UNIVERSIDADE FEDERAL DO PARANÁ

ALLAN COSTA VIEIRA

FERNANDO DE OLIVIEIRA MELO

BASE DE DADOS - FUNGOS PATOLÓGICOS

CURITIBA

2016

ALLAN COSTA VIEIRA

FERNANDO DE OLIVIEIRA MELO

BASE DE DADOS - FUNGOS PATOLÓGICOS

Trabalho de conclusão de curso apresentado como requisito à conclusão do curso de Tecnologia em Análise e Desenvolvimento de Sistemas, Setor de Educação Profissional e Tecnologia, Universidade Federal do Paraná.

Orientador: Jeroniza Marchaukoski

CURITIBA

2016

**RESUMO**

Arquivos GFF e GBK guardam as mais diversas informações genéticas de um organismo, sendo aplamente utilizados na Bioinformática, mas apesar de possuírem a mesma informação, cada formato as armazena de forma diferente. Este trabalho apresenta um sistema web capaz de guardar e converter os arquivos entre esses dois formatos, assim como disponibiliza-los para download. O sistema também funcionará como uma plataforma de trabalho, facilitando o armazenamento e vizualização de arquivos genéticos e possibilitando que o andamento dos projetos seja facilmente consultado, arquivos disponibilizados e usuários gerenciados de acordo com uma hierarquia estabelecida.

Palavra Chave: Bioinformática, Arquivos genéticos, Visualização de dados biológicos, GFF, GBK

**ABSTRACT**

GFF and GBK files store the most diverse genetic information of an organism, being widely used in bioinformatics, but despite having the same information, each format has different ways of storing. This paper presents a web system capable of store and convert files between these formats, and offers them for download. The system will also function as a work platform, facilitating the storage and visualization of genetic files and enabling the progress of each project to be easily consulted, files made available and users managed in accordance with an established hierarchy.

Key-word: Bioinformatics, Genetic files, Visualization of biological data, GFF, GBK

**LISTA DE FIGURAS**

FIGURA 1 – ARQUIVO FASTA.................................................................................14

FIGURA 2 – DIAGRAMA WBS..................................................................................21

FIGURA 3 – DIAGRAMA DE GANTT........................................................................22

FIGURA 4 – LINHA DO TEMPO DO DIAGRAMA DE GANTT..................................23

FIGURA 5 – LINHA DO TEMPO DO DIAGRAMA DE GANTT..................................23

FIGURA 6 – FOTO TIRADA NO NÍVEL 1 DE COMPRESSÃO.................................33

FIGURA 7 – FOTO TIRADA NO NÍVEL 5 DE COMPRESSÃO.................................34

FIGURA 8 – COMPARAÇÃO ENTRE A FOTO ORIGINAL E A IMAGEM APÓS O PROCESSAMENTO DA IMAGEM.......................................................36

FIGURA 9 - TELA DE ABERTURA............................................................................37

FIGURA 10 - TELA INICIAL DO APLICATIVO..........................................................38

FIGURA 11 - TELA DA CÂMERA..............................................................................38

FIGURA 12 - TELA PARA SELECIONAR UMA IMAGEM.........................................39

FIGURA 13 - TELA DE RESULTADOS.....................................................................40

FIGURA 14 - TELA DE RESULTADOS COM HSPS.................................................40

FIGURA 15 - IMAGEM ORIGINAL NA ABA “DEBUG”..............................................41

FIGURA 16 - IMAGEM ORIGINAL PRÉ-PROCESSADA NA ABA “DEBUG”...........42

FIGURA 17 - SEQUÊNCIA ACHADA PELO OCR NA ABA “DEBUG”......................42

FIGURA 18 - BOTÃO PARA LIMPAR OS RESULTADOS........................................43

FIGURA 19 - ALERTA DE CONFIRMAÇÃO..............................................................44

FIGURA 20 - MENU NA TELA INICIAL DO APLICATIVO.........................................45

FIGURA 21 - TELA DE INFORMAÇÕES GERAIS....................................................46

FIGURA 22 - TELA DE INFORMAÇÕES GERAIS....................................................46

FIGURA 23 - TELA DE CONFIGURAÇÕES..............................................................47

FIGURA 24 - TELA DO APLICATIVO NA PLAY STORE...........................................48

FIGURA 25 – DIAGRAMA DE CASOS DE USO.......................................................54

FIGURA 26 – DIAGRAMA ENTIDADE RELACIONAMENTO....................................62

FIGURA 27 – DIAGRAMA DE CLASSES – PACOTE ACTIVITIES...........................63

FIGURA 28 – DIAGRAMA DE CLASSES – PACOTE DATA.....................................64

FIGURA 29 – DIAGRAMA DE CLASSES – PACOTE PROCESSING......................65

FIGURA 30 – DIAGRAMA DE CLASSES – PACOTE SYSTEM................................66

FIGURA 31 – DIAGRAMA DE ATIVIDADES – USO GERAL....................................67

FIGURA 32 – DIAGRAMA DE ATIVIDADES – ALTERAR CONFIGURAÇÕES........68

FIGURA 33 – DIAGRAMA DE COMPONENTES……………………………………….69

**LISTA DE SIGLAS**

DNA -Ácido desoxirribonucleico

FASTA -Formato utilizado para armazenar sequências de bases e de

aminoácidos em arquivo texto

GenBank -Banco de dados público do National Center for Biology Information

GBK -GenBank File

GFF -General Feature Format

HTML -HyperText Markup Language

IDE -Integrated Development Environment

mRNA -RNA mensageiro

NCBI -National Center for Biotechnology Information

NR -Banco de dados não redundante disponibilizado pelo NCBI

RNA -Ácido ribonucleico

RUP -Rational Unified Process

SILA -Genome Automated Annotation System

UML -Unified Modeling Language

WBS -Work Breakdown Structure

XML -Extensible Markup Language

**LISTA DE TABELAS**

TABELA 1 – LISTA DE ALGORITMOS DO BLAST...................................................15

TABELA 2 – PLANO DE RISCOS..............................................................................24

TABELA 3 – RESULTADOS DOS TESTES DE OCR................................................31

TABELA 4 – RESULTADOS DOS TESTES DE COMPRESSÃO DO VÍRUS EBOLA...................................................................................................32

TABELA 5 – RESULTADOS DOS TESTES DE COMPRESSÃO DA BACTÉRIA HERBBASPIRILLUM.........................................................................32

SUMÁRIO

[1 INTRODUÇÃO 11](#_Toc405835694)

[1.1 OBJETIVOS 12](#_Toc405835695)

[1.2 OBJETIVOS ESPECÍFICOS 12](#_Toc405835696)

[2 FUNDAMENTAÇÃO TEÓRICA 14](#_Toc405835697)

[2.1 BIOINFORMÁTICA 14](#_Toc405835698)

[2.2 CONCEITOS BIOLÓGICOS 14](#_Toc405835699)

[2.3 CONSULTA DE INFORMAÇÕES BIOLÓGICAS 15](#_Toc405835700)

[2.3.1 Armazenamento 15](#_Toc405835701)

[2.3.2 Banco de Dados 15](#_Toc405835702)

[2.3.3 BLAST 16](#_Toc405835703)

[2.4 OCR 17](#_Toc405835704)

[2.5 PROCESSAMENTO DE IMAGENS 18](#_Toc405835705)

[2.6 SQLITE 18](#_Toc405835706)

[2.8 UML 19](#_Toc405835707)

[3 METODOLOGIA 20](#_Toc405835708)

[3.1 MODELO DE PROCESSO DE ENGENHARIA DE SOFTWARE 20](#_Toc405835709)

[3.2 PLANO DE ATIVIDADES 20](#_Toc405835710)

[3.2.1 Diagrama WBS 20](#_Toc405835711)

[3.2.2 Diagrama de Gantt 23](#_Toc405835712)

[3.3 PLANO DE RISCOS 25](#_Toc405835713)

[3.4 MATERIAIS 26](#_Toc405835714)

[3.4.1 Ambiente de Desenvolvimento 26](#_Toc405835715)

[3.4.2 Linguagem de Programação 26](#_Toc405835716)

[3.4.3 Banco de Dados 27](#_Toc405835717)

[3.4.4 Ambiente Integrado de Desenvolvimento 27](#_Toc405835718)

[3.4.5 Ferramenta para Criação do Diagrama WBS 27](#_Toc405835719)

[3.4.6 Ferramenta para Criação do Diagrama de Gantt 27](#_Toc405835720)

[3.4.7 Ferramenta para Modelagem UML 28](#_Toc405835721)

[3.4.8 Ferramenta para Criação do Diagrama Entidade Relacionamento 28](#_Toc405835722)

[3.5 FERRAMENTA PARA CONTROLE DE VERSÃO 28](#_Toc405835723)

[3.6 BIBLIOTECAS 28](#_Toc405835724)

[3.6.1 BioJava 29](#_Toc405835725)

[3.6.2 Tess-two 29](#_Toc405835726)

[4 DESENVOLVIMENTO DO PROJETO 30](#_Toc405835727)

[4.1 VALIDAÇÃO DO SOFTWARE 31](#_Toc405835728)

[4.1.1 Testes de OCR 31](#_Toc405835729)

[4.1.2 Testes de Compressão 33](#_Toc405835730)

[4.1.3 Testes de Iluminação 36](#_Toc405835731)

[5 APRESENTAÇÃO DO SOFTWARE 39](#_Toc405835732)

[5.1 INSTALAÇÃO 50](#_Toc405835733)

[6 CONSIDERAÇÕES FINAIS 51](#_Toc405835734)

[REFERÊNCIAS 52](#_Toc405835735)

[APÊNDICE A - DIAGRAMA DE CASOS DE USO 56](#_Toc405835736)

[APÊNDICE B - ESPECIFICAÇÃO DOS CASOS DE USO 57](#_Toc405835737)

[APÊNDICE C - DIAGRAMA ENTIDADE RELACIONAMENTO 64](#_Toc405835738)

[APÊNDICE D- DIAGRAMA DE CLASSES 65](#_Toc405835739)

[APÊNDICE F - DIAGRAMA DE COMPONENTES 71](#_Toc405835740)

# 1 INTRODUÇÃO

A Bioinformática é uma ciência que atua na manipulação e extração de conhecimento de dados biológicos. O crescimento da área se deve à necessidade de manipular quantidades enormes de dados genéticos e bioquímicos, utilizando recursos computacionais para catalogá-las, organizá-las e estruturá-las (RASHIDI, 2000).

# 1.1 OBJETIVOS

Desenvolver uma plataforma web capaz de guardar arquivos genéticos, organizando-os de acordo com o organismo estudado e o projeto em que se encontra. Além disso, fazer com que essa plataforma tenha um sistema de gerenciamento hierarquico de usuários, possibilitando seguir os moldes de organização do grupo de pesquisa da Bioinformática.

1.2 OBJETIVOS ESPECÍFICOS

Este trabalho tem os seguintes objetivos específicos:

* + Possibilitar ao sistema fazer upload de arquivos GFF e GBK
  + Possibilitar ao sistema fazer a conversão de arquivos GFF para GBK e vice-e-versa
  + Possibilitar ao visitante o acesso a dados públicos
  + Possibilitar ao visitante submeter um projeto e cadastro como coordenador a um administrador
  + Possibilitar ao visistante solicitar acesso de pesquisador a um projeto a um coordenador
  + Possibilitar ao administrador do sistema aceitar ou rejeitar solicitações públicas para inclusão de projetos e de acesso como coordenador
  + Possibilitar ao administrador redesignar um projeto a outro coordenador
  + Possibilitar ao administrador editar e excluir coordenadores
  + Possibilitar ao coordenador cadastrar, editar e excluir um projeto
  + Possibilitar ao coordenador editar um equipe
  + Possibilitar ao coordenador aceitar ou rejeitar solicitação de pesquisadores ou convidados para um projeto
  + Possibilitar ao coordenador editar um pesquisador
  + Possibilitar ao coordenador fazer upload de arquivos genéticos de um organismo
  + Possibilitar ao pesquisador acesso aos arquivos em que faz parte do projeto
  + Possibilitar ao pesquisador solicitar acesso de convidado a um projeto a um coordenador

# 2 FERRAMENTAS UTILIZADAS

O objetivo deste capítulo é apresentar os recursos que foram utilizados para o desenvolvimento do projeto.

2.1 BALSAMIQ MOCKUPS

Para a criação da estrutura das telas, foi utilizado o software Balsamiq Mockups. Com ele, conseguiu-se uma forma ágil e realista de desenvolvimento de layout modelo devido às facilidades trazidas pela ferramenta.

# 2.8 UML

A UML (*Unified Modeling Language*) é uma linguagem visual utilizada para modelar software baseados no paradigma de orientação a objetos. É uma linguagem de modelagem de propósito geral que pode ser aplicada a todos os domínios de aplicação. Essa linguagem tornou-se, nos últimos anos, a linguagem-padrão de modelagem adotada internacionalmente pela indústria de engenharia de software (GUEDES, 2011).

Neste trabalho utilizamos o diagrama de caso de uso, o diagramas de classe, o diagrama de atividades e o diagrama de componentes. O diagrama de caso de uso permite organizar os requisitos do sistema dando uma visão geral. O diagrama de classes representa a estrutura do sistema orientado a objetos. O diagrama de atividades apresenta o fluxo das atividades do aplicativo e o diagrama de componentes mostra a componentes e suas dependências.

2.3 ASTAH

Astah é uma ferramenta para criação de diagramas UML. Com ela foi possível criar os diagrams de classe, caso de uso e telas de uma forma simples e padronizada.

2.4 ORACLE VM VIRTUAL BOX

2.5 MY SQL WORKBENCH

# 2.6 SQLITE

SQlite é uma biblioteca desenvolvida na linguagem C que implementa um banco de dados SQL. Aplicações que utilizam a biblioteca do SQLite tem acesso a um banco de dados SQL sem a necessidade de um SGBD externo (SQLITE, 2014).

Ao invés de conectar-se a um servidor de banco de dados, o SQLite executa o papel de servidor, grava e consulta os dados diretamente para um arquivo em disco com um limite de 2 terabytes (SQLITE, 2014).

O SQLite é um software livre de domínio público e multiplataforma, além de não possuir dependências externas. Esta biblioteca foi escolhida pela equipe por estar embutida ao Android e por sua simplicidade, já que não há necessidade de instalação, configuração ou administração.

# 3 METODOLOGIA

A função deste capítulo é apresentar as metodologias e ferramentas que foram utilizadas para o desenvolvimento deste aplicativo. Também serão mostradas as responsabilidades dadas a cada membro.

# 3.1 MODELO DE PROCESSO DE ENGENHARIA DE SOFTWARE

Neste projeto foi utilizado o modelo RUP (*Rational Unified Process*)  que fornece diretrizes para definir as tarefas e atribuir responsabilidades aos membros do projeto. O RUP possui quatro fases: iniciação, elaboração, construção e transição. Na iniciação define-se o escopo do projeto. Na elaboração é obtida uma visão abrangente do sistema, através da construção de protótipos e também é definida a arquitetura do sistema. Na construção, o foco está no desenvolvimento do sistema. Por fim, na transição, o produto é transferido ao usuário e o projeto é avaliado e pode ser concluído. Em todas as fases há o gerenciamento dos requisitos e dos recursos do projeto. Por isso o RUP é baseado no desenvolvimento iterativo, que por sua vez é mais flexível quanto às mudanças de escopo durante o desenvolvimento do projeto (IBM, 2014).

# 3.2 PLANO DE RISCOS

No plano de riscos foram definidos os impactos e a probabilidade de determinados eventos acontecerem, bem como as possíveis ações que poderiam ser tomadas. O plano de riscos do projeto pode ser observado na TABELA 2.

TABELA 2 – PLANO DE RISCOS

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nº** | **Risco** | **Consequência** | **Ação** | **Probabilidade** | **Impacto** | **Classificação** |
| 1 | Mudança de requisitos do aplicativo. | Necessidade de mudar o direcionamento do projeto. | Mudanças na estrutura do projeto. | Alto | Alto | 7 |
| 2 | Falta de tempo para o desenvolvimento do aplicativo. | Impossibilidade de entregar ou atraso na entrega. | Divisão de tarefas e monitoramento do andamento. | Alto | Alto | 7 |
| 3 | Conhecimento limitado em bioinformática. | Dificuldade no desenvolvimento do projeto. | Estudo e pesquisa na internet e em livros. | Alto | Moderado | 6 |
| 4 | Perda de dados. | Impossibilidade de continuar o projeto. | Fazer uma rotina de backup dos arquivos importantes. | Moderado | Alto | 6 |
| 5 | Infraestrutura insuficiente. | Falta de equipamentos para desenvolvimento e testes. | Utilizar emuladores para as necessidades específicas. | Moderado | Moderado | 4 |
| 6 | Desistência de um membro da equipe. | Atrasos no andamento e necessidade de refazer a divisão de tarefas. | Dividir mais uma vez as tarefas. | Moderado | Moderado | 4 |

# 3.3 MATERIAIS

O objetivo deste capítulo é descrever as ferramentas utilizadas para o desenvolvimento do projeto.

# 3.3.1 Ambiente de Desenvolvimento

Para o desenvolvimento do aplicativo, foram utilizados dois computadores:

* Notebook CyberPower MS16-GC, processador Intel Core i7-4810MQ 2.80GHz, memória RAM 16GB, placa de vídeo Nvidia GTX 860M 4GB, sistema operacional Windows 10 Home;
  + Oracle VM Virtual Box, Ubuntu 16.04
* Notebook Dell Latitude 3450, processador Intel Core i5 5200U 2.20GHz, memória RAM 8GB, placa de vídeo Intel HD Graphics 5500, sistema operacional Windows 8.1 Pro;
  + Oracle VM Virtual Box, Ubuntu 16.04

# 3.3.2 Linguagem de Programação

A linguagem de programação escolhida pelos usuários foi Ruby. É uma linguagem baseada na orientação a objetos, utilizada em diversas plataformas, inclusive o Android. Atualmente essa linguagem pertence à Oracle (JAVA, 2014).

Ela foi escolhida pelos membros, pois é a única opção para criação de um aplicativo em Android. Mesmo sendo a única opção da equipe devido à plataforma escolhida, o Java possui uma quantidade grande de material sobre ela.

# 3.3.3 Banco de Dados

Para implementar o banco de dados foi escolhido o SQlite, pois é uma ferramenta que já vem instalada no Android e como os dados ficam num arquivo local,  não há necessidade de servidores ou sistemas gerenciadores.

# 3.3.4 Ambiente Integrado de Desenvolvimento

A ferramenta Eclipse foi utilizada neste projeto como IDE (*Integrated Development Environment*). O Eclipse possui licença gratuita e suporta diversas linguagens como Java, além de ser uma das poucas opções de IDE para se trabalhar com Android.

A IDE Eclipse foi escolhida porque os membros da equipe já possuem experiência com a ferramenta, além dela estar disponível em vários sistemas operacionais.

# 3.4.7 Ferramenta para Modelagem UML

A ferramenta utilizada para criação dos diagramas UML foi o Software Ideas Modeler e o Violet UML Editor. Apesar de serem ferramentas simples, gratuitas e exclusivas do Windows, elas atenderam bem as necessidades do projeto.

# 3.4.8 Ferramenta para Criação do Diagrama Entidade Relacionamento

A ferramenta utilizada para criação do diagrama entidade relacionamento foi DBDesigner4, software de código aberto e desenvolvido pela fabFORCE. Está disponível para Windows e Linux.

# 3.5 FERRAMENTA PARA CONTROLE DE VERSÃO

O git é um sistema de controle de versão de softwares desenvolvidos por um ou mais desenvolvedores. Ele também é um sistema de gerenciamento de código fonte, ou seja, todos os membros da equipe podem enviar alterações. Todos os arquivos e todo histórico é armazenado num repositório (GIT, 2014).

Neste projeto foi usado o GitHub, que além de gerenciar as versões e as alterações no código, possui um repositório online, facilitando as alterações para os membros da equipe.

# 3.6 PACOTES

Pacotes são um conjunto de funções pré-escritas para resolver diversos tipos de problema. Os desenvolvedores utilizam pacotes para não precisarem criar uma solução já existente.

  Através de pesquisas foi decidido pelos membros da equipe que 2 bibliotecas externas seriam usadas no projeto, o BioJava para o uso do BLAST, e o tess-two para o processamento digital da imagem e para aplicação do OCR.

# 3.6.1 BioJava

O BioJava é um framework desenvolvido para processar dados biológicos em Java. Contêm rotinas analíticas e estáticas, leitura dos formatos comuns de arquivo, permite a manipulação de sequências e de estruturas 3D e acesso aos webservices (BIOJAVA, 2014).

# 3.6.2 Tess-two

O tess-two é uma biblioteca desenvolvida para juntar duas ferramentas para o Android. Estas ferramentas são o Tesseract OCR e o Leptonica (RMTHEIS, 2014).

O Tesseract OCR é provavelmente uma das ferramentas de OCR mais precisas dentre as disponíveis, consegue ler vários tipos de imagem e converte para texto em mais de sessenta línguas. Estava entre as três melhores no teste de precisão da UNLV (*University of Nevada, Las Vegas*) (TESSERACT, 2014).

Entre 1995 e 2006 foi pouco trabalhada, mas desde então foi consideravelmente melhorada pela Google. Ela foi lançada pela Apache (TESSERACT, 2014). Essa ferramenta foi escolhida por sua precisão, velocidade e compatibilidade.

Leptonica é uma biblioteca de código aberto para processamento e análise de imagens orientada para pedagogos (LEPTONICA, 2014).

**4 DESENVOLVIMENTO DO PROJETO**

A ideia inicial do projeto era criar um aplicativo em Android que se comunicaria com um webservice, que também seria desenvolvido, para realizar consultas do BLAST por dispositivos móveis. Foram feitas diversas mudanças de escopo e na escolha de linguagens e bibliotecas ao decorrer do projeto, que causaram alguns problemas com o cronograma.

Como já dito, seria desenvolvido um webservice para comunicar-se com o BLAST e o dispositivo móvel. Inicialmente feito no Octave, o processamento da imagem e o OCR estavam funcionando de forma instável. Mesmo assim, foi desenvolvido um servidor em Python para ligar os processos do Octave à web.

  Após alguns testes, a equipe concluiu que seria inviável continuar com o Octave e decidiu implementar o processamento da imagem e o OCR em Python, assim o webservice ficaria em apenas uma linguagem. O Tesseract-OCR foi escolhido como biblioteca para o OCR, era de uso fácil e rápido. Para consulta ao BLAST foi usada a biblioteca BioPython, que estava funcionando sem problemas, mas o processamento da imagem ainda estava instável e havia a necessidade de um servidor.

A equipe decidiu testar todos esses processos no próprio dispositivo móvel para verificar se não haveria limitações de hardware. Os testes com a biblioteca Tess-two foram satisfatórios, pois obteve bons resultados tanto no processamento da imagem quanto no OCR.

Os resultados do BioJava também foram satisfatórios, necessitando apenas da leitura do XML devolvido pelo BLAST que foi facilmente suprida pela linguagem. Assim o projeto migrou-se totalmente para o Java. Posteriormente houve a necessidade de um banco de dados simples, logo a equipe optou pelo SQLite pela praticidade da ferramenta.

A divisão de tarefas foi mudando de acordo com as mudanças de linguagem, graças ao conhecimento da linguagem dos membros da equipe. Apesar das diversas mudanças, a equipe conseguiu se adaptar a elas.

A equipe decidiu tirar os diagramas de sequência e colocar diagramas de atividade, pois são diagramas mais fáceis de serem lidos e ilustram o que ocorre de forma mais simples.

# 4.1 VALIDAÇÃO DO SOFTWARE

A equipe realizou algumas validações após o processamento da imagem,

**5 APRESENTAÇÃO DO SOFTWARE**

Neste capítulo será apresentada uma visão geral do aplicativo desenvolvido, mostrando suas telas e funcionalidades.

Antes de entrar no aplicativo, uma tela de abertura é mostrada na FIGURA 9 enquanto o aplicativo carrega seus componentes. A tela inicial mostrada na FIGURA 10 lista as consultas, pode-se verificar o status do processamento ou caso esteja finalizado, verificar os resultados e dá as opções da câmera e de tirar foto.

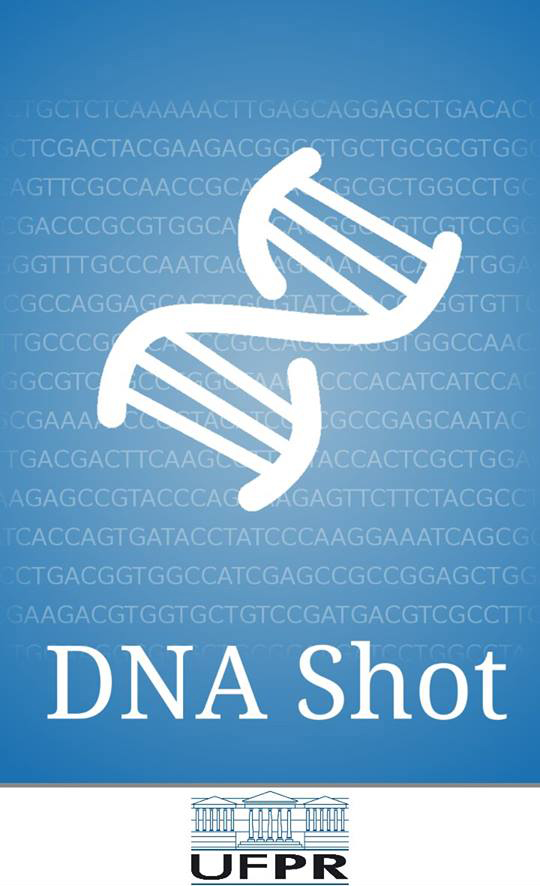


FIGURA 9 - TELA DE ABERTURA

Fonte: Os autores (2016).

# 5.1 INSTALAÇÃO

Para instalar o aplicativo, deve-se entrar no link do aplicativo na Play Store (https://play.google.com/store/apps/details?id=com.dnashot.v0), mostrado na FIGURA X1, toque em “Instalar” e depois em “Aceitar”. Após o download e a instalação do aplicativo, ele já estará pronto para uso. A equipe recomenda o uso deste aplicativo em celulares com processador de 1 GHz ou mais, além de 1GB de memória RAM.



FIGURA 24 - TELA DO APLICATIVO NA PLAY STORE

Fonte: Os autores (2016).

**6 CONSIDERAÇÕES FINAIS**

O desenvolvimento do aplicativo trouxe um contato inicial com a bioinformática para os usuários. Ao trabalhar com apenas um dos programas ofertados pelo BLAST, foi possível perceber que ainda podem-se integrar muitos algoritmos da bioinformática para os dispositivos móveis.

O aplicativo consegue cumprir suas funções de processar sua imagem, aplicar o OCR e realizar uma pesquisa no blastn num dispositivo móvel. A interface gráfica do aplicativo foi feita para que o usuário também possa ver o que aconteceu durante o processo.

  A maioria das funções do aplicativo tem um processamento custoso à nível de dispositivos móveis, logo é possível que celulares mais antigos não funcionem perfeitamente ou tenham a performance afetada.

Durante o desenvolvimento, várias ideias de como o aplicativo poderia ser melhorado e ampliado surgiram, porém não foi possível implementá-las devido a complexidade e tempo. Algumas destas ideias para uma futura versão estão listadas abaixo:

* Melhorias na câmera: É possível melhorar o foco da câmera para que toda vez que haja alguma mudança brusca, ela foque automaticamente ao invés de esperar um segundo.
* Integração ao SILA (*Genome Automated Annotation System*): Através de um webservice é possível fazer uma integração a este sistema de anotação automática de genomas. Como é uma ferramenta rápida, os resultados viriam mais rápido que os resultados do BLAST.
* Integração com o blastp: É necessário um filtro diferente para a sequência de proteínas e também um tempo maior de espera do usuário.

**REFERÊNCIAS**

ABBYY. **What is OCR and OCR Tecnology.** Disponível em: <http://finereader.abbyy.com/about\_ocr/whatis\_ocr/>. Último acesso: 17/11/2014.

ACHARYA, T.; RAY, A. **Image Processing: Principles and Applications**. New Jersey: John Wiley & Sons, 2005.

ANDROID. **Introduction to Android | Android Developers.** Disponível em: < https://developer.android.com/guide/index.html>. Último acesso: 22/11/2014.

BAXEVANIS, A.; OULLETTE, B. **Bioinformatics: A pratical Guide to the Analysis of Genes**. New Jersey: John Wiley & Sons, 2005.

BERG, Jeremy Mark. **Biochemistry**. Nova York: Freeman, 2006.

BIOJAVA. **BioJava.** Disponível em: <http://biojava.org/>. Último acesso: 17/11/2014.

ENGINEERSGARAGE. **What is Android: Introduction, Features & Applications.** Disponível em: <http://www.engineersgarage.com/articles/what-is-android-introduction>. Último acesso: 17/11/2014.

FOX, J. **WHAT IS BIOINFORMATICS?** Disponível em: <<http://www.scq.ubc.ca/what-is-bioinformatics/>>. Último acesso: 24/10/2014.

GANTT. **What is a Gantt Chart? Gantt Chart Information, history and Software.** Disponível em: <http://www.gantt.com/>. Último acesso: 18/11/2014.

GIT. **About - Git.** Disponível em: <http://git-scm.com/about/>. Último acesso: 18/11/2014.

GUEDES, G. **UML 2: uma abordagem prática**. São Paulo: Novatec, 2011.

IBM. **IBM – Definição de RUP.** Disponível em: <http://www-01.ibm.com/software/rational/rup/>. Último acesso: 29/10/2014.

JAVA. **Obtenha informações sobre a Tecnologia Java.** Disponível em: <https://www.java.com/pt\_BR/about/>. Último acesso: 18/11/2014.

KORF, I.; YANDELL, M.; BEDELL, J. **BLAST: An Essential Guide to the Basic Local Alignment Search Tool**. Sebastopol: O’Reilly, 2003

LEPTONICA. **Leptonica.** Disponível em: <http://www.leptonica.com/>. Último acesso: 17/11/2014.

LEPTONICA. **Leptonica: src/skew.c File Reference.** Disponível em: <https://tpgit.github.io/Leptonica/skew\_8c.html>. Último acesso: 17/11/2014.

MATHWORKS. **Image Thresholding.** Disponível em: <http://www.mathworks.com/discovery/image-thresholding.html>. Último acesso: 17/11/2014.

MOUNT, David W. **Bioinformatics: Sequence and Genome Analysis**. Nova York: CSHL Press, 2004.

NCBI. **BLAST Program Selection Guide.** Disponível em: <http://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE\_TYPE=BlastDocs&DOC\_TYPE=ProgSelectionGuide>. Último acesso: 14/11/2014.

NCBI. **Zaire ebolavirus isolate Ebola virus/H.sapiens-tc/COD/1976/Yambuku-May - Nucleotide - NCBI.** Disponível em: <http://www.ncbi.nlm.nih.gov/nucleotide/10313991>. Último acesso: 20/11/2014

NCBI. **Herbaspirillum seropedicae SmR1, complete genome - Nucleotide - NCBI.** Disponível em: <http://www.ncbi.nlm.nih.gov/nucleotide/300072131>. Último acesso: 20/11/2014

NCBI. **DNA sequence coding dengue fever virus I(DEN1)-S275/90 - Nucleotide - NCBI.** Disponível em: <http://www.ncbi.nlm.nih.gov/nucleotide/2175014>. Último acesso: 20/11/2014

NCBI. **Hevea brasiliensis hydroxynitrile lyase (hnl) mRNA, complete cds - Nucleotide - NCBI.** Disponível em: <http://www.ncbi.nlm.nih.gov/nucleotide/1223883>. Último acesso: 20/11/2014

PHONESCOOP. **Smartphone definition (Phone Scoop).** Disponível em: <http://www.phonescoop.com/glossary/term.php?gid=131>. Último acesso: 19/11/2014.

PROSDOCIMI, F. **Bioinformática: Manual do Usuário.** Disponível em: <http://www2.bioqmed.ufrj.br/prosdocimi/divulgacao/bioinfo\_manual.pdf Último acesso: 14/11/2014.

RASHIDI, H.; BUEHLER, L. **Bioinformatic Basics – Applications in Biological Science and Medicine.** Londres: CRC Press, 2000.

RMTHEIS. **rmtheis/tess-two.** Disponível em: <https://github.com/rmtheis/tess-two>. Último acesso: 17/11/2014.

SAUVOLA. J.; PIETIKAINEN M. **Adaptive document image binarization**. Disponível em: <http://www.mediateam.oulu.fi/publications/pdf/24.pdf>. Último acesso: 18/11/2014.

SQLITE. **About SQLite.** Disponível em: <http://www.sqlite.org/about.html>. Último acesso: 09/11/2014.

TESSERACT. **tesseract-ocr.** Disponível em: <https://code.google.com/p/tesseract-ocr/>. Último acesso: 17/11/2014.

VIALLE, R. A. et al. **SISTEMA INTEGRADO PARA ANOTAÇÃO AUTOMÁTICA DE GENOMAS**. [ S. I.]    Universidade Federal do Paraná, 2013.

WBSTOOL. **WBS Tool.com - Software web gratuito para WBS (EAP), Organogramas e Hierarquias.** Disponível em: <http://www.wbstool.com/whatIsWBS.php>. Último acesso: 18/11/2014.

ZHANGLAB. **FASTA format.** Disponível em: <http://zhanglab.ccmb.med.umich.edu/FASTA/>. Último acesso: 24/10/2014.

DBDesigner4

Disponível em: < http://www.fabforce.net/dbdesigner4/>. Último acesso: 20/11/2014.

Eclipse

Disponível em: <https://www.eclipse.org/home/index.php>. Último acesso: 10/11/2014

ProjectLibre

Disponível em: <http://www.projectlibre.org/>. Último acesso: 29/10/2014.

Software Ideas Modeler

Disponível em: <http://www.softwareideas.net/>. Último acesso: 30/10/2014.

WBS Tool

Disponível em: <http://www.wbstool.com/>. Último acesso: 29/10/2014.

Violet UML Editor

Disponível em: < http://alexdp.free.fr/violetumleditor/page.php>. Último acesso: 24/11/2014.

**APÊNDICE A - DIAGRAMA DE CASOS DE USO**

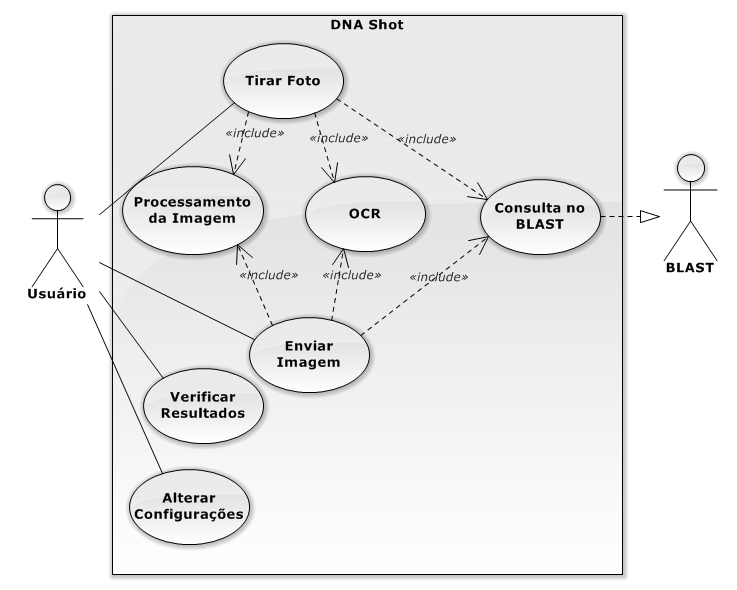


FIGURA 25 – DIAGRAMA DE CASOS DE USO

Fonte: Os autores (2016).

**APÊNDICE B - ESPECIFICAÇÃO DOS CASOS DE USO**

**Caso de uso 001 - Tirar foto**

**Descrição**

Este caso de uso descreve como tirar foto no aplicativo.

**Pré-condições**

Não há.

**Pós-condições**

O processamento da imagem inicia.

**Ator Primário**

Usuário

**Fluxo Principal de Eventos**

1. Na tela inicial, o usuário seleciona a “CAMERA”;
2. O aplicativo irá abrir a tela da câmera;
3. O usuário foca no texto desejado e toca em tirar.
4. O aplicativo irá salvar a imagem para processamento;
5. O aplicativo volta à tela da câmera;
6. O caso de uso é finalizado.

**Fluxo de Exceção**

Não há.

**Regras de Negócio**

Não há.

**Caso de uso 002 - Enviar Imagem**

**Descrição**

Este caso de uso descreve como enviar imagem no aplicativo.

**Pré-condições**

Não há.

**Pós-condições**

O processamento da imagem inicia.

**Ator Primário**

Usuário

**Fluxo Principal de Eventos**

1. Na tela inicial, o usuário seleciona a “LOAD”;
2. O aplicativo irá abrir a galeria do dispositvo;
3. O usuário seleciona a imagem desejada;
4. O aplicativo irá salvar a imagem para processamento;
5. O aplicativo volta à tela inicial;
6. O caso de uso é finalizado.

**Fluxo de Exceção**

Não há.

**Regras de Negócio**

Não há.

**Caso de uso 003 - Processamento da Imagem**

**Descrição**

Este caso de uso descreve como ocorre o processamento da imagem no aplicativo.

**Pré-condições**

O aplicativo deve ter salvado uma imagem para processamento, seja pela câmera ou pela galeria.

**Pós-condições**

O OCR inicia.

**Ator Primário**

Usuário

**Fluxo Principal de Eventos**

1. O aplicativo carrega a imagem;
2. Thresholding é aplicado na imagem;
3. O ângulo de inclinação do texto na imagem é detectado;
4. A imagem será girada de acordo com o ângulo encontrado;
5. O caso de uso é finalizado.

**Fluxo de Exceção**

Não há.

**Regras de Negócio**

Não há.

**Caso de uso 004 - OCR**

**Descrição**

Este caso de uso descreve como ocorre o OCR é aplicado na imagem.

**Pré-condições**

O aplicativo deve ter processado a imagem devidamente.

**Pós-condições**

A consulta ao BLAST inicia.

**Ator Primário**

Usuário

**Fluxo Principal de Eventos**

1. O aplicativo carrega a imagem processada;
2. O OCR é aplicado na imagem;
3. O caso de uso é finalizado.

**Fluxo de Exceção**

E1 O texto reconhecido pelo OCR é menor que 20.

        E1.1 Será colocado no status “An OCR error has ocurred” na lista de resultados na tela inicial.

E2 O OCR não reconheceu nenhum caractere.

        E2.1 Será colocado no status “An OCR error has ocurred” na lista de resultados na tela inicial.

**Regras de Negócio**

Não há.

**Caso de uso 005 - Consulta no BLAST**

**Descrição**

Este caso de uso descreve como ocorre a consulta no BLAST.

**Pré-condições**

O aplicativo deve ter terminado o OCR e achado uma sequência com mais de 20 caracteres e o dispositivo deve estar conectado à internet.

**Pós-condições**

O resultado ficará disponível na tela inicial.

**Ator Primário**

Usuário

**Fluxo Principal de Eventos**

1. O aplicativo inicia uma consulta no BLAST com a sequência achada;
2. O aplicativo espera 8 segundos;
3. O aplicativo verifica se o BLAST já finalizou sua consulta;
4. O aplicativo repete os passos 2 e 3 até a consulta finalizar;
5. O aplicativo interpreta o XML da consulta e salva no banco de dados;
6. O aplicativo coloca “Pronto” no status do aplicativo;
7. O caso de uso é finalizado,

**Fluxo de Exceção**

E1 O dispositivo não está conectado à internet.

        E1.1 Ele irá tentar realizar a consulta 6 segundos depois.

    E2 O BLAST não encontrou nenhum resultado.

        E2.1  Será colocado no status “An BLAST error has ocurred” na lista de resultados na tela inicial.

E3 O BLAST não estava com a resposta pronta após 50 tentativas.

E3.1  Será colocado no status “An BLAST error has ocurred” na lista de resultados na tela inicial.

**Regras de Negócio**

Não há.

**Caso de uso 006 - Verificar Resultados**

**Descrição**

Este caso de uso descreve como se verifica os resultados da pesquisa a partir da imagem enviada.

**Pré-condições**

O aplicativo deve ter encerrado a consulta no BLAST e salvo os dados no banco de dados e seu status na tela inicial deve estar “Pronto”.

**Pós-condições**

Não há.

**Ator Primário**

Usuário

**Fluxo Principal de Eventos**

1. O usuário deve tocar no resultado com status “Pronto” na lista da tela inicial;
2. O aplicativo irá abrir a tela de resultados;
3. O caso de uso é finalizado.

**Fluxo de Exceção**

Não há.

**Regras de Negócio**

Não há.

**Caso de uso 007 - Alterar Configurações**

**Descrição**

Este caso de uso descreve como ocorre a alteração de configurações no aplicativo.

**Pré-condições**

Não há.

**Pós-condições**

Não há.

**Ator Primário**

Usuário

**Fluxo Principal de Eventos**

1. O usuário toca em Configurações;
2. O aplicativo abre a tela de configurações;
3. O usuário seleciona as configurações desejadas;
4. O caso de uso é finalizado.

**Fluxo de Exceção**

Não há.

**Regras de Negócio**

Não há.

**APÊNDICE C - DIAGRAMA ENTIDADE RELACIONAMENTO**

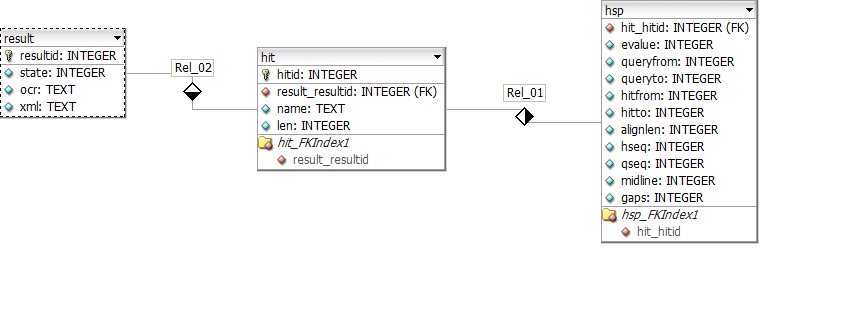


FIGURA 26 – DIAGRAMA ENTIDADE RELACIONAMENTO

Fonte: Os autores (2016).

**APÊNDICE D- DIAGRAMA DE CLASSES**

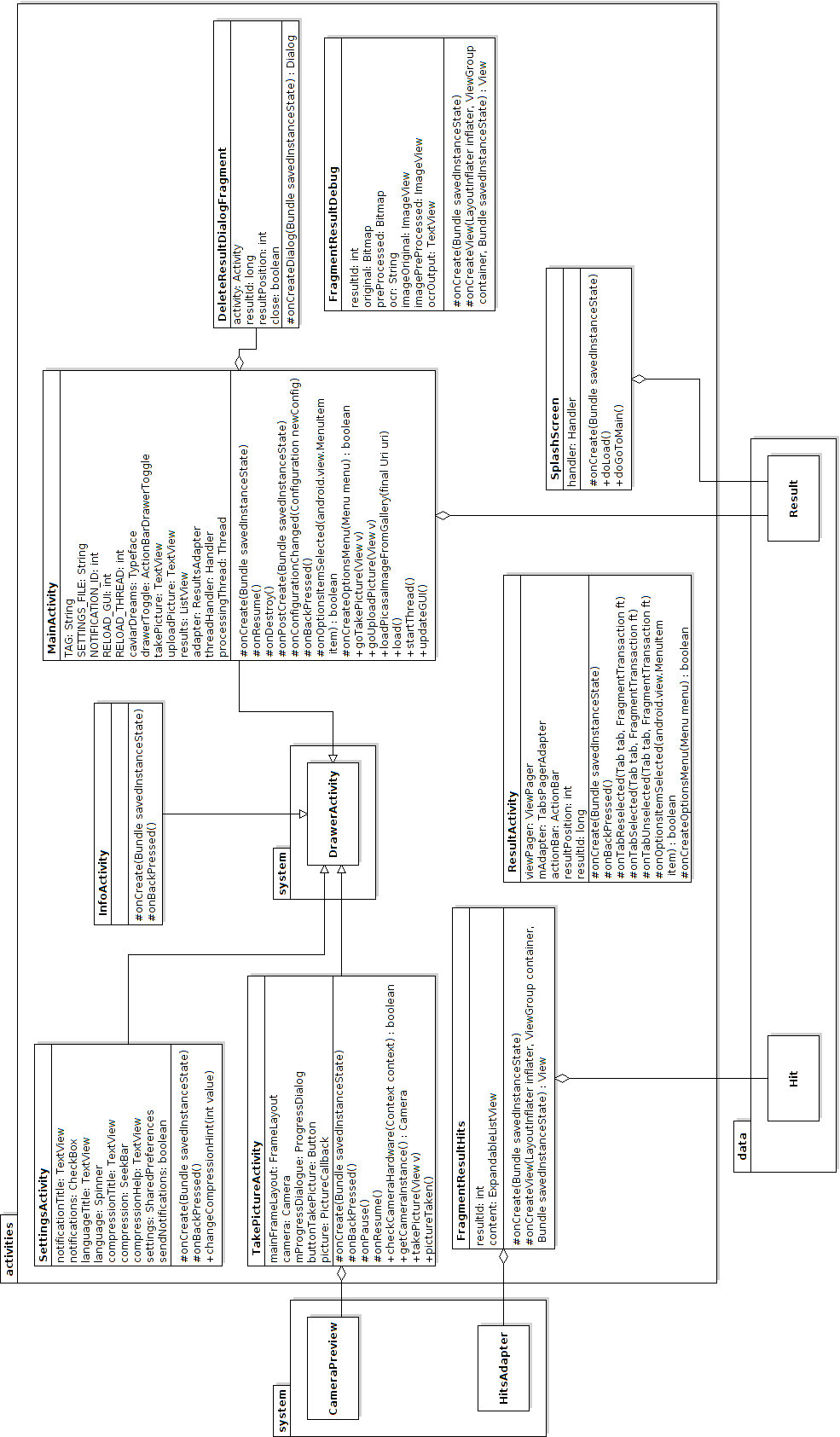
****

FIGURA 27 – DIAGRAMA DE CLASSES – PACOTE ACTIVITIES

Fonte: Os autores (2016).

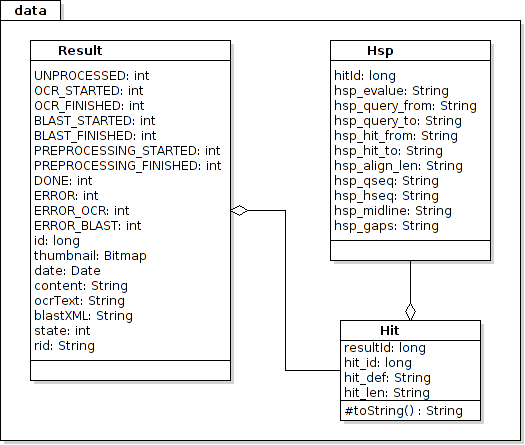


FIGURA 28 – DIAGRAMA DE CLASSES – PACOTE DATA

Fonte: Os autores (2016).

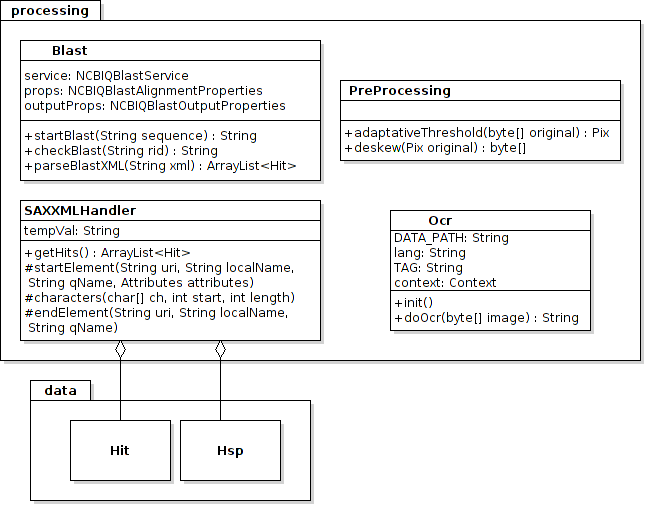


FIGURA 29 – DIAGRAMA DE CLASSES – PACOTE PROCESSING

Fonte: Os autores (2016).

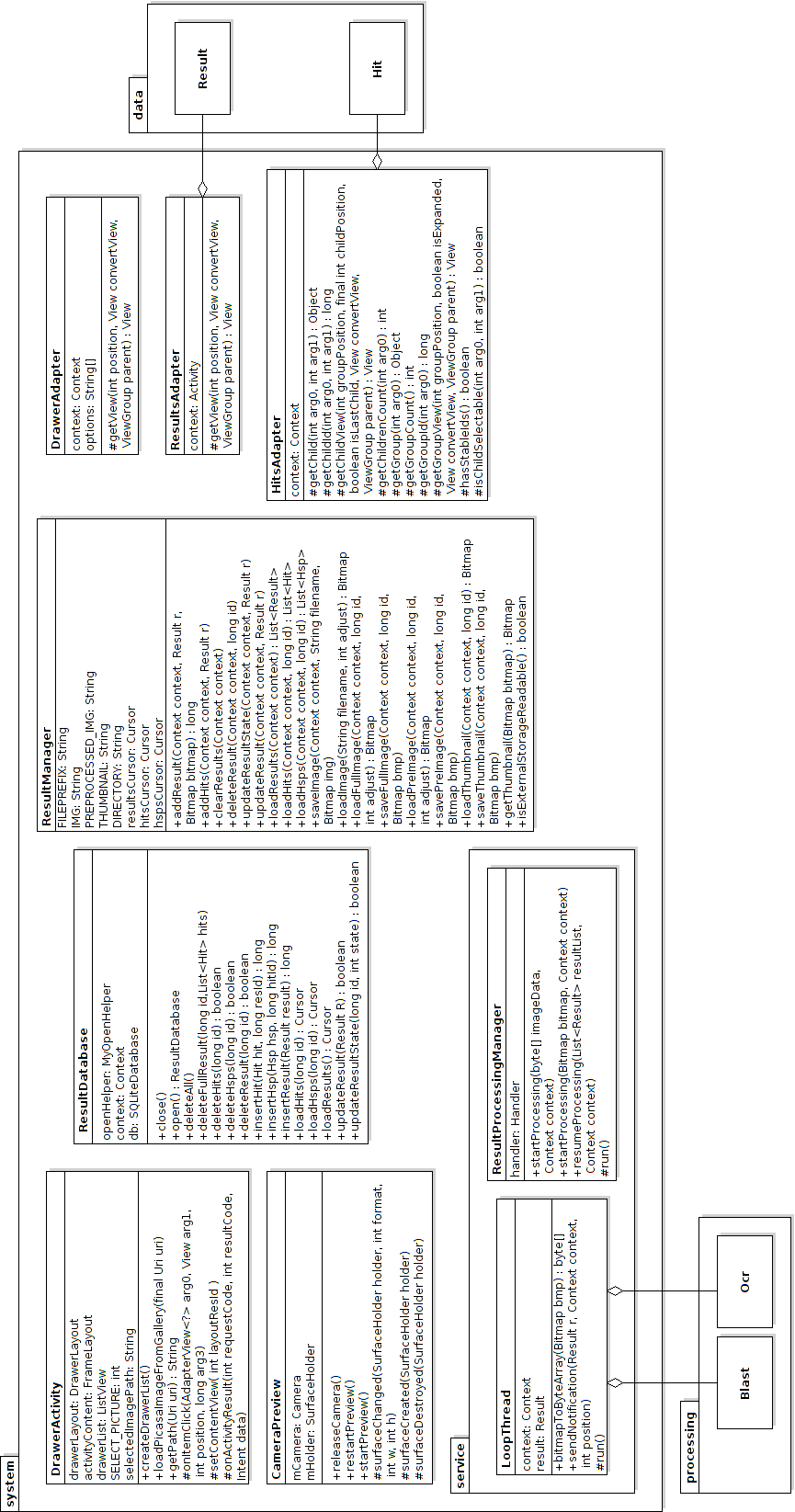


FIGURA 30 – DIAGRAMA DE CLASSES – PACOTE SYSTEM

Fonte: Os autores (2016).

**APÊNDICE E - DIAGRAMA DE ATIVIDADES**

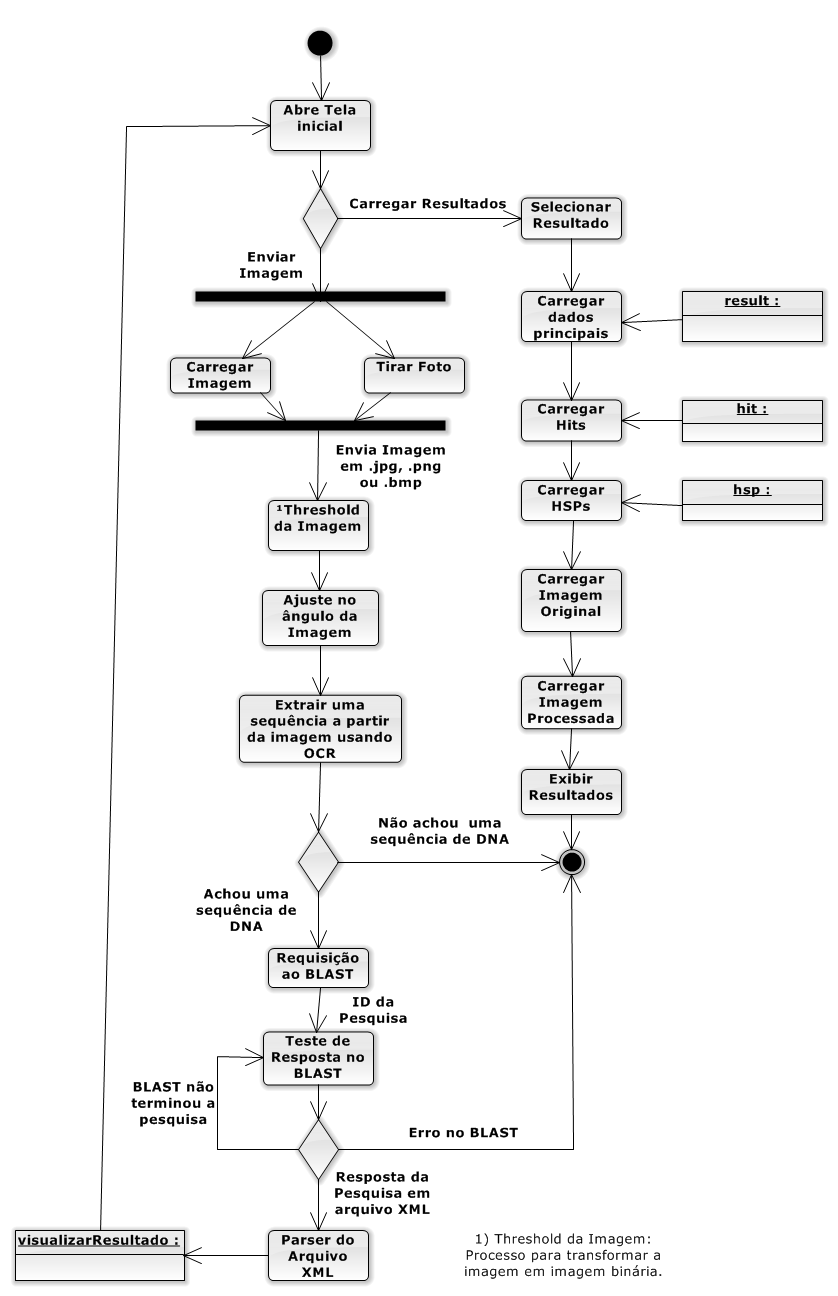
****

FIGURA 31 – DIAGRAMA DE ATIVIDADES – USO GERAL

Fonte: Os autores (2016).

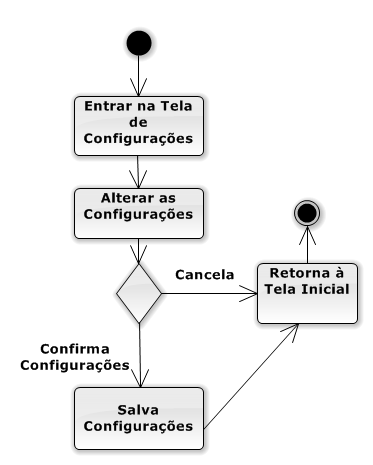


FIGURA 32 – DIAGRAMA DE ATIVIDADES – ALTERAR CONFIGURAÇÕES

Fonte: Os autores (2016).

# APÊNDICE F - DIAGRAMA DE COMPONENTES

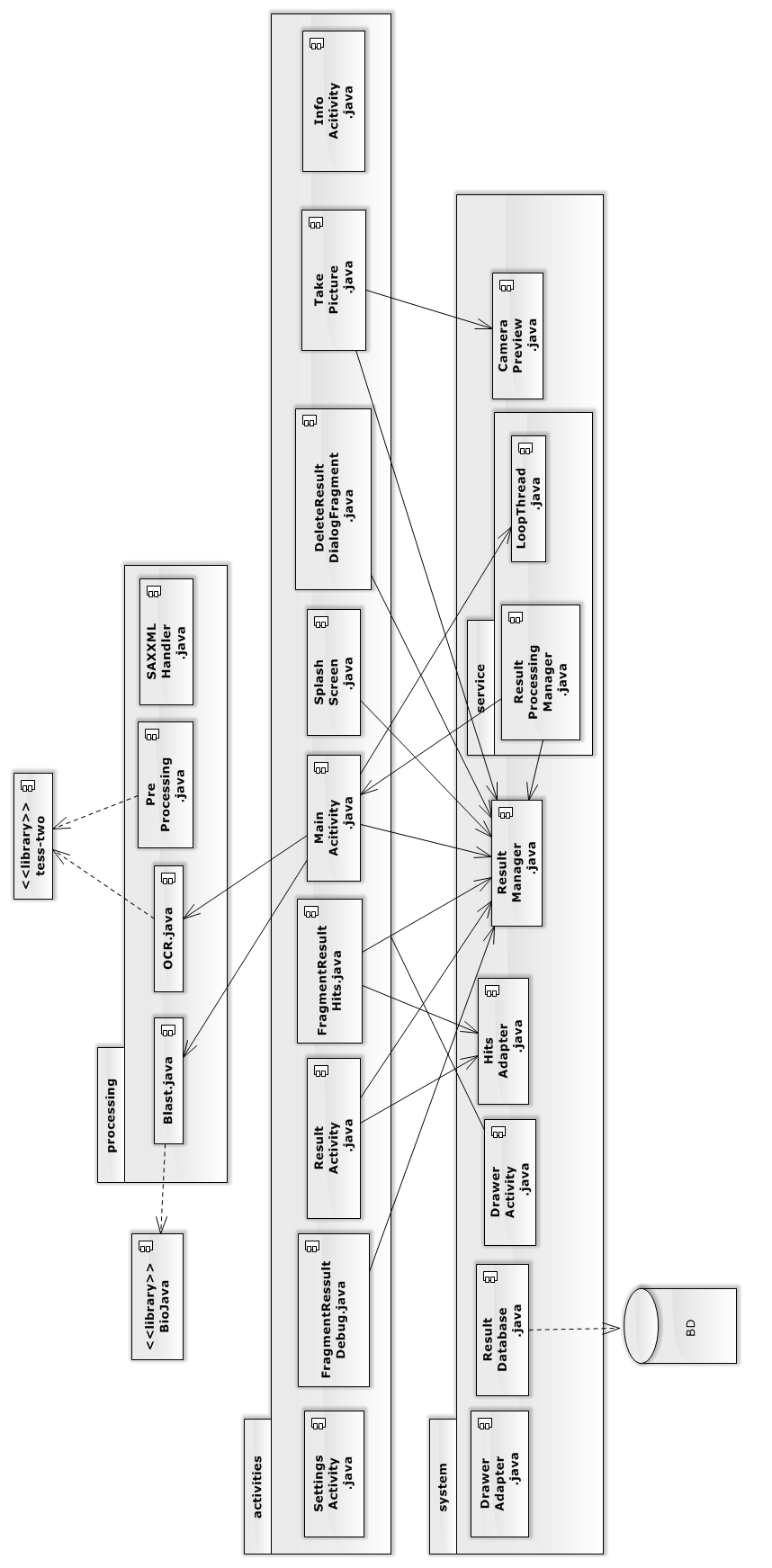


FIGURA 33 – DIAGRAMA DE COMPONENTES

Fonte: Os autores (2016).