

Project: SketchApp

Team No.: 11

Class: CSE 3310.003 - Fall 2020

Module: System Requirements Analysis (SRA)

Deliverable: SRA Document

Version: [1.1]

Date: [10/22/2020]

Contributors

Gabriel de Sa
Lucas Streanga
Luke Brown
Saugat Pandey
David Rademacher

Revision History

<i>Version number</i>	<i>Date</i>	<i>Originator</i>	<i>Reason for change</i>	<i>High level description of changes</i>
1.0	10/06/2020	David Rademacher	Initial draft	
1.1	11/30/2020	David Rademacher	Increment 2 complete	Update UML Documentation and Added User Manual and Test Cases.

TABLE OF CONTENTS

1. INTRODUCTION AND PROJECT OVERVIEW	IV
2. OBJECTIVES	V
2.1 BUSINESS Objectives.....	v
2.2 SYSTEM Objectives	v
3. PROJECT CONTEXT DIAGRAM	VI
4. SYSTEMS REQUIREMENTS.....	VII
4.0 Requirements Overview	vii
4.1 “Login” Requirements	vii
4.2 “Using the Canvas” Requirements	vii
4.3 “Drawing Categorization” Requirements	viii
4.4 “Comments, Voting, and Reporting” Requirements.....	viii
4.5 “Database search” Requirements.....	viii
4.6 “Following and Filtering” Requirements.....	ix
4.7 “Profile” Requirements.....	ix
5. SOFTWARE PROCESSES AND INFRASTRUCTURE.....	X
5.1 Hardware and Infrastructure.....	x
5.2 Conceptual Data Model - Database	x
5.3 UML Diagrams.....	xi
5.4 Screen Shots	xx
5.5 Test Plan	xx
6. ASSUMPTIONS AND CONSTRAINTS.....	XXI
6.1 ASSUMPTIONS	xxi
6.2 CONSTRAINTS.....	xxi
6.3 Out of Scope material.....	xxi
7. DELIVERY AND SCHEDULE	XXII
8. STAKEHOLDER APPROVAL FORM.....	XXIII
APPENDIX:	XXIV

1. Introduction and Project Overview

We aim to create a fun and expansive application for Android by the name of SketchApp. The full specifications of the application will be explained in this document, but in a high-level sense we will create an environment for users to converse with each other and share images and drawings as well as allow users to create drawings using our tool. The app will be easy to use and will be a more relaxed social platform for our users. In addition to the requirements listed in this document, we are open to adding more features as our user base requests them. The project will be completed and presented by December 8th of 2020.

2. Objectives

2.1 BUSINESS OBJECTIVES

The following is a list of business objectives:

Objective 1:

• **User Registration:** Users must provide personal information in order to access and view the system. The personal information requested will be:

- Username
- Password
 - Minimum 8 Characters, 1 Capital Letter, 1 Number.
- First Name, Last Name
- E-mail address

Objective 1a:

• **Category Selection:** Part of the user registration process will be to follow certain categories of Animals:

categories:

Bear, Bee, Bird, Cat, Cow, Dog, Dolphin, Duck, Elephant, Fish, Frog, Horse, Mouse, Penguin, Rabbit, Sheep, Snake, Whale.

Objective 2:

• **Login:** Users must login using username and password provided during customer registration.

Objective 3:

• **Feed:** After login users will be directed to their “feed,” a section of the app that shows images scrapped from Reddit, as well as images drawn by friends.

- Allows users to view “real” pictures of followed animals.
- Allows users to see pictures drawn by other users.
- Allows users can like posts.
- Allows users to select post.

Objective 4:

• **Posts:** In the feed users will be allowed to select posts, and view comments, like or report.

- Users will click post, which will open post from feed and take over the view board.
- User can then like, comment (undecided), or report post.
 - Report post if post is not appropriate or tagged incorrectly.

Objective 5:

• **Draw:** User will then be allowed to draw a post, where the app will try to guess what they are drawing as they draw.

- User is prompted to drawing board.
- User has option to clear the board.
- Submit and Cancel buttons available.
- If submitted the drawing is processes through the neural network, and the category is determined. User is asked to confirm category.
 - Once category is confirmed drawing is posted to users' feed.

Objective 6:

• **User Homepage:** In the Users homepage, the user will have the option to see all posts made by user. As well as see the posts that the user has liked.

- See owned posts.
- Delete Post.
- See liked posts.
- Change followed Categories.
- Discover New Categories.

Objective 7:

• **Settings:** User will have the ability to control color scheme of app, change post view settings, and delete user account.

- User can change color scheme of SketchApp.
 - Light Mode
 - Dark Mode
- User can change types of posts seen.
 - Only drawn posts
 - Only scraped posts.
 - Both.
- Users can delete their account, including all account data and posts from SketchApp database.

2.2 SYSTEM OBJECTIVES

The following is a list of system objectives:

Objective 1: SketchApp will be an Android OS application.

Objective 2: Web scraping functionality will utilize Reddit for animal images.

Objective 3: M.L. will utilize Google Doodles data for training drawing images.

Objective 4: M.L. will use TensorFlow for Neural Network Implementation.

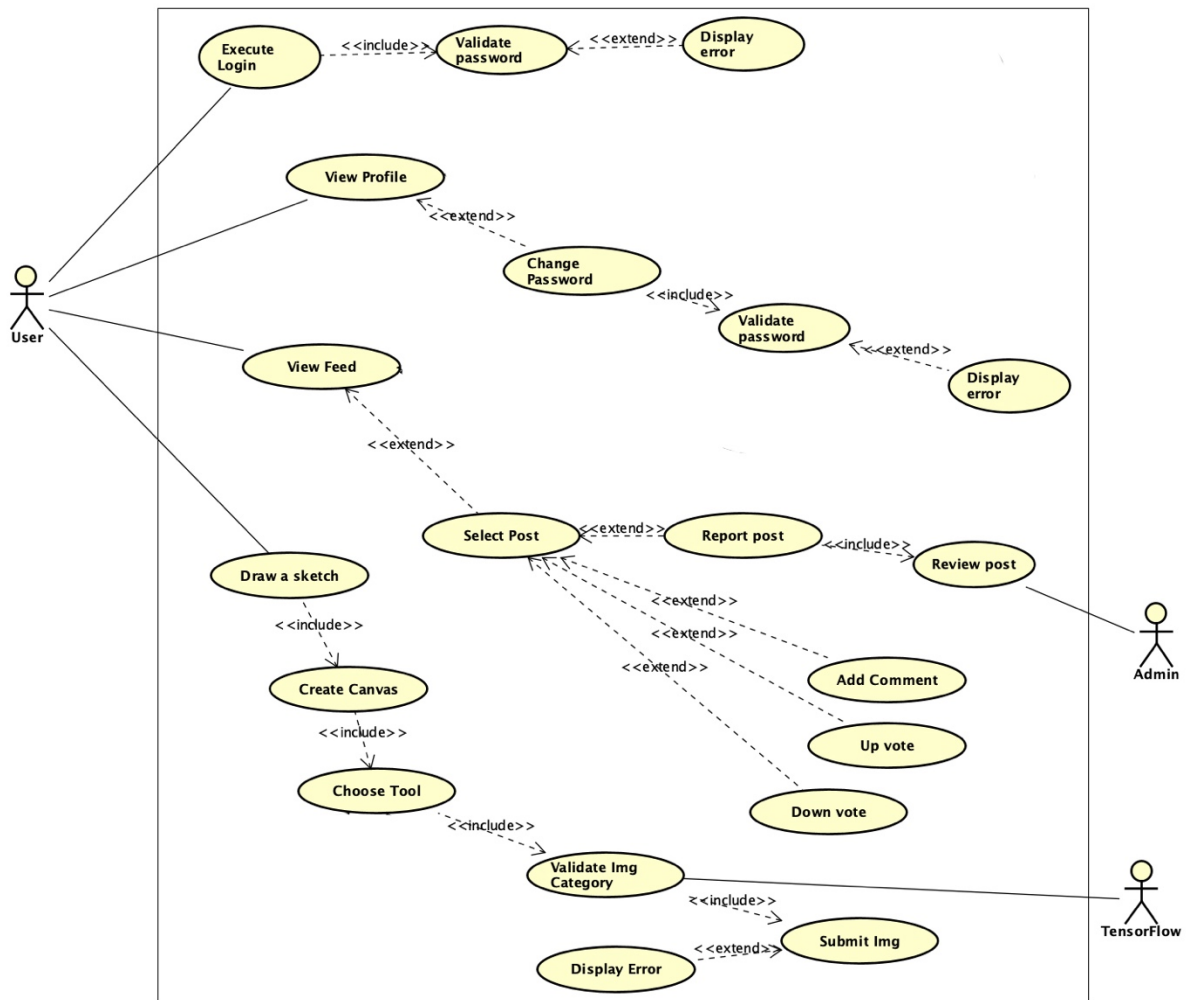
Objective 5: App will communicate with Firebase, and SQLite to gather and contain app data.

Objective 6: SketchApp will require no further development or maintenance after production date.

Objective 7: SketchApp will use proprietary tools for web scraping. Reddit will be used to gather images for user feeds.

- Credit to Reddit and Redditor will be placed in every post.

3. Project Context Diagram



4. Systems Requirements

4.0 REQUIREMENTS OVERVIEW

The following requirement forms provide details about how each specific area of the application will accomplish its objective. The forms are grouped by areas of the application in which they operate. These subsystems are: Login, Using the Canvas, Drawing Categorization, Comments, Voting, and Reporting, Database Search, Following and Filtering, and Profile.

4.1 “LOGIN” REQUIREMENTS

Requirement title	Login
Sequence No.	1
Short description	How the system accepts login information
Description	The system will have two entry boxes, email/username, and the password. Each box will have a header of respective name i.e. Username and Password. Minimum of 8 entry is required on the password box.
Preconditions	The user will need to open the application for login page and needs to be connected to the internet.
Postconditions	Successful entry of username and password will trigger to the homepage of the application.
Other attributes	There will be a “login” button for user to submit his/her username and password.

4.2 “USING THE CANVAS” REQUIREMENTS

Requirement title	Using the Canvas
Sequence No.	1
Short description	How the user interacts with the canvas interface and submits a drawing
Description	The Canvas interface will provide tools for the user to create a Drawing. The user will be able to select the brush and create strokes in the provided region. The user can reset his/her Drawing or submit it to the system for categorization.
Preconditions	The user must be logged in to a valid account before creating a Drawing.
Postconditions	The user must have a connection to the internet to submit a Drawing. The Drawing will then be categorized and stored in the database.
Other attributes	N/A

4.3 “DRAWING CATEGORIZATION” REQUIREMENTS

Requirement title	Drawing Categorization
Sequence No.	1
Short description	How the user interacts with the canvas to draw the sketch.
Description	The user will have various tools for drawing. Also, there will be an option to upload sketch. User can save his/her sketch and continue later.
Preconditions	Drawing canvas will needed to be open to draw on it.
Postconditions	On completion of sketch user can share his/her sketch, save it or delete it.
Other attributes	N/A

4.4 “COMMENTS, VOTING, AND REPORTING” REQUIREMENTS

Requirement title	Comments, Voting, and Reporting
Sequence No.	1
Short description	Accept user selection based on icon chosen
Description	When viewing a different user's post, the user will be able to view other user's or their own comment on the chosen post. The user will have the option, a single button, to put a comment on a post or reply to other comments on posts. The input area for comments will max out at 140 characters and require at least a single character. There will be three buttons placed next to any comment. The buttons will be arranged in a column and the top button will be for increasing the vote by a single increment, the middle button will be for lowering the vote by a single increment, and the bottom most button will be for reporting the comment. When clicking the report button a new activity will open offering check box selections the user can choose to describe why the comment needs to be evaluated by admin.
Preconditions	The user must be logged in to their account.
Postconditions	Successful data input will trigger
Other attributes	If an attempt to do any of the actions occurs without being logged on, prompt user to login and provide shortcut to login screen

4.5 “DATABASE SEARCH” REQUIREMENTS

Requirement title	Database Search
Sequence No.	1
Short description	How the user can search for information in the database
Description	The server-side database will store all user Drawings and Profile information. Drawings can be queried by category or by the Profile that created them. Profiles can be found by username or an optionally associated personal name.
Preconditions	The user must have an internet connection to search the database.
Postconditions	Database search results will be sent to the client for use in filtering the user's feed or displaying search results from the Profile tab.
Other attributes	N/A

4.6 “FOLLOWING AND FILTERING” REQUIREMENTS

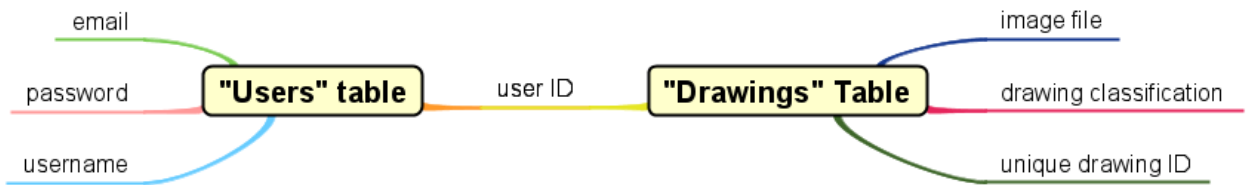
Requirement title	Following and Filtering
Sequence No.	6
Short description	Allow the user to customize their feed with following and filtering.
Description	The system will allow a user to follow another user by navigating to their profile and pressing follow. Once followed, that followed user's new content will appear on the feed of everyone following them in real time. In addition, the user will be able to filter their feed with keywords. The user can filter by category, for example, they can choose to only view posts of dogs, and their feed will update accordingly and only show posts in that category. If a user is following no one and no category is given, then random posts shall be shown.
Preconditions	Both users (following/followed) must have a valid account. Posts in the system should be logged into a category, either by the machine or by user input.
Postconditions	The user must have an active internet connection to get real time feed updates. The “following” status should stay on the relevant accounts until removed by the user, to ensure the user gets the proper feed.
Other attributes	The feed is subject to improvement. Currently, there is no plan for a method to choose the most relevant for our users, and so we rely on filtering and following to achieve this. In the future, we could implement a learning machine to understand the user's interests and customize their feed like many platforms do.

4.7 “PROFILE” REQUIREMENTS

Requirement title	User Profiles
Sequence No.	7
Short description	Unique user profiles for database storage.
Description	<p>Each user will be required to create a profile before using the app. The user must provide a unique username. The username will be checked against all saved usernames in the database to ensure it is unique. The user must also provide a password. These are the base requirements, but users can also add/edit a bio as well as a profile picture. Each user will have a page for their profile, displaying their username at the top and their bio and profile picture (if they have one). A default profile picture will be used if the user does not provide one. On the user's profile, there will be multiple tabs. One tab will be that user's posts, and will display posts the user has made in chronological order with the most recent at the top. The user will also have a tab for liked content, in which posts the user has given a like to will be shown. Any user can see an other's profile by searching their username or by clicking on their username on a post they made or a comment they made on another post.</p>
Preconditions	The database must be able to store profile information and must be able to retrieve profile information.
Postconditions	Posts must be linked to a profile so that they can be displayed in the appropriate profiles and so that users can find someone's profile by seeing a post they made.
Other attributes	N/A

5. Software Processes and Infrastructure

5.1 CONCEPTUAL DATA MODEL - DATABASE



5.2 UML DIAGRAMS

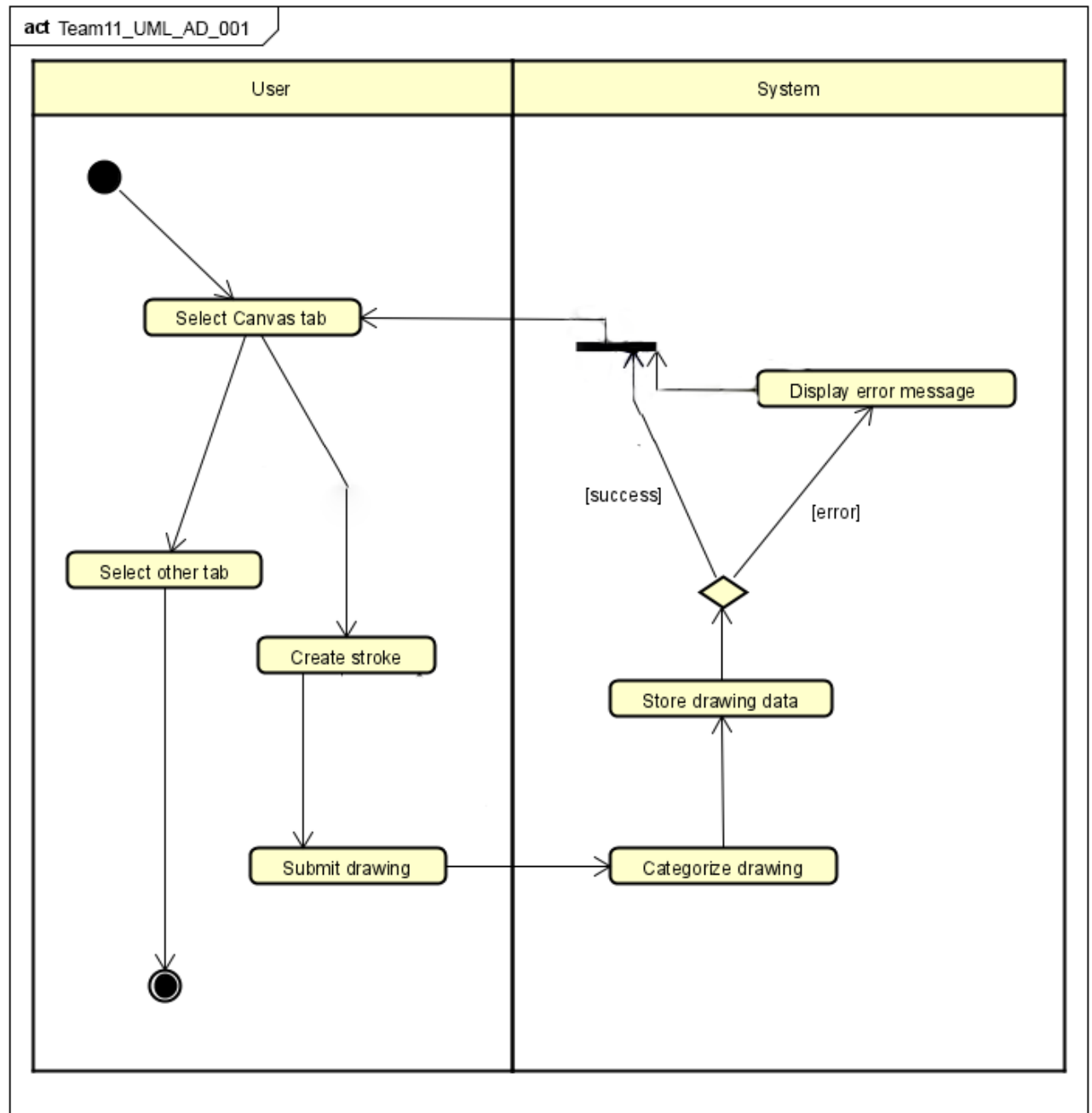


FIGURE: ACTIVITY DIAGRAM -CANVAS

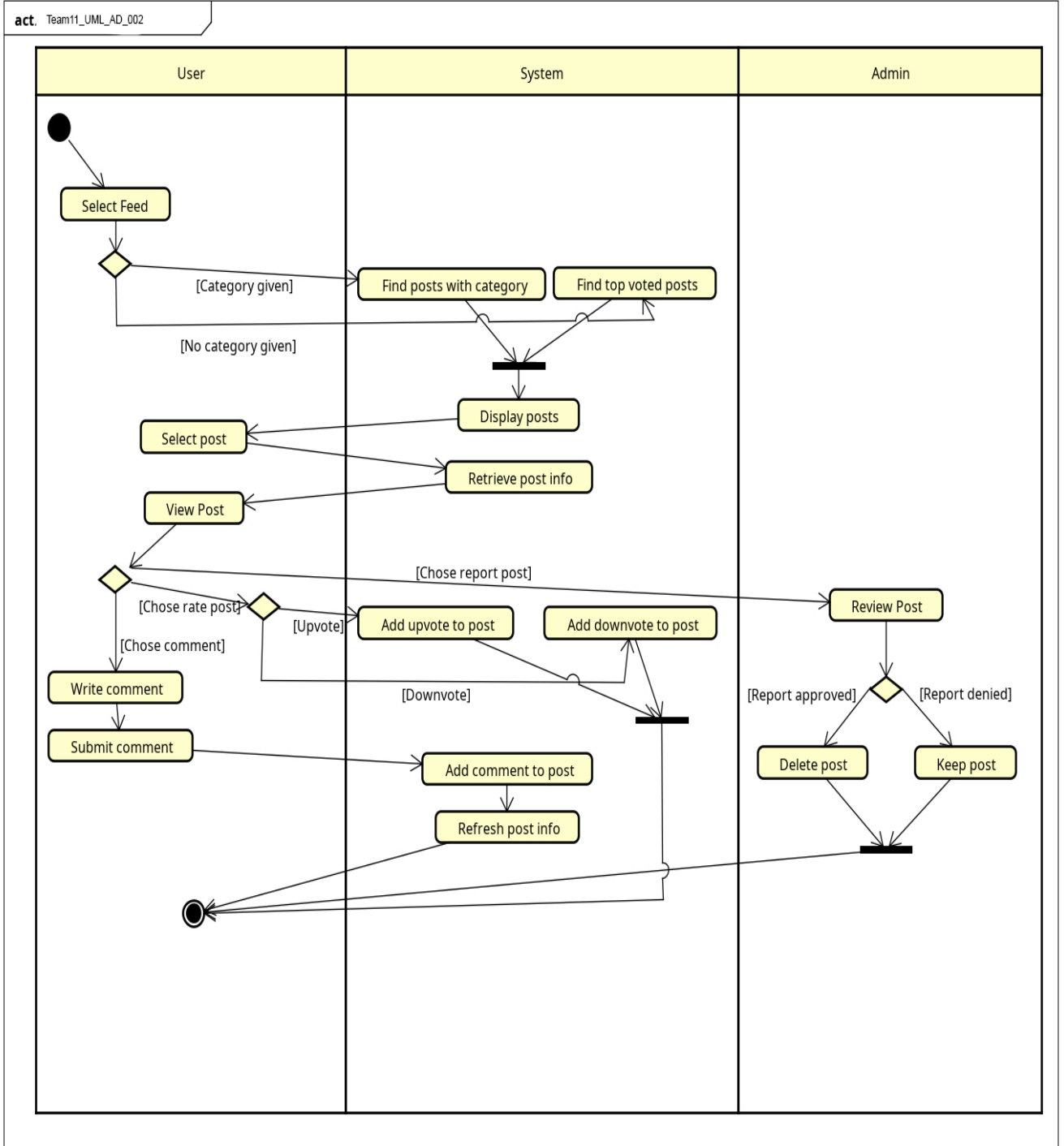


FIGURE: ACTIVITY DIAGRAM- USER FEED

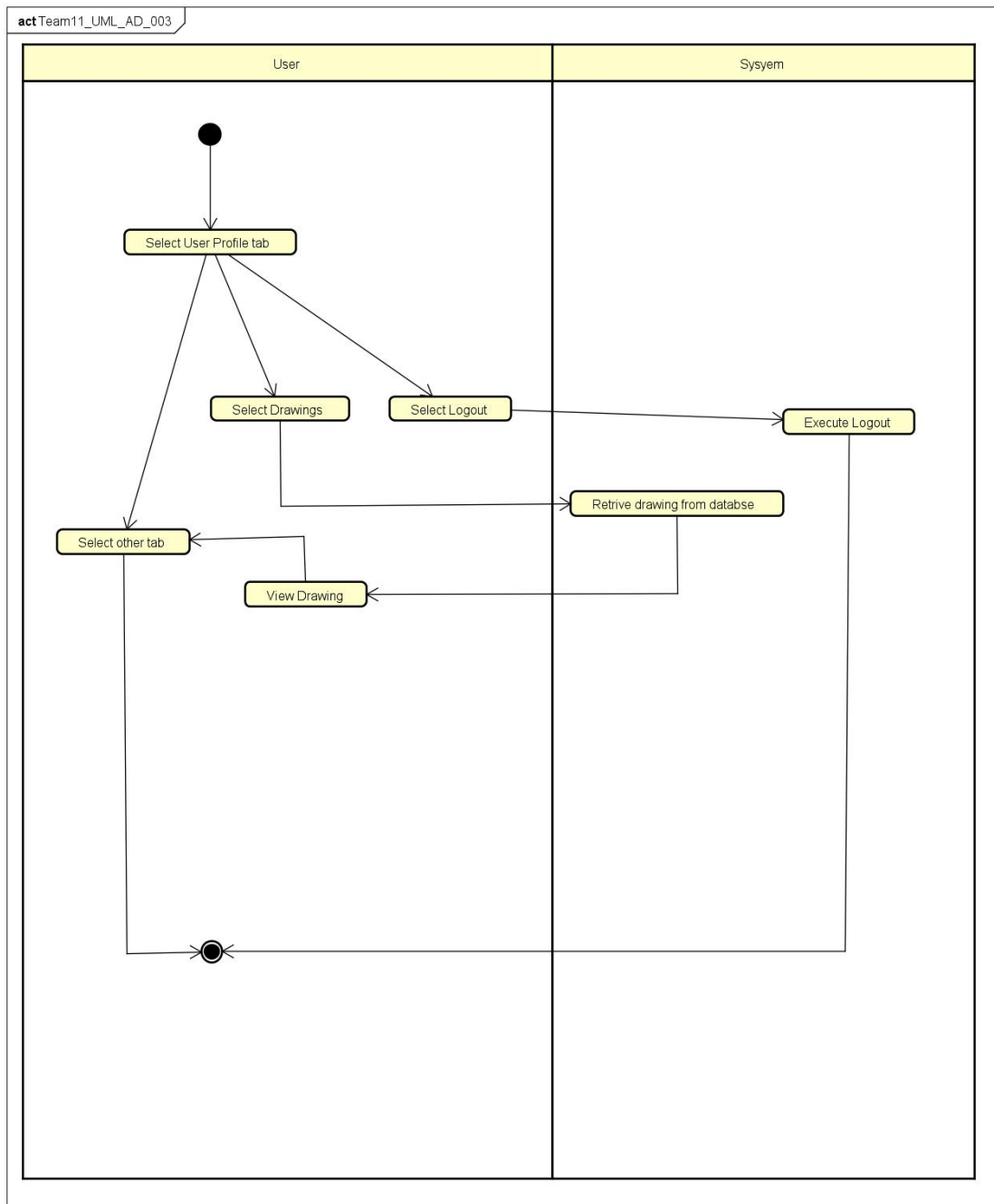


FIGURE: ACTIVITY DIAGRAM- USER PROFILE

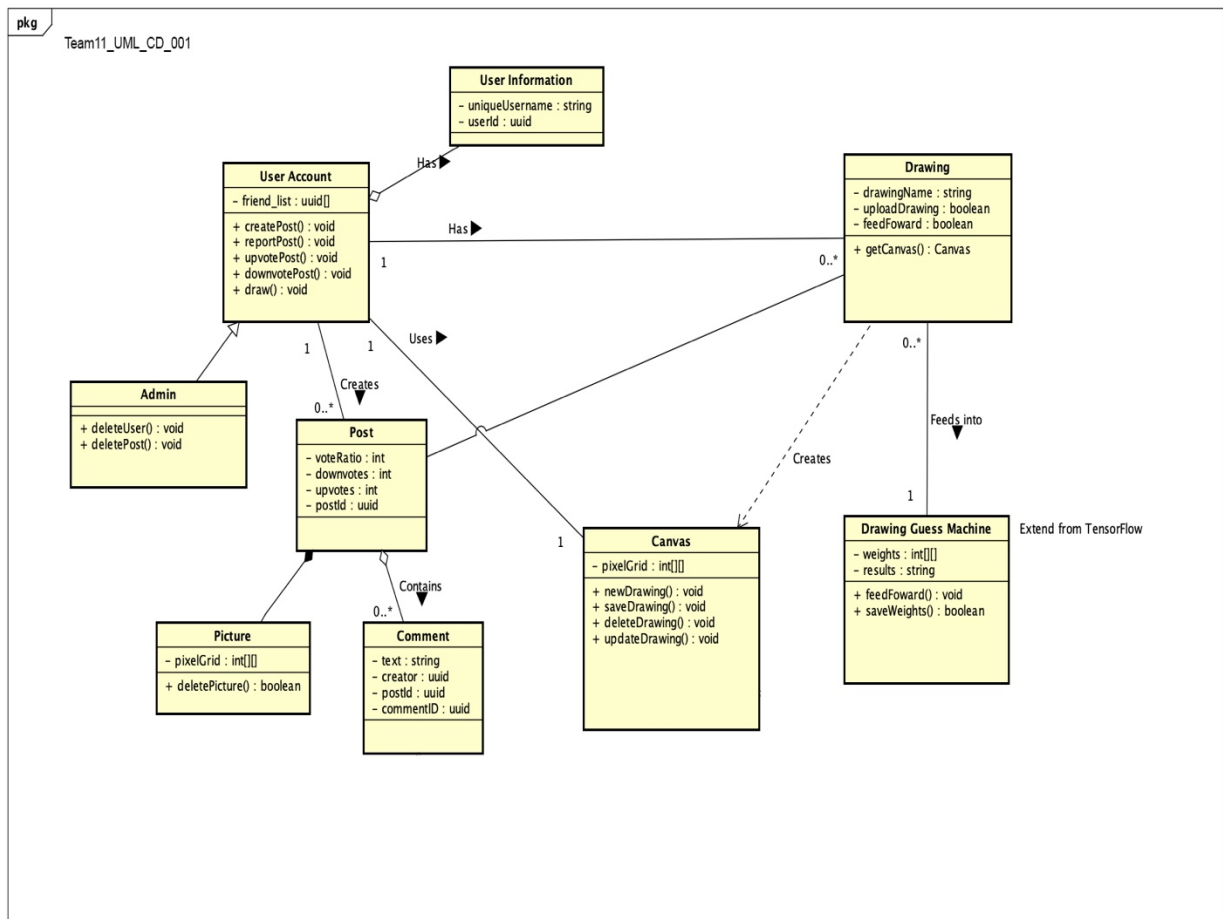


FIGURE: CLASS DIAGRAM

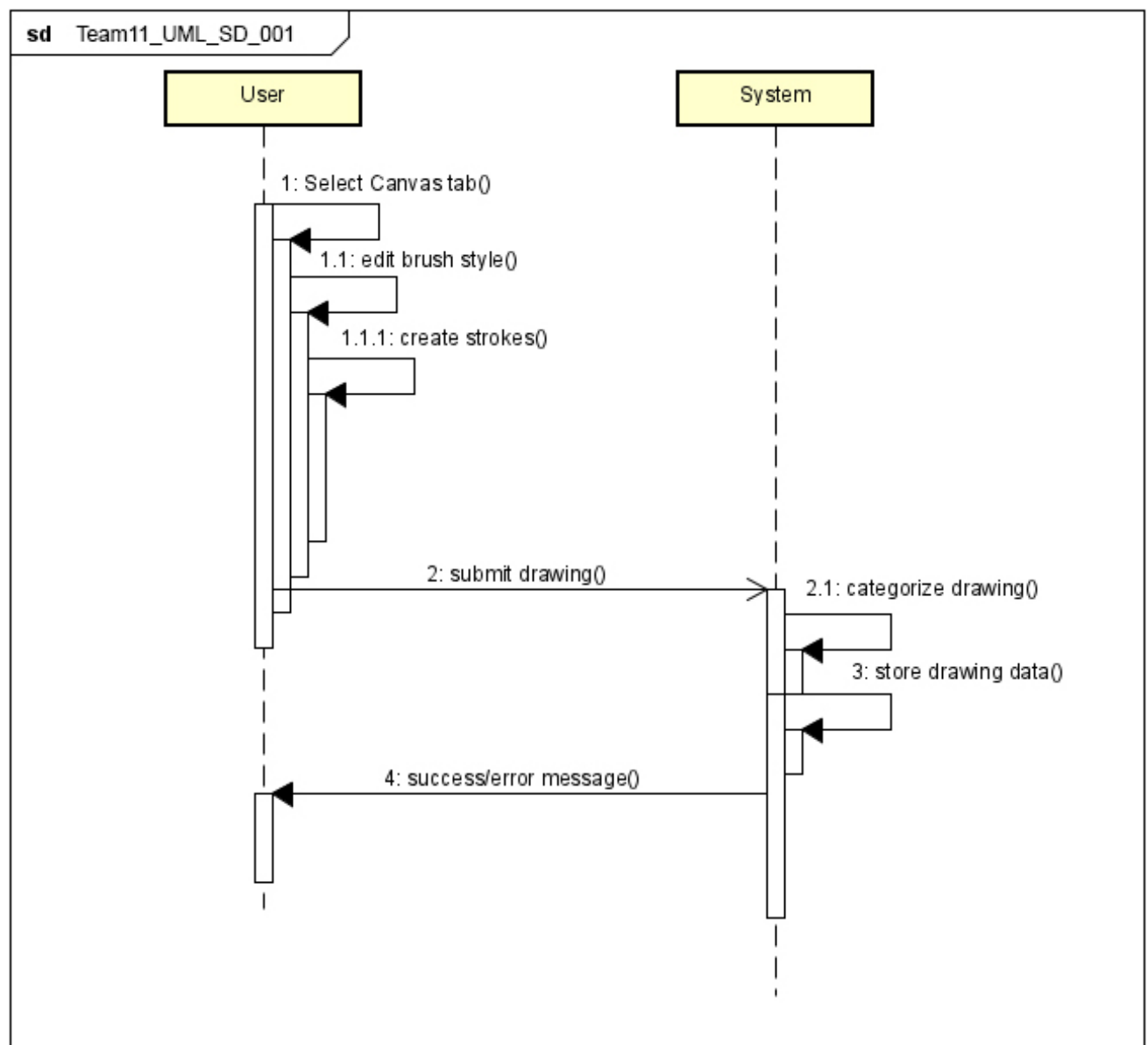


FIGURE: SEQUENCE DIAGRAM- CANVAS

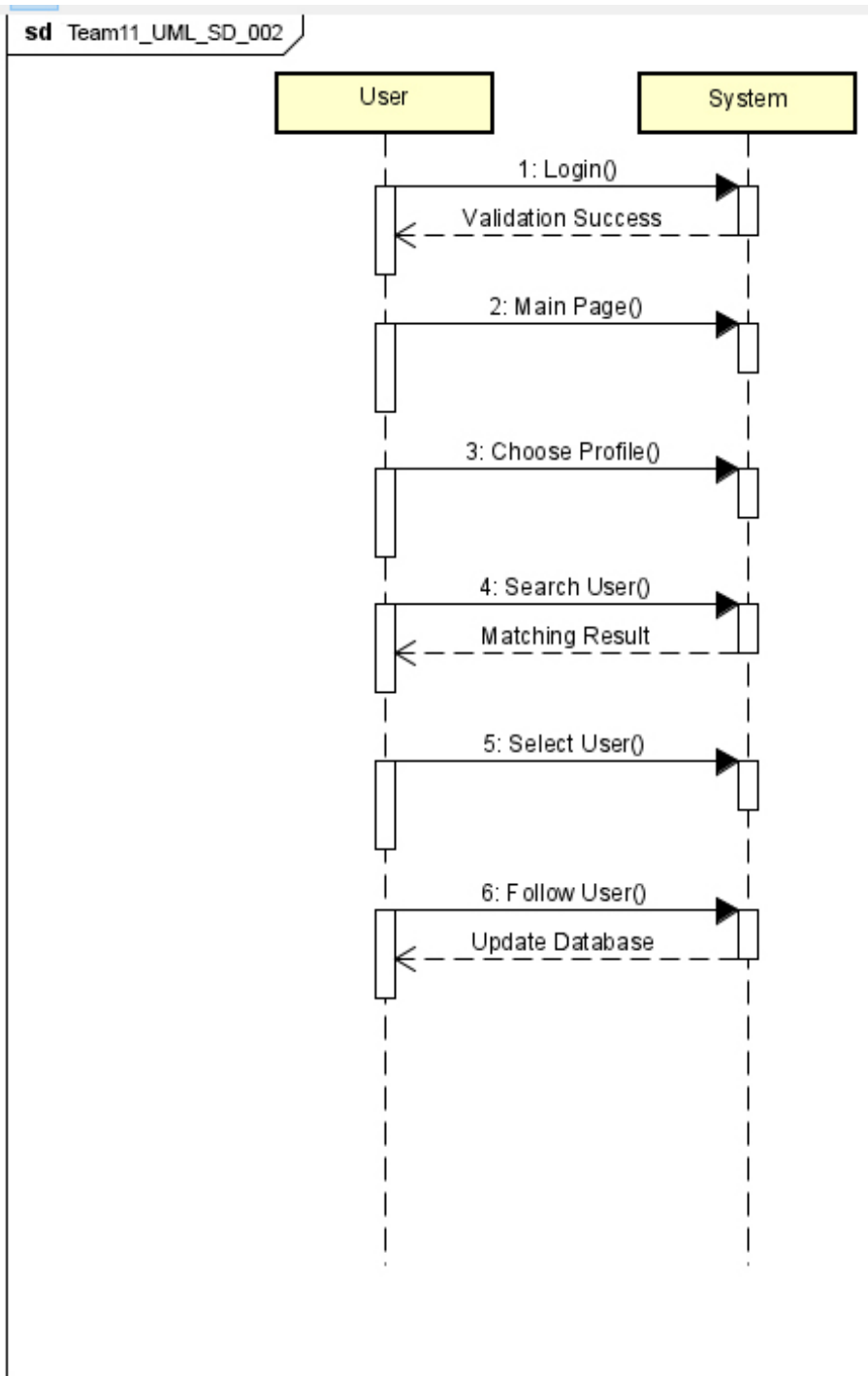


FIGURE: SEQUENCE DIAGRAM- LOGIN

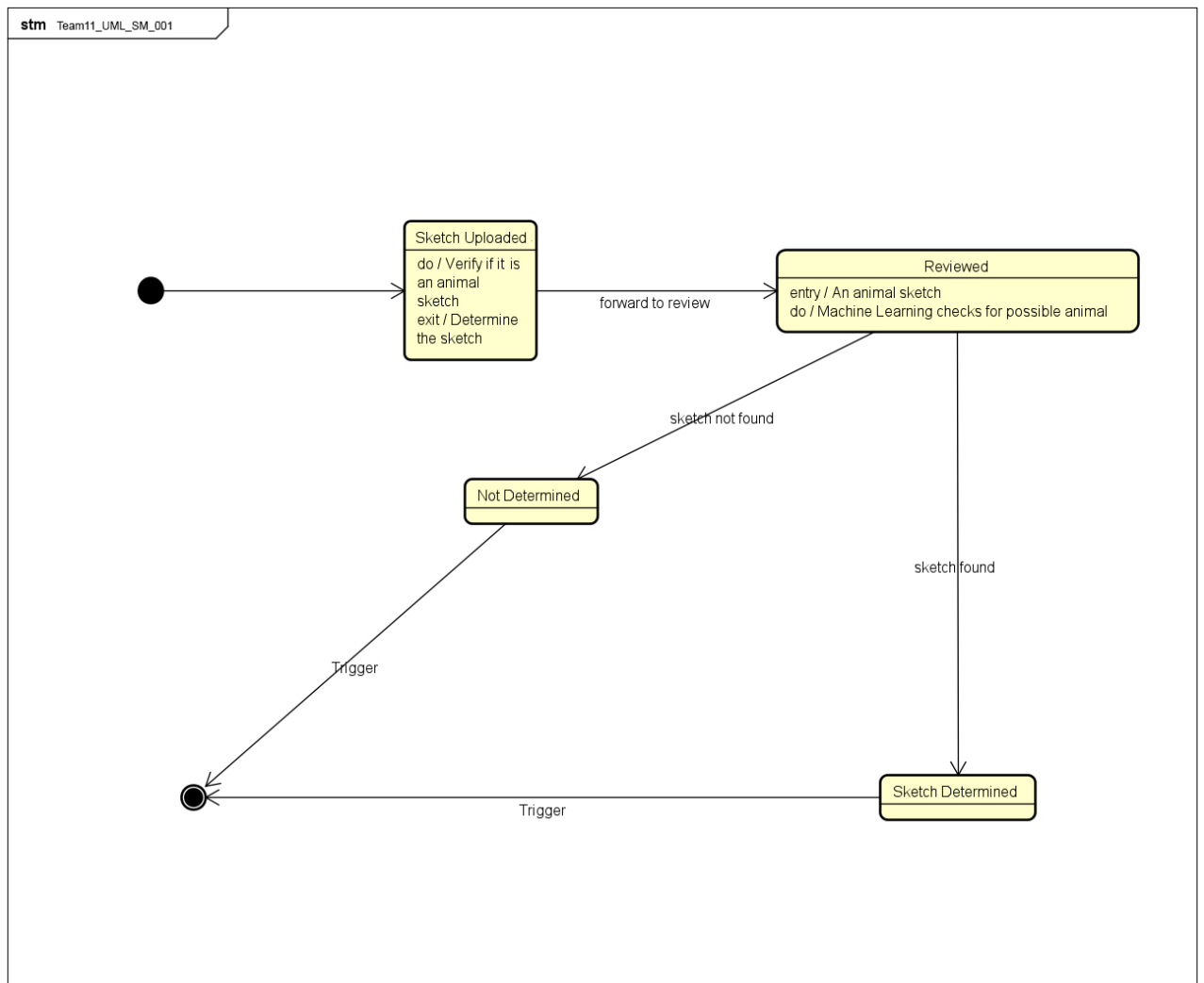


FIGURE: STATE MACHINE DIAGRAM

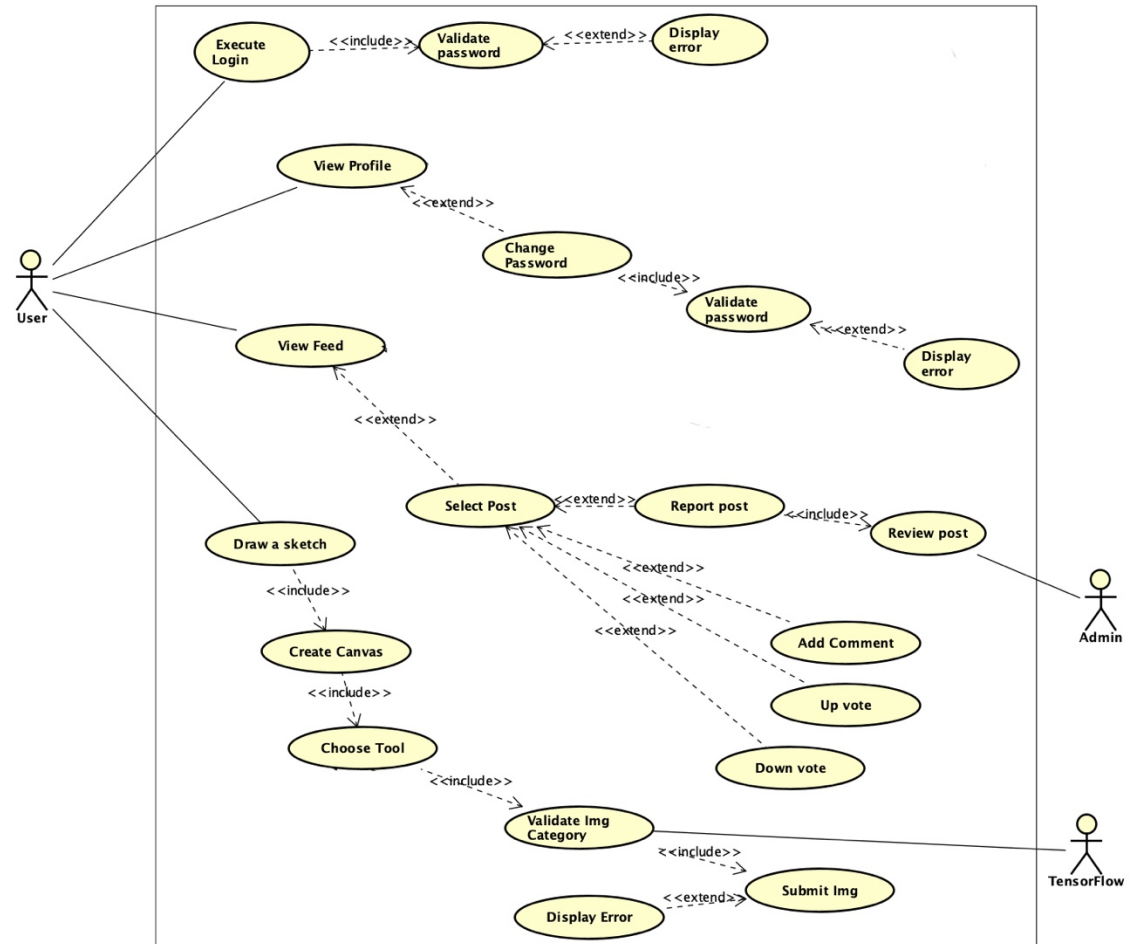


FIGURE: USE CASE DIAGRAM

5.5 Test Case

5.5.1. Introduction and Plan of Approach

Provide a brief description for the following areas:

- Project overview
- List components that will be covered in your Test Plan
- Include any assumptions and anomalies that you have in your Test Plan (e.g. missing components)

Our project is an app for drawing and viewing images online. There are a few different components. Users can create accounts and login and will be able to view posts. In addition, users will be able to draw with a canvas and the ML will be able to guess what animal their drawing is. Currently, most of the app is up and running. The ML has not been integrated with the app but there is a model made and being worked on for accuracy. The canvas and posts are integrated in the app. Some features are missing:

- Comments and voting are not up yet. This is planned to be added, but the base for the app and offline components are priority.
- Infinite scrolling is a work in progress. It should be finished soon

The components covered in this test plan are as follows:

- Registration and Login
- Canvas drawing and saving
- Viewing and refreshing posts
- Testing the ML model

5.5.2. Test Cases: “Login”

Project Name: SketchApp
Test Case Name: Login
Test Case Id: CSE3310/Fall 2020/Team11/ Login

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Enter in a valid and saved email address and password into the correct fields	System should verify and let the user in.	Pass
TC2	Enter invalid email and password and click sign in	System should display a “denied” token	Pass
TC3	Enter a valid User Id and password save to database	System should save the credentials to the database on account made so the user can log back in	Pass
TC4	Deny existing user	If a user with a redundant email tries to register, the system will notify the user	Pass
TC5	Click on login tab	Clicking on login tab will allow user to relogin or register an account	Pass

5.5.3. Test Cases: “Canvas”

Project Name: SketchApp
Test Case Name: Canvas
Test Case Id: CSE3310/Fall 2020/Team11/ Canvas

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Clicking “Canvas” tab on the bottom of the screen	Canvas should open, ready for user drawing.	Pass
TC2	Drawing with finger	Drawing will be shown on screen in real time.	Pass
TC3	Clicking on the save button will save drawing.	Drawing should be saved in its current state to the database as a 255x255 image	pass
TC4	Clicking on erase button	User should be able to erase what they have done, in a similar manner to drawing.	Pass
TC5	Drawing being disposed after clicking away	When user clicks on a different tab in the app, the drawing canvas should be cleared.	Pass

5.5.4. Test Cases: “Viewing Posts”

Project Name: SketchApp
Test Case Name: Viewing Posts
Test Case Id: CSE3310/Spring 2018/Team?/Search

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Click browser button	Show list of recyclerview with images and titles	Pass
TC2	Infinite scrolling: load more posts as user scrolls	Load in more posts when reaching bottom of list	pass
TC3	Click a single item from list	Shows enlarged view of selected item	pass

5.5.5. Test Cases: “ML Model”

Project Name: SketchApp
Test Case Name: ML Model
Test Case Id: CSE3310/Spring 2018/Team?/Form_Club

Test Case No.	Test Case Description	Expected results	Outcome Pass, Fail, Other (comments)
TC1	Proper model setup	Model should be formatted with 255x255 jpg images with 18 classes	Pass, proper amount of classes
TC2	Continuous model training	Model should be made and reloaded and trained with new images to improve accuracy	Pass
TC3	Model conversion to tflite	Model should be able to be converted to .tflite for use in app	Pass
TC4	Model validation set	Model should use 80 percent for training, and 20 percent for validation	Pass
TC5	Model accuracy	Model should achieve 76 percent accuracy with 18 classes	pass
TC6	Model verification	Model should be able to guess a random image outside of the data training set	pass

6. Assumptions and Constraints

6.1 ASSUMPTIONS

The following is a list of assumptions:

- All users are 13+ in age.
- The project will not need maintenance/ evolution after initial release.

6.2 CONSTRAINTS

The following is a list of constraints:

- Team does not have enough experience with database implementation.
- Team does not have enough experience with Machine Learning.

6.3 OUT OF SCOPE MATERIAL

The following is a list of “out of scope” material:

- Post Project activities are not covered.

7. Delivery and Schedule

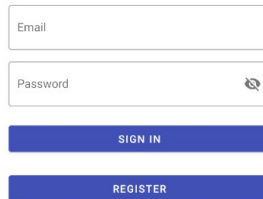
Task/Milestone Description	Anticipated Start Date	Anticipated End Date	Status	Comments
UML diagrams	9/1/2020	10/1/2020	Completed	
SRA document (Includes project objectives, Requirements and UML diagrams)	10/6/2020	10/22/2020	Completed	Deliverable will be the SRA document. All stakeholders agree on the content of the SRA by signing in section 8. Increment 2 Deliverable
Web-scraping tool finished.	10/22/2020	10/29/2020	Completed	Complete Tool to start implementing SketchApp.
Initial App Layout	10/22/2020	10/29/2020	Completed	Finish skeleton for App XML.
Implement Database	10/30/2020	11/6/2020	Completed	Finish database table and ORM setup.
Implement Drawing Feature	10/30/2020	11/15/2020	Completed	Implement drawing board and tools.
Train N.N.	11/10/2020	11/15/2020	Completed	Train NN using Google Doodle data.
Test Plan and CPRs	11/10/2020	11/12/2020	Completed	Deliverables will be the Test Plan and CPRs.
Implement User Page	11/10/2020	11/20/2020	Completed	Implement User Pages.
Final Product and Presentation	11/20/2020	12/10/2020	Completed	Final Presentation and Packing of Product. Deployment to Android Play Store.

8. Stakeholder Approval Form

Stakeholder Name	Stakeholder Role	Stakeholder Comments	Stakeholder Approval Signature and Date
Rodrigo Augusto	Development Manager		
Prajwal Gautam	Project Assistant		
Gabriel de Sa	Developer		Gabriel de Sa 10/22/2020
Lucas Streanga	Developer		Lucas Streanga 10/22/2020
Luke Brown	Developer		Luke Brown 10/22/2020
Saugat Pandey	Developer		Saugat Pandey 10/22/2020
David Rademacher	Developer		David Rademacher 10/22/2020

10 – User Manual

1. Once the user opens the application, user will be prompted to login screen if

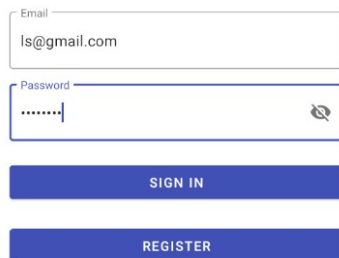


The image shows the default login page of an application. It features two input fields: 'Email' and 'Password'. The 'Email' field is empty, and the 'Password' field is also empty with a toggle icon on the right. Below the input fields are two blue buttons: 'SIGN IN' and 'REGISTER'.

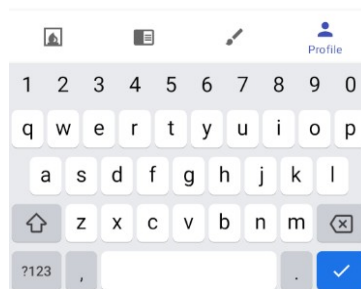
not logged in.

Default login page

2. If user already has an account, user could enter their username and password and then login.



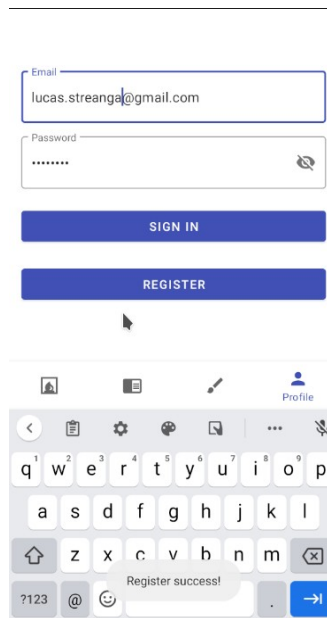
The image shows the login page with the 'Email' field containing 'ls@gmail.com' and the 'Password' field containing '.....'. The 'SIGN IN' and 'REGISTER' buttons are still present.



The image shows the login page with a virtual keyboard displayed over the bottom half. The keyboard is a standard QWERTY layout with a numeric row at the top. The 'Email' field contains 'ls@gmail.com' and the 'Password' field contains '.....'. The 'SIGN IN' and 'REGISTER' buttons are still present.

Entering in login credentials

3. If user needs to create an account, they will enter their email and password to register an account.



Registering Account

Register success toast appears on successful registration

4. Once the user login, user will be directed to a browser view. In this view user can see drawings drawn by other users and Reddit posts.

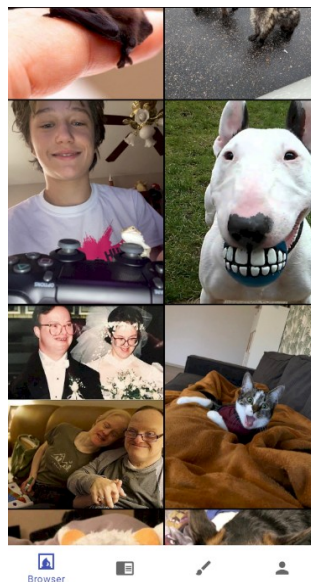
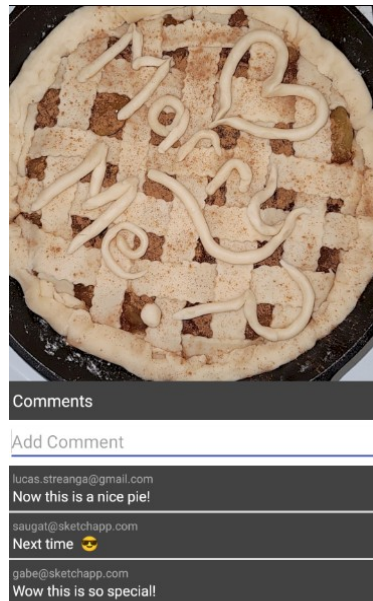


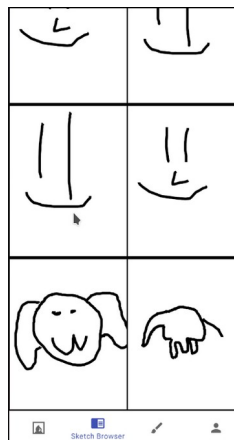
Image Browser

5. Users could click on the drawings in the browser. It will expand the post where user could add a comment or and see other's comments as well as zoom in on the images.



Posts and comments

- There is bottom navigation bar on the application which could be used to switch between browser view, sketch browser, canvas, and the user profile.



- Selecting Sketch Browser on the bottom navigation bar will take the user to the sketch browser window.

Sketch Browser showing all user's drawings

- Selecting a sketch will allow the user to leave comments and see other's comments as well as zoom in on the sketches.



View of a sketch and comments

9. Selecting the canvas on the bottom navigation bar will take the user to the canvas window.



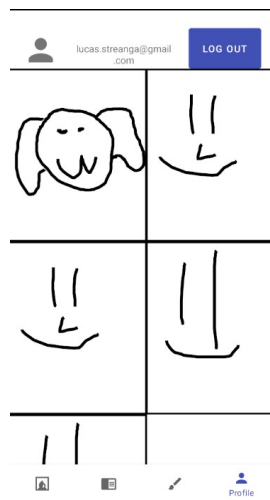
Canvas view

10. In the canvas window, user can draw an animal sketch and submit it, the application will guess the animal with the probability of it being correctly guessed. The user is also able to clear and undo their drawings.



Machine Learning guessing penguin sketch

11. The uploaded animal will be available in the user's user profile and will also be available in the sketch browser view for other users to see and add comments.



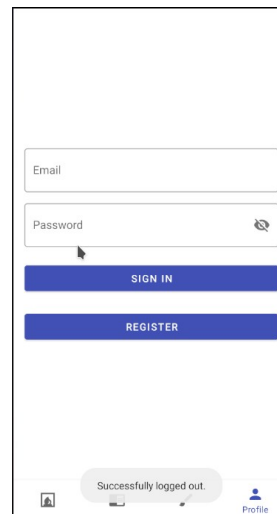
Drawings in user profile
Drawings in Sketch Browser

12. Selecting the User Profile on the bottom navigation bar will take user to the user's profile screen. In the User Profile user can see their username and email. Also, user can view all the drawings drawn by them.



User profile with all images drawn

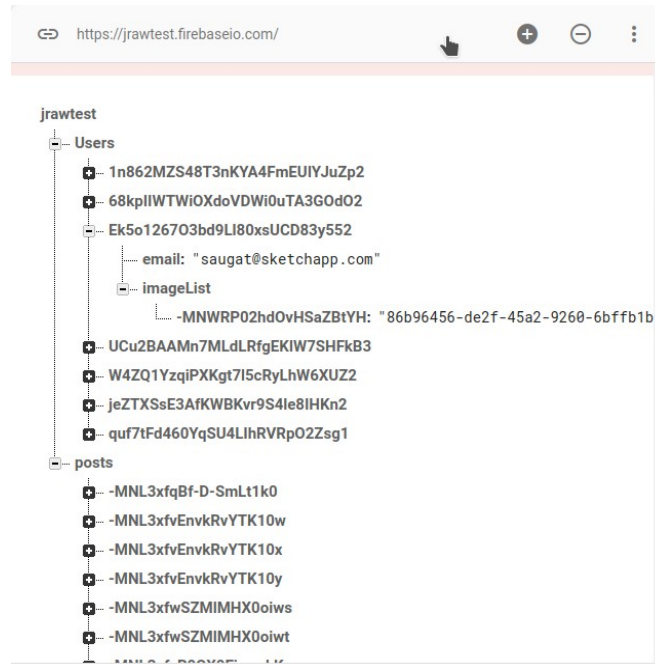
13. There is also a logout button on the user profile which will log them out of the account.



Logout with successful logout toast

11. Source Code

The following is a screenshot of firebase database for pictures and user info



Below are screenshots of the most significant code pieces.
LoginFragment.java

```
import java.util.ArrayList;

public class LoginFragment extends Fragment{

    private static final String TAG = "LoginFragment";

    private FirebaseAuth mAuth;

    private TextInputLayout editTextEmail, editTextPassword;

    private Button registerButton;
    private Button signInButton;

    //These are for the profile!!
    private Toast loginStatus;
    private Button logoutButton;
    private TextView profileEmail;
    private RecyclerView profilePicturesRV;
    private Adapter mAdapter;
    private String urlBase = "gs://jrawtest.appspot.com";
    private final ArrayList<Item> itemList = new ArrayList<>();

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        Log.d(TAG, msg: "LoginFragment onCreateView()");
        View view;

        if(mAuth.getCurrentUser() != null) {
            Log.d(TAG, "LOGIN", msg: "Currently logged in as: " + mAuth.getCurrentUser().getEmail());
            view = inflater.inflate(R.layout.profile, container, attachToRoot: false);
            profileEmail = (TextView) view.findViewById(R.id.profile_email);
            profileEmail.setText(mAuth.getCurrentUser().getEmail());
            logoutButton = (Button) view.findViewById(R.id.logout_button);
            logoutButton.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    signOut();
                }
            });
            profilePicturesRV = view.findViewById(R.id.profile_RV);
            profilePicturesRV.setHasFixedSize(true);
            profilePicturesRV.setLayoutManager(new GridLayoutManager(view.getContext(), spanCount: 2));

            //Get images only from current user.
```

RedditViewerFragment.java

```
import ...

public class RedditViewerFragment extends Fragment {

    private static final String TAG = "RedditViewerFragment";

    private RecyclerView mRecyclerView;
    private Adapter mAdapter;
    private ArrayList<Item> mItemList;
    private Bundle itemBundle;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {

        itemBundle = new Bundle();
        mItemList = new ArrayList<>();

        View view = inflater.inflate(R.layout.fragment_reddit_viewer, container, (attachToRoot: false));

        // setup recycle viewer
        mRecyclerView = view.findViewById(R.id.recycler_view);
        mRecyclerView.setHasFixedSize(true); // saves memory because size doesn't change
        // mRecyclerView.setLayoutManager(new LinearLayoutManager(view.getContext()));
        mRecyclerView.setLayoutManager(new GridLayoutManager(view.getContext(), spanCount: 2));

        // get Bundle passed from MainActivity
        itemBundle = getArguments();

        if (itemBundle != null) {

            // get ArrayList out of the Bundle from MainActivity
            mItemList = itemBundle.getParcelableArrayList(key: "redditItemList");
            mAdapter = new Adapter(view.getContext(), mItemList);
            mRecyclerView.setAdapter(mAdapter);
        } else {
            Log.d(TAG, "msg: setAdapter:failed");
        }
        return view;
    }
}
```

PaintView.java

```
public class PaintFragment extends Fragment implements View.OnClickListener {

    public static final String TAG = "PaintFragment";

    FirebaseAuth mAuth;

    PaintView paintView;
    public Button uploadButton, undoButton, clearButton;

    Bitmap bmp;

    private Classifier tf;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        Log.d(TAG, msg: "PaintFragment onCreateView()");

        View view = inflater.inflate(R.layout.fragment_paint, container, attachToRoot: false);

        paintView = view.findViewById(R.id.PaintView);
        // testButton = view.findViewById(R.id.test_button);

        // used for testing upload functionality
        uploadButton = view.findViewById(R.id.upload_button);
        uploadButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    storeBitmapFirebase();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });

        undoButton = view.findViewById(R.id.undo_button);
        undoButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { paintView.undo(); }
        });

        clearButton = view.findViewById(R.id.clear_button);
        clearButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { paintView.clear(); }
        });
    }
}
```

Classifier.java

```

public class Classifier {

    public static final String [] classes = {"bear", "bee", "bird", "cat", "cow",
        "dog", "dolphin", "duck", "elephant", "fish", "frog", "horse",
        "mouse", "penguin", "rabbit", "sheep", "snake", "whale"};
    private final String filename = "model.tflite";

    private Interpreter tf;
    private final Interpreter.Options tfOptions = new Interpreter.Options();
    private TensorBuffer probabilityBuffer;
    private Activity activity;

    Classifier(Activity activity) { this.activity = activity; }

    public class ModelOutput {
        String identifier;
        float probability;

        ModelOutput(String identifier, float probability) {
            this.probability = probability;
            this.identifier = identifier;
        }
    }

    private MappedByteBuffer loadModelFile(Activity activity) throws IOException {
        AssetFileDescriptor fileDescriptor = activity.getAssets().openFd(filename);
        FileInputStream inputStream = new FileInputStream(fileDescriptor.getFileDescriptor());
        FileChannel fileChannel = inputStream.getChannel();
        long startOffset = fileDescriptor.getStartOffset();
        long declaredLength = fileDescriptor.getDeclaredLength();

        return fileChannel.map(FileChannel.MapMode.READ_ONLY, startOffset, declaredLength);
    }

    public ModelOutput classify(Bitmap bitmap) throws IOException {

        if(bitmap == null) {
            Log.d("tag: \"Classifier Error\", \"msg: \"bitmap is NULL\"");
            return null;
        }

        tf = new Interpreter(loadModelFile(activity), tfOptions);

        TensorImage image = loadImageFromBitmap(bitmap);
        probabilityBuffer =
            TensorBuffer.createFixedSize(new int[]{1, Integer.BYTES * classes.length}, DataType.UINT8);
        tf.run(image.getBuffer(), probabilityBuffer.getBuffer());
        //We will print the output
    }
}

```

```

121 #         drawing = next(drawings[y])
122
123 model = None
124
125 # Outerloop. How many times will we train?
126 for z in range(total_training_iterations):
127     print('recreating drawing jpg...')
128     threads = []
129     for j in range(len(drawings)):
130         t = Thread(target=create_drawings, args=(classes[j],drawings[j]))
131         t.start()
132     threads.append(t)
133     for t in threads:
134         t.join()
135
136     data_dir = 'jpg'
137     batch_size = 32
138     img_height = 255
139     img_width = 255
140
141     num_classes = len(classes)
142
143     train_ds = tf.keras.preprocessing.image_dataset_from_directory(
144         data_dir,
145         validation_split=0.2,
146         subset="training",
147         seed=123,
148         image_size=(img_height, img_width),
149         batch_size=batch_size)
150
151     val_ds = tf.keras.preprocessing.image_dataset_from_directory(
152         data_dir,
153         validation_split=0.2,
154         subset="validation",
155         seed=123,
156         image_size=(img_height, img_width),
157         batch_size=batch_size)
158
159     data_augmentation = keras.Sequential(
160     [
161         layers.experimental.preprocessing.RandomFlip("horizontal",
162                                                     input_shape=(img_height,
163                                                         img_width,
164                                                         3)),
165         layers.experimental.preprocessing.RandomRotation(0.1),
166         layers.experimental.preprocessing.RandomZoom(0.1),
167     ]
168     )
169
170     class_names = train_ds.class_names
171     print(class_names)
172
173     AUTOTUNE = tf.data.experimental.AUTOTUNE
174
175     train_ds = train_ds.cache().shuffle(200).prefetch(buffer_size=AUTOTUNE)
176     val_ds = val_ds.cache().prefetch(buffer_size=AUTOTUNE)
177
178     if model is None:
179         model = Sequential([
180             data_augmentation,
181             layers.experimental.preprocessing.Rescaling(1./255),
182             layers.Conv2D(16, 3, padding='same', activation='relu'),
183             layers.MaxPooling2D(),
184             layers.Conv2D(32, 3, padding='same', activation='relu'),
185             layers.MaxPooling2D(),
186             layers.Conv2D(64, 3, padding='same', activation='relu'),
187             layers.MaxPooling2D(),
188             layers.Dropout(0.2),
189             layers.Flatten(),
190             layers.Dense(128, activation='relu'),
191             layers.Dense(num_classes)
192         ])
193         model.compile(optimizer='adam',
194                     loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
195                     metrics=['accuracy'])
196
197     epochs=10
198     history = model.fit(

```


Appendix:

None