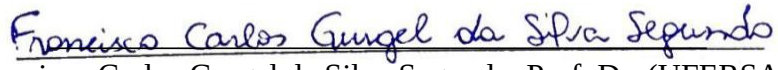
Trabalho de conclusão de curso apresentada a Universidade Federal Rural do Semi-Árido como requisito para obtenção do título de Bacharel em Engenharia de Computação.

Defendida em: 07 / 02 / 2020.

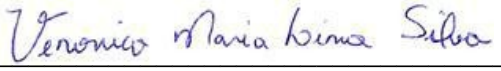
BANCA EXAMINADORA



Cecilio Martins de Sousa Neto, Prof. Dr. (UFERSA)
Presidente



Francisco Carlos Gurgel da Silva Segundo, Prof. Dr. (UFERSA)
Membro Examinador



Veronica Maria Lima Silva, Profa. Dra. (UFERSA)
Membro Examinador

RESUMO

A mobilidade urbana e a disponibilidade de estacionamentos não acompanharam o crescimento da frota de veículos, fazendo com que em grandes centros urbanos, motoristas tenham dificuldade para encontrar uma vaga disponível em estacionamento, diante do cenário atual e com o avanço tecnológico, o crescimento da internet das coisas, se torna possível realizar diversas aplicações, que venham a ser inseridas no dia-a-dia das pessoas. Dessa forma, neste trabalho foi desenvolvido um sistema *mobile*,utilizando o *framework* Flutter e banco de dados Firebase. Na parte de hardware foram utilizados a plataforma de prototipagem de livre Arduino, juntamente com os sensores ultrassônico e o módulo de *wi-fi* esp8266. Com este trabalho, foi possível obter um protótipo satisfatório que realize o que controle de vagas de um estacionamento, verificando a presença de objetos na vaga delimitada e informando na aplicação *mobile*.

Palavras-chave: Arduíno. *Mobile*.Estacionamento;

ABSTRACT

Urban mobility and parking availability do not keep pace with the growth of the vehicle fleet, making large urban centers difficult for drivers to find an available parking space, given the current scenario and with technological advances, or the growth of the Internet of Things , it becomes possible to make several applications, which will be inserted in people's daily lives. Thus, in this work a mobile system was developed, using the Flutter framework and Firebase database. On the hardware side, we used the Arduino's free prototyping platform, including the ultrasonic sensors and the esp8266 wifi module. With this work, it was possible to obtain a satisfactory model that performs or controls parking spaces, verifying the presence of objects in the defined space and information in the mobile application.

Keywords: Arduino. Mobile. Parking;

LISTA DE FIGURAS

Figura 1: Placa Arduino Uno…..……..…… 12

Figura 2: Funcionamento do sensor ultrassônico.....……. 15

Figura 3: Módulo ESP-01…….....……. 15

Figura 4: Hierarquia de Classes …….....…….. 17

Figura 5: Camadas do Flutter …….....…….....17

Figura 6: Conexão entre o sensor e o Arduino …… 19

Figura 7: Hardware Integrado…..……..……..... 20

Figura 8: Coleção usuario_master……..……..... 21

Figura 9: Tela de *Login*…….……..…. 22

Figura 10: Tela principal …….……..... 23

Figura 11: Tela de *histórico do sensor*……...…….……... 24

SUMÁRIO

1. INTRODUÇÃO **8**

1.1. OBJETIVOS 9

1.1.1. Objeto geral 9

1.1.2. Objetivos específicos 9

1.2. ORGANIZAÇÃO DO TRABALHO 9

2. REFERENCIAL TEÓRICO **11**

2.1 SISTEMA EMBARCADOS E IoT 11

2.2 MICROCONTROLADOR ARDUINO 12

2.3 DESENVOLVIMENTO MOBILE PARA ANDROID 13

3. DESCRIÇÃO DO SISTEMA **14**

3.1. HARDWARE DO SISTEMA 14

3.1.1 Sensor Ultrassônico 14

3.1.2 Módulo WiFi ESP8266 ESP-01 15

3.2. SOFTWARE DO SISTEMA 16

3.2.1 Flutter 16

3.2.2 Firebase Cloud Firestore 18

4 RESULTADOS E DISCUSSÕES **19**

4.1 BANCO DE DADOS 20

4.2 SOFTWARE MOBILE 22

5 CONSIDERAÇÕES FINAIS **25**

REFERÊNCIAS **26**

1. INTRODUÇÃO

O Brasil possui uma frota formada por mais de 100 milhões de veículos, sendo mais da metade automóveis, e aproximadamente 20 milhões de motocicleta (IBGE, 2018). O aumento massivo da frota de veículo ocorreu devido ao maior poder aquisitivo dos brasileiros, em especial, na ascensão da classe média (LIMA, 2020).

A mobilidade urbana e a disponibilidade de estacionamentos não acompanharam o crescimento da frota de veículos, fazendo com que em grandes centros urbanos, motoristas tenham dificuldade para encontrar uma vaga disponível em estacionamento.

Diante do cenário no qual encontrar vagas livres em estacionamento pode se tornar uma tarefa desgastante, o mercado vem desenvolvendo novas tecnologias com o propósito de facilitar esse processo de encontrar vagas, surgindo novas aplicações como o Parknet. O Parknet é uma aplicação mobile que ajuda motorista a encontrar possíveis localizações com vagas disponíveis para estacionar o veículo, a aplicação utiliza uma inteligência artificial que estima com base em diversos fatores se existe vagas próximas à localização do motorista (BISSI, 2016).

Embora o Parknet seja uma aplicação que facilite a busca por vagas, o aplicativo não garante total confiabilidade visto que não tem como ter certeza das vagas livres. O ideal é uma solução automatizada que verifique em tempo real a disponibilidade de vagas e informe ao motorista.

No intuito de solucionar problemas e facilitar o cotidiano das pessoas, novas tecnologias têm sido inseridas em áreas que tradicionalmente não eram automatizadas. Kevin Ashton descreveu uma tecnologia capaz de obter dados sem a que seja necessário a interferência humana, isso se tornou possível com o advento do sensor, capacitando as máquinas de sentir o ambiente com maior precisão que o homem (ASHTON, 2009).

Com tal tecnologia descrita por Kevin Ashton e o avanço da *internet* das coisas, é possível desenvolver milhares de sistemas que estarão no nosso cotidiano extraindo informações e nos provendo análise à distância.

Tal avanço tecnológico possibilitou a idealização do projeto discutido no presente trabalho, que é um protótipo de controle de vagas de um estacionamento inteligente, utilizando um sistema físico em hardware capaz de coletar dados sobre a disposição ou não de

uma vaga, analisar e enviar o resultado para um servidor em nuvem, que será acessado por uma aplicação mobile visível para o usuário real.

Portanto o protótipo implementado é uma solução automatizada que contribuiria tanto para os proprietários de vagas de estacionamento, que poderiam ter um controle em tempo real, além de um histórico para coleta de dados. É um estudo que pode ser utilizado para que um usuário de estacionamentos pudessem verificar a disponibilidade em tempo real, contribuindo na mobilidade urbana.

1.1. OBJETIVOS

1.1.1. Objeto geral

Desenvolver uma solução mobile integrada com hardware para solução de automação de estacionamento, facilitando a visualização de vagas ocupadas e livres.

1.1.2. Objetivos específicos

- Construir um protótipo, com a plataforma Arduino;
- Desenvolver um sistema mobile, utilizando o framework flutter;
- Integrar a parte de hardware feita em Arduino e sensores, com a parte da aplicação de software feita em flutter;

1.2. ORGANIZAÇÃO DO TRABALHO

No capítulo 2, o referencial teórico descreve os conceitos de sistemas embarcados e a relação com internet das coisas (IoT), também descreve um pouco sobre o microcontrolador Arduíno e por fim a seção relata sobre o desenvolvimento mobile para Android;

No capítulo 3, tem-se os materiais e métodos deste trabalho, em que descreve os materiais utilizados para o sistema, como o sensor ultrassônico, o módulo WiFi ESP8266 ESP-01, para fazer a conexão do hardware com a aplicação. Na parte de software utilizou-se do framework flutter, que utiliza a linguagem dart para construir as telas da aplicação. Para o armazenamento de dados, utilizou-se do banco de dados não relacional, Firebase Cloud Firestore;

No capítulo 4, contém os resultados e discussões deste trabalho, nele é apresentado o sistema desenvolvido;

No capítulo 5, são apresentadas as considerações do trabalho.

2. REFERENCIAL TEÓRICO

2.1 SISTEMA EMBARCADOS E IoT

Sistemas embarcados são sistemas microprocessados que executam uma função específica, por meio de uma programação (ALMEIDA; MORAES; SERAPHIN, 2016). Tais sistemas são facilmente encontrados no cotidiano, como em impressoras, semáforos, calculadores, roteadores, entre outros.

Geralmente sistemas embarcados apresentam restrição de recursos, seja lógico (memória e processamento) ou físico (número de terminais e interface de exibição/entrada de dados). Essas restrições ocorrem para que os sistemas apresentem baixo custo, seja financeiro ou de energia, além de robustez (ALMEIDA; MORAES; SERAPHIN, 2016).

Quando se conecta um sistema embarcado à rede *internet*, pode-se originar uma aplicação de *internet* das coisas (*Internet of Things*,ou, IoT). A IoT não é somente conectar "algo" à *internet*,mas além disso, tornar esse "algo" inteligente, capaz de coletar e processar informações (OLIVEIRA, 2017).

O termo *Internet of Things* foi abordado pela primeira vez por Kevin Asthon, justamente pela ideia central de objetos ("coisas") conectados à rede mundial (STEVAN JUNIOR, 2018). Entretanto a primeira aplicação conhecida de IoT foi o RFID (*Radio Frequency Indetification* - Identificação por Radiofrequência), que surgiu em 1940 durante a Segunda Guerra Mundial. O RFID foi um dispositivo utilizado para enviar por radiofrequência uma identificação única, sendo utilizado na guerra para identificar outros aviões próximos (OLIVEIRA, 2017).

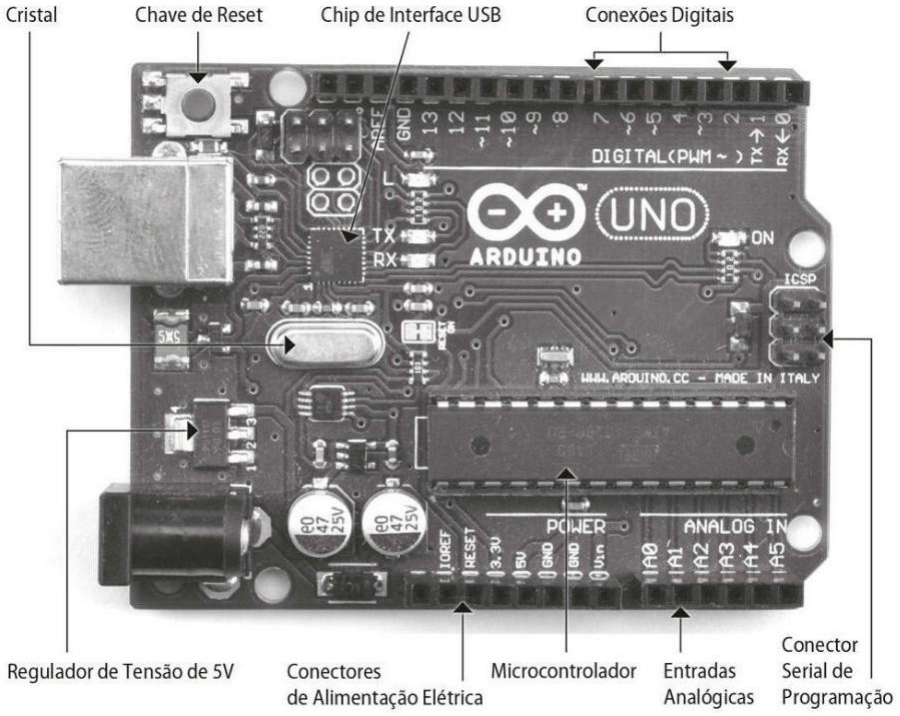
A utilização de sistemas embarcados conectados à *internet* criou um precedente de dispositivos presentes no dia-a-dia que podem enviar e receber informações úteis à distância. Tais tecnologias possibilitaram a criação das casas inteligentes, que é formada por um conjunto de objetos controlados por um sistema embarcados que estão conectados à uma rede enviando e recebendo dados, ou seja, a própria definição de IoT.

2.2 MICROCONTROLADOR ARDUINO

O arduino é uma plataforma de programação barata, funcional e de fácil programação, que possui na sua composição um microcontrolador Atmel e circuitos de entrada/saída (THOMSEN, 2014). Um microcontrolador é um pequeno computador em um chip, apresentando um processador, memória (RAM e ROM) e pinos que ligam o chip aos demais componentes do circuito (MONK, 2017).

O Arduino Uno é um dos mais utilizados, devido ao seu custo e simplicidade, sendo ideal para projetos que não requerem muito recurso. Conforme é apresentado na Figura 1, ele possui 6 entradas analógicas que medem a tensão aplicada. Possui 14 conexões digitais, sendo os dois primeiros pinos (0 e 1) para recepção e transmissão, eles podem fornecer 5V de tensão. O microcontrolador do Arduino Uno é o ATmega328 e possui 28 pinos de conexão (MONK,2017).

Figura 1: Placa Arduino Uno



Fonte: MONK,2017

A plataforma arduino é bastante popular entre os estudantes justamente pelas suas características, sendo barato e fácil de programar com a linguagem C/C++. Para programá-lo basta conectar o arduino a um computador que possua a IDE por meio de um cabo USB.

2.3 DESENVOLVIMENTO MOBILE PARA ANDROID

O sistema operacional Android foi desenvolvido no ano de 2005 pela startup Android Inc, porém seu lançamento ocorreu em outubro de 2008. O Android compartilha o mesmo kernel do Linux, escrito em C e C++ com um pouco de Assembly (DUARTE, 2016).

A grande vantagem dos dispositivos mobiles em relação aos desktops tradicionais está na comodidade trazida pelo seu tamanho, além de possuir um preço de mercado menor por apresentar menos recurso físicos que os desktop.

Segundo uma pesquisa realizada pela SimiliarWeb, o acesso de sites por aplicações mobiles é responsável pelo total de 58 % de todos os acessos, enquanto que dispositivos desktops são responsáveis pelos outros 42 % (ENGE, 2019).

Diante dos dados, o mercado de desenvolvimento de software tem entrado cada vez mais no mercado de aplicações mobiles, seja criando apps nativas para mobile ou adaptando sites para serem responsivos em todas as telas.

O Android é o sistemas operacional mobile mais utilizado atualmente. Não existe linguagens ou IDEs padrão para desenvolver para Android, entretanto é necessário o Android SDK, sendo o oficial o que engloba o ADT (Android Development Toolkit), possui ferramentas para compilação, conexão e simulação (DUARTE, 2016).

3. DESCRIÇÃO DO SISTEMA

Neste capítulo, são apresentados os materiais e métodos para o desenvolvimento do sistema. O trabalho trata-se de um sistema que simula um estacionamento inteligente com controle de vagas. Este sistema é um protótipo em hardware que verifica a presença ou não de objeto em um espaço delimitado e, caso haja, envia as informações via *WiFi*.A informação enviada pelo protótipo físico será tratada por uma aplicação em software composta de uma aplicação *backend* e *mobile*.

3.1. HARDWARE DO SISTEMA

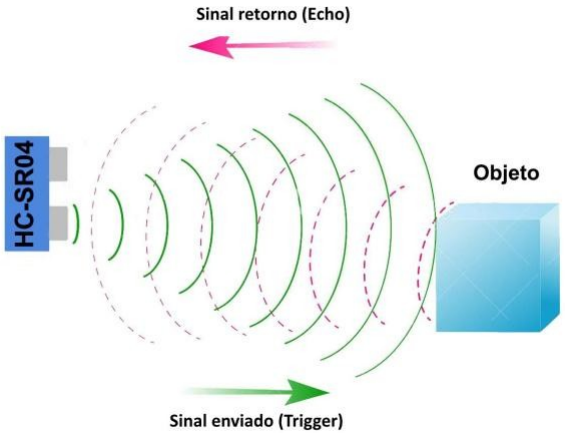
Para a construção do hardware do protótipo foram utilizados os seguintes materiais:

- Arduino Uno;
- Sensores Ultrassônicos;
- Módulo *WiFi* ESP822 ESP-01;
- Protoboard e conectores;

3.1.1 Sensor Ultrassônico

O sensor ultrassônico HC-SR04 é um componente compatível com o Arduino, que possui a capacidade de fazer leitura de distâncias entre 2 cm e 4 m. O sensor envia sinais ultrassônicos no objeto que serão refletidos ao se colidir com o objeto, conforme apresenta a Figura 2. O tempo em que o sinal percorre ao ir e retornar ao sensor é utilizado para o cálculo da distância até o objeto (THOMSEN,2011).

Figura 2: Funcionamento do sensor ultrassônico

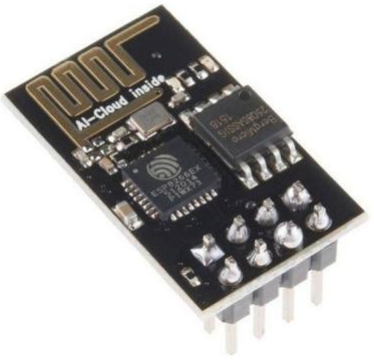


Fonte: THOMSEN,2011

3.1.2 Módulo WiFi ESP8266 ESP-01

O ESP8266 é um sistema de baixo custo e alto desempenho em chip que possui como recurso o WiFi, possibilitando que o Arduino se conecte à rede *internet*. O chip pode ser encontrado no mercado associado à módulos ou placas de desenvolvimento, como por exemplo, no módulo ESP-01 que foi utilizado para a realização do projeto descrito neste trabalho, como também pode vir associado NodeMCU que é uma placa de desenvolvimento que pode substituir um Arduino simples (BAUERMEISTER, 2018).

Figura 3: Módulo ESP-01



Fonte: OLIVEIRA,2019

Pela Figura 3, nota-se que o módulo ESP-01 possui 8 pinos, dentre os quais dois são destinados para alimentação e aterramento. Os demais pinos são:

- RST: sinal de *reset*.
- CH_PD: sinal de habilitação do chip.
- TX e RX: usado para comunicação com microcontrolador.
- GPIO0 e GPIO2: para controle de sinais de entrada/saída.

3.2. SOFTWARE DO SISTEMA

Para a parte de mobile da aplicação foram utilizados as seguintes tecnologias:

- Framework Flutter;
- Firebase;

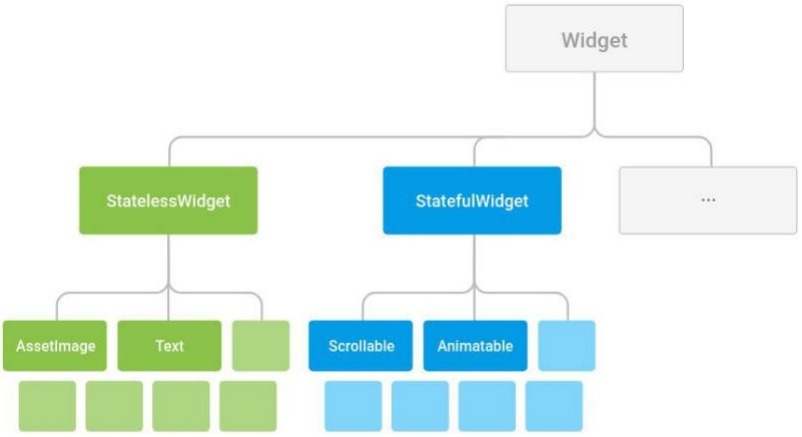
3.2.1 Flutter

O Flutter é um framework para desenvolvimento de aplicações web criado pelo Google em 2015, inicialmente sob o nome de “Sky”. Entretanto a sua primeira versão estável, o Flutter 1.0, foi a mercado no dia 4 de dezembro de 2018 (ZAMMETTI, 2020).

A principal vantagem de utilizar o Flutter está no widgets que são componentes de designs que já vão associado com o framework, ou seja, não fica da responsabilidade do sistema operacional definir como os componentes da tela irão se comportar. Portanto, a utilização do Flutter garante que a aplicação irá se comportar da mesma forma em diferentes dispositivos (ZAMMETTI, 2020).

No Flutter os widgets normalmente são compostos por vários outros widgets, em que os widgets “filhos” herdam características do seu “pai”. Portanto podemos ter numa tela um widget responsável por criar um bloco centralizado na tela e dentro dele pode ser inserido um widget que exibe um texto. Há dois tipos de widgets, os que alteram o seu estado dinamicamente (*StatefulWidget*) e os que permanecem imutáveis a não ser que sejam criados uma novas instâncias (*StatelessWidget*). A Figura 4 apresenta o sistema de hierarquia do Flutter.

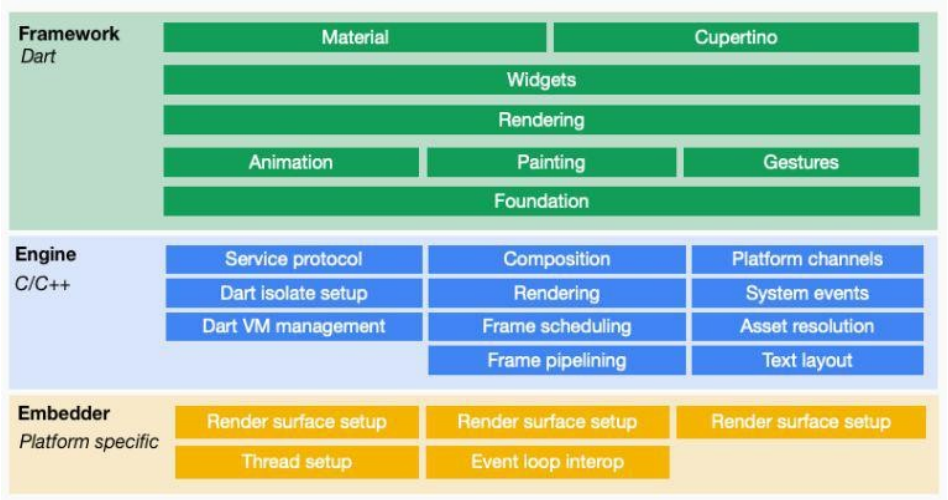
Figura 4: Hierarquia de Classes



Fonte: FLUTTER,2020

O Flutter possui três camadas conforme mostra a Figura 5. A camada de baixo (*Embedder*) é a camada de comunicação com a plataforma Android ou IOS. A camada intermediária (*Engine*) é implementada na linguagem C/C++ e é responsável pela execução do Flutter. Por fim a camada de framework é a camada no qual o desenvolvedor irá agir, utilizando a linguagem de programação do Flutter Dart para criar sua aplicação.

Figura 5: Camadas do Flutter



Fonte: FLUTTER,2020

3.2.2 Firebase Cloud Firestore

O Firebase Cloud Firestore é um banco de dados NoSQL hospedado em nuvem e de fácil acesso por aplicativos Web e móveis. Os banco de dados NoSQL são modelos de representação mais recentes no mercado e que não seguem o modelo relacional tradicional. Segundo Renato Groffe (2016), existe. quatro principais tipos de banco de dados NoSQL, que são:

- Banco de dados orientado a documento: neste tipo de banco de dados, os dados são representados por documentos dentro de uma coleção, embora seja semelhante ao modelo relacional, a principal diferença é que os documentos não possuem colunas pré-definidas (estrutura flexível).
- Banco de dados do tipo chave-valor: neste banco de dados, os dados são formados por um conjunto de chaves e seus respectivos valores. Cada conjunto possui uma chave que funciona como identificador. A sua estrutura também é flexível.
- Banco de dados orientado a colunas: este modelo é bastante divergente do tradicional, nele uma linha representa um conjunto de dados relacionados, ocupando várias colunas. Portanto a organização dos dados ocorre tendo como base as colunas.
- Banco de dados orientado a grafos: neste banco de dados, os grafos representam o relacionamento entre os conjuntos de dados.

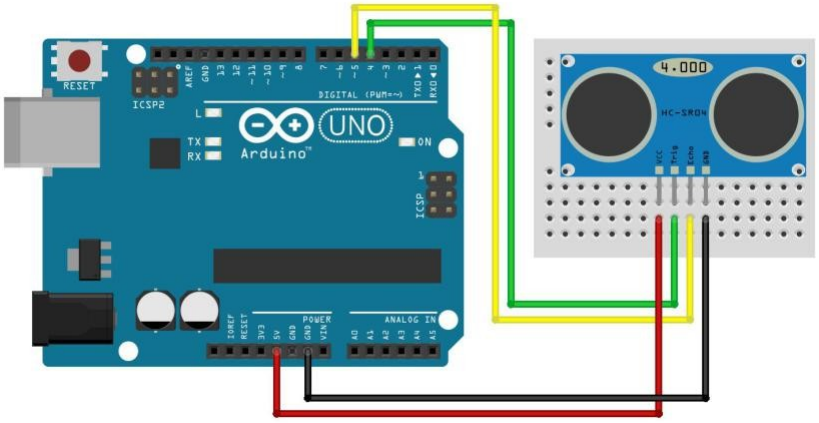
O modelo de dados do Cloud Firestone é o do tipo documento, ou seja os dados são tratados como documentos flexíveis e são agrupados em coleções. Tais documentos são compatíveis com os tipos de dados básicos, como strings, inteiros, booleanos, array, data, entre outros. Um dos recursos mais conhecidos da banco é a sua sincronização em tempo-real entre o banco e aplicação, garantindo que os dados atualizados serão enviados para aplicação sem que seja necessário recuperar todo o banco (CORAZZA, 2018).

4 RESULTADOS E DISCUSSÕES

O protótipo proposto simula uma aplicação de *internet* da coisas, com um servidor na nuvem, uma aplicação mobile e um sistema em hardware com sensores. Portanto, seu desenvolvimento ocorreu dividido em duas etapas, sendo a etapa de implementação física e uma etapa de implementação do software.

A primeira ação a ser realizada foi a integração entre os sensores ultrassônicos e arduino realizando os testes dos sensores e simulando o funcionamento do protótipo, ao verificar a presença ou não de um objeto. A esquematização do funcionamento do sensor integrado ao Arduino é apresentado na Figura 7:

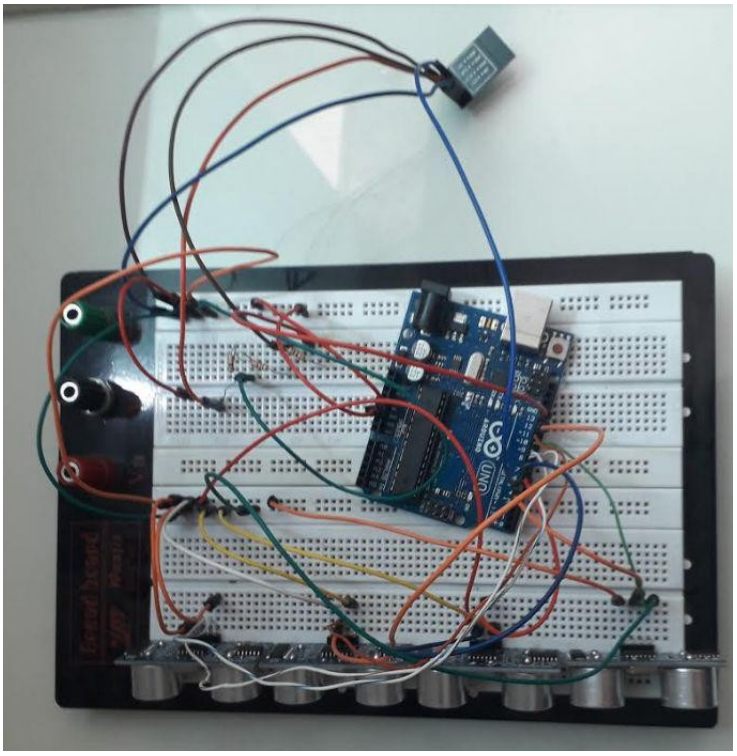
Figura 6: Conexão entre o sensor e o Arduino



Fonte: THOMSEN, 2011

Posteriormente foi realizado o teste do módulo WiFi ESP-01 e a integração com os sensores ultrassônicos. O protótipo é apresentado na Figura 7, nele foram utilizados quatro sensores ultrassônicos, sendo que cada sensor representa uma vaga do estacionamento. A vaga fica no estado ocupado quando o sensor detecta um objeto à 4 cm de distância, após isso é enviado uma requisição HTTP pelo módulo WiFi, que será visualizada na aplicação *mobile*.

Figura 7: Hardware integrado



Fonte: Autor, 2020

Com todos os dispositivos necessários no hardware em funcionamento, foi feito o estudo do framework Flutter, utilizado na implementação do sistema mobile. O primeiro passo na implementação do software foi construir o banco de dados em nuvem por meio do Firebase e a configuração do Flutter na IDE Visual Studio. Após a criação do projeto foi feito a conexão entre o Firebase e o projeto em Flutter.

4.1 BANCO DE DADOS

Para armazenar os dados gerados pela aplicação foi utilizado o Firebase, visto a fácil integração tanto com o Arduino como com a aplicação mobile e por já estar disponibilizado na nuvem. O banco de dados pode ser criado facilmente sem a necessidade de realizar instalação, basta acessar o site e criar uma conta, com isso vários serviços estarão disponíveis, entre eles, o Firebase.

O Firebase dispõe de dois tipos de banco de dados, o *Cloud Firestore* e o *Realtime Database*. Para o projeto foi escolhido o *Cloud Firestore* pela facilidade em trabalhar com

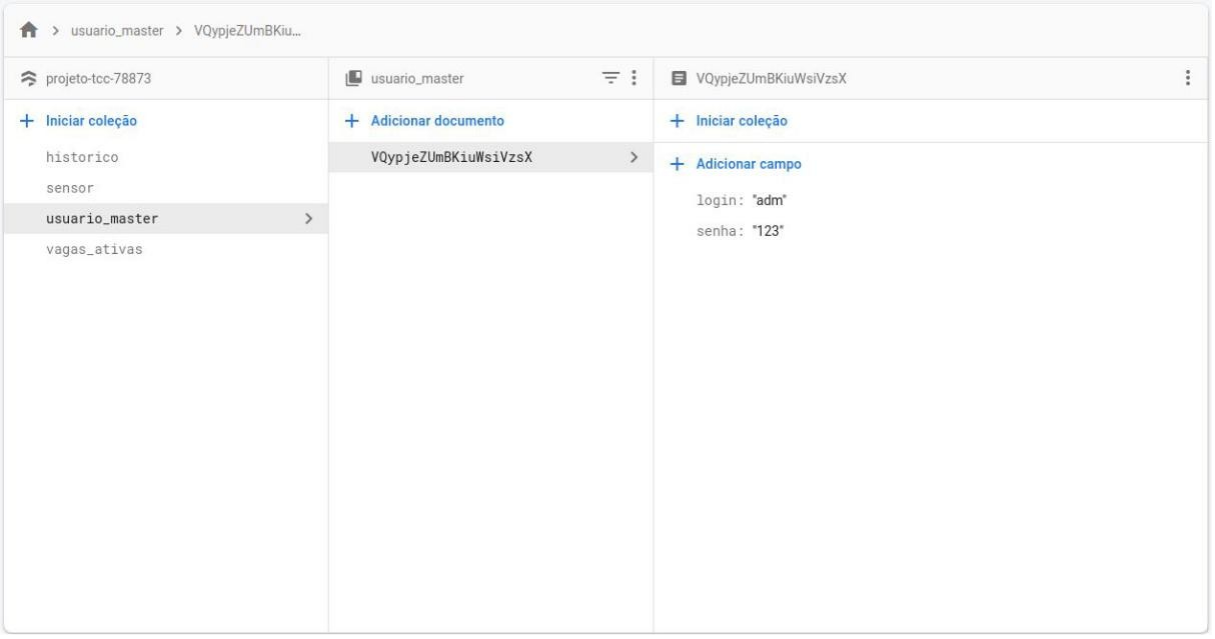
dados com hierarquia mais complexa e pela facilidade em realizar manipulações nesses dados do ponto de vista *back-end*.

Para o projeto foi criado as seguintes coleções:

- usuário_master: armazena dados do usuário que acessa a aplicação, os dados são o *login* e a senha.
- vagas_ativas: armazena dados do estado atual das vagas ocupadas. Possui dois atributos: uma identificação do sensor e a data em formato *timestamp* inicial em que a vaga foi ocupada.
- historico: armazena todo o histórico da aplicação, os atributos são os mesmos da vagas_ativas com a adição de um atributo em formato de data em que o objeto foi retirado da vaga.
- sensor: armazena o nome atribuído ao sensor, ou vaga.

Vale ressaltar que todos objetos de coleção possui por padrão um identificador que pode ser gerado automaticamente pelo firebase. A Figura abaixo mostra a coleção usuário_master. Na primeira coluna são apresentadas as coleções, em seguido tem os documentos das coleções que representam cada instância no banco, sendo a segunda coluna os identificadores para cada documento e a terceira os atributos do documento selecionado.

Figura 7: Coleção usuario_master



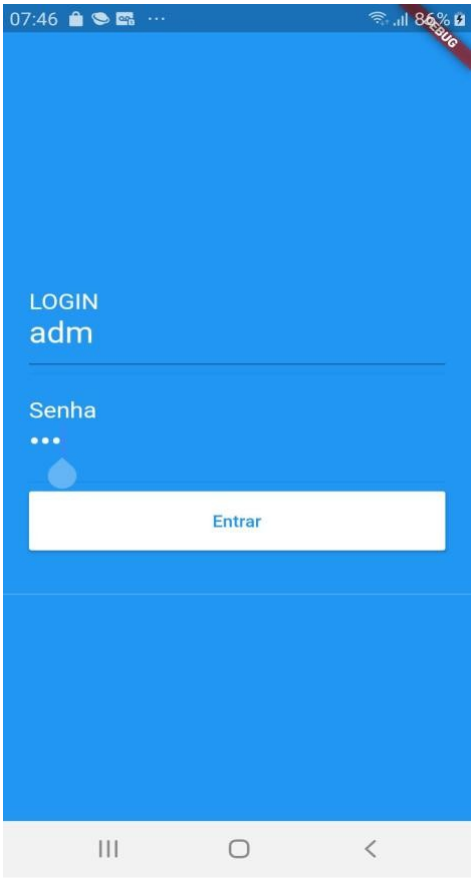
Fonte: Autor, 2020

4.2 SOFTWARE MOBILE

A aplicação *mobile* é responsável por apresentar os dados do banco de dados organizado para o operador. Portanto, tem-se três telas, sendo a primeira para que o operador entre no sistema, seguido de uma tela com os estados atuais da vagas e, por fim, uma tela que apresenta o histórico de cada vaga.

A primeira tela da aplicação, mostrada na Figura 9, é a tela de *login* do sistema, a função desta tela é garantir que apenas usuários com permissão possam acessar as informações disponíveis no sistema, para isso é necessário que ele seja um documento na coleção usuario_master no banco de dados.

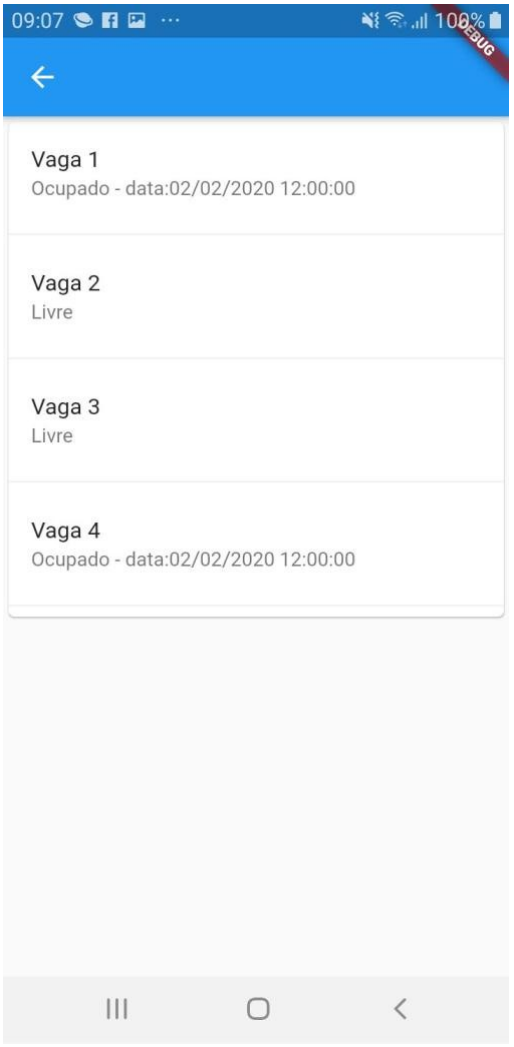
Figura 9: Tela de *Login*



Fonte: Autor, 2020

Após obter o acesso ao sistema, o usuário será apresentado à tela principal, apresentada na Figura 10, que irá conter todo o estado atual das vagas observadas. Nesta tela será informada se a vaga se encontra no estado livre, sem um objeto sendo detectado pelo sensor, ou estado ocupado. No caso da vaga estar ocupada é informado a data com horário na qual ela passou do estado livre para ocupado.

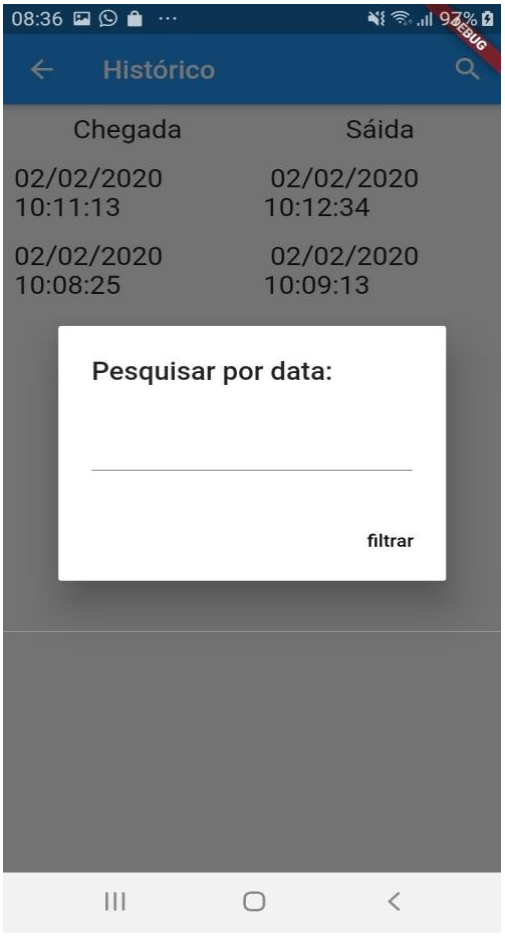
Figura 10: Tela *principal*



Fonte: Autor, 2020

Caso seja selecionado uma vaga na tela principal por meio do click no *card* de exibição da vaga, o operador é apresentado à tela de histórico, mostrada na Figura 11. Na tela do histórico é apresentado todo o histórico de entrada e saída na vaga selecionada, ordenado pela data. Também há a possibilidade do operador selecionar a data na qual deseja realizar um filtro, ao selecionar o *icon*de busca e digitar a data a ser filtrada.

Figura 11: Tela com histórico do sensor



Fonte: Autor, 2020

5 CONSIDERAÇÕES FINAIS

A automação em processos do cotidiano é uma tendência no mercado de tecnologia e que exige desenvolvedores qualificados para suprir a necessidade do mercado. O protótipo abordado no presente trabalho é um exemplo de automação no cotidiano, permitindo um controle sobre vagas de estacionamentos, possibilitando a um operador verificar o estado da vaga e o seu histórico.

Uma grande problemática do protótipo é a utilização de sensor ultrassônico para verificar a presença de vaga livre, visto que o sensor pode detectar não somentes carros ou objetos grandes, mas também objetos pequenos que estejam de passagem, por exemplo, uma pessoa que passe próximo ao sensor.

Para trabalhos futuros, pode-se expandir o sistema trazendo uma visualização em mapa das vagas do estacionamento, assim se torna mais fácil identificar a localização de cada vaga a medida que seu número aumenta. Outra sugestão é melhorar o sistema em hardware, para permitir que o sistema seja mais preciso na indicação de vaga ocupada. Uma sugestão é aumentar o número de sensores em uma vaga, para verificar a dimensão do objeto antes de enviar uma confirmação para o banco em nuvem.

REFERÊNCIAS

ALMEIDA, Rodrigo Maximiano A.; MORAES, Carlos Henrique V.; SERAPHIN, Thatyana F. Piola. **Programação de sistemas embarcados:** Desenvolvendo software para microcontroladores em linguagem C. Rio de Janeiro: Elsevier, 2016. 488 p.

ASHTON, Kevin. **Essa coisa da "Internet das Coisas".** 2009. Disponível em: <https://www.rfidjournal.com/articles/view?4986>. Acesso em: 14 fev. 2020.

BAUERMEISTER, Giovanni. **Guia do Usuário do ESP8266.** 2018. Disponível em: <https://www.filipeflop.com/blog/guia-do-usuario-do-esp8266/>. Acesso em: 2 fev. 2020.

CORAZZA, Paulo Victor. **Um aplicativo multiplataforma desenvolvido com Flutter e NoSQL para o cálculo da probabilidade de apendicite.** 2018. 64 f. TCC (Graduação) - Curso de Ciência da Computação, Ufrs, Porto Alegre, 2018.

DUARTE, Luiz. **Criando apps para empresas com Android.** Gravataí:Luiztools, 2016.243 p.

ENGE, Eric. **Mobile vs Desktop Traffic in 2019.** 2019. Disponível em: <https://www.perficientdigital.com/insights/our-research/mobile-vs-desktop-usage-study>. Acesso em: 3 fev. 2020.

FLUTTER. **Technical overview.** Disponível em: <https://flutter.dev/docs/resources/technical-overview>. Acesso em: 03 fev. 2020.

GROFFE, Renato. **Banco de dados NoSQL**: Uma visão geral**.** Disponível em: <https://imasters.com.br/banco-de-dados/bancos-de-dados-nosql-uma-visao-geral>. Acesso em: 03 fev. 2020.

OLIVEIRA, Euler. **Como usar com Arduino - Módulo WiFi ESP8266 ESP-01.** 2019. Disponível em: <https://blogmasterwalkershop.com.br/arduino/como-usar-com-arduino-modulo-wifi-esp8266-esp-01/>. Acesso em: 2 fev. 2020.

OLIVEIRA, Sérgio de. **Internet das Coisas:** com ESP8266, Arduino e Raspberry Pi. São Paulo: Novatec, 2017. 240 p.

LIMA, Samantha. Frota de veículos cresce mais onde há menos dinheiro no país. Disponível em: <http://www.ufjf.br/ladem/2010/02/09/frota-de-veiculos-cresce-mais-onde-ha-menos-din hei ro-no-pais/>. Acesso em: 14 fev. 2020.

MONK, Simon. **Programação com Arduino::** Começando com Sketches. 2. ed. Porto Alegre: Bookman, 2017. 200 p.

STEVAN JUNIOR, Sergio Luíz. **IOT - Internet das coisas::** Fundamentos e aplicações em Arduino e NodeMCU. São Paulo: Érica, 2018. 224 p.

THOMSEN, Adilson. **Como conectar o sensor ultrassônico HC-SR04 ao Arduino.** 2011. Disponível em: <https://www.filipeflop.com/blog/sensor-ultrassonico-hc-sr04-ao-arduino/>. Acesso em: 3 fev. 2020.

THOMSEN, Adilson. **O que é Arduino.** 2014. Disponível em: <https://www.filipeflop.com/blog/o-que-e-arduino/>. Acesso em: 2 fev. 2020.

VICENTE, Thiago Pires Romanelli. **Controle Inteligente de Vagas para Estacionamento Utilizando o Conceito de Internet das Coisas.** 2016. 92 f. TCC (Graduação) - Curso de Engenharia Elétrica, USP, São Paulo, 2016.

ZAMMETTI, Frank. **Flutter na prática::** Melhore seu desenvolvimento mobile com o SDK open source mais recente do Google. São Paulo: Novatec, 2020.