Treasure Hunt

Let's start

01

Introduction

Treasure hunt app gameplay

There are 2 main modes for a user:

- **Game mode**: a player searches and selects a game to play, where each game is composed by a set of steps that need to be solved.
- **Make mode**: a user can create a game for other player by defining the steps of which it is made of.

Step types

Camera

The player has to guess an object by a question and make a photo of it

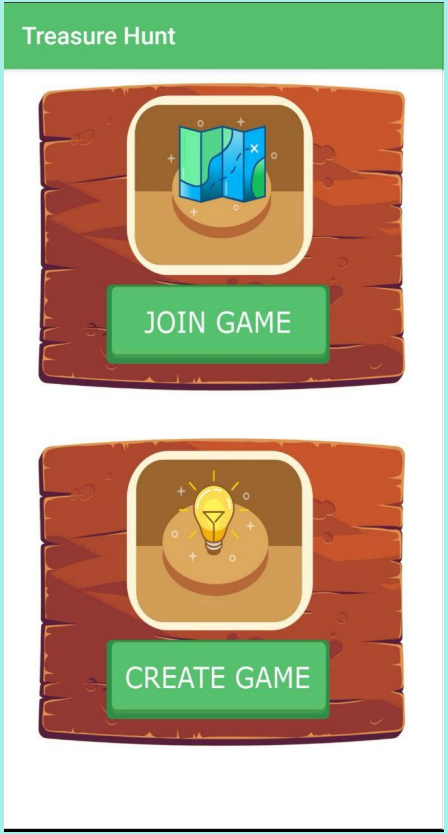
Position

The player has to guess a place to go by a question

Hints

The player has a maximum of 3 hints
in each game to answer a question

Initial Activity

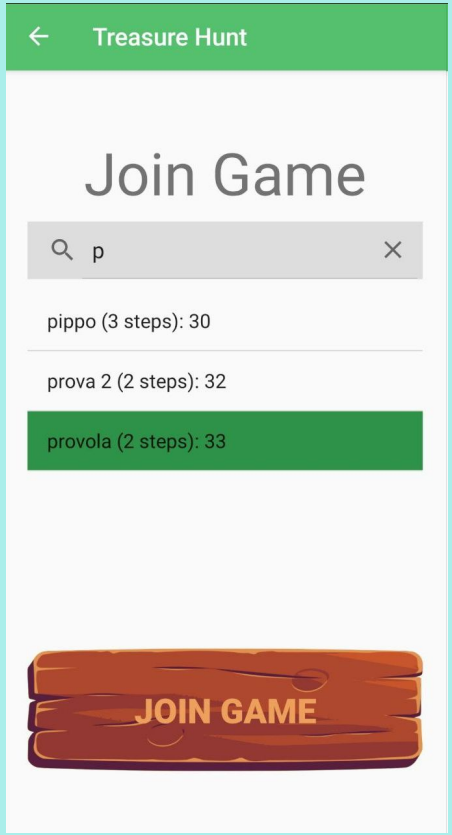


02

Game Frontend

Search Game Activity

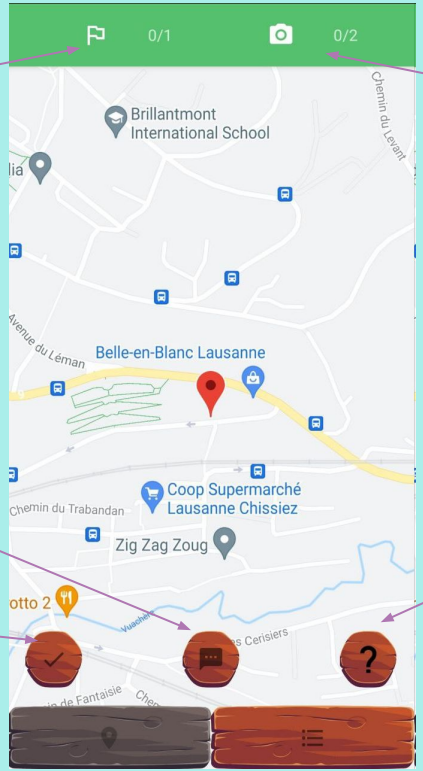
Search Game Activity



Game Activity

Position step counter

Image step counter



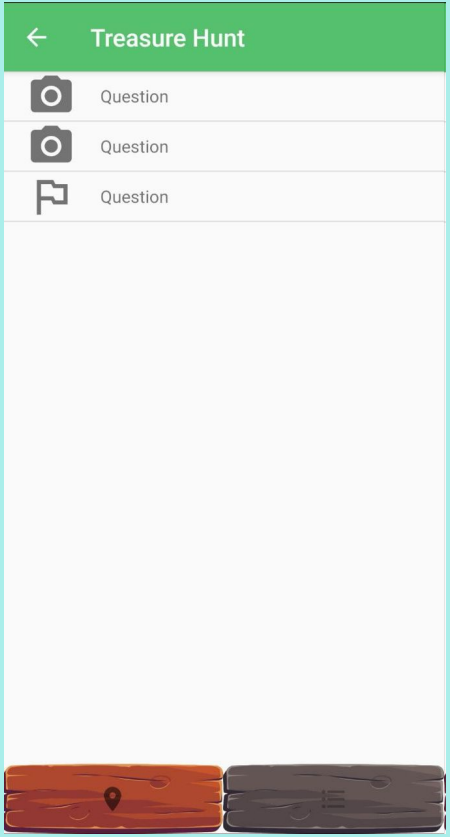
Steps question

Step hint

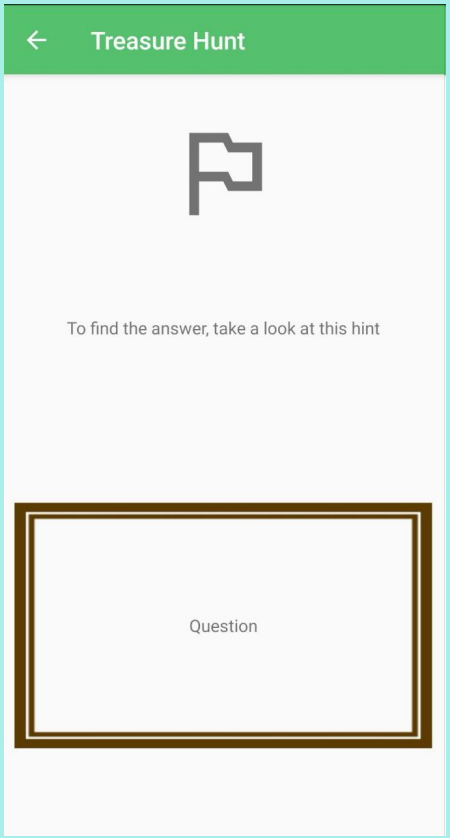
Check step

Steps list

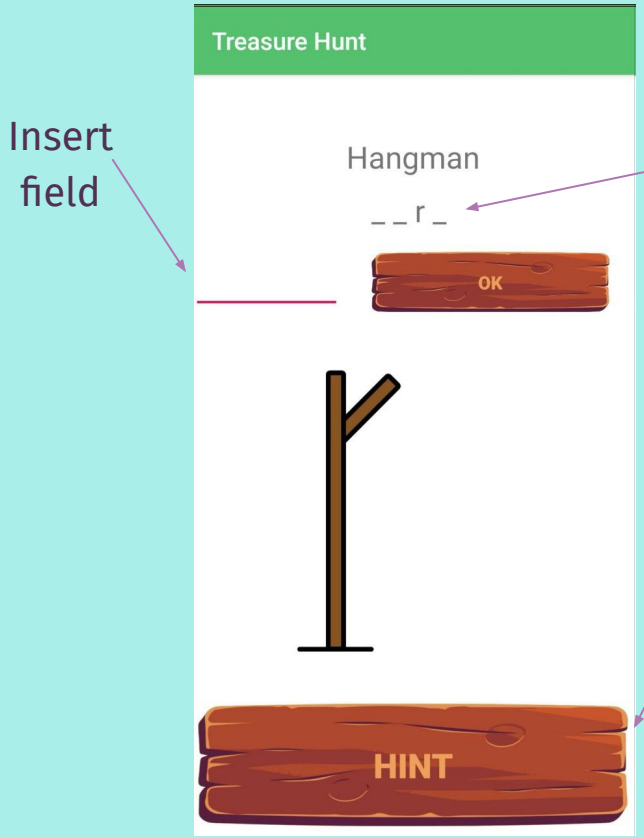
Steps list Activity



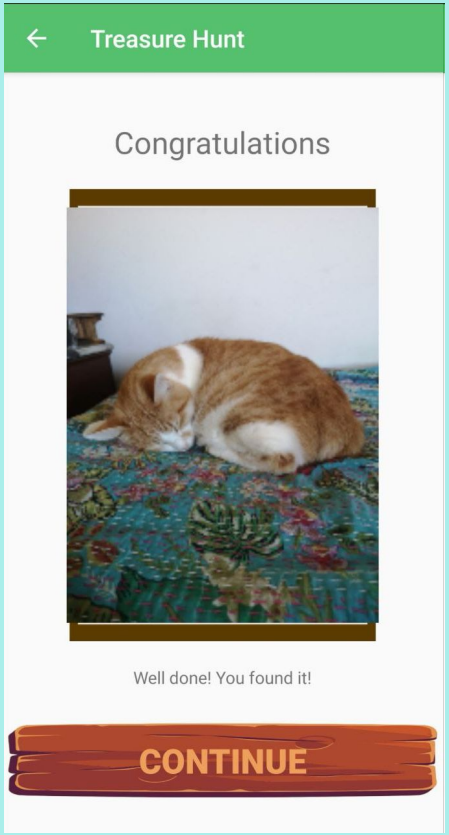
Question Activity



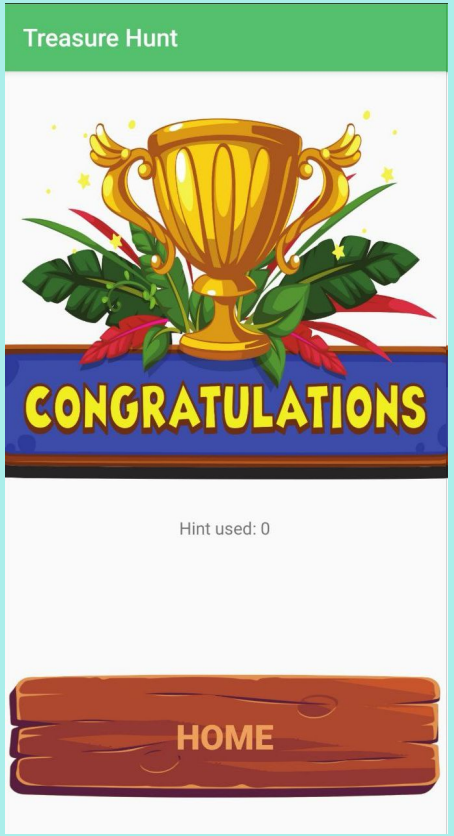
Camera Hint Activity



Camera check Activity



Success Activity

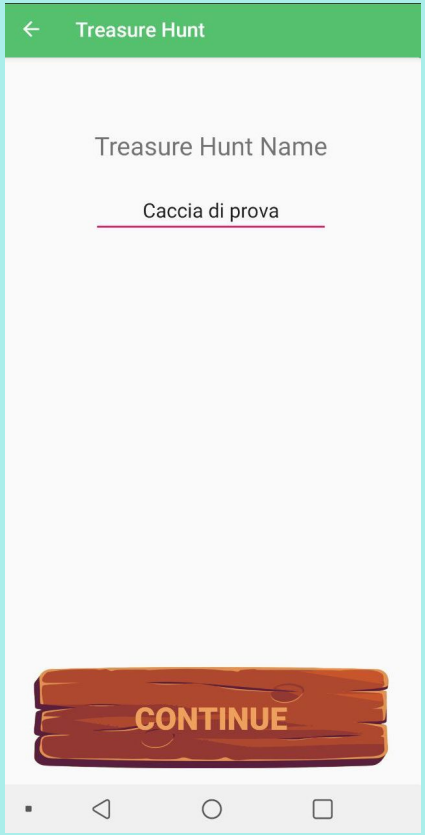


03

Make Frontend

Game Name form Activity

Everyone can also create own educational game just clicking on Make button

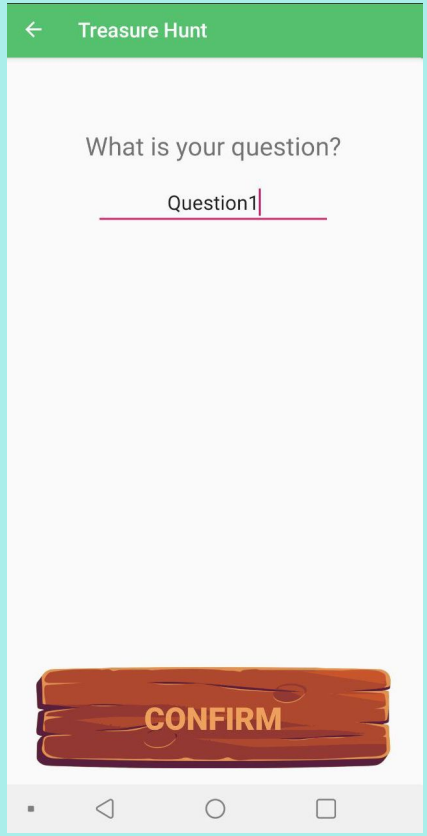


Maker Map Activity

Most important activity for the maker
Three buttons to interact with:

- *Place for place steps*
- *Camera for recognition steps*
- *Finish to conclude*

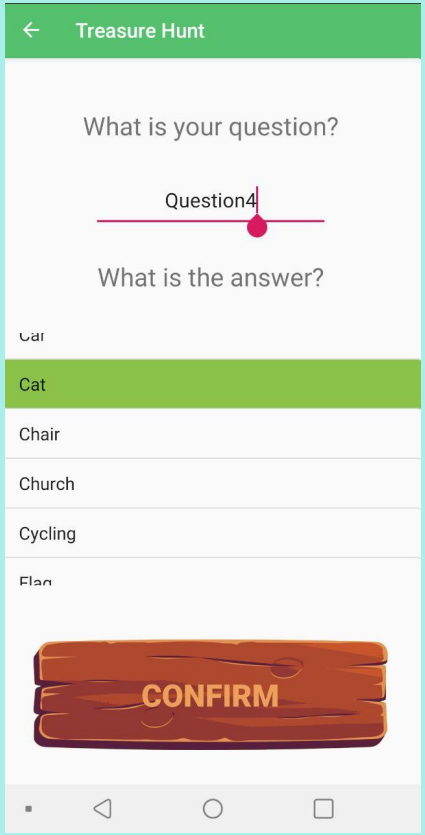
Place Activity



Camera Activity

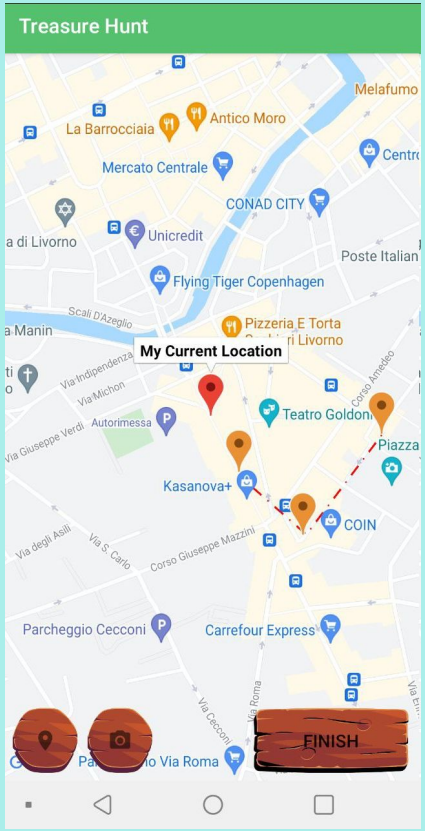
ScrollView contains some label contained in ML Kit
Constraints:

- *Not empty question*
- *No selected options*



Maker Map Activity

Constraint: At least two steps

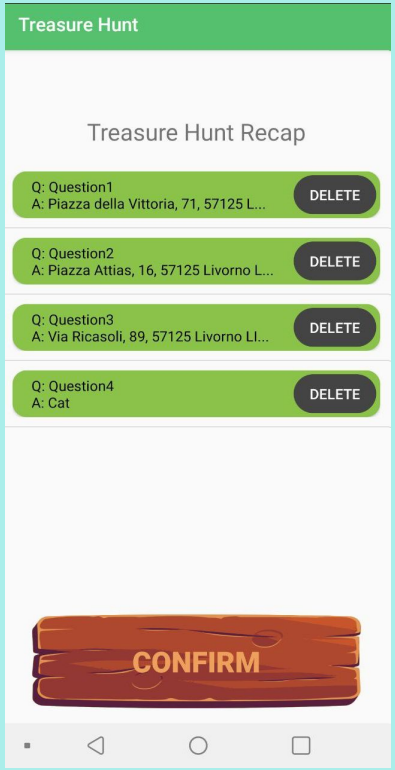


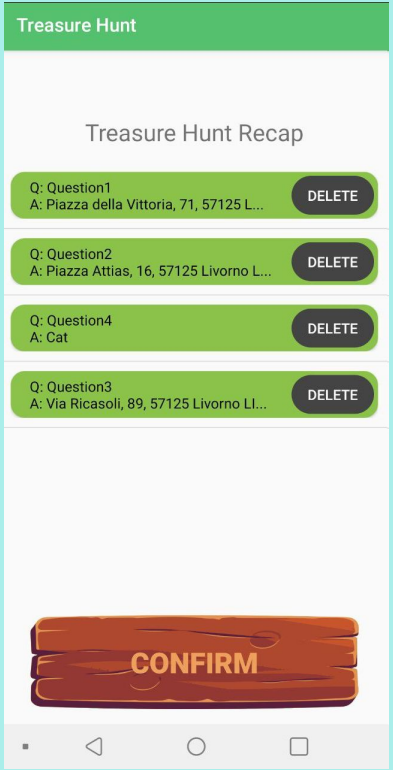
Recap Activity

Consistency handle

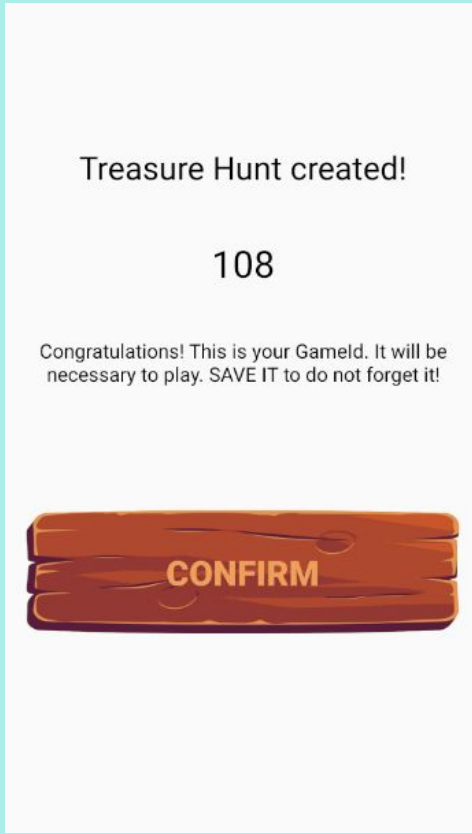
- *drag and drop*
- *delete*
- *back button*

Clicking on finish: POST request and gameId response using Volley





Success Creation Activity



Database

MySQL database contains a simple table

Each row of this table represents a step of a given Treasure Hunt

gameID	step	gameName	question	answer	stepType
109	1	Test piazza	Question	lat/lgn: (37.4219...	1
109	2	Test piazza	Question	Car	0
108	1	Test	Question	Bird	0
108	2	Test	Question	School	0

Backend

Maven project with the following dependencies:

- Spring boot: allows to realize the backend application and define the REST calls using annotations.
- Mysql-connect: allows connecting backend and database
- Json-simple: allows creating Json objects
- Hibernate: ORM service to handle data persistence in the MySql database using annotations and to maintain a relational database formed by Java classes. Allows definition of EntityManagers

REST API

- **GET /game?gameId=x**: returns every step of the game with gameId x
- **POST /game** : saves all the steps of a new game in the database. The response contains the GameId used to save it
- **DELETE /game/gameId=x** : deletes all the record of the treasure hunt with gameid x. Used in debugging.
- **GET /games?initName=x** : returns gameId, gameName and number of steps for every treasure hunt that starts with x
- **DELETE /step?gameId=x&step=y** : deletes a record of a given game and deal with the step inconsistencies. Used in debug

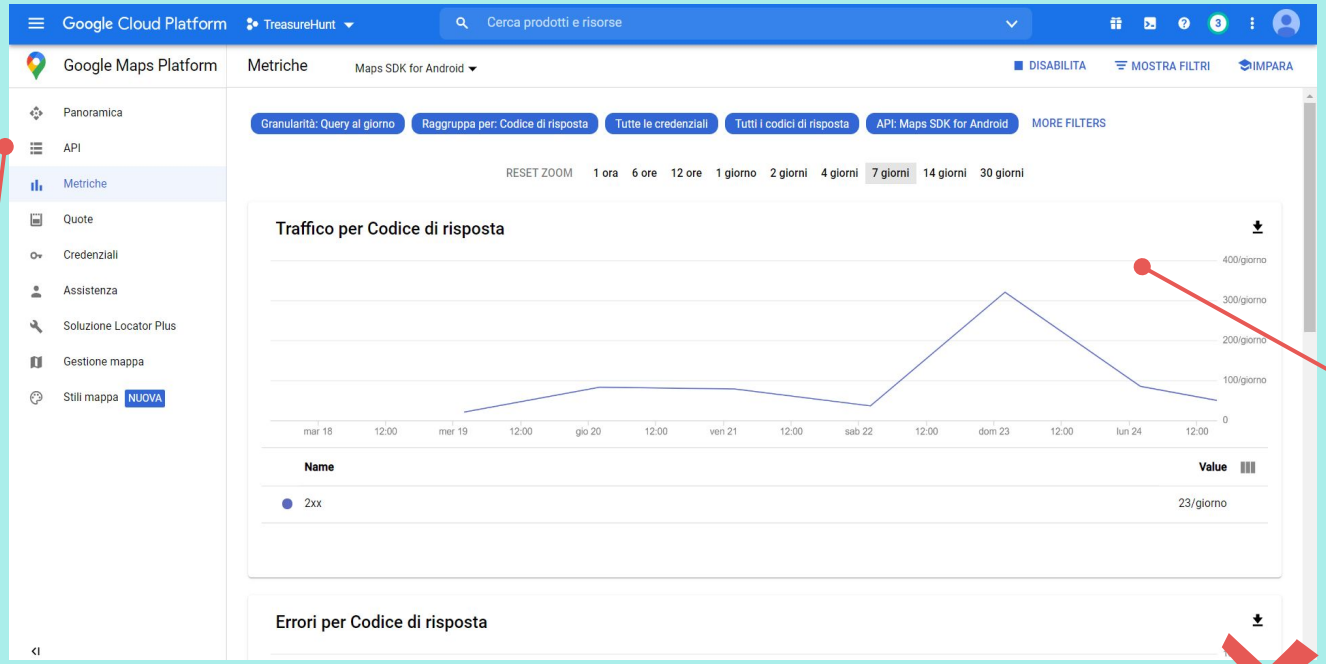
04

Location-based functionalities

Google map API integration

Map setup and init

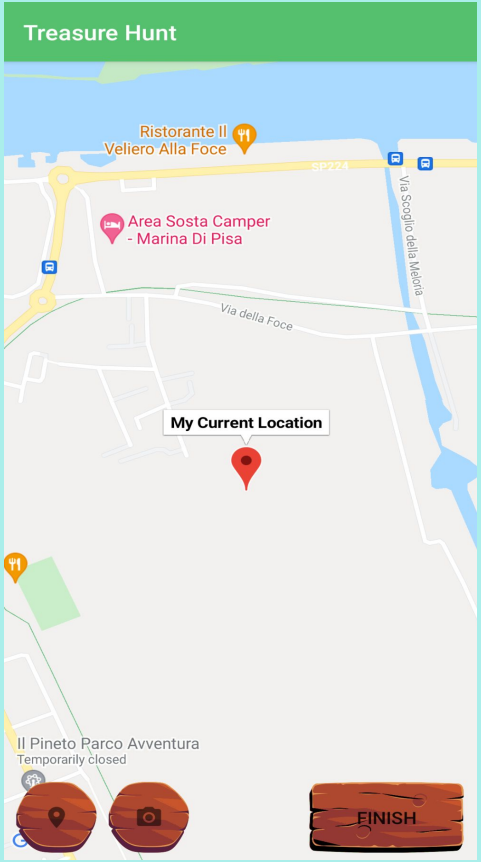
How to start using Google API



API ENABLING AND KEY ACTIVATION

ANALYTICS AROUND TRAFFIC RATE

Map Activity Initialization



Graphic recap of game steps

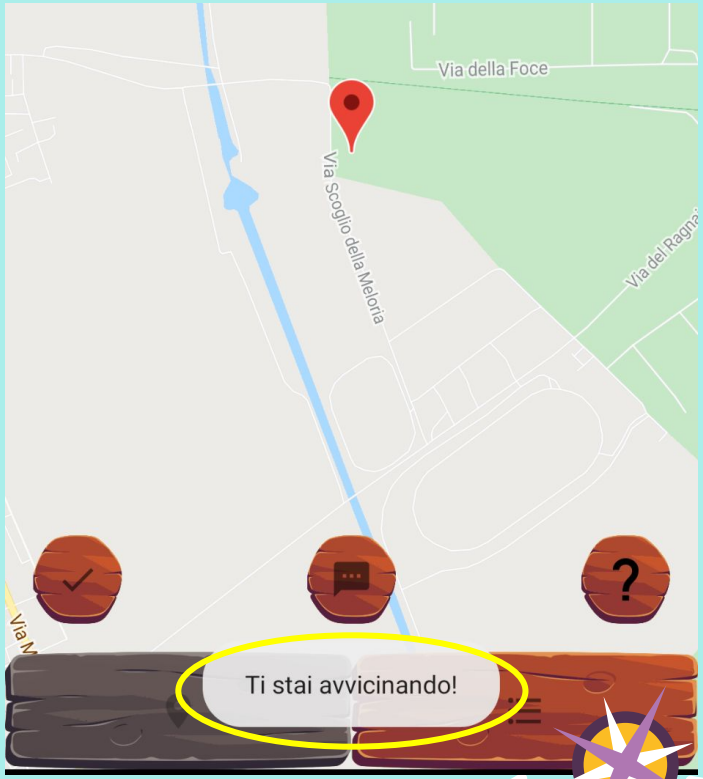
Location check

DIFFERENT LOCATION-BASED FEEDBACK ARE GIVEN:

Very close (< 100m)

On the right way (< 1 km)

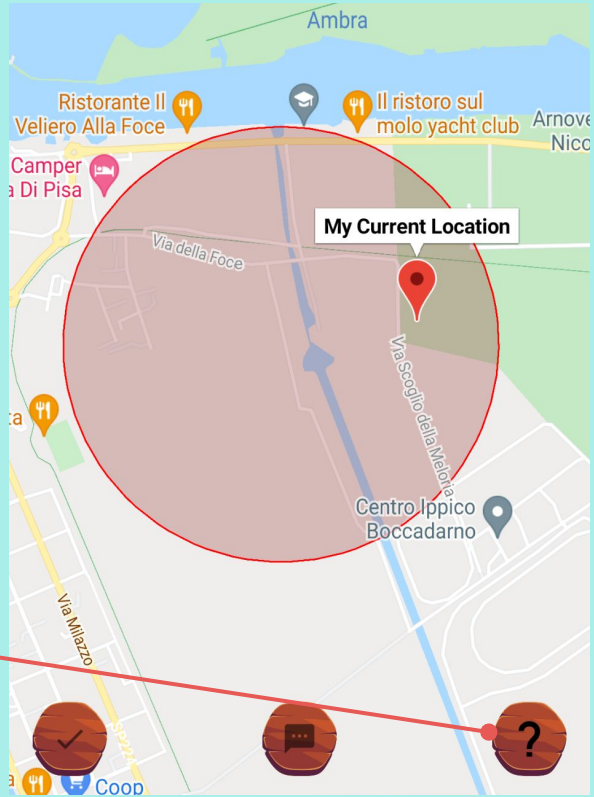
On the wrong way



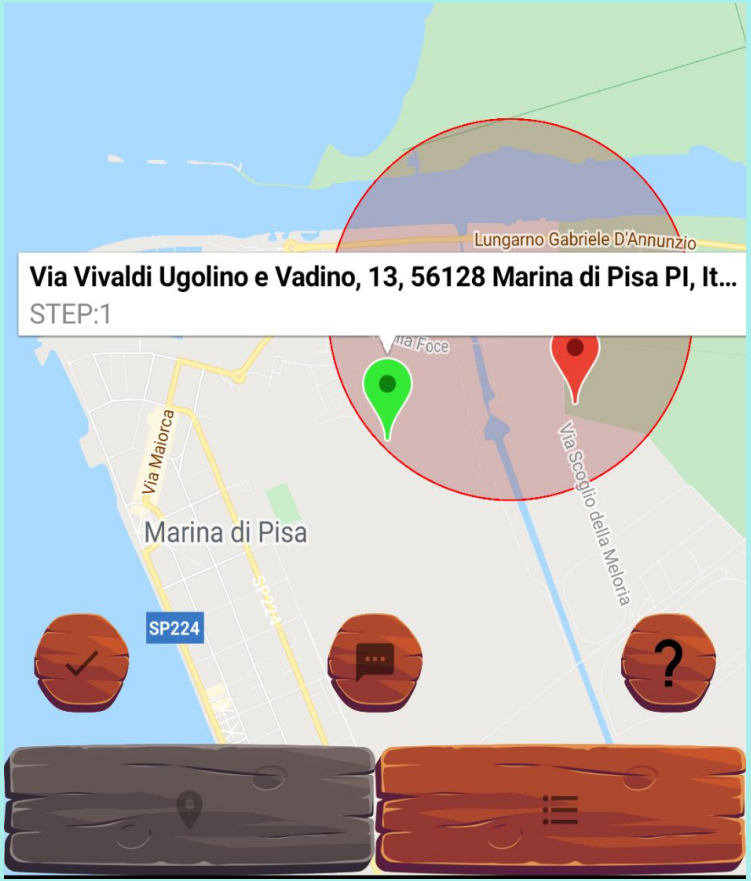
Location-based user support

USER SUPPORT FOR LOCATION
Whenever a user need support to find a location can use a hint and a red circle surrounding the target place is shown. Circle radius was set to be 500 meters long.

ON CLICK:
Area of interest is displayed



Step completion



05

Machine-Learning functionalities

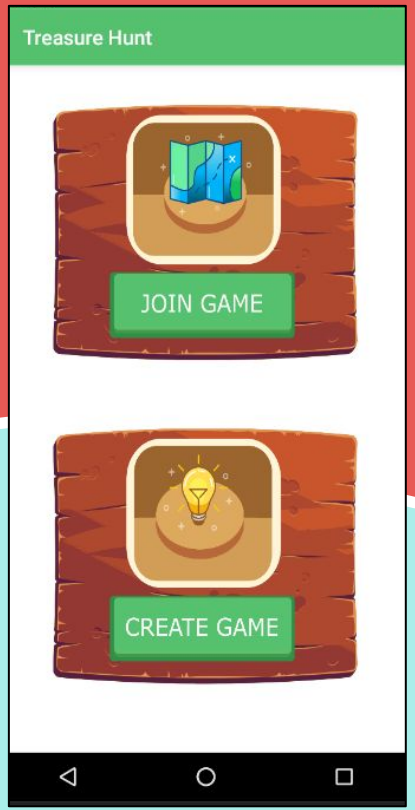
Google ML SDK for Mobile integration

Material Design

Components: interactive building blocks for creating a user interface.

Experience: User flows, error handling.

Theming: Color, shape, typography.



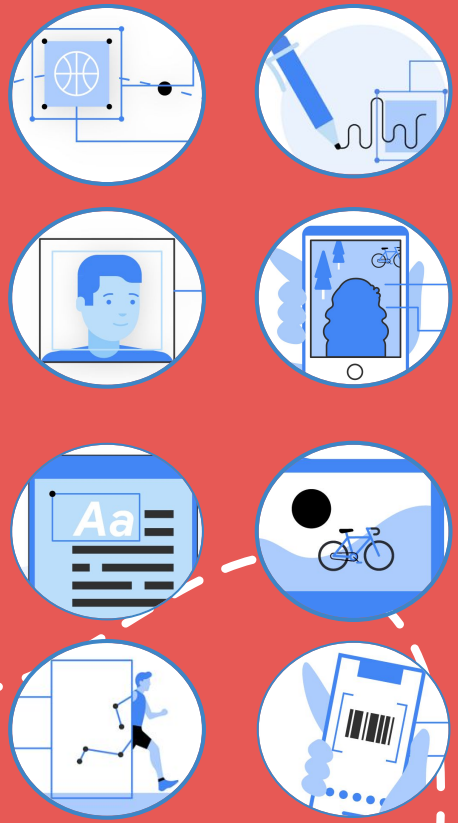
ML KIT

Tensor Flow Lite

Android

iOS

Vision



Neural Language

Custom

Machine Learning Kit

Vision APIs from google

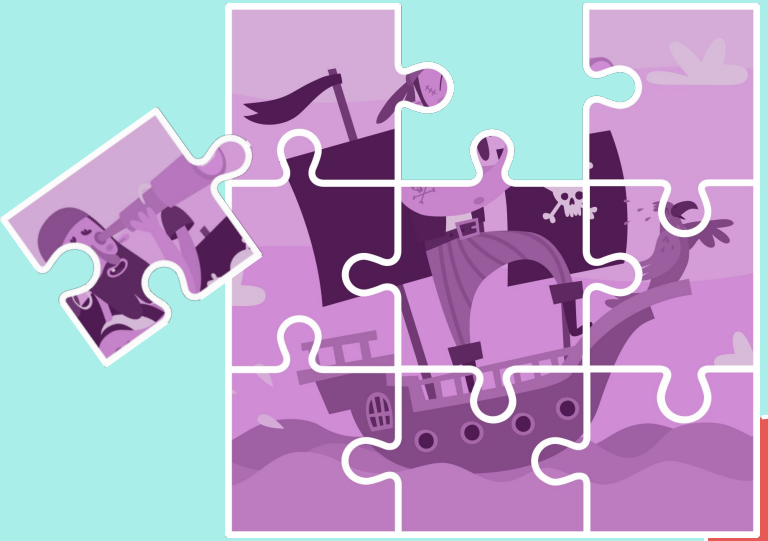
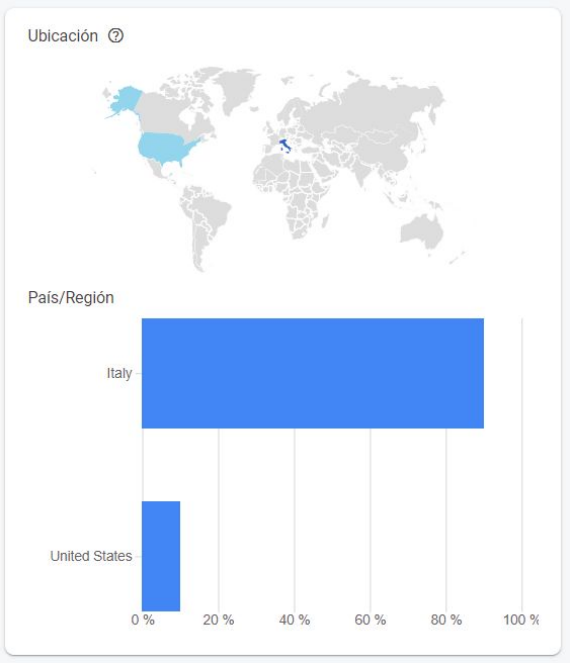


Image labeling

1. Support more 400+ labels
2. Identify people, things, places, activities, etc

Google Analytics



Arrrrrrrr!!

You are going to love the game!!
Be the best pirate!!