CAMTAG

Oskar Neuman

Developer

neos1666@student.ju.se

Gustaf Levinsson

Developer

legu1666@student.ju.se

Project Work in Android Development

Jönköping University 2019

Table of Contents

Introduction

Recent progress in the field of AI development has brought great power to digitally analyzing picture content. The possibility of detecting objects in a picture is increasing as machine learning algorithms are becoming more fed with object data.

Camtag offers the player the experience of testing state of the art AI image technology through a comprehensive challenging game experience.

The game

After a quick signup process the player can start a challenge against a friend registered in the game. They are matched into a game introducing the players and their posted pictures so far. The player can now start a round by clicking on the "TAG" button. Tags such as: "cow", "rock" or "person" presents to the user through the familiar camera view. The player will then try to capture the objects in his surroundings by taking an image with as many of the given tags as possible. The score is calculated by reading the image and analyzing the confidence score of how well the capture represents the given tags for each object requested. The player submits the photo and the score thereby ending her turn in the round. When the last capture is submitted by a player, he must wait for the opponent to finish all rounds.

The game finishes when all rounds are finished, the total score is calculated, and the winner is presented to the players.

Purpose

The hot topic of image recognition and AI in general, intrigues not only gamers and tech savvy people, it also reaches the scope of attention for the ordinary mobile phone user. Camtag seeks to fulfill the urge to experience this upcoming technology in a way that is compelling to everyone, a game.

Engage creativity

Another of the main purposes of Camtag is to encourage creativity among the players. The player forces to make certain decisions like evaluating whether she should try to capture a great photo with just one tag, or a decent photo with multiple tags.

Enhance socializing

Camtag offers not only the possibility to play against friends battling logically, it also presents the captured images for the other player so that the players come to know each other by the other persons choice of object to capture and the surroundings.

Activate the player

Many of the tags are connected to the outdoors making the player go outside to capture more tags. Tags such as "tree" or "cow" are easier to capture outside, and other tags such as "sea" or "cloud" are almost impossible to capture indoors. This will make the player go out and explore the world and not only capture great images, but also capture great memories.

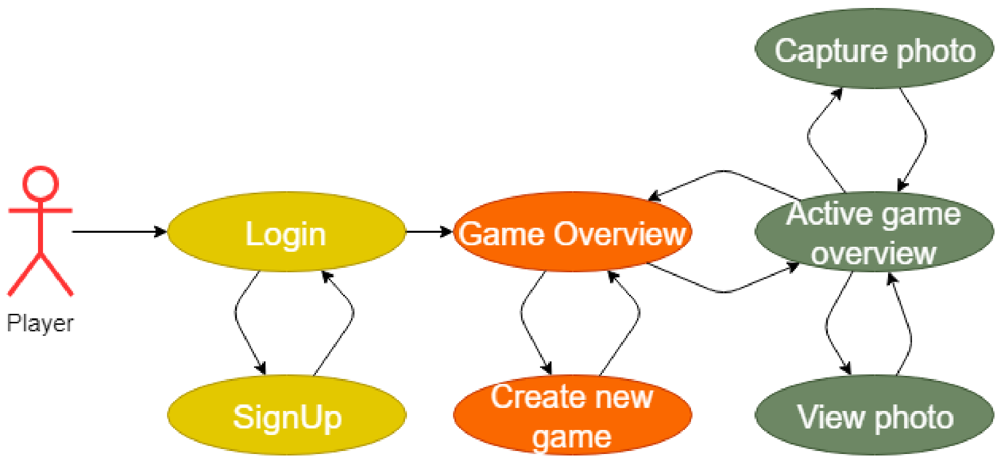
Audience
Due to being very versatile and not age-bound thanks to the great verity of the tags, Camtag is for everyone. The tags are general enough so that anyone can play and understand what they should try to capture. Everyone, regardless of age, can play against each other and create great connections.

Future
In later implementations Camtag aims to be able to select a category when starting a new game. This is to allow players to fit the game-tags to their environment. This could also open for players to play against others with the same interest, thus, sparking new friendships.

Camtag has a lot of growing potential due to the social tendency of the game. Features that would enrich the player social experience like private messaging is something to think about for later improvement of the game.

Use case description
The use case diagram bellow shows the players interaction with the application.

- The first view a player interacts with is the login screen. From here the player can login with their account or create an account using the sign-up view. After the sign up is complete the player has to login. If the player is logged in recently then the first view will be the game overview.
- This view displays all the current games and all finished games. The player will also be able to start new games by clicking the “Find new game”- button and open a new window.
- This window is used for searching for opponents and starting a new game with the desired settings. After starting a game, the window will disappear and the game overview appears again with the newly started game.
- When the player selects one of the games a new view, the active game overview, appears showing the current statistics regarding the game such as score, pictures and rounds, and see the state of the game. If it is the players turn the player can see the tags for the round and then open up the camera view and snap a photo.
- After the picture is taken the player will get back to the active game overview with the new picture and its score next to it. If the player would like to look at a picture the player can select one and it will open in a new window for further inspection.



Overview

The application uses Amazon Rekognition, an image and video analyzing Application programming interface (API) created by Amazon. The API uses machine learning developed by Amazon to analyze images or videos stored on Amazons file storing service, Amazon S3 Bucket, and "tags" a picture corresponding to its contents.

The API is easily implemented and does not require former machine learning knowledge. The API has many applications such as facial recognition, pathing and reading out text from an image. We only use it to receive tags from uploaded images.

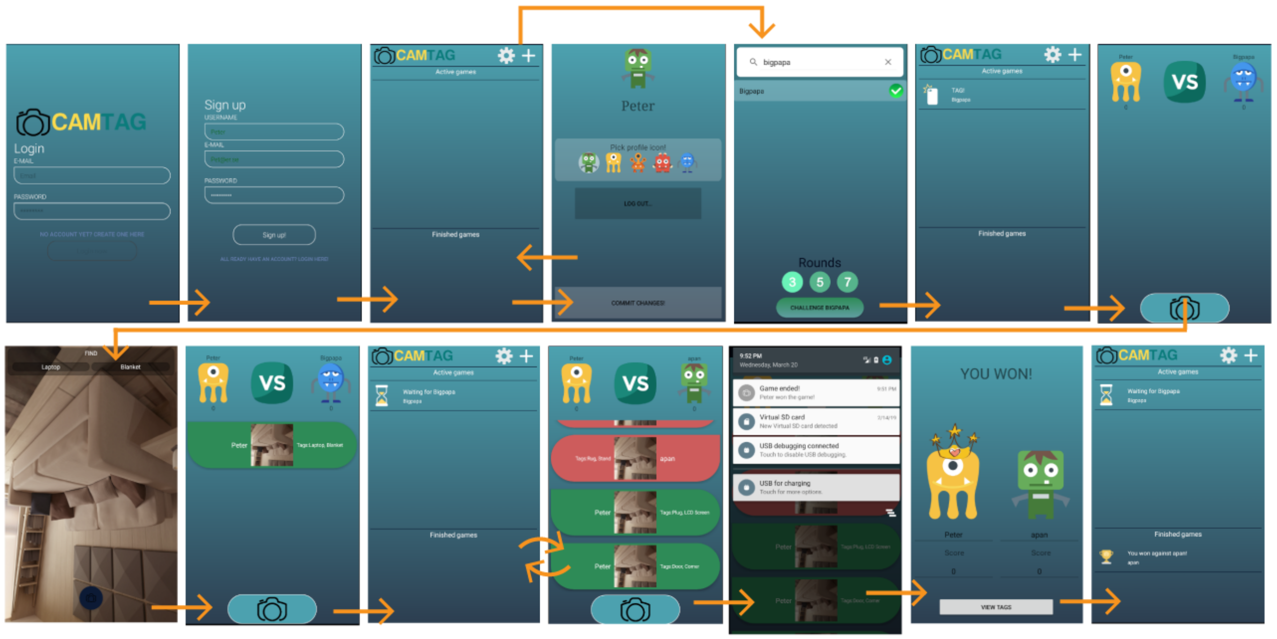
Amazon S3 Bucket is used for storing images. Amazon S3 is a cloud storing service with an API for managing the uploaded files.

The data is stored in a database Amazon Relational Database Service (RDS) and the images are stored with Amazon S3. The communication to the database and file storing service uses our own CamTag-API. The backend is developed in JavaScript with node express. Deployed on Amazon Beanstalk for continuous uptime and global connection range. This suite the purpose of game releasing by enabling on demand upgrades (scaling), thereby limiting production and maintenance costs associated with wrong expectancy of usage of the service.

For notifications and signaling functions we utilize Firebase, googles cloud storage service. Firebase allows for notifications. Firebase sends signals to the application for updates and refreshing data from the database. It also sends notifications regarding new games and when a game is finished.

Application

(Figure 1)



The application is built with 3 activities, 8 fragments and 2 services. The first and default activity is the login activity with 2 fragments.

Login and signup fragments

A player can login with email and password or go to the next fragment where a player can sign up and create an account. The account creation progress checks the input against the backend and signals the player if the email or username is vacant or occupied, only enabling a signup request if all requirements are satisfied.

After the player has logged in or has signed up, they get to the next activity. The application stores a login token in local storage. When the player opens the application next time it checks if the application has an active token stored. If the player has a valid token they are greeted and brought to the game overview activity right away.

Gamelist activity

The game list activity is the home of the player and acts as the main activity of the application. It holds most of the app contents with 8 fragments. These fragments are displaying information and taking minor player input. The main menu has 2 recycler views where the ongoing and finished games are displayed and 2 image buttons that leads to account setting and starting of new games. When an item in the recycler view is clicked it opens a new fragment to show game information. The same fragment is opened with both recycler view and adapts depending on the status of the game.

Profile fragment

The account settings fragment lets the player pick between different images to select a profile image. The player can also log out to switch accounts or sign up for a new. When the player logs out the login token is removed, and the player is brought back to the first login activity with the fields filled with the information of the most recent player signed in for easy login / logout.

New challenge fragment

To start a new game the player search for a username of another player in the new game fragment. The search view that the player types in to takes a search term that matches a part of a username. The results of the search are displayed in a recycler view. The player then selects a player to challenge, selects the number of rounds through selecting one of the 3 alternative image views and start game by pressing a button with adapting text depending on selection.

Game information fragment

The game information fragment shows player and opponent selected profile image with image views. When the opponent picture is pressed it opens the account settings fragment with an adapted view showing email and account image. A recycler view shows each post that the players have posted. If a post is clicked it opens a fragment that shows the post full screen with information of the post. If the player can post a new image the tagbutton for a new post is visible. The button opens the camera activity when clicked. If the player has posted all posts for the game the tagbutton is hidden and the player must wait for their opponent to finish all rounds.

Camera activity

The camera activity requests and requires access to camera and storage and is a full screen camera view with 2 overlays. One overlay shows the tags that the player is aiming to take a photo of, and the other overlay is the capture picture button. When the photo is taken an overlay is activated and informs player that the picture is being posted.

Analyzing the picture

The picture is uploaded to Amazon S3 bucket to be able to be analyzed by Amazon Rekognition. Amazon Rekognition is sent an URL to the picture and the Rekognition response contains information about the analyzed pictures. The Camtag backend extracts the tags and the confidence score (accuracy) of the tags that are recognized within the picture to calculate the score. The score is sent to the player application and displayed to the player which then is brought back to the game information fragment.

Game ending

When all rounds have been posted a winner is calculated and the players in the game gets notifications. When opened the player is brought to the winner fragment displaying the winner and stats for the game.

Data managing

Camtag is very dependent on the backend to function properly and therefor makes use of third-party libraries and updating services to make sure the connection and refreshing of games are reliable and convenient.

Retrofit for API calls

Camtag uses a third-party library called “retrofit” a type-safe HTTP-client for Android and Java which has the power to create objects straight from the API responses. By generating “pojo” classes from plain text API responses that can be used like ordinary Java classes. The pojo classes are used in the interface that declares all the applications calls to the backend. Using the interface methods in the services makes the process of making calls very convenient.

Firebase messaging

Upon login the player retrieves a firebase token written to a row in the Member SQL table. This token is used by the backend and firebase cloud to communicate between the clients and the backend.

Upon player posting a new picture or starting a new game Camtag service classes receives a message telling the client to perform a call to the API for the newly created information. This leads to less stress for the backend server and offers the client device less workload. Another benefit is the notifications sent when player posting occurs or a new game starts. Providing the player, the possibility to keep updated on the active games while performing other tasks on the device.

Services

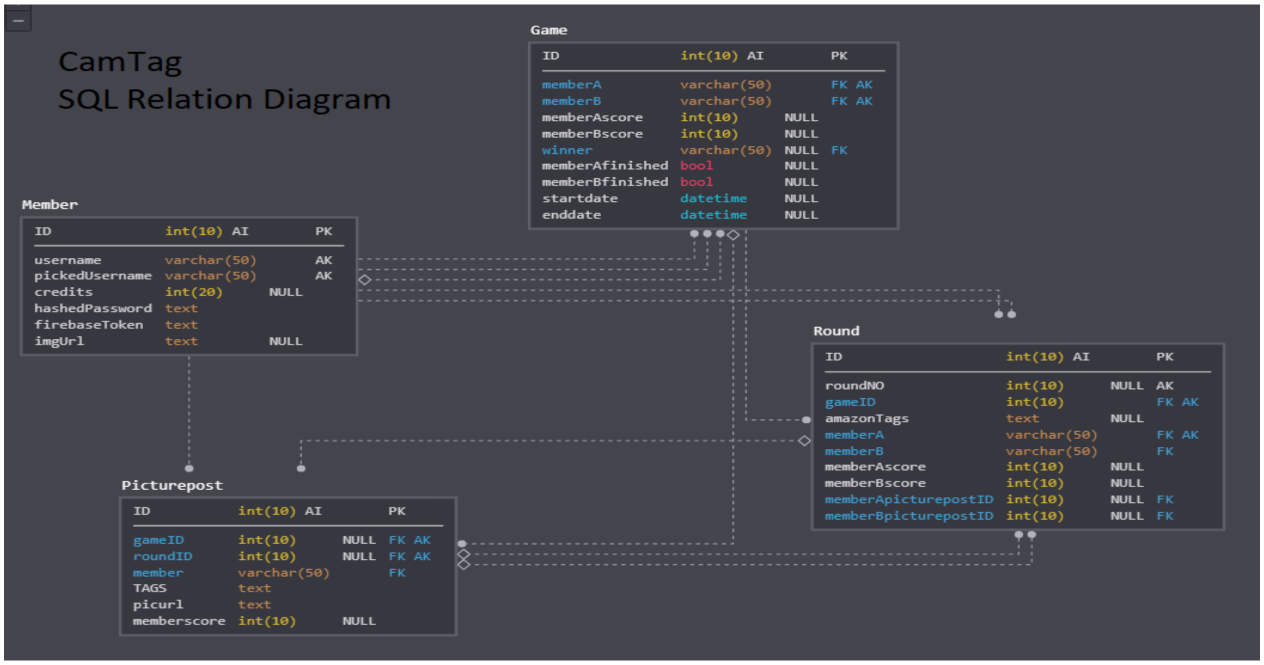
Camtag has two services running in the background, one for firebase messages and one for game updating. The game updating service both receives and sends broadcasts across the application and signals the fragments when new data is available so that the recycler views can update. The service holds a retrofit API interface that downloads data from the backend when signaled from firebase. All calls that are dependent on new data is being made in the service, fragments make calls on their own when it's necessary for the player to wait for a progress (like signup and login).

Client-side data storing (caching)

Camtag currently writes lists upon updates to static variables and does not cache images locally. This leads to unnecessary data downloading on behalf of bandwidth. The feature of offline caching and hash maps for lists is scheduled for upcoming releases.

The SQL database

Game variables are stored on a SQL database consistent of 4 tables: Member, Game, Round, and Picturepost. Game holds the game related information for a game, such as the members and the score. Rounds are connected to Games and holds information of each round of tags that are generated to the players.



Game

ID	int(10) AI		PK
memberA	varchar(50)		FK AK
memberB	varchar(50)		FK AK
memberAscore	int(10)	NULL	
memberBscore	int(10)	NULL	
winner	varchar(50)	NULL	FK
memberAfinished	bool	NULL	
memberBfinished	bool	NULL	
startdate	datetime	NULL	
enddate	datetime	NULL	

Member

ID	int(10) AI		PK
username	varchar(50)		AK
pickedUsername	varchar(50)		AK
credits	int(20)	NULL	
hashedPassword	text		
firebaseToken	text		
imgUrl	text	NULL	

Round

ID	int(10) AI		PK
roundNO	int(10)	NULL	AK
gameID	int(10)		FK AK
amazonTags	text	NULL	
memberA	varchar(50)		FK AK
memberB	varchar(50)		FK
memberAscore	int(10)	NULL	
memberBscore	int(10)	NULL	
memberApicturepostID	int(10)	NULL	FK
memberBpicturepostID	int(10)	NULL	FK

Picturepost

ID	int(10) AI		PK
gameID	int(10)	NULL	FK AK
roundID	int(10)	NULL	FK AK
member	varchar(50)		FK
TAGS	text		
picurl	text		
memberscore	int(10)	NULL	

Picturepost organizes the pictures and connects to the round and the member posting the picture. Member holds the member variables like the hashed password, picked username, chosen image and firebase token received upon login.
(Figure 2)

Supported devices

Camtag requires a minimum SDK version of 21. The targeted SDK version is 28. With a minimum of version 21 we can reach 88.9% of Android players. We started of supporting version 19, but due to issues with the camera activity we decided to go up to version 21 where a lot of improvements regarding camera handling occurred. During the project we encountered difficulties where going to an even higher SDK would be a simple solution but going up to SDK 23 would only cover around 70%. We were able to work around the problems and keep the functionality that were necessary for our application.

Current state of development

Camtag is currently available as an alpha release for internal testing, this version is solely for proof of concept and the to-do list is long. The alpha provides a basic functionality testing of the main components of the game but there is still some work to be done to make a stable release suitable for a greater audience. The Camtag gitlab repo provides the developer attending the project with a board of issues and a structured outlook on the project.

The developing future of Camtag

Although the application and backend are prepared for future implementations. Camtag's focus is still on closing issues related to basic functionality like the previous mentioning of caching images feature. Milestones include releasing a closed beta when the application progress has reached a point where the necessities are fulfilled, and the application is working without reoccurring issues. After that is done new features will be introduced.

There is a lot of power within Amazon Rekognition that we would like to use, such as receiving the picture analyzed with tags and facial recognition. The game itself can be improved with new features and game modes.

Due to the social nature of the game there is a lot to build upon, introducing private messaging and sharing functionalities could create a community around the activity of Camtagging.