

SISTEMA DE CONTROLE FINANCEIRO UTILIZANDO RECONHECIMENTO DE IMAGEM



Orientadora: Prof. Dra. Simone das Graças Domingues Prado

Thiago Hoffart Vieira

RA:161026524

Sumário

- Introdução
- Fundamentação Teórica
- Ferramentas
- Desenvolvimento
- Conclusão
- Próximos Passos

Introdução

Introdução

Smartphones

- Processamento
- Portabilidade

Machine Learning

- Otimização
- *Datasets*
- Reconhecimento de
imagens

App Financeiro

- Orçamento

Problema

- Integração do desenvolvimento urbano com a tecnologia da informação, comunicação e internet;
- Modelos inteligentes para a resolução de problemas cotidianos;
- Soluções eficientes para o reconhecimento de objetos em um ambiente inteligente;

Objetivos

- Construir a infraestrutura necessária para o desenvolvimento;
- Desenvolver API para a comunicação do aplicativo com o banco de dados;
- Elaborar aplicativo em Flutter;
- Implementar modelo de treinamento dos dados no TensorFlow.



Justificativa

- Otimizar o gerenciamento de gastos pessoais;
- Integrar a tecnologia de reconhecimento de imagem em smartphones;
- Diminuir o tempo dispendido pelos usuários para a realização de tarefas cotidianas como a confecção de orçamentos e listas de compras.

Fundamentação Teórica

Fundamentação Teórica

Inteligência Artificial

Executar tarefas inteligentes como reconhecer objetos, nossa linguagem e tomar decisões de maneira independente.

Machine Learning:

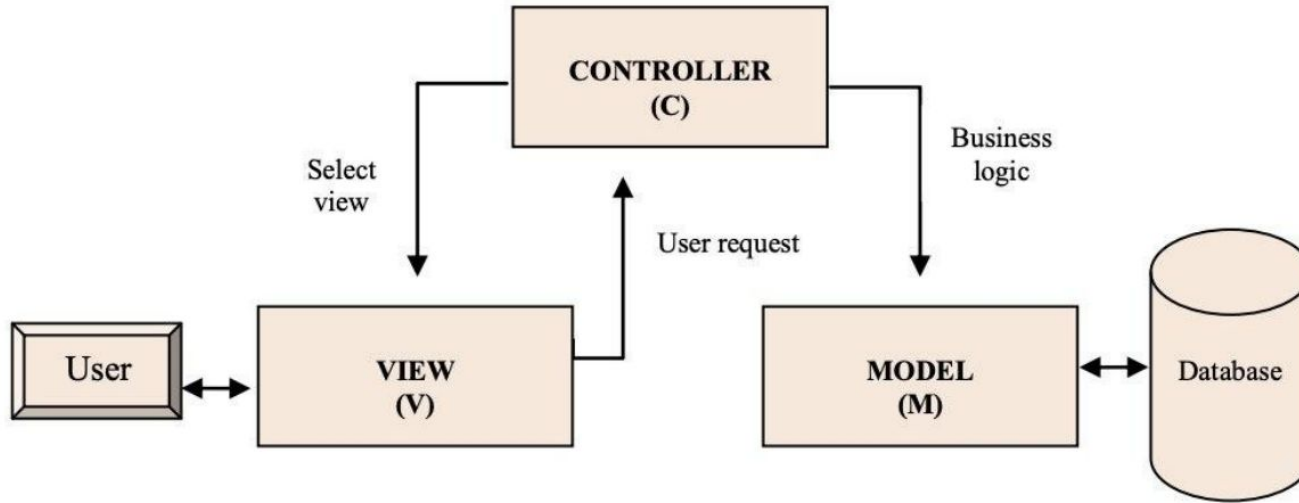
- Melhorar seu próprio desempenho utilizando dados de exemplos ou experiências passadas.

Arquitetura de Software

Entender como os sistemas de software são desenhados e construídos, por meio de um conjunto de decisões de design do sistema.

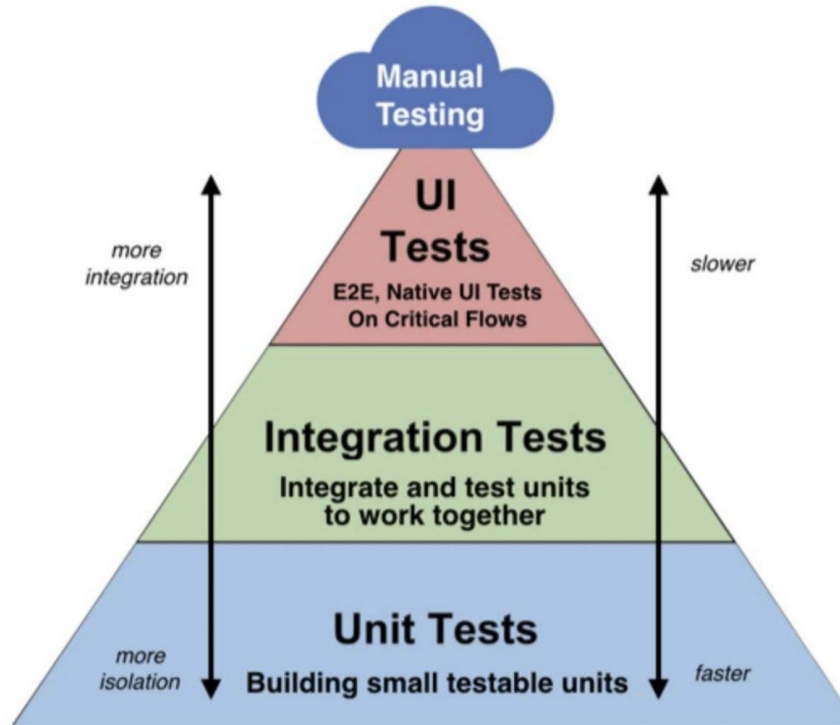
- Arquitetura MVC
- Framework
- State Management
- Testes de Software

Arquitetura MVC



Fonte: Sarker; Apu (2014).

Relação de testes de software



Fonte: https://miro.medium.com/max/1494/1*1qWygfNJqWQ4VCyjecQ6Eg.png. Acesso em: 29 de novembro de 2020.

Fundamentação Teórica

Banco de dados

Coletânea de dados virtual, utilizado para salvar informações.

Linguagem de Query:

- Linguagem de programação para recuperar dados do banco enviando queries, por meio de requests.

Sistema de controle de versões

Salva as mudanças feitas nos arquivos com o passar do tempo, para assim, ser possível restaurar versões anteriores.

- Local Version Control Systems
- Centralized Version Control Systems
- Distributed Version Control Systems

Fundamentação Teórica

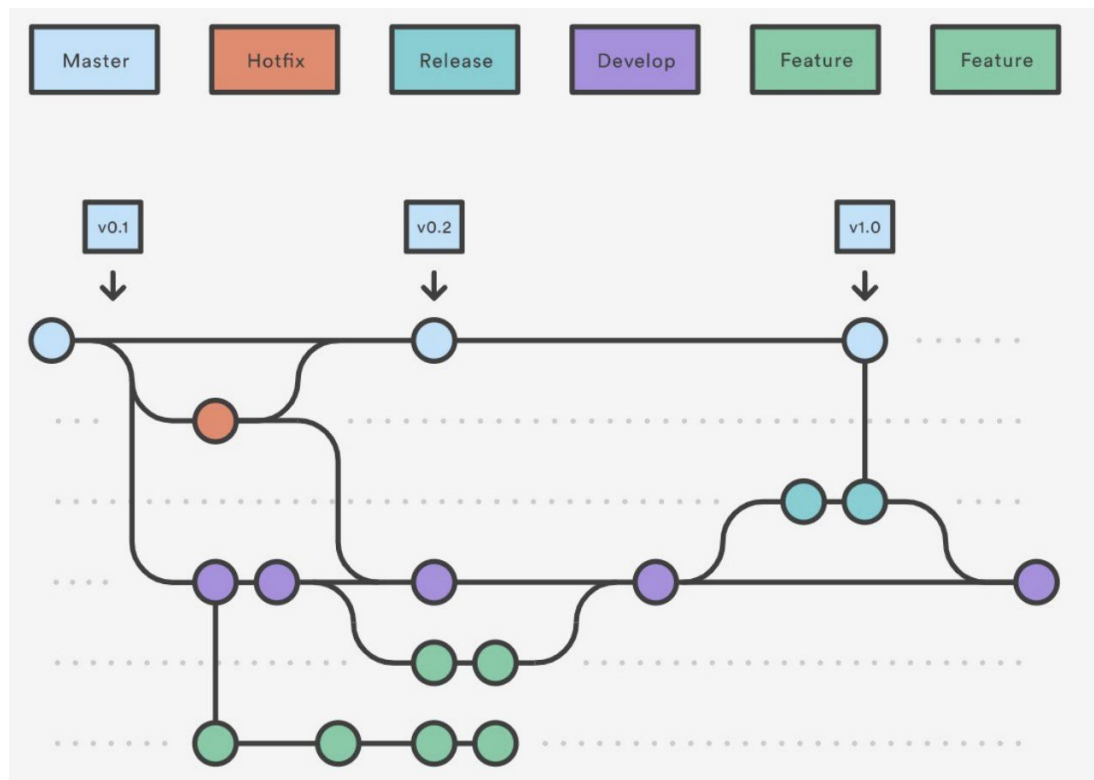
Gitflow

Fluxo de controle de versão definido por uma estrutura de branches, onde cada branch tem sua funcionalidade específica, com o objetivo de prover uma estrutura mais definida de desenvolvimento.

CI/CD

Conjunto de princípios que buscam garantir uma maior eficiência no fluxo de entrega do software e na integração da equipe de desenvolvimento, com o objetivo final de gerar valor da maneira mais segura possível.

Estrutura de branches utilizando Gitflow



Fonte: Atlassian (2020).

Ferramentas

Ferramentas

Linguagens

- Dart
- PHP
- GraphQL

Softwares

- Git
- Docker
- Pivotal Tracker
- PostgreSQL

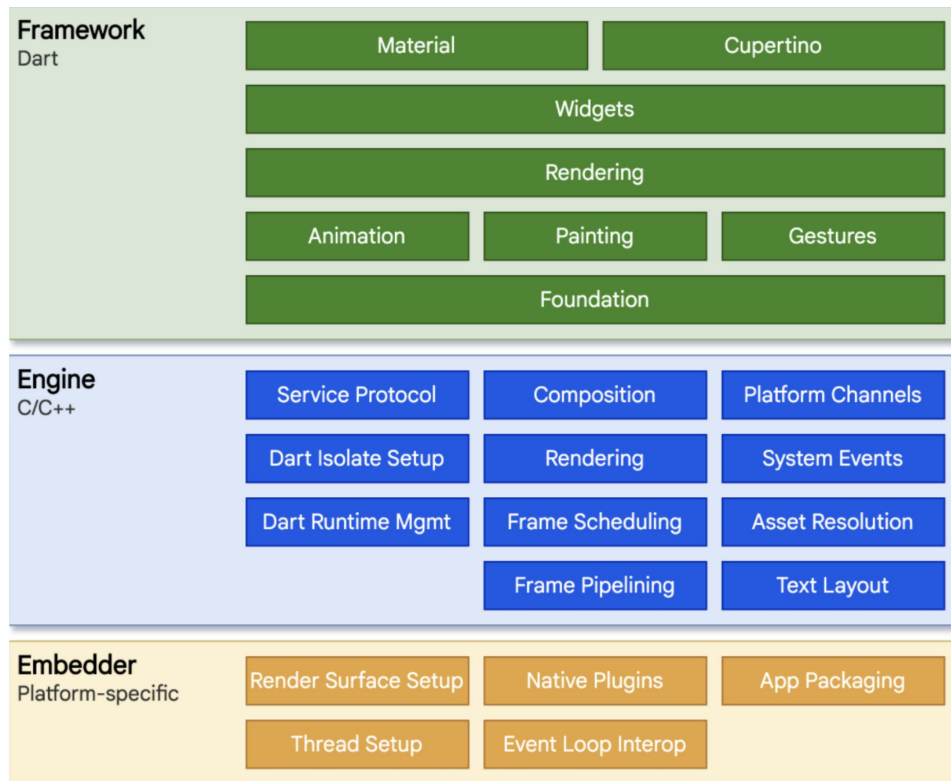
Ferramentas

Frameworks

- Flutter
- Laravel
- Lighthouse

Bibliotecas

- Tensorflow

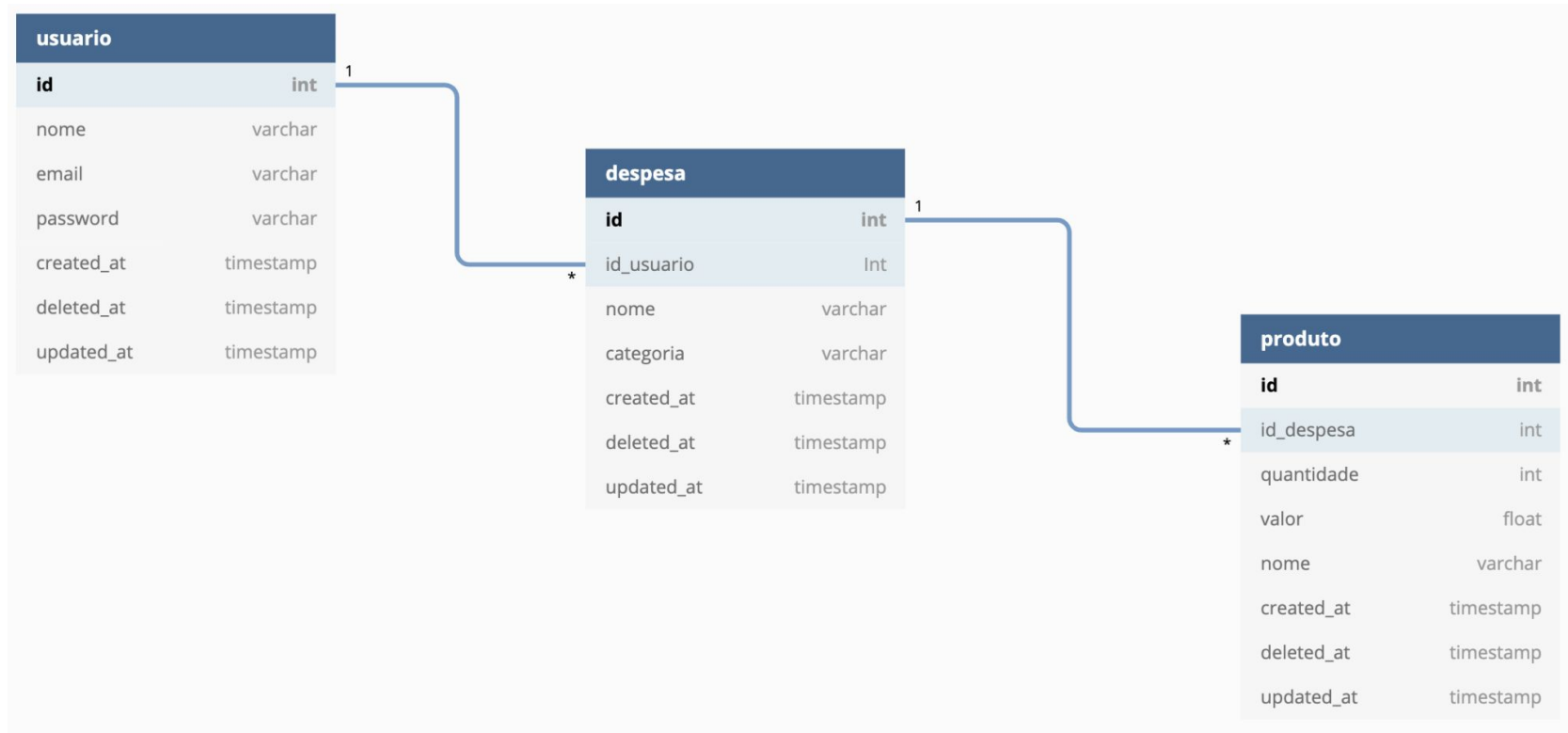


Arquitetura do
Flutter

Desenvolvimento

Pré-desenvolvimento

- Levantamento dos requisitos
 - Telas
 - Funcionalidades
- Modelagem do banco de dados
- Geração dos projetos
 - Flutter
 - Laravel
- Infraestrutura
 - Docker
 - CI



MER do banco de
dados

Projeto

- Desenvolvimento da API
 - Migrations: criação das tabelas dentro do banco de dados;
 - Models: realiza as configurações de cada tabela, como definir campos privados e públicos, chaves primárias, timestamps e relações entre as tabelas;
 - GraphQL: definição de queries, mutations e types das tabelas;
 - Testes de integração: garante o funcionamento da API.

```

class CreateProdutoTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('produto', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->integer('id_despesa')->unsigned();
            $table->foreign('id_despesa')->references('id')->on('despesa')->onDelete('restrict');
            $table->bigInteger('quantidade');
            $table->float('valor', 8, 2);
            $table->string('nome');
            $table->timestamps();
            $table->softDeletes();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('produto');
    }
}

```

```

Route::post('login', 'Auth\LoginController@login');
Route::get('logout', 'Auth\LoginController@logout')->middleware('auth:api');

```

```

class ArquivoController extends Controller
{
    public function uploadImagem(Request $request)
    {
        $files = $request->files->all();

        StorageService::storeImagens($files);

        return Response::send(true);
    }
}

```

```

class Despesa extends Model
{
    use SoftDeletes,
        CastsEnums;

    protected $table = 'despesa';

    protected $dates = ['created_at', 'updated_at', 'deleted_at'];

    protected $guarded = [
        'id',
        'created_at',
        'updated_at',
        'deleted_at',
    ];

    protected $fillable = [
        'id_usuario',
        'nome',
        'categoria',
    ];

    //===== Mapping =====

    public function usuario(): BelongsTo
    {
        return $this->belongsTo(Usuario::class, 'id_usuario');
    }
}

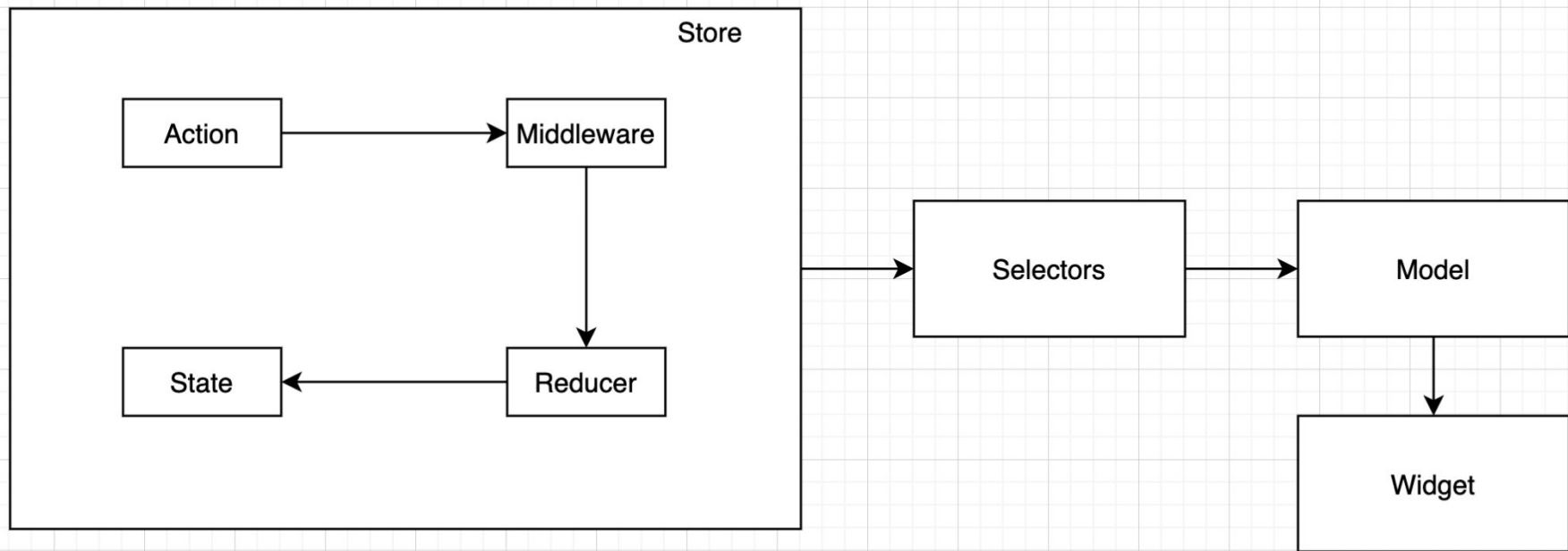
```

Exemplos de código da API

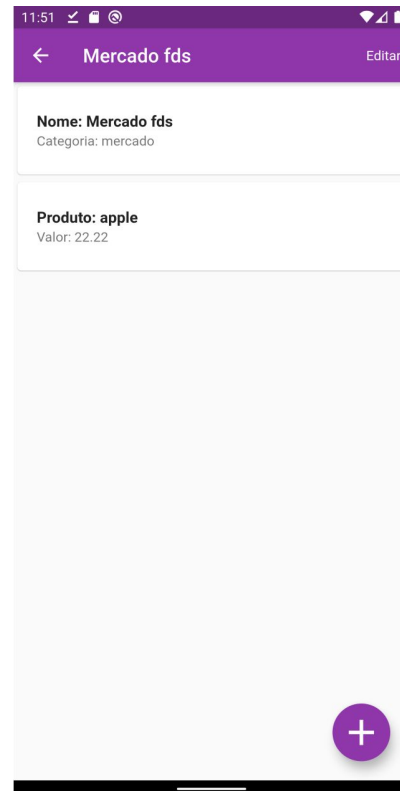
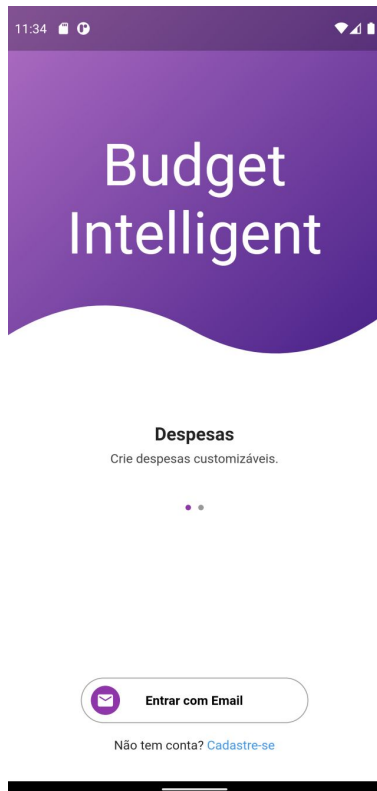
Projeto

- Desenvolvimento do aplicativo
 - Modelagem do State Management: definição da arquitetura do estado da aplicação;
 - Construção das páginas: seguiram o fluxo de interação do usuário dentro do aplicativo;
 - Requisições à API: criação de uma classe base, com métodos auxiliares, utilizando a biblioteca GraphQL para o Flutter;
 - Integração com o Tensorflow: criação uma classe base para fazer a conexão entre o aplicativo e a biblioteca.

Flutter_redux



Fluxo do estado do
aplicativo



Telas do aplicativo

Tensorflow

- Biblioteca TFLite;
- Modelos SSD_MobileNet e YOLO;
- Modelo pré-treinado com dataset disponibilizado pelo TFlite;
- Implementação de classe base.

```

class _ImagePickerTfliteState extends State<ImagePickerTflite> {
  File imageURI;
  String result;
  String path;
  final ImagePicker picker = ImagePicker();

  Future getImageFromCamera() async {
    var image = await picker.getImage(source: ImageSource.camera);
    var imageFile = File(image.path);
    setState(() {
      imageURI = imageFile;
      path = imageFile.path;
    });
    await classifyImage();
  }

  Future classifyImage() async {
    await Tflite.loadModel(
      model: 'assets/ssd_mobilenet.tflite',
      labels: 'assets/ssd_mobilenet.txt'
    );
    var output = await Tflite.detectObjectOnImage(path: path);
    setState(() {
      result = output.toString();
    });
    widget.updateName(output.first['detectedClass']);
  }

  Future getImageFromGallery() async {
    var image = await picker.getImage(source: ImageSource.gallery);
    var imageFile = File(image.path);
    setState(() {
      imageURI = imageFile;
      path = imageFile.path;
    });
    await classifyImage();
  }
}

```

Classe base de integração com o
Tensorflow


11:52

<

Cadastro Produto

Selecionar imagem da camera

Selecionar imagem da galeria



Nome
apple

5/250

Quantidade
2

1/250

Valor
R\$ 22,22

✓

Reconhecimento de imagem no
aplicativo

Conclusão

Conclusão

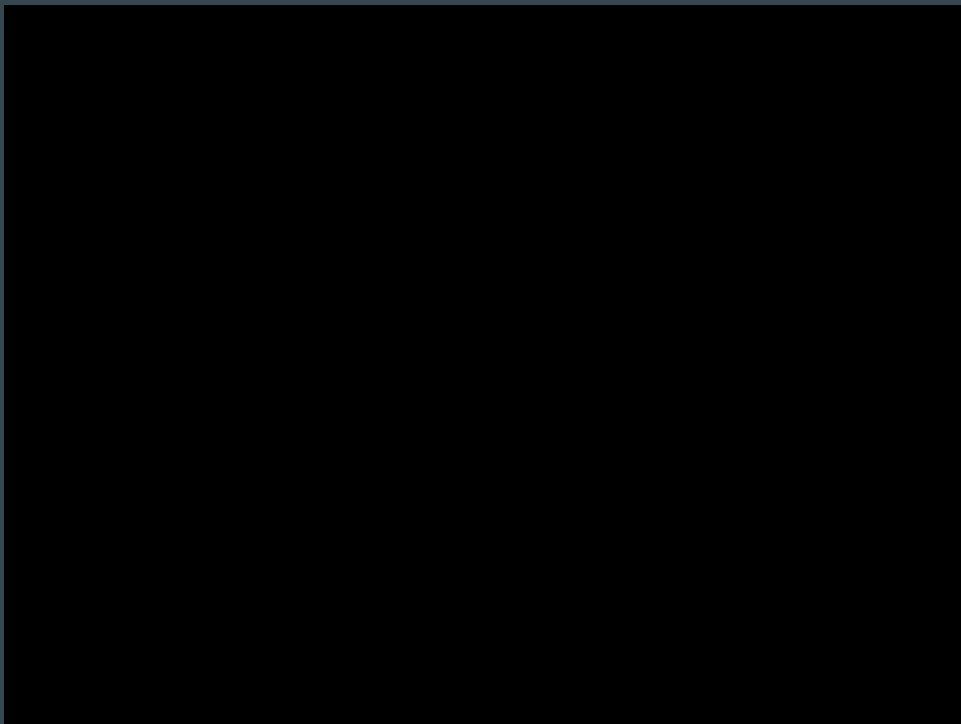
- O uso de softwares que podem facilitar a vida do usuário, sendo utilizado em seu cotidiano, refletem os rumos da Ciência da Computação;
- Machine learning na melhoria da qualidade de vida da sociedade;
- Utilização de tecnologias modernas mostram como o surgimento de novos conceitos e paradigmas são importantes para o desenvolvimento da Ciência da Computação.

Próximos Passos

Próximos Passos

- Realização de mais testes de reconhecimento de imagem, para que a ferramenta tenha uma melhor precisão e desempenho;
- Utilizar outros datasets e treinar o modelo utilizado para reconhecer outras categorias de objetos e aprimorar a presente categoria;
- Implementar o Tensorflow na API, tendo em vista que o poder de processamento dos celulares ainda é limitado;
- Realizar melhorias no aplicativo e publicá-lo nas lojas AppStore e PlayStore.

Aplicativo



Referências

ALPAYDIN, E. Introduction to machine learning. Cambridge, Massachusetts: The MIT Press, 2020. 683 p. (4 Ed).

ALSING, O. Mobile Object Detection using TensorFlow Lite and Transfer Learning. 2018. 78 p. — Trabalho de conclusão de curso (Engenharia e Ciência da Computação) – KTH Royal Institute of Technology, School of Electrical Engineering and Computer Science, Estocolmo, Suécia, 2018.

ATLASSIAN. Gitflow Workflow. 2020. Disponível em: <<https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>>. Acesso em: 23 de outubro de 2020.

FREITAS, T. Os três tipos de aprendizado no machine learning, um ramo da inteligência artificial. 2019. Disponível em: <<https://www.startse.com/noticia/nova-economia/machine-learning-inteligencia-artificial-aprendizado>>. Acesso em 22 de Março de 2020.

MEDVIDOVIC, N.; TAYLOR, R. Software architecture: foundations, theory, and practice. In: ACM/IEEE International Conference on Software Engineering. Cape Town, South Africa: ICSE, 2010.

MULFARI, D.; MINNOLO, A. L.; PULIAFITO, A. Building tensorflow applications in smart city scenarios. In: IEEE INTERNATIONAL CONFERENCE ON SMART COMPUTING. n. 3, 2017. Hong Kong, China, Anais, Hong Kong: SMARTCOMP, 2017.

POP, D. P.; ALTAR, A. Designing an mvc model for rapid web application development. Procedia Engineering, v. 69, p. 1172–1179, 2014.

TENSORFLOW. Tensorflow. 2020. Disponível em: <<https://www.tensorflow.org/>>. Acesso em: 29 de outubro de 2020.

Obrigado!