Technological Educational Institute of Crete

Department of Informatics Engineering

Bachelor Thesis

Design and Development of a Treasure Hunt Environment

Spyridon Bartokaymenos (AM: 4417)

Zacharias Fragkiadakis (AM: 4347)

Supervising Professor: Dr. Vidakis Nikolaos

Acknowledgments

Firstly, we would like to express our thanks and deep gratitude to our supervisor for this project, Dr. Nikolaos Vidakis for his invaluable guidance, trust, and support throughout the creation of this thesis. We would also like to thank the team at NiLE Lab for their feedback in improving and help in testing the software. Lastly, our eternal gratitude goes to our families for their love, support, and sacrifices, without which we would surely not be where we are today.

Abstract

Back in the age of piracy, treasure hunting was the practice of searching for sunken shipwrecks in hopes of salvaging gold and artifacts with market value. Nowadays, this activity has evolved past its once dangerous roots into a sport of sorts, where people use GPS coordinates to look for hidden caches of trinkets and baubles, commonly referred to as scavenger hunts.

The goal of this thesis was to take a more abstract look at the subject. To design an environment where treasure hunts are not bound to the concept of looking for a literal item. An environment where a treasure hunt's goal can be adapted to suit the needs of the one creating it.

To that end, we have created multi-platform environment that can be used to design treasure hunts consisting of a wide variety of highly customizable activities in the browser. Which can then be experienced by multiple teams of players through a mobile application. The creation system was designed with customizability in mind, allowing hunts to be used for practically any subject. While the mobile application promotes communication and team play through a variety of social features.

Σύνοψη

Στην εποχή της πειρατείας, το κυνήγι θησαυρού ήταν η διαδικασία κατά την οποία πειρατές έψαχναν για ναυάγια με σκοπό να βρουν χρυσάφι και άλλα αγαθά με αξία στην αγορά. Σήμερα, αυτή η δραστηριότητα έχει εξελιχθεί από τις επικίνδυνες της ρίζες και έχει μεταμορφωθεί σε ένα είδος αθλήματος κατά το οποίο συντεταγμένες GPS χρησιμοποιούνται για την αναζήτηση κρυμμένων κρυπτών με διάφορα μπιχλιμπίδια.

Ο σκοπός αυτής της πτυχιακής ήταν μια πιο αφηρημένη εξέταση αυτού του αντικειμένου. Ο σχεδιασμός ενός περιβάλλοντος, όπου ένα κυνήγι θησαυρού δεν έχει αναγκαία ως στόχο την αναζήτηση ενός αντικειμένου. Ένα περιβάλλον με ευέλικτους στόχους, που μπορούν να προσαρμοστούν στις ανάγκες του δημιουργού τους.

Για τον σκοπό αυτό, δημιουργήσαμε ένα περιβάλλον το οποίο μπορεί να χρησιμοποιηθεί για τον σχεδιασμό και την δημιουργία κυνηγιών, με μεγάλη ποικιλία από προσαρμόσιμες δραστηριότητες στον «browser». Τα οποία μπορούν στην συνέχεια να πραγματοποιηθούν από ομάδες παικτών μέσω μίας κινητής εφαρμογής. Το σύστημα σχεδιάστηκε έτσι, ώστε να επιτρέπει την δημιουργία και χρήση κυνηγιών για σχεδόν οποιαδήποτε περίσταση.

Table of Contents

List of Figures

List of Acronyms

UI	User Interface
UX	User Experience
GUI	Graphical User Interface
API	Application Programming Interface
JSON	JavaScript Object Notation
REST	Representational State Transfer
HTTP	Hypertext Transfer Protocol
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
HMS	Hunt Management System

1. Introduction

Odyssey is a game where users can design hunts consisting of a list of activities in which other players compete to see who will complete the most by the end of the hunt. Odyssey is usually played in a team setting where teams of players compete to see who can gather the most points, but depending on the rules set by the hunt's organizer, individual players can also participate.

1.1 Main Objective

The goal of this thesis was to develop a treasure hunt environment with accessibility, customizability, freedom of expression, and interactivity in mind. We wanted an environment where anyone could create interesting and engaging experiences without requiring vast amounts of technical knowledge. We wanted to take away the burden of implementation and let people unleash their creative freedom. At the same time, we wanted to create a social environment, where people could work together and participate in those unique experiences while having as much fun as possible.

To achieve these goals, we ended up creating two applications. One which is based on the browser and is used to design and manage the hunts. And a mobile application with which the hunts can be played. To allow for team-play and ensure security and integrity, a RESTful API provided both applications with the necessary data to function.

1.2 Catalyst

Organizing activity-oriented events with a high number of participants, is quite frequently, a colossal undertaking. One has to design engaging and fun activities, find a way to explain the rules for each activity to the participants, prepare the necessary space and materials for each activity, and figure out a way to monitor and communicate with players during the event. This thesis aims to directly address the complexity of such tasks through the creation of an activity-oriented, hunt creation and management system, and a mobile application in which these hunts can be experienced.

1.3 Similar Endeavors

There have been a few attempts to create objective-focused scavenger hunts. However, none of them offered all the features we envisioned. The following is a list of platforms that are similar to Odyssey but did not quite match our vision:

Actionbound:

"Actionbound is an app for playing digitally interactive scavenger hunts to lead the learner on a path of discovery. We call these multimedia based hunts 'Bounds'. The program quite literally augments our reality by enhancing peoples' real-life interaction whilst using their smartphones and tablets. Create your app-based DIY escape game, a digital timeline of events or a places of interest tour, with the use of GPS coordinates and pre-placed codes and mysteries"

GooseChase:

“Build your game on our website, giving it a name, picture and description to mark it as yours. Choosing a unique and memorable name makes it easier for your participants to join Once you login to our iPhone or Android apps, the first thing to do is join a game. Just search for the name of the game or enter the specific game code within the app and away you go!”

Scavify:

“Create a list of challenges, choose your settings, and get ready for lift-off. It's that easy. When you’re ready to launch your program you can publish it to the app with the click of a button. Once your program is live, people can use the app to start the adventure and complete challenges.”

1.4 Outline

Chapter 1: Introduction

This chapter consists of an introduction to the thesis, the catalyst that sparked its creation, as well as a small showcase of similar endeavors.

Chapter 2: Technology Stack

This chapter provides a summary of the programming languages, tools, and other technologies used to develop the software of the thesis.

Chapter 3: Implementation

This chapter offers insights into the thought and development processes behind the thesis’ backend and frontend software.

Chapter 4: User Interface

This chapter focuses on the web and mobile application’s user interface and user experience, providing a detailed analysis of the core design philosophies used to create them.

Chapter 5: Conclusion

The last chapter includes the results of our development efforts, as well as any goals for the future of the project.

2. Technology Stack

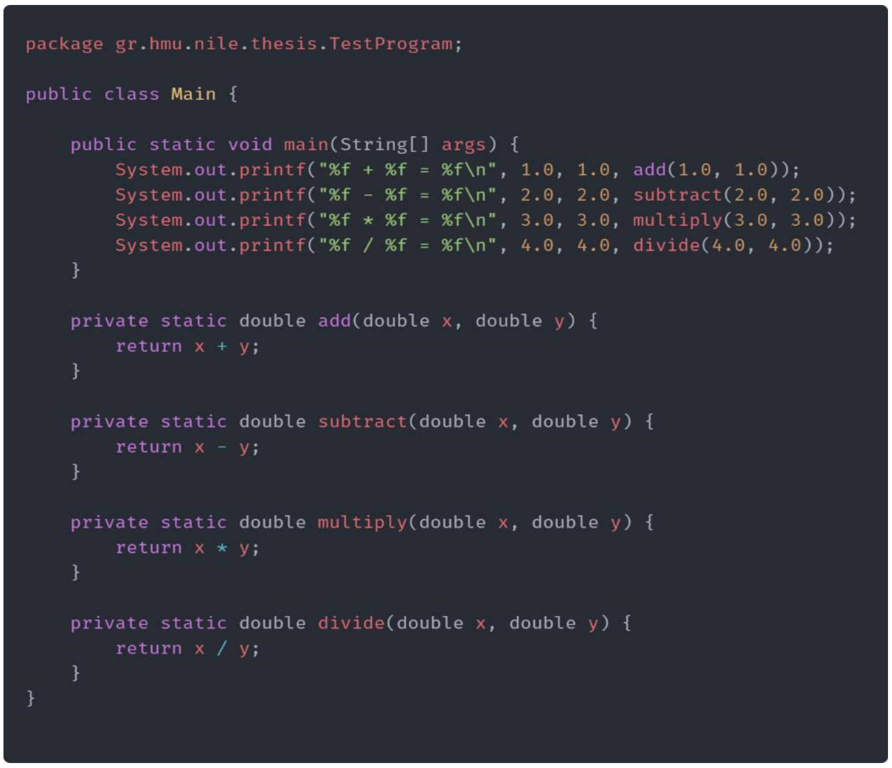
This chapter will focus on the primary technology stack used throughout the design and development of this project. More specifically it will offer insight into our choices of programming languages and software frameworks used for this project.

2.1 Programming Languages

Programming languages are in essence collections of instructions and directions used by programmers to develop software. Programmers use “high-level languages” write code that is later compiled into “low-level languages” that computer hardware can be understand [1]. This section offers a list of the languages used to design and develop this project. It explains the rationale behind why these languages were chosen.

2.1.1 Java

Java is an object-oriented, general purpose programming language which is maintained by Oracle but originally developed by Sun Microsystems in 1995. It is influenced by C++, Objective C and SmallTalk among other languages. One of its advantages is that it eliminates the need to recompile the code for different platforms, allowing programmers to write once, and run anywhere (provided that the system which will run the program supports Java). To accomplish this level of portability, the program is compiled to bytecode and executed in a Java virtual machine. [2]



```java
package gr.hmu.nile.thesis.TestProgram;

public class Main {

    public static void main(String[] args) {
        System.out.printf("%f + %f = %f\n", 1.0, 1.0, add(1.0, 1.0));
        System.out.printf("%f - %f = %f\n", 2.0, 2.0, subtract(2.0, 2.0));
        System.out.printf("%f * %f = %f\n", 3.0, 3.0, multiply(3.0, 3.0));
        System.out.printf("%f / %f = %f\n", 4.0, 4.0, divide(4.0, 4.0));
    }

    private static double add(double x, double y) {
        return x + y;
    }

    private static double subtract(double x, double y) {
        return x - y;
    }

    private static double multiply(double x, double y) {
        return x * y;
    }

    private static double divide(double x, double y) {
        return x / y;
    }
}
```

Figure 1 A Java code example

2.1.2 JavaScript

JavaScript is a high-level, just-in-time compiled, multi-paradigm programming language used mainly for web development. Released in September of 1995 JavaScript is one of the three fundamental languages comprising the World Wide Web [3]. JavaScript is used to add interactivity to a web page by enabling the display of dynamic content as well as the implementation of complex behaviors.

While originally designed to run only in browsers, its popularity over the years has caused it to be adapted to work in many diverse environments. Nowadays JavaScript is increasingly used to develop web and mobile applications, real-time networking applications such as chats, command-line tools, and even games.

One of the main drawbacks of JavaScript is the fact that it is implemented slightly differently by every browser. Meaning it is somewhat challenging to write cross-browser code. Other than that, the fact that it is a scripting language causes its performance to be lacking when performing heavy computations.

2.1.2.1 TypeScript

TypeScript is a language that builds on JavaScript by adding static type definitions. TypeScript simplifies JavaScript code, making it easier to understand and debug. This in turn makes working with it more predictable, reducing errors and speeding up development time, especially in larger projects such as this one.

Considering the scale and complexity of this project, TypeScript was chosen as the main language for the web application as well as the mobile application.

2.1.2.2 JSON

JSON (JavaScript Object Notation) is a data-interchange format which is as its name implies based on JavaScript’s object notation, which it uses to store and transmit data objects which consist of key-value pairs and arrays [4]. It is the most common data interchange format on the web due to it being easier to read and shorter than other data interchange formats like XML, plus it is parsed very efficiently by JavaScript. Since this project makes heavy use of JavaScript and for all the factors mentioned previously JSON was an easy choice for this project.

2.1.3 HTML

Hypertext Markup Language (HTML) is the standardized markup language used by documents displayed on the World Wide Web. It forms the structural foundation of a web page and it is common among all web browsers. Originally released in 1993 HTML has seen many revisions over the years with the most recent being HTML 5 which was published in 2014 as a W3C Recommendation [5].

2.1.4 CSS

Cascading Style Sheets (CSS) is a style sheet language that describes the appearance of an HTML document. CSS provides the visual style of a web page and is essential for a better

User Experience. It allows for the separation of presentation and content such as fonts, colors, and layout. This separation is key in reducing complexity and repetition in the HTML file.

2.2 Software Frameworks

A software framework is an abstraction in which a programming language or software providing generic functionality can be changed to provide application-specific software. It is in essence a software environment based on a larger software platform (such as a programming language) to provide particular functionality and streamline the development of software, products, and solutions. Examples of software frameworks include code libraries, APIs, compilers, toolsets, and support programs all of which consolidate the different components needed to enable the development of a system or project.

The frameworks mentioned below are the ones we utilized for the development of this project. Each of which was chosen for specific reasons mentioned in their respective sections.

2.2.1 Spring

Spring is an open-source Java framework that was created by Rod Johnson in 2002. The Spring framework can be useful when developing any Java program, but it is popular for creating web applications. It includes modules for providing different kinds of services, such as inversion of control, data access, aspect-oriented programming, testing, and many more. [6]

2.2.2 ReactJS

ReactJS is a robust, front end, JavaScript library used for building graphical practical, and efficient user interfaces and/or user interface components. Developed by Facebook and released in May of 2013 [7], ReactJS has been actively supported and has been further developed over the years, with the latest stable version being 17.0.1 [8] as of the time of writing. ReactJS is one of the most popular JavaScript frameworks/libraries alongside AngularJS and VUE.JS.

A ReactJS application is in essence a collection of components composed in a specific way. A ReactJS component is a piece of the user interface such as a navigation bar, a button, etc. The reason ReactJS makes use of components is that user interface applications, in general, lend themselves to being split up into several independent, isolated, and reusable components, which can then be composed to build complex user interfaces. By splitting an application in such a way, we can not only save space by reusing code, which also reduces page loading times in the case of a website, but by also making the code much easier to read since it is divided into smaller, digestible chunks.

The main reason ReactJS was chosen for this project over other JavaScript frameworks, is in large part due to it being the most popular. This in turn means it is more likely to have wider support in the form of reading material, guides, solutions for problems, etc. Apart from it being the most popular, ReactJS is also backed by one of the biggest tech companies in the world which means it is very likely that it will keep being supported for the foreseeable future, making it a great skill to add to one's skillset. Last but certainly not least, unlike other JavaScript frameworks, ReactJS can be used to develop mobile applications through React Native (see below).

2.2.2.1 Material-UI

Material UI is a UI framework that provides React components that follow Google’s Material Design guidelines and patterns [9]. The style chosen for both the web and mobile applications was Google’s Material Design. Material Design’s widespread usage in the current application landscape assured us that it would be a good fit for our project since people would already be familiar with it making the user experience intuitive. To that end Material-UI provided us with reliable, tried, and tested tools to implement Material Design in our web application.

2.2.3 React Native

React Native is a mobile application framework used to develop applications for various mobile operating systems such as Android and iOS. It was developed by Facebook and originally released in March of 2015 [10], Facebook’s primary goal with React Native was to bridge the gap between developing web applications and mobile applications. It achieves this goal by allowing developers to use ReactJS’s core features alongside native platform capabilities.

There are quite a few mobile frameworks that allow the developer to create cross-platform mobile applications using web technology stacks. The difference between them and React Native is that React Native as its name implies uses agnostic native components that map directly to the platform’s native UI building blocks. This ensures that apps built with React Native can be as fast as apps built with the platform’s recommended technology stack without compromising on the user’s experiences.

We chose React Native for two reasons. Firstly, because of the efficiency, it would provide this project. Since we had already chosen ReactJS, using React Native would mean that large chunks of the code from the web application could be reused for the mobile application and vice versa, cutting down development time significantly. The other reason being cross-platform development. Being able to create and maintain one codebase for Android and iOS means we could spend more time on designing, testing, and improving the application rather than learning one to two extra programming languages. Apart from saving a significant amount of time, having a single codebase also cuts down on the complexity of the project, making it easier to maintain, modify, and expand.

2.2.3.1 Expo

Expo is an open-source platform for making universal native apps with React. Expo provides several useful functions for working with React Native, which help streamline the development process, allowing for rapid prototyping, as well as fast and reliable distribution and updating of the mobile application.

3. Implementation

This chapter will focus on the thought process and design decisions behind the implementation of the backend and frontend side of the development of this system. Section 3.1 describes how a hunt is meant to be designed and played. While sections 3.2 and 3.3 describe the implementation of the backend and frontend parts of the system respectively.

3.1 Design: The Application’s Use Case Scenario

This section will describe the basic flow of using the system. It will outline how to use the web application to create, manage, and end a hunt. As well as how to use the mobile application to find, join, and finally play hunts.

3.1.1 Hunt Design Flow

To design a hunt, the user first needs to log in to the web application. Afterward, they can navigate to their hunts list page, from where they click on the “New Hunt” button, prompting them to fill a form with the hunt’s information. After filling and submitting the form, the user is transported to the hunt management system page. From there they edit the hunt’s information, upload the hunt’s featured image, or navigate to the activity management page. From there they can create activity groups and activities using the drag and drop interface, assigning their points and completion order among other things. After designing the activity groups and activities, the user may then proceed to the member management page where they may optionally create teams for players to join.

Upon finishing designing the hunt, the user may then choose to publish it, so that other users can start joining in. In this stage, the user may once again visit the member management page whereupon he can organize the users into teams, or kick members from the hunt if they so desire. The user can also proceed to the announcements page and start broadcasting messages to all members of the hunt. Once the user deems it is time for the hunt to start, they can press the “Start Hunt” button in the hunt management page, or the mobile application. Throughout the hunt, the user retains the option of managing the members and making announcements. Finally, the user can at any time by pressing the “End Hunt” button in the hunt management screen, which officially ends the hunt.

3.1.2 Hunt Play Flow

The first step in joining and later playing a hunt is logging in to the mobile application. Upon successfully logging in the user is presented with a feed of hunts they have joined. To find hunts to join the user needs to navigate to the “Discover” section of the feed, from which all public hunts are available to choose. Upon selecting a hunt, the user can tap on it to proceed to its screen. From there the user can navigate to the Actions tab and select the “Join Hunt” options to join the hunt. After joining the hunt, the user will be prompted to select or create a team to join.

After the hunt has started the user can navigate to the actions tab of the hunt, and select the “Play Hunt” option. This will transport them to the game screen. From there they can start completing activities with their fellow team members, communicating through the chat, reading hunt announcements, and viewing the progress of each team through the leaderboard. Once the hunt has finished, the user can see how well each team did through the leaderboard.

3.2 Backend Development

The backend is essentially the data access layer of this system. This section will focus on describing how the backend of Odyssey was developed.

3.2.1 Database

The database is an important component of the system, which is used to store all data that should be persisted (User information, hunt, gameplay related information, etc.). In Odyssey we are using MySQL, an open-source relational database management system for our database. In the following image you can see the entity – relationship diagram that describes the structure of our database.

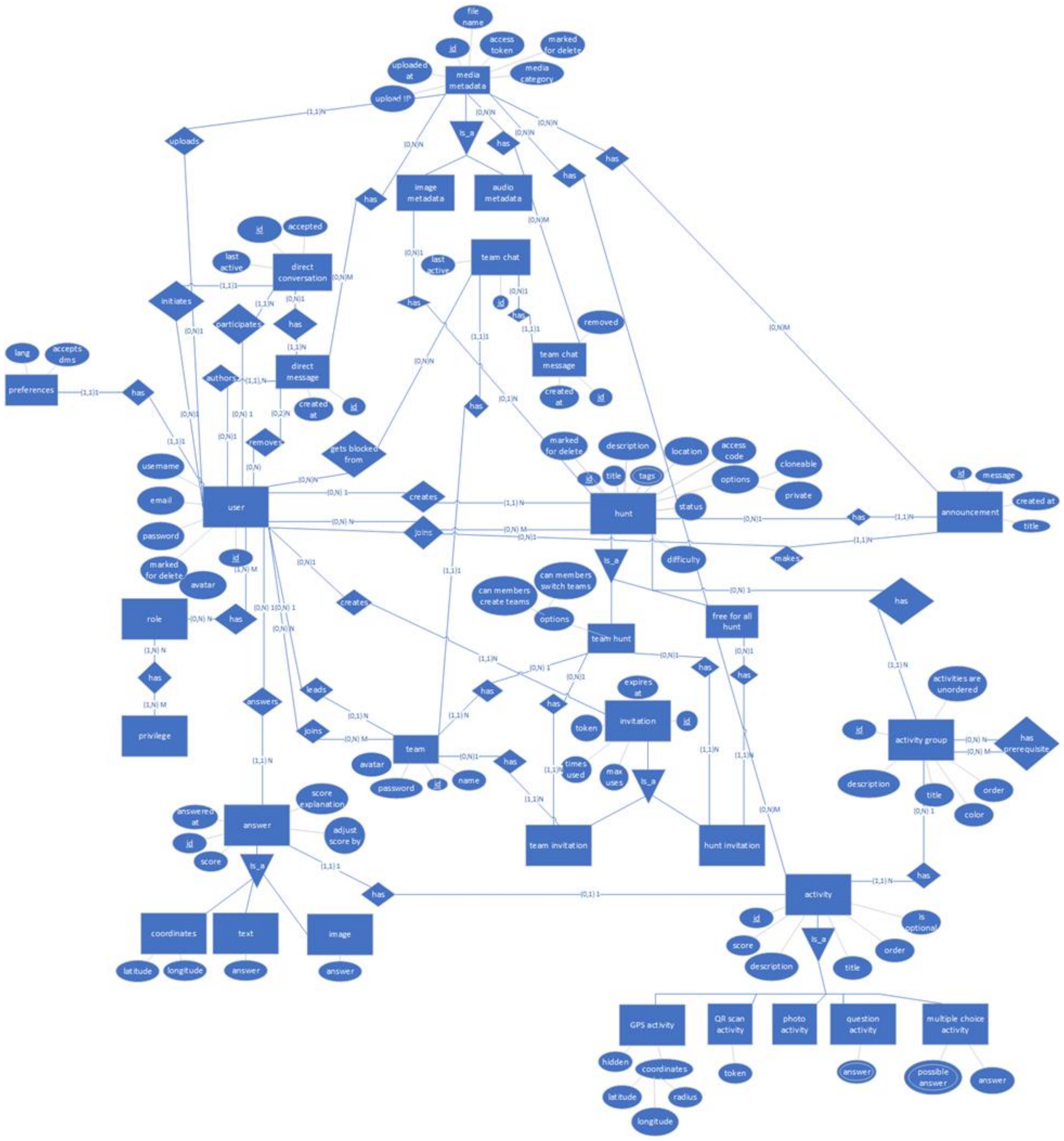


Figure 2 The entity - relationship diagram of Odyssey

3.2.2 REST API

3.2.2.1 API

An Application Programming Interface (API) enables different software applications to communicate with each other. It defines the rules for the communication, for example which requests can be made and how. An API can also be used to extend the functionality of a system. [11] In “Odyssey”, we have created a REST API (we will discuss what a REST API is shortly) that is used by both the website and the mobile application and allows them to communicate with the server. As an example, when the mobile application wants to display a list of all available hunts, it connects to the internet and uses the API to send data to the server, which in turn processes the request, compiles the list of available hunts, and sends it back to the app.

3.2.2.2 HTTP

The Hypertext Transfer Protocol, HTTP, is an application layer protocol which was introduced in 1989 by Tim Berners-Lee to allow the retrieval resources, such as HTML documents. It is the foundation for any data exchange in the World Wide Web. HTTP is a client – server protocol, meaning that the client (for example a web browser or a mobile application) initiates the communication by sending an HTTP request to the server, which in turn processes the request and sends a response message containing information about the completion status of the request and possibly any additional content requested back to the client. [12]

3.2.2.2.1 HTTP Request Methods

Every action that can be performed on a resource has a corresponding method, defined by the HTTP protocol. [12] The most common are the following:

- GET
  Using this method, a client can request a representation of the requested resource. GET requests should only retrieve data and should not make any changes to the state of the server.
- HEAD
  HEAD is similar to the GET method, but it only includes the headers and not the response body
- POST
  Using this method, a client can create or update a resource
- PUT
  Put replaces a resource or creates a new one if it does not exist. The difference between PUT and POST is that PUT is idempotent
- DELETE
  This method is used to delete a resource
- PATCH
  This method can be used to partially modify an existing resource
- OPTIONS
  Using this method, a client can be informed about the supported methods for the specified URL

3.2.2.3 REST

The Representational State Transfer, REST, is a collection of software architectural constraints that an API for web services should follow in order to be considered RESTful. These constraints were specified by Dr Roy Fielding for his doctorate dissertation with the goal of improving some aspects of the systems that follow the defined constraints, such as reliability, scalability, and performance. A REST API commonly uses the HTTP protocol, but due to its flexibility can take advantage of many existing protocols. [13]

3.2.2.3.1 Architectural Constraints

- Client – Server
  The system must follow the client – server model in order to achieve separation of concerns
- Stateless
  This means that application state must be stored on the client, and the client is responsible for sending the state to the server when necessary
- Cacheability
  In order to improve scalability and performance, storage of cacheable data should be encouraged
- Layered System
  Every different type of server that comprises the system should be organized in a hierarchical manner. These intermediary servers should be invisible to the client
- Uniform Interface
  The API should provide a standardized way for the communication between the client and the server
- Code on Demand (optional)
  This is an optional constraint that, if implemented, enables the server to send executable code (such as JavaScript code) to the client in order to extend its functionality

3.2.2.3.2 Common HTTP methods

The following HTTP methods are the most used methods for REST APIs:

- POST
  With POST a client can request the creation of a resource
- GET
  With GET a client can request a representation for a resource
- PATCH / PUT
  With these methods a client can update an existing resource either partially or completely
- DELETE
  With DELETE a client can request the deletion of an entity

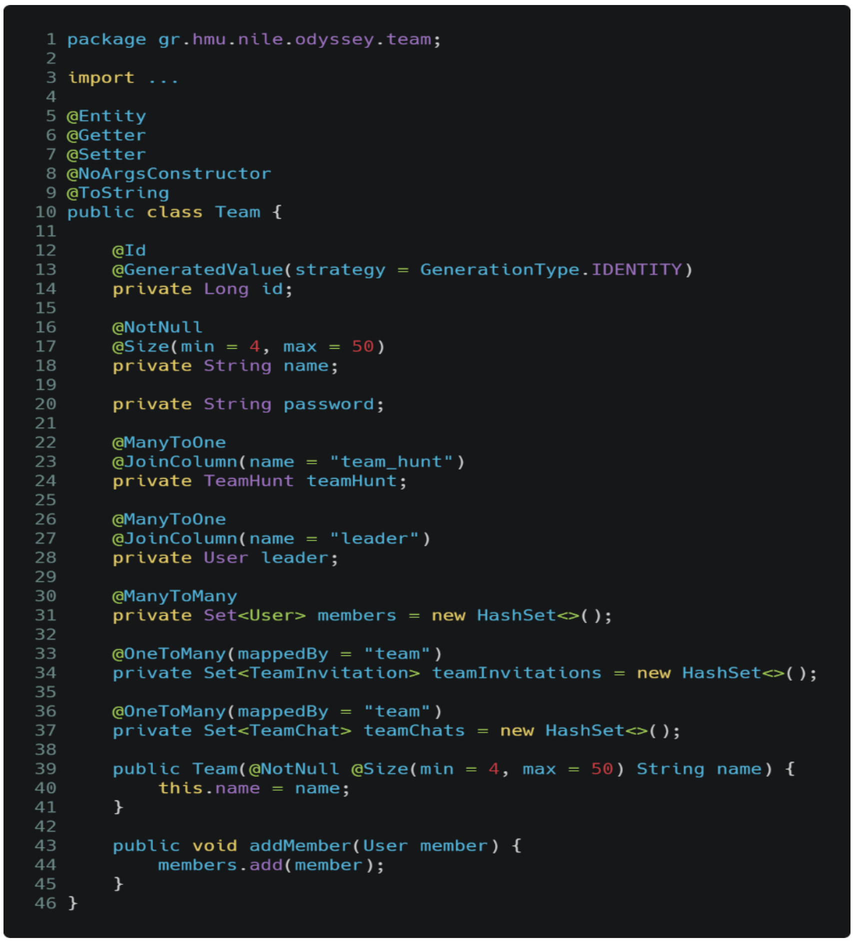
3.2.2.4 Creating REST APIs using the Spring framework

As discussed in chapter 2, for the implementation of our REST API we used the Spring Framework. We will now take a look on how Spring can be used to create REST APIs.

3.2.2.4.1 MODELS

A model contains information about an entity, such as a user, a hunt, or a team. Models simply hold the data that describe an entity, and do not contain any business logic. Models can serve several purposes. Some of the most common are the following: they can be used by controllers by serializing them and sending them to a client as the response body for a specific request, they can represent the request body of a request, or if you are using an ORM they can be used to describe the database table for a specific entity.

To create a Model, all you must do is create a class and add a field for every property of your entity. To better describe a model, you can add annotations that can be used by different parts of your program to execute specific functionality (for example validation).

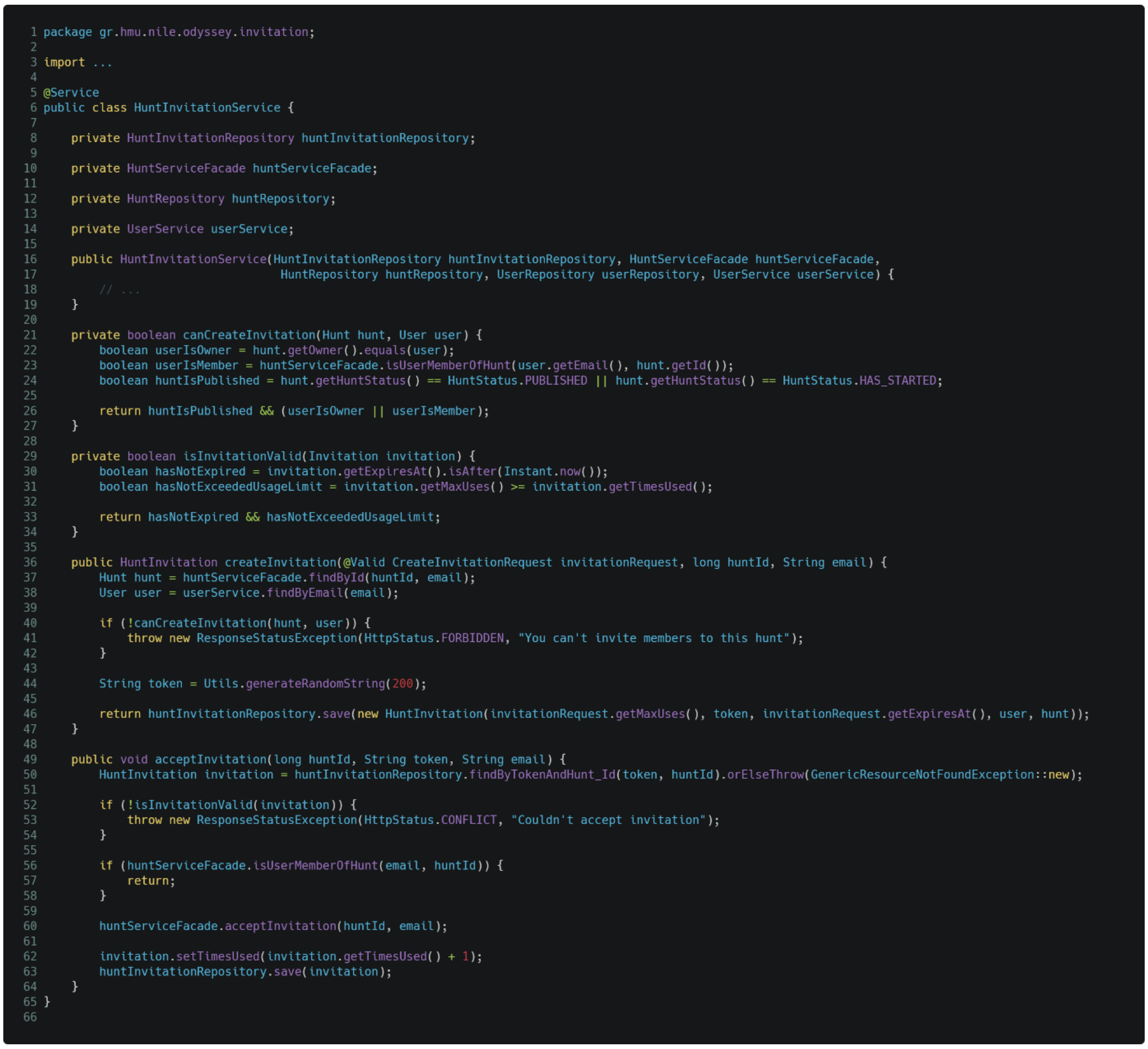


```java
package gr.hmu.nile.odyssey.team;

import ...

@Entity
@Getter
@Setter
@NoArgsConstructor
@ToString
public class Team {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @NotNull
    @Size(min = 4, max = 50)
    private String name;

    private String password;

    @ManyToOne
    @JoinColumn(name = "team_hunt")
    private TeamHunt teamHunt;

    @ManyToOne
    @JoinColumn(name = "leader")
    private User leader;

    @ManyToMany
    private Set<User> members = new HashSet<>();

    @OneToMany(mappedBy = "team")
    private Set<TeamInvitation> teamInvitations = new HashSet<>();

    @OneToMany(mappedBy = "team")
    private Set<TeamChat> teamChats = new HashSet<>();

    public Team(@NotNull @Size(min = 4, max = 50) String name) {
        this.name = name;
    }

    public void addMember(User member) {
        members.add(member);
    }
}
```

Figure 3 An example model class

3.2.2.4.2 Services

A service contains all the business logic for a specific part of the system. For example, a Service for hunt invitations could contain methods for creating a new invitation for a hunt, accepting an invitation, or checking if an invitation is valid. That service could then be used by, for example, an invitation controller to handle invitation related user requests.

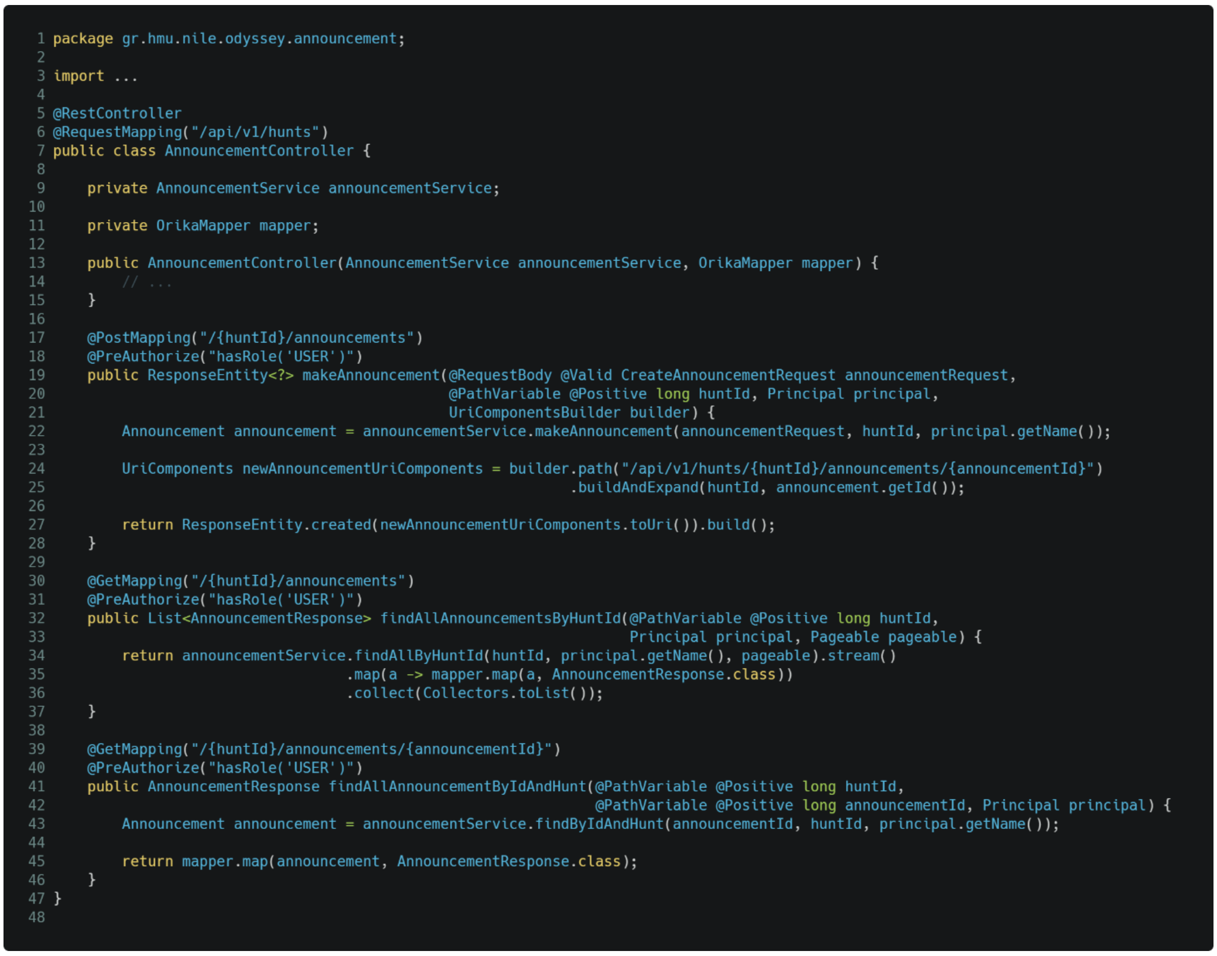


```java
package gr.hmu.nile.odyssey.invitation;

import ...

@Service
public class HuntInvitationService {

    private HuntInvitationRepository huntInvitationRepository;

    private HuntServiceFacade huntServiceFacade;

    private HuntRepository huntRepository;

    private UserService userService;

    public HuntInvitationService(HuntInvitationRepository huntInvitationRepository, HuntServiceFacade huntServiceFacade,
                                 HuntRepository huntRepository, UserRepository userRepository, UserService userService) {
        // ...
    }

    private boolean canCreateInvitation(Hunt hunt, User user) {
        boolean userIsOwner = hunt.getOwner().equals(user);
        boolean userIsMember = huntServiceFacade.isUserMemberOfHunt(user.getEmail(), hunt.getId());
        boolean huntIsPublished = hunt.getHuntStatus() == HuntStatus.PUBLISHED || hunt.getHuntStatus() == HuntStatus.HAS_STARTED;

        return huntIsPublished && (userIsOwner || userIsMember);
    }

    private boolean isInvitationValid(Invitation invitation) {
        boolean hasNotExpired = invitation.getExpiresAt().isAfter(Instant.now());
        boolean hasNotExceededUsageLimit = invitation.getMaxUses() >= invitation.getTimesUsed();

        return hasNotExpired && hasNotExceededUsageLimit;
    }

    public HuntInvitation createInvitation(@Valid CreateInvitationRequest invitationRequest, long huntId, String email) {
        Hunt hunt = huntServiceFacade.findById(huntId, email);
        User user = userService.findByEmail(email);

        if (!canCreateInvitation(hunt, user)) {
            throw new ResponseStatusException(HttpStatus.FORBIDDEN, "You can't invite members to this hunt");
        }

        String token = Utils.generateRandomString(200);

        return huntInvitationRepository.save(new HuntInvitation(invitationRequest.getMaxUses(), token, invitationRequest.getExpiresAt(), user, hunt));
    }

    public void acceptInvitation(long huntId, String token, String email) {
        HuntInvitation invitation = huntInvitationRepository.findByTokenAndHunt_Id(token, huntId).orElseThrow(GenericResourceNotFoundException::new);

        if (!isInvitationValid(invitation)) {
            throw new ResponseStatusException(HttpStatus.CONFLICT, "Couldn't accept invitation");
        }

        if (huntServiceFacade.isUserMemberOfHunt(email, huntId)) {
            return;
        }

        huntServiceFacade.acceptInvitation(huntId, email);

        invitation.setTimesUsed(invitation.getTimesUsed() + 1);
        huntInvitationRepository.save(invitation);
    }
}

```

Figure 4 An example service class

3.2.2.4.3 Controllers

Controllers are used to define the endpoints of your API. When a client makes a request to a specific endpoint, the appropriate controller handles the request and sends a response back to the client. For instance, in Odyssey when a user wants to view all announcements for a hunt, the mobile app makes a GET request to /api/v1/hunts/:huntId/announcements. Then, the announcements controller receives the request, compiles the list of announcements using the announcements service, and sends it to the app which then displays the list of announcements to the user.



```java
package gr.hmu.nile.odyssey.announcement;

import ...

@RestController
@RequestMapping("/api/v1/hunts")
public class AnnouncementController {

    private AnnouncementService announcementService;

    private OrikaMapper mapper;

    public AnnouncementController(AnnouncementService announcementService, OrikaMapper mapper) {
        // ...
    }

    @PostMapping("/{huntId}/announcements")
    @PreAuthorize("hasRole('USER')")
    public ResponseEntity<?> makeAnnouncement(@RequestBody @Valid CreateAnnouncementRequest announcementRequest,
                                              @PathVariable @Positive long huntId, Principal principal,
                                              UriComponentsBuilder builder) {
        Announcement announcement = announcementService.makeAnnouncement(announcementRequest, huntId, principal.getName());

        UriComponents newAnnouncementUriComponents = builder.path("/api/v1/hunts/{huntId}/announcements/{announcementId}")
                                                             .buildAndExpand(huntId, announcement.getId());

        return ResponseEntity.created(newAnnouncementUriComponents.toUri()).build();
    }

    @GetMapping("/{huntId}/announcements")
    @PreAuthorize("hasRole('USER')")
    public List<AnnouncementResponse> findAllAnnouncementsByHuntId(@PathVariable @Positive long huntId,
                                                                   Principal principal, Pageable pageable) {
        return announcementService.findAllByHuntId(huntId, principal.getName(), pageable).stream()
                                  .map(a -> mapper.map(a, AnnouncementResponse.class))
                                  .collect(Collectors.toList());
    }

    @GetMapping("/{huntId}/announcements/{announcementId}")
    @PreAuthorize("hasRole('USER')")
    public AnnouncementResponse findAllAnnouncementByIdAndHunt(@PathVariable @Positive long huntId,
                                                               @PathVariable @Positive long announcementId, Principal principal) {
        Announcement announcement = announcementService.findByIdAndHunt(announcementId, huntId, principal.getName());

        return mapper.map(announcement, AnnouncementResponse.class);
    }
}
```

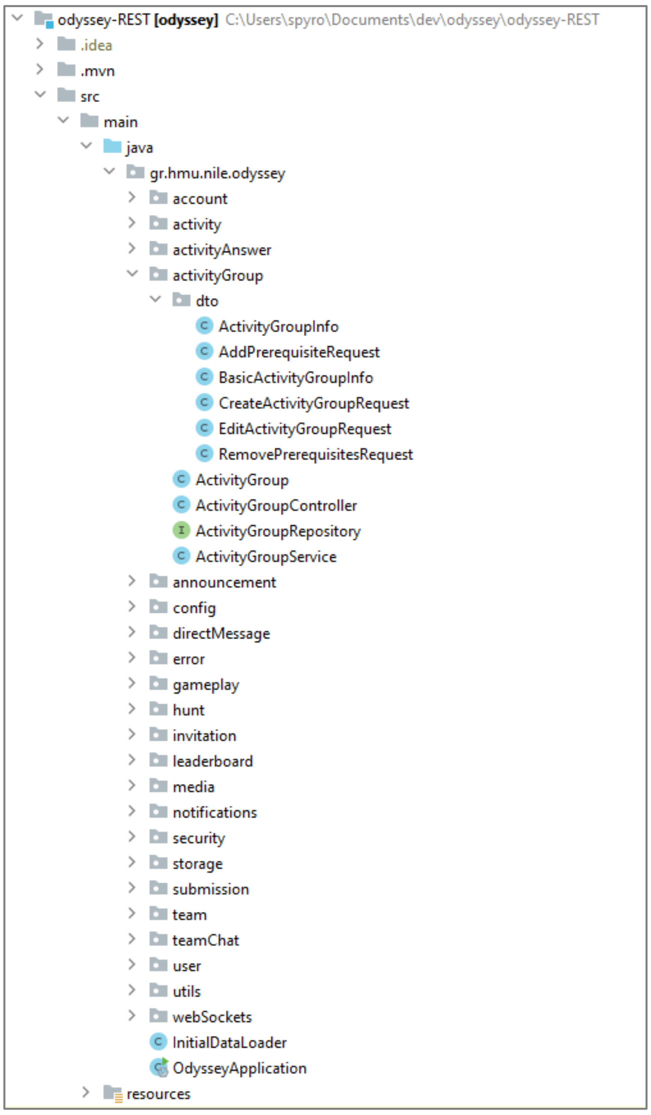
Figure 5 An example controller class

3.2.2.4.4 Real – Time Communication

REST APIs allow clients to send a request to the server, which then sends a response back to the client. It does not allow the server to send data to the client if the client has not initiated the communication by sending a request. In odyssey, we needed a way for the server to send data to the clients when certain events happened, such as when a user sends a message to another user, the recipient of the message should be notified immediately without the need of performing a refresh, or when a player completes an activity, the app must let their fellow players know and immediately present the new activity without the need of a refresh. A very simple way of solving these problems could be to automatically send a request to the server at a specified interval, but that would be very inefficient. To allow real time communication in Odyssey, we made use of the WebSocket protocol which creates a full duplex connection between the client and the server and allows them to send data to each other at any time.

3.2.2.5 Implementing the REST API for Odyssey

As discussed earlier, to create a REST API using Spring, you basically need A Model class which will be used to pass information about an entity around the app, a service, which will handle the business logic, and finally a controller which will create the endpoints and handle the requests made to them. Therefore, to create the REST API for Odyssey, we created a package for each feature of the app (e.g., hunts, users, activities) which contained all necessary classes for that feature. The following image illustrates the structure of our project.



```
odyssey-REST [odyssey] C:\Users\spyro\Documents\dev\odyssey\odyssey-REST
  .idea
  .mvn
  src
    main
      java
        gr.hmu.nile.odyssey
          account
          activity
          activityAnswer
          activityGroup
            dto
              ActivityGroupInfo
              AddPrerequisiteRequest
              BasicActivityGroupInfo
              CreateActivityGroupRequest
              EditActivityGroupRequest
              RemovePrerequisitesRequest
            ActivityGroup
            ActivityGroupController
            ActivityGroupRepository
            ActivityGroupService
          announcement
          config
          directMessage
          error
          gameplay
          hunt
          invitation
          leaderboard
          media
          notifications
          security
          storage
          submission
          team
          teamChat
          user
          utils
          webSockets
          InitialDataLoader
          OdysseyApplication
      resources
```

Figure 6 The project structure of the REST API project

3.3 Frontend Development

This section will focus on describing how the languages, systems, and technologies mentioned in chapter 2 were used to develop the web and mobile applications.

3.3.1 Designing the Mobile Application

The mobile application is the primary frontend software of this project and is thus what the bulk of the frontend time and resources were spent on. It is what allows the users to take part in treasure hunts and complete the activities it asks of them.

The upcoming sections will list and detail the various features of the mobile application and how they were designed and implemented, as well as the challenges we faced in implementing them. Some context for the following sections: screens are the term used for describing the user interface segments of a mobile application. They are equivalent and serve a similar purpose to a website's webpages.

3.3.1.1 Authentication

The mobile application’s authentication is contained within two screens. The “Sign Up” and “Log In” screens. They provide a quick and secure way for the user to sign up and log in to the application respectively, allowing them to use the rest of the application’s features. To spare the user from having to log in every time they reopen the app, after their first login, a token from the server is stored safely in the device’s memory and is used to log the user in automatically whenever they open the app.

3.3.1.2 The Feed Screen

The feed screen is the one showed to the user right after logging in. It is responsible for presenting the treasure hunts to the user. The feed screen has two states. In the “joined” state the feed screen acts as a quick navigation hub between the hunts the user has already joined. In the “discover” state, on the other hand, the screen becomes more of an exploration hub, presenting the user with all available hunts.

The feed screen is comprised of five unique components:

1) “Hunt list items”, which are compact elements displaying the core information about a given hunt. Each item displays the hunt’s image, title, creator, and location, as well as tags showing the state of the hunt and the user’s association with it. Tapping on an item opens the hunt screen for the tapped hunt. Lastly, the hunt items have a quick-options button, which opens a modal with the options of “share the hunt” and “report hunt”.
2) The “hunt list”, which, as the name implies, is a list of hunt list items. The list is supports refreshing, as well as infinite scrolling for the user’s convenience.
3) The “search bar” provides a way for players to search for a particular hunt.
4) Navigation tabs with which to switch between the “joined” and “discover” states of the feed. The “joined” state being the default state when the user opens the application.
5) A floating action button opens the phone’s camera, enabling the user to scan invitation QR codes.

3.3.1.3 The Hunt Screen

The hunt screen is responsible for displaying the hunt’s details as well as offering some hunt-related actions to the user. In addition, the screen features a quick-options button with the same actions as the hunt item quick-options button.

Designing the hunt screen was the first major challenge we faced during the creation of the mobile application. The reason was, we had to fit a potential full page’s worth of text, as well as upwards of six actions in the limited space a smartphone screen provides. Thankfully after some research into how other popular apps solved this issue and some experimentation, we came to the solution. We decided to use tabs to separate the hunt’s information from its actions while having its featured image remain prevalent throughout. The tabs used are the following.

3.3.1.3.1 Details Tab

The details tab features all the hunt’s information. The title is displayed at the top, followed by the creator. Afterward, the hunt’s tags are presented in a horizontally scrollable line. The description follows letting the players know what the hunt is about. Finally, the hunt’s type, location, and difficulty are displayed vertically.

3.3.1.3.2 Actions Tab

The actions tab displays all currently available actions to the user. The actions available change depending on the hunt’s state as well as on the user’s association with the hunt. The actions the user can perform are the following:

1) “Join Hunt”. This action is pretty self-explanatory; it allows the user to join the selected hunt. Note that this action is only available when the user has not already joined the hunt, and the hunt has not ended.
2) “Leave Hunt”. Through this action, the user can leave the hunt, provided he has already joined it.
3) “Play Hunt” opens the game portion of the application allowing the user to participate in the hunt. This action requires that the hunt has been started by its creator.
4) “View Members/Teams” allows the user to see the members and teams of the hunt. This action opens the “members” or “teams” screen depending on the type of the hunt. Allowing the user to interact with them.
5) “Invite to Hunt” produces a QR code invitation on screen, which can then be scanned (through the floating action button in the feed screen) by other players, allowing them to join the hunt. This action becomes unavailable once the hunt has ended.

3.3.1.3.3 Admin Tab

The admin tab, similar to the actions tab, presents the creator of the hunt with all available hunt administrative actions. This tab is only present to the creator of the hunt, ensuring no players can tamper with sensitive hunt actions.

The reason behind the inclusion of the admin tab was to allow the creator to perform the actions required to “run” the hunt without having to leave the application. This promotes quick management, which in turn allows the hunt creator to actively participate in the hunt as a player without having to worry about leaving the app to perform their managerial duties. The admin tab allows the creator of the hunt to perform these actions:

1) The **“Publish Hunt”** action, which changes the hunts state from “draft” to “published” in the backend. That in turn allows players to discover and start joining the hunt. This action is only available if the hunt is in the “draft” state, meaning it has not been published, started, or ended before.
2) The **“Start Hunt”** action, which locks the hunt’s information and activities, preventing further editing and initializes the hunt. The hunt’s information and activities are locked to prevent the creator from changing them while the hunt is active, which could cause confusion between the players and could even cause cheating. This action is only available for hunts in the “published” state.

3) The **“End Hunt”** action, which ends the hunt. Notifying the players and preventing them from completing any more activities, thus finalizing each player’s or team’s score and final spot on the leaderboard.
4) The **“Delete Hunt”** action, which kicks any players that have joined the hunt and then deletes it.

3.3.1.3.4 The Members/Teams Screen

The members/teams screen provides a way for players to view and interact with the members and/or teams of the hunt. Accessed through the “View Members/Teams” action in the Actions tab, this screen’s contents change depending on the hunt’s type and options.

In the case of a “free for all” hunt, the screen shows all users that have joined the hunt. By tapping a user, an options dialog pops up with the option of either sending a message to the player or reporting them.

In the case of a “team hunt”, the screen shows all teams of the hunt. The team element consists of the team’s icon (randomly generated at creation), the team’s name, and a list of all players in the team with the team leader having a crown next to his name. If the creator has chosen to allow members to create teams, a button placed at the top of the screen, allows members to create new teams. To join a team all the user has to do is tap on its element if the team has no password, the user joins automatically whereas if it does, the user is prompted to enter it, upon entering the password correctly the user then joins the team. Tapping on a team’s element while already being a member of the team, brings up an options dialog allowing the user to either leave the team or edit it in case they are the leader.

3.3.1.4 The Treasure Hunt Game Screen

The treasure hunt is the most important part of this system. As such, designing the way players experience it, was perhaps the most important task at hand. We had to make sure that the user interface for the treasure hunt was accessible, seamless, and above all intuitive.

3.3.1.4.1 The Activities Screen

The activities screen is the first one shown to the players when they start hunting. It provides a way for players to select the activity they would like to complete. It does that by displaying the hunt’s activities intuitively and concisely.

The thought process behind the activities screen was, that we wanted a place where players can quickly access the hunt’s activities. Since activities are not necessarily linear, there had to be a way for players to select the activity they would like to complete at any given time. It was for that reason that we made it so tapping a not yet completed activity selects it, and brings the player to the objective screen, allowing them to immediately start working on the activity. Apart from that, tapping on a completed activity, shows the activities information, as well as its solution, giving players a chance to recall any information about that activity. Optional activities feature a subheading mentioning they are optional. And lastly, any activities not available (due to their activity group having a prerequisite group that has not been completed or their group requiring them to be completed in order) are shown though are not accessible.

3.3.1.4.2 The Activity Logs Screen

The activity logs screen shows a list of all completed activities by the player or the player's team. It serves to reference the players can come back to if they need to if they ever need any information provided by previously completed activities. This way hunt creators can hide clues about future activities in previous ones.

Each activity element in the list displays the activity's type, title, the username of the player who completed it, the points the activity was worth, how long ago it was completed, as well as the answer is given. Finally, tapping on the activity will open a dialog with any other information the activity had.

3.3.1.4.3 The Objective Screen

The objective screen is the most important in the application and is where players will spend most of their time. It presents players with the objective of the activity they are currently trying to complete, and it is through this screen that they are called to give their answer.

When designing the objective screen, we had to consider that each activity type has a different objective. However, they also have common elements, such as a title, a description, and an optional image and/or audio clip. This meant that it was possible to reuse the UI elements for the common elements while using different elements for the objective. And indeed, we were able to take advantage of ReactJS's component-based system and reuse the code for the common elements while swapping out the code for the objective, based on the selected activity's type.

The user interface we ended up with for the objective screen is as follows. At the top, rests the title with a large bold font, followed by the description, the image which is set to fit the screen size while maintaining its aspect ratio, and beneath that, an audio player element that features the activity's audio clip. Finally, at the bottom, lies the objective section, which changes based on the activity's type. What is common between objective sections, is the submit button. A button, which allows the player to submit their answer to the server. If the answer is correct, a notification snackbar appears letting them know they have answered correctly, and the next activity is automatically selected if available. On the other hand, if they answered incorrectly, a snackbar appears indicating their answer was wrong and prompting them to try again.

The objective sections for the different activity types are the following:

1) For question activities, the objective section consists of an input field, followed by a button that submits the answer in the input field.
2) For multiple-choice activities, the possible answers are shown in selectable fields. Tapping on one of these fields selects it, after which the submit button is enabled and can be tapped to submit the selected answer.
3) For photo activities, the objective section displays a button which when tapped, opens the camera of the phone, allowing the player to take the desired photo. After taking the photo, the submit button appears allowing the player to submit the photo.
4) Similar to photo activities, the QR scan activities feature the same button, although in this case, the player needs to scan a QR code. After scanning it the player can submit it through the submit button.
5) Lastly, the GPS activities objective section consists of a button that opens a map displaying the user's current location, and in case the destination is set to be visible, the location and radius of the destination are displayed as well. As soon as the player

believes they are in the range of the destination, they can press a button to lock their current position as the answer, and then submit it through the submit button.

3.3.1.4.4 The Leaderboard Screen

Upon deciding that we wanted hunts to have a scoring system, we knew we would need some kind of leaderboard in place, to let the players know how well they have performed compared to each other. We elected to implement a leaderboard that would compare scores on a hunt-by-hunt basis rather than a global leaderboard since hunt scores can vary wildly.

The leaderboard screen is used to keep track of player and team progress throughout the hunt. It displays all participating members or teams showing how many points they have accrued, how many activities they have completed, and how fast they have completed them. It also features a "podium" displaying the top three players/teams. The leaderboard screen is available both during a hunt, and after it has ended, allowing players to compare scores in real-time as well as commemorating the placements at the end.

3.3.1.4.5 The Notifications Screen

After the decision was made to let the creator communicate with all of the hunt's members through announcements, we had to figure out how we would display those announcements to the players. The notifications screen was implemented as a result of that. Through the notifications screen, players can see all announcements related to the hunt. Meaning they can stay up to date with any changes made to it.

3.3.1.5 Miscellaneous Features

This section will focus on features the application's miscellaneous features. Showcasing the various small features that help the application provide the best user experience it can.

3.3.1.5.1 Real-Time Duplex Communication

In a team hunt, when a player completes an activity, the rest of his team needs to be notified of the activity's completion. When the hunt creator makes an announcement, all of the users registered for the hunt need to be notified. The same holds for when the hunt begins or ends. To achieve this, the server needs a way to communicate with the application. To that end, the mobile application features two-way communication with the server through the WebSocket protocol. It allows the server to send data to the application without the application requesting that data, this enables a fast and secure exchange that solves the mobile application's real-time data exchange requirements.

3.3.1.5.2 Visual Feedback Through Snackbars

Visual feedback is key in designing an intuitive interface. It provides visual input to the user in real-time, letting them know that their actions are happening. Basic forms of visual feedback include the change in the color of a button when it is pressed or the highlighting of a link when hovered. In some cases, though there is a delay from when the user performs an action, and when that action is completed. In such cases, it is hard to provide the user with an immediate visual input since that input may need to change depending on the results of the action. For example, when submitting an answer for an activity to the server, that answer can either be correct or incorrect. Since the application does not know beforehand it needs to wait on confirmation from the server. To alleviate this issue, we decided on using Snackbars. Snackbars are like small notification popups that appear inside the application. Through them, we can provide accurate visual feedback on the completion of such actions.

3.3.2 Designing the Web Application

The web application is the second major frontend software of the Odyssey environment. It is a web application designed to allow users to create and manage hunts. It offers users the necessary tools with which to build fun, unique, and creative hunts and activities.

The following sections will describe the function, design philosophy, and thought process behind each of the web application's pages. The focus will specifically be on the hunt detail and activity managers since they are where the bulk of the web application's function lies.

3.3.2.1 Designing the Homepage

The homepage is the first page a user sees when first visiting the website. It is perhaps one of the most important pages of a website. It is a homepage's job to create a strong first impression, to grab a user's attention, and in this case, explain the function and features of the application in a comprehensive yet succinct fashion. As such, we took great care in designing a homepage which fulfills those criteria.

3.3.2.2 Authentication

The web application's authentication consists of three pages. The Sign-Up page, the Log-In page, and the User Profile page. The Sign-Up and Log-in pages follow the latest design principles and ask the user to provide the necessary data clearly and securely. As a result, creating an account and logging in is fast, secure, and reliable. Since the bulk of the application's features require the user to be logged in, the authentication process needed to be as unintrusive as possible. To that end, after a user has logged in once in a certain browser, they are automatically authenticated in any subsequent visits.

The user profile page offers users three functions with which to modify their account. Firstly, they can change the email associated with their account. Secondly, they can change their password, which provides an extra layer of security to the account. And lastly, they can delete their account, which deletes all data associated with that account from our system, ensuring the user's privacy.

3.3.2.3 Designing the Hunt Detail Manager

The hunt detail manager is one of the most important parts of the web application. It is a form that is used when first creating a hunt, as well as when editing one. It presents the user with all the necessary fields, options, and actions required to build a hunt.

When designing the Hunt Detail Manager, we had already established the fields and features a hunt would have in the backend, so all that was left was to design a form that would include them all in a way that would not overwhelm the user. To achieve that, we agreed to split those fields and features into three distinct segments.

3.3.2.3.1 The Info Segment

The **Info Segment** is the one in which the user is asked to provide information that defines the hunt. It consists of seven fields:

1) The **hunt type** is a field that decides on whether the type of the hunt will be teams or free for all. It is only present on the dialog which appears when first creating a hunt and cannot be changed after the hunt is created. This decision was made to keep the hunt creation streamlined and prevent confusion down the line.
2) The **title**, as its name implies is the most prevalent text of a hunt.
3) The **description**, which is the field where a hunt creator can explain what the hunt is about to potential players.
4) The **difficulty** is used to indicate how challenging a hunt is to complete and is one of the fields used to filter hunts to the user's satisfaction.
5) The **location** indicates where a hunt takes place and can be as broad or specific as the hunt creation requires. This field is optional and is only supposed to be used with hunts that are designed to be played in a specific place. In cases where a hunt does not need to take place in a specific location, this field can be omitted.
6) The **tags** field accepts a variety of user-defined keywords that better help defines the hunt. It is both a quick and easy way to let players know about the content in the hunt as well as a great way for players to filter through hunts they might be interested in.
7) The **image** of the hunt. In this field the create of the hunt is asked to provide an image that will be featured for the hunt. An image allows the user to brand their hunt and can be used to make a first impression as well as provide a lot of information in a short amount of time. This field is not accessible in the dialog that appears when first creating a hunt, but only in the menu that appears when editing one. This was done for clarity, and to ensure the user is not overwhelmed. In case an image is not provided, the web application will use the default image.

3.3.2.3.2 The Options Segment

The **Options Segment** contains all the administrative options of a hunt. Through the options, the creator of the hunt can manage how other users interact with the hunt. The options a hunt has, are dependent on its type. The Options Segment consists of four options:

1) The **"private"** option determines whether a hunt can be discovered by the search feature in the mobile and web applications. It can be set to either private or public. If set to private the only way people are going to be able to join the hunt is through invitations. This allows for finer control over who can see the hunt. It is set to private by default and is available for both free for all and team hunt types.
2) The **"cloneable"** option, if set to on, allows for users other than the creator of the hunt to clone the hunt. Cloning a hunt gives users the ability to replay an existing hunt again with different people. A good use case for this feature would be a hunt tied to a tour of a museum or an archaeological space. It would allow for multiple tour guides to use a hunt over and over with each group, without having to rebuild the hunt from scratch. When set to off, only the hunt's creator will be able to clone the hunt. It is set to off by default and is available for both "free for all" and "teams" hunt types.
3) The **"can members create teams"** option, as its name implies controls whether players of the hunt can create their own teams. When set to on it allows players to create teams making the hunt more open-ended, while setting it to off grants the creator the ability to predetermine the teams of the hunt. This option is specific to the "teams" hunt type and is set to "on" by default.

4) The **“can members switch teams”** option, determines the players’ ability to switch between teams. This, alongside the aforementioned “can members create teams” allows for very granular control of how the teams in a hunt are created and structured. This option is specific to the “teams” hunt type and is set to “on” by default.

3.3.2.3.3 The Actions Segment

The **Actions Segment** includes all actions the hunt’s creator can perform, that are directly related to the hunt. These actions are:

1) The **“Publish Hunt”** action, which changes the hunts state from “draft” to “published” in the backend. That, in turn, allows players to discover and start joining the hunt. This action is only available if the hunt is in the “draft” state, meaning it has not been published, started, or ended before.
2) The **“Start Hunt”** action, which locks the hunt’s information and activities, preventing further editing and initializes the hunt. The hunt’s information and activities are locked to prevent the creator from changing them while the hunt is active, which could cause confusion between the players and could even cause cheating. This action is only available for hunts in the “published” state.
3) The **“End Hunt”** action, which ends the hunt. Notifying the players and preventing them from completing any more activities, thus finalizing each player’s or team’s score.
4) The **“Copy Access Code”** action, which copies the hunt’s access code to the clipboard. This access code can then be shared and used to join the hunt, even if the hunt is set to private.
5) The **“Duplicate Hunt”** action, which makes a clone of the hunt and its activities, enabling its reuse.
6) The **“Delete Hunt”** action, which kicks any players that have joined the hunt and then deletes it.

3.3.2.4 Designing the Activity Manager

The activity manager is one of the cornerstone pages of the web application. The activity manager is the tool which enables the user to create the activities player will be called to clear during the hunt. It is designed around giving the creator as much flexibility as possible so that they can unleash their creativity, bring their ideas to life, and create a truly unique experience for the players of their hunt. More specifically, the activity manager is used to manage two elements.

3.3.2.4.1 Activity Groups

Activity Groups are in essence “containers” that house activities. They provide a way for the hunt creator to group a hunt’s activities as they see fit. A common use case for activity groups is using them as chapters by grouping activities by theme or location. Thus, allowing players to experience a more immersive, cohesive, and digestible experience. An activity group has the following features:

1) A title and color, to distinguish them between other activity groups, and a description, which can be used to give players an overview of what the activities in this activity group have in store for them.

2) Prerequisite activity groups. Meaning an activity group's activities can be in essence locked until specified activity groups are completed. Allowing for fine control over the pace of the hunt.
3) The ability to order activities within an activity group, and the option to require players to complete the activity group's activities in sequential order. This was implemented to give the creator even greater freedom in expressing their vision. Reordering the activities in an activity group is achieved with a seamless, intuitive, user-friendly drag and drop interface.

3.3.2.4.2 Activities

Activities, which are arguably the most important component of the hunt as a whole. Activities are the tasks players will be called to complete once the hunt has begun. They play a big role in shaping how the hunt progresses.

Designing the user interface for creating and managing activities proved to be one of the greatest challenges of the frontend software part of this project. The form of each activity had to have the elements that are present within each activity as well as each activity's unique objective. This was not an easy task, however using what we had learned from our previous attempt as well as a lot of trial and error, we managed to create a user interface that we feel confident about.

Each activity was given each own form. However, through ReactJS's component-based system, we were able to reuse the code for the inputs for the common activity fields, namely the title, score, description, and the optional checkmark. As for each activity's objective, their forms were designed as follows:

1) The **"Photo Activity"** form was the easiest to design since its objective is to take any photo, requiring no unique field for its form.
2) The **"Question Activity"** form was given a field that could accept any number of possible correct answers, allowing the hunt creator to construct questions or riddles with more than one correct answer.
3) The **"Multiple Choice"** form was similar to the "Question Activity" in that it also required a field that would accept multiple possible answers; however, it also required a field that would designate one of the possible answers as the correct one.
4) The **"QR Scan"** form has two different states. During the activity's creation, the form is identical to the base form, without any extra fields. The reason for this is because the answer string is created after the activity is created. When editing the activity, the answer string is then displayed as a QR code along with a download button which downloads an image of the QR code for use by the creator.
5) The **"GPS Activity"** proved to be quite challenging. This form required four extra fields. Latitude, longitude, and radius to establish the center and radius of the circle, as well as the option to have the position of the circle shown to players. While adding three numbers and one boolean field to the form would be easy, it would mean the user would have to provide the coordinates by hand. That would most likely mean that to create a "GPS Activity" would have to use some third-party map software to find the spot these coordinates represent. The same would be true for every time they would like to edit the coordinates. Since a hunt can potentially have upwards of a dozen "GPS Activities" that would be a terrible user experience causing the hunt creator to waste a lot of time. To prevent that, we decided to add an interactive map to the form. The interactive map supports zoom in and out, clicking anywhere on the map, sets the coordinates to the corresponding spot on the map, the selected coordinates, as well as the radius, is also

displayed, giving the user the appropriate feedback for the location of the circle they have created.

3.3.2.5 The Member Manager

The member manager provides hunt creators the tools required to moderate the players of their hunt. In “free for all” hunts, the member manager allows the hunt creator to remove members from the hunt. While in “teams” hunts, the member manager enables the creation and deletion of teams, changing team leaders, as well as freely moving and removing members to and from teams.

When designing the member manager, it was of utmost importance not only to give the players the tools required to manage the members and teams of the hunt, but to also make doing so intuitive, quick, and straightforward. That is why we decided to use a drag and drop interface similar to the one used in the activity manager.

3.3.2.6 Announcements Page

The announcements page provides the hunt creator with an interface through which they can write and send announcements. There is not much to say about the design of the announcements page. It consists of an announcement form that asks for a title, a message, and an optional image for the announcement. And a list of the announcements for the current hunt. All things that had been done before in some way or another. All had to do was put them together in a cohesive fashion.

4. User Interface

This chapter will showcase the user interface of both the mobile, as well as the web application's user interface.

4.1 Mobile Application

When designing the user interface, we were focused on making it robust and intuitive as well as providing the best user experience possible. To that end, we studied and took inspiration from various famous and trending mobile applications on the market and followed the latest principles in user experience design.

4.1.1 Authentication Screens

The authentication user interface of the mobile application consists of three screens. Each of these screens serves a distinct and important role, in allowing the user to proceed to the main part of the application.

4.1.1.1 Welcome Screen

The welcome screen is the first screen that appears when a user first opens the app. It is a hub meant to guide the user to either signing up or signing in. The user is not meant to linger on this screen for very long. To that end, a simplistic design was used, consisting of a welcome message and two buttons that lead the user to the other authentication screens.

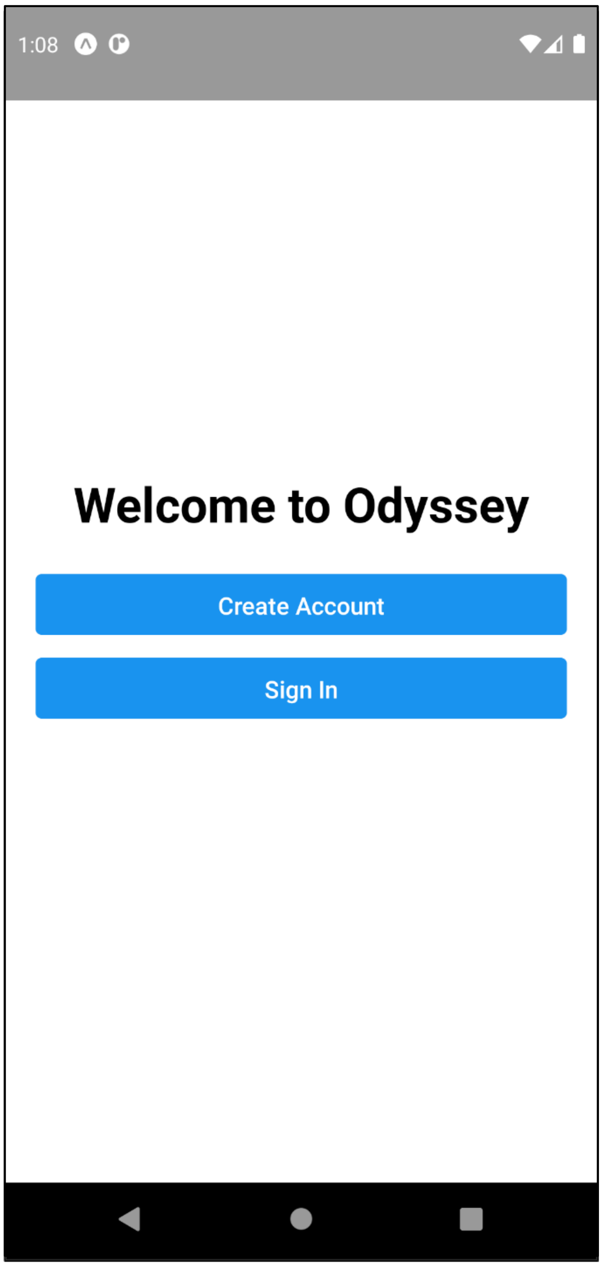
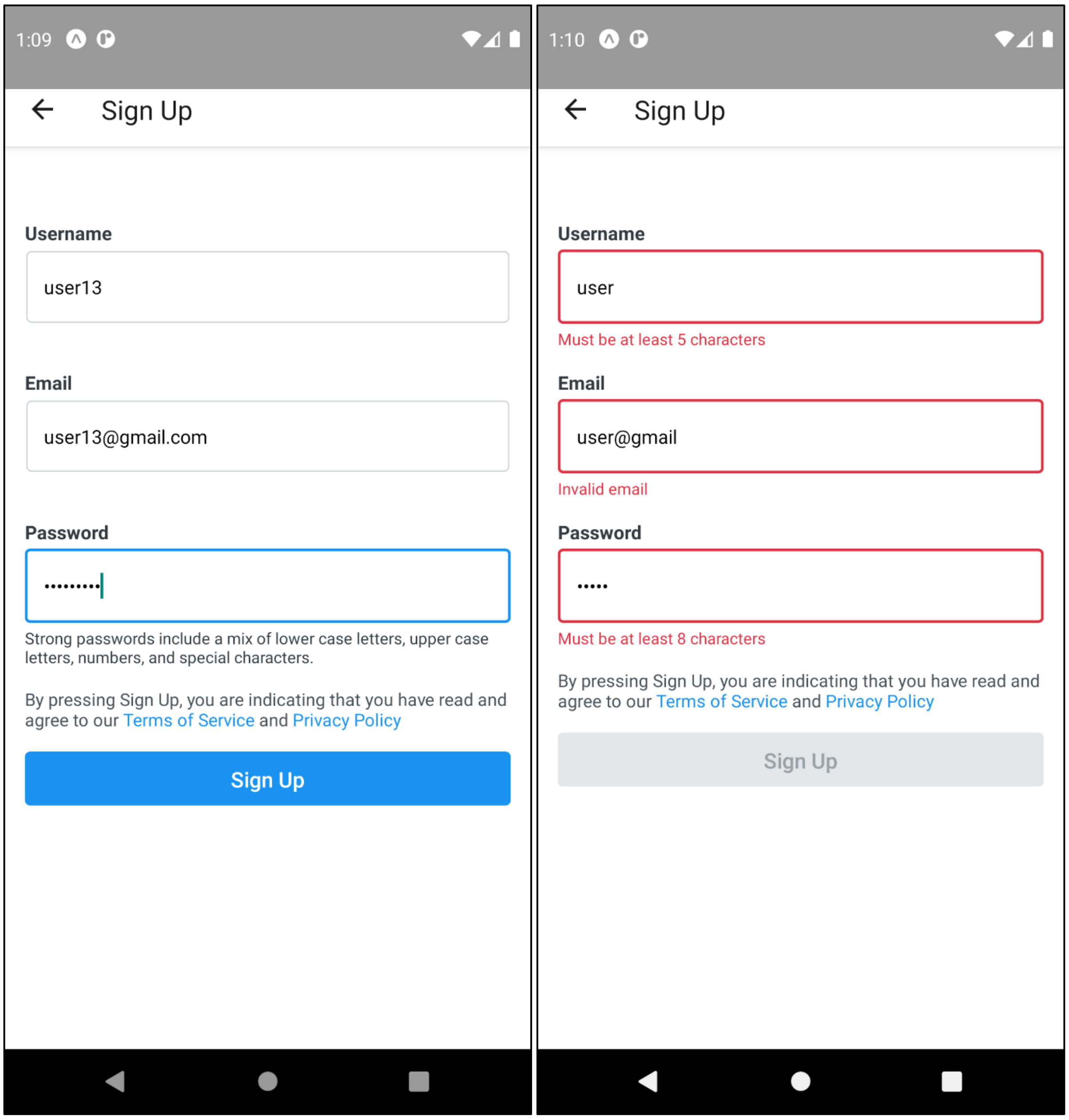


Figure 7 The Welcome Screen

4.1.1.2 Sign-Up Screen

The sign-up screen allows the user to create an account and proceed to the main part of the application. It features a form with just three fields as we want the user to be able to set up an account and get to playing as fast as possible. The form features error checking and feedback for the type of error.


1:09

Sign Up

Username

user13

Email

user13@gmail.com

Password

•••••••••

Strong passwords include a mix of lower case letters, upper case letters, numbers, and special characters.

By pressing Sign Up, you are indicating that you have read and agree to our Terms of Service and Privacy Policy

1:10

Sign Up

Username

user

Must be at least 5 characters

Email

user@gmail

Invalid email

Password

•••••

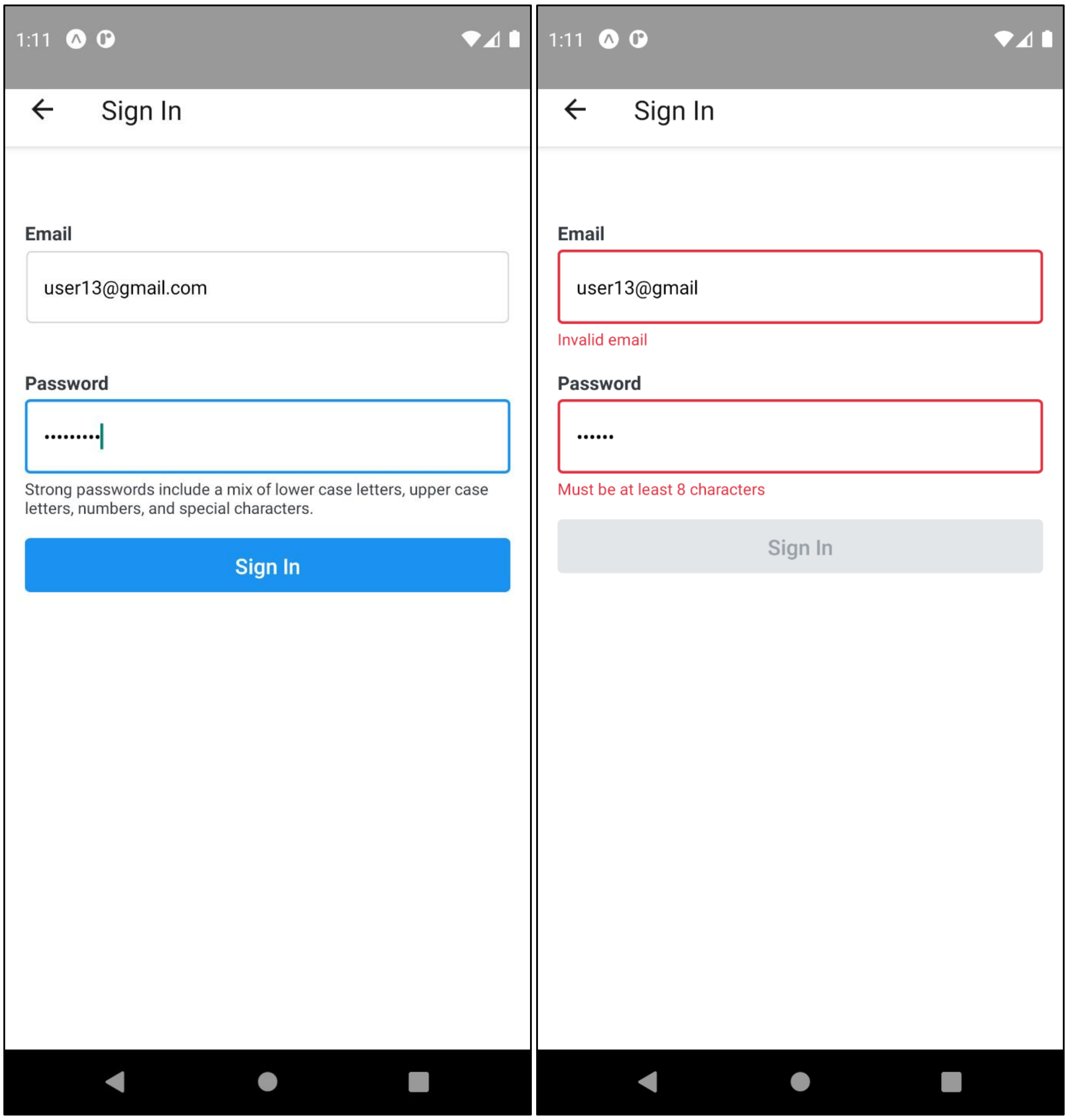
Must be at least 8 characters

By pressing Sign Up, you are indicating that you have read and agree to our Terms of Service and Privacy Policy

Figure 8 The Sign-Up Screen

4.1.1.3 Sign-In Screen

Like the sign-up screen, the sign-in screen features a form with two fields. Enabling quick access to the hunts. It also features error checking and feedback.


1:11

Sign In

Email

user13@gmail.com

Password

••••••••••

Strong passwords include a mix of lower case letters, upper case letters, numbers, and special characters.

Sign In

1:11

Sign In

Email

user13@gmail

Invalid email

Password

••••••

Must be at least 8 characters

Sign In

Figure 9 The Sign-In Screen

4.1.2 Hunts Feed Screen

The hunts feed screen is a gallery of hunts from which the player can pick take a glance at the main points of a hunt and choose whether they want to proceed. It features three main elements. The hunts list, list items, and QR scanner which when combined make for an appealing introduction to the hunts.

4.1.2.1 Hunts List

The hunts list as its name implies is a list that displays hunts to the user. It features two different modes represented by the tabs on the bottom, infinite scrolling for a seamless browsing experience, drag-to-refresh, as well as search bar with which to filter for the desired hunts.

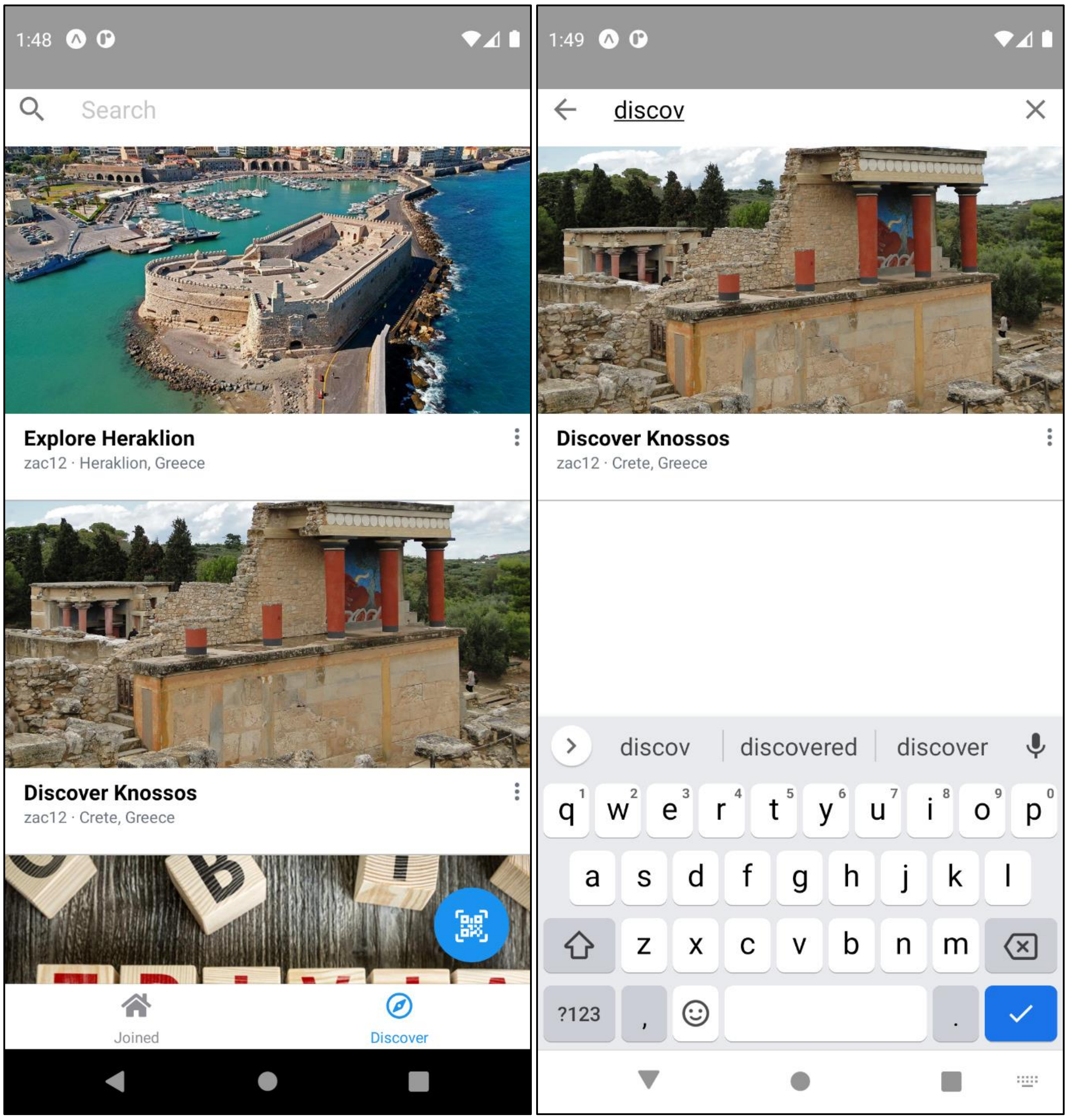


Figure 10 The Hunts Feed Screen

4.1.2.2 Hunt List Item

The hunt list item offers a preview of a hunt. It is meant to display a hunt's basic information to get users interested It prominently features the hunt's image, followed by tags representing the hunt's state, ownership, and joined status. Directly below, the hunt's title, creator, and location are displayed. Lastly, the hunt list item has an options button which opens up further quick options.

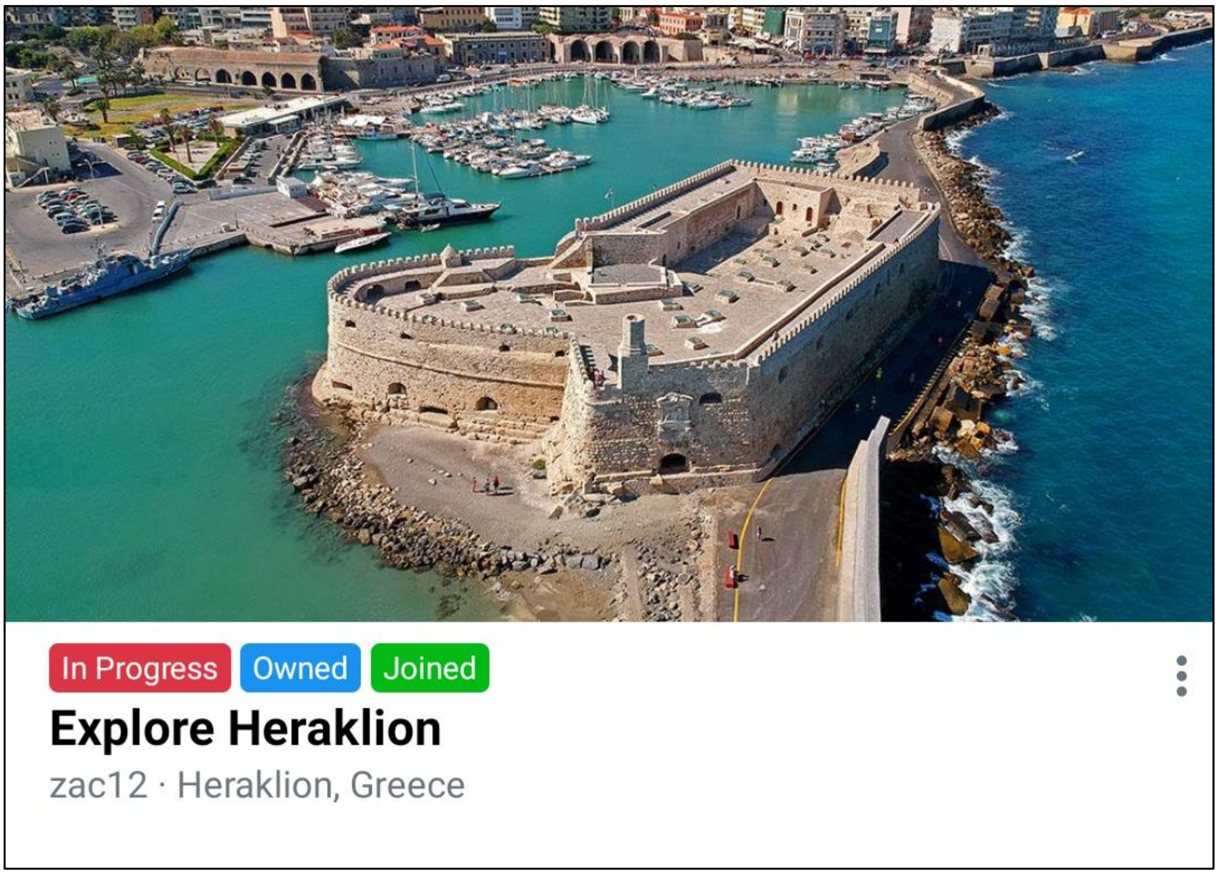


Figure 11 A Hunt Item

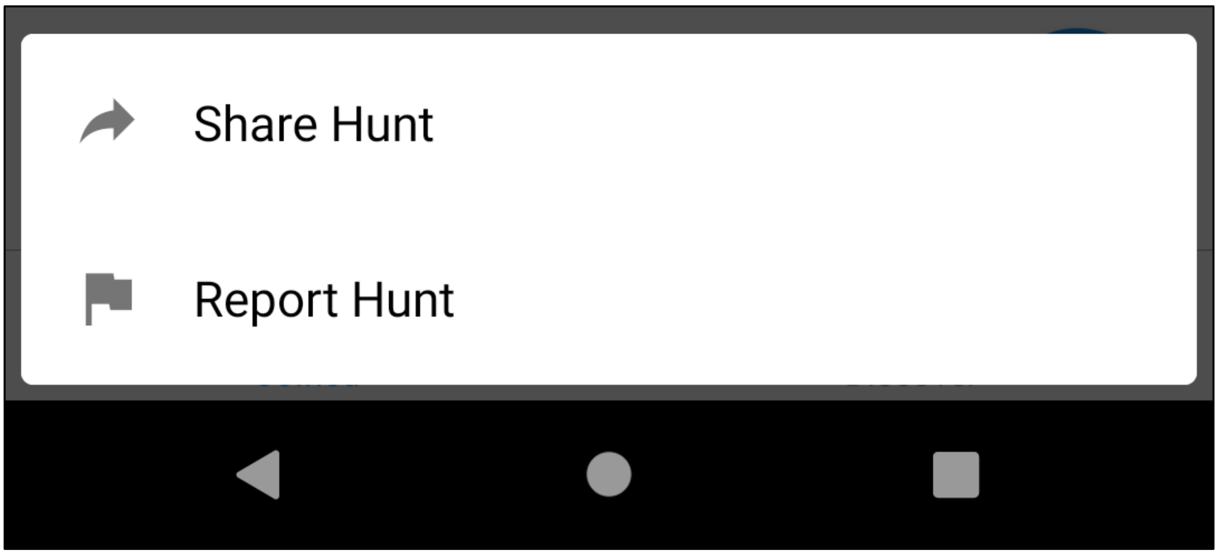
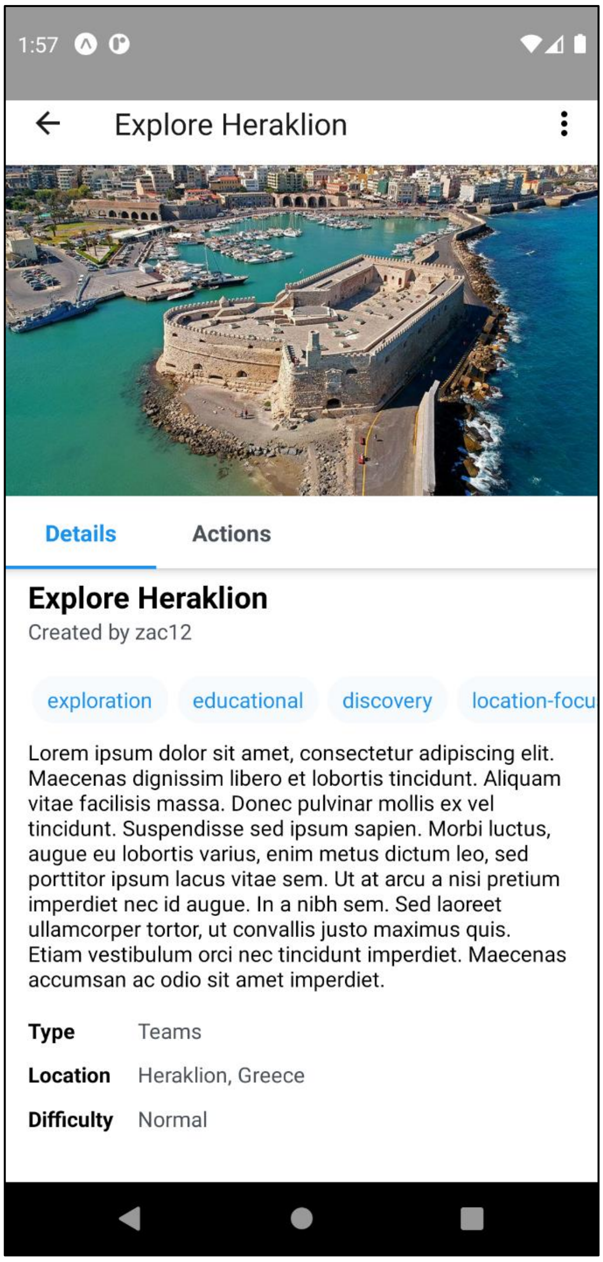


Figure 12 A Hunt's Quick Option Menu

4.1.3 Hunt Screen

The hunt screen is the main way a user can interact with the hunt besides playing it. In this screen, the user can read all available information regarding the hunt, as well as perform any pre-game actions. From top to bottom, the hunt screen features a header, with an options button that opens a quick options menu as seen in (FIGURE). The hunt's image is prominently featured next. Finally, the tabs section consists of either two or three tabs depending on whether the user is an administrator, which display the hunt’s information actions and administrative actions.



1:57

Explore Heraklion

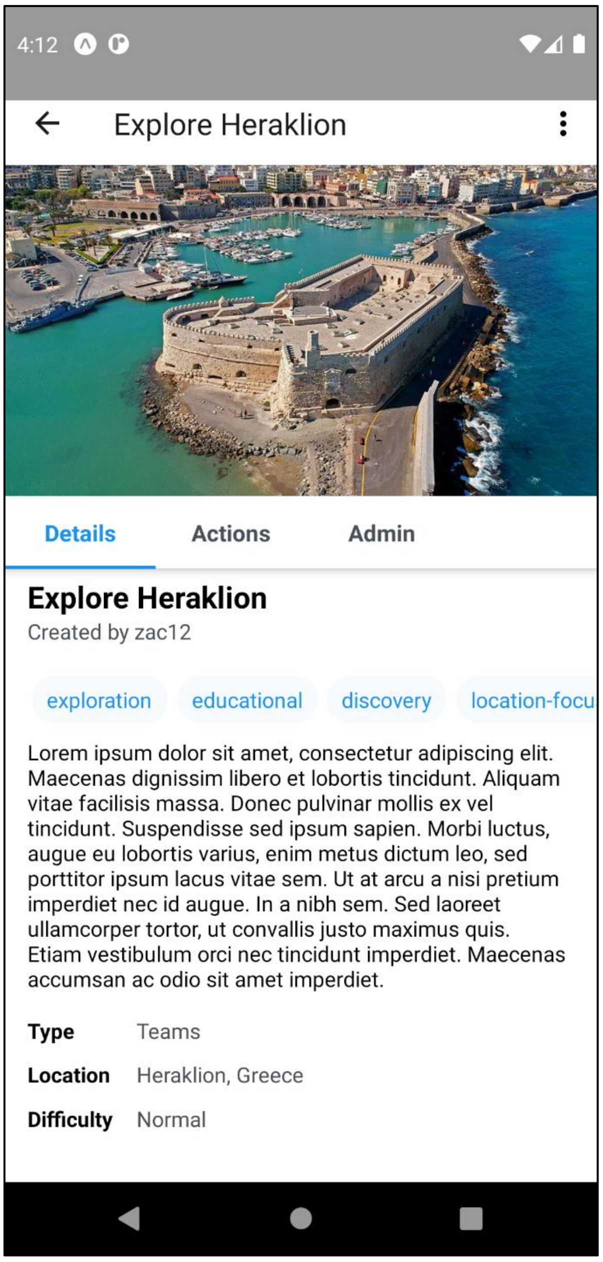
Details | Actions

Explore Heraklion
Created by zac12

exploration | educational | discovery | location-focu

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas dignissim libero et lobortis tincidunt. Aliquam vitae facilisis massa. Donec pulvinar mollis ex vel tincidunt. Suspendisse sed ipsum sapien. Morbi luctus, augue eu lobortis varius, enim metus dictum leo, sed porttitor ipsum lacus vitae sem. Ut at arcu a nisi pretium imperdiet nec id augue. In a nibh sem. Sed laoreet ullamcorper tortor, ut convallis justo maximus quis. Etiam vestibulum orci nec tincidunt imperdiet. Maecenas accumsan ac odio sit amet imperdiet.

Type	Teams
Location	Heraklion, Greece
Difficulty	Normal



4:12

Explore Heraklion

Details | Actions | Admin

Explore Heraklion
Created by zac12

exploration | educational | discovery | location-focu

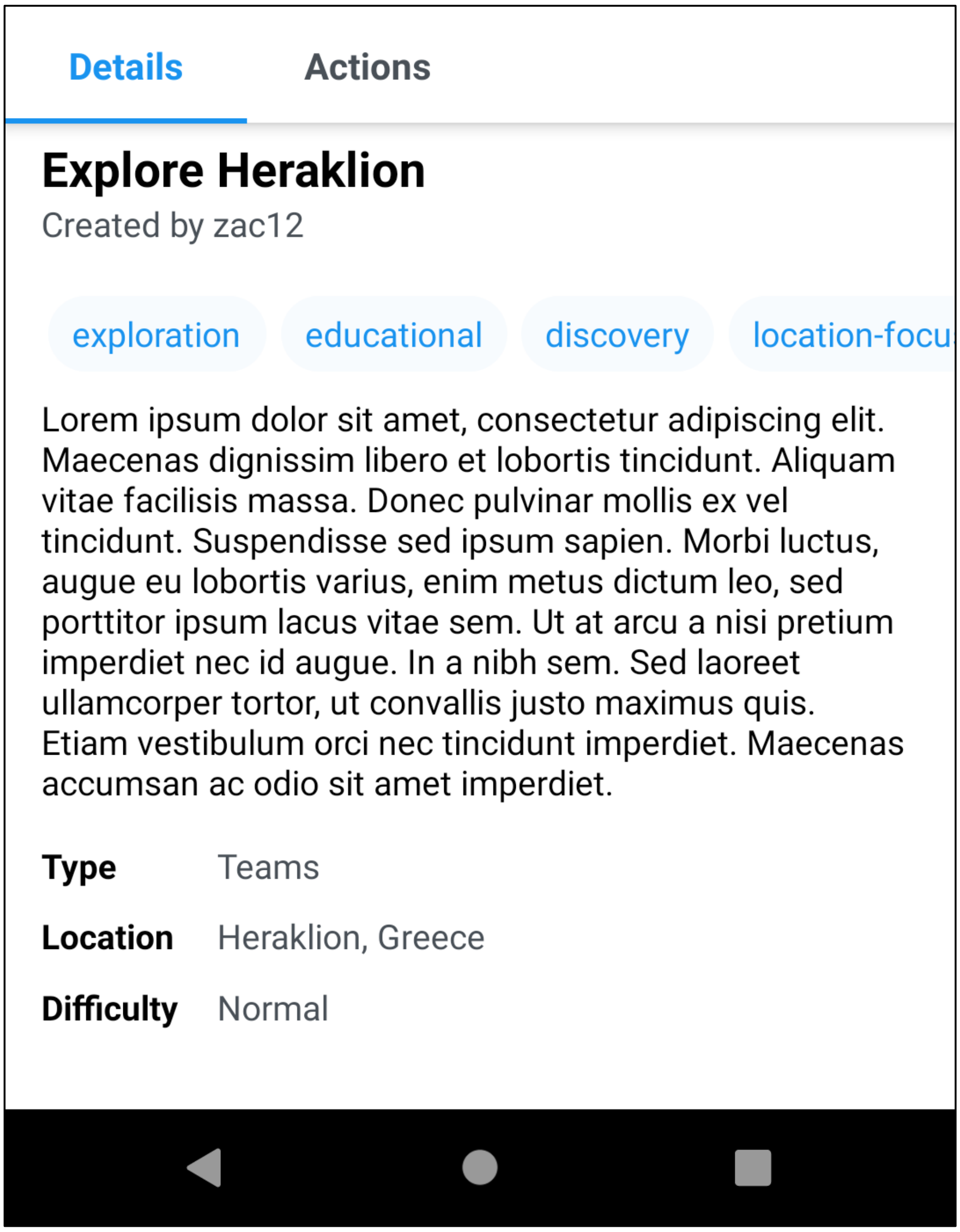
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas dignissim libero et lobortis tincidunt. Aliquam vitae facilisis massa. Donec pulvinar mollis ex vel tincidunt. Suspendisse sed ipsum sapien. Morbi luctus, augue eu lobortis varius, enim metus dictum leo, sed porttitor ipsum lacus vitae sem. Ut at arcu a nisi pretium imperdiet nec id augue. In a nibh sem. Sed laoreet ullamcorper tortor, ut convallis justo maximus quis. Etiam vestibulum orci nec tincidunt imperdiet. Maecenas accumsan ac odio sit amet imperdiet.

Type Teams
Location Heraklion, Greece
Difficulty Normal

Figure 13 The Hunt Screen

4.1.3.1 The Details Segment

The details segment displays the hunt’s information. The title and creator are shown at the top, followed by a horizontally scrollable list of the hunt’s tags. Next is the description, followed by the hunt’s type location and difficulty.



Explore Heraklion

Created by zac12

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas dignissim libero et lobortis tincidunt. Aliquam vitae facilisis massa. Donec pulvinar mollis ex vel tincidunt. Suspendisse sed ipsum sapien. Morbi luctus, augue eu lobortis varius, enim metus dictum leo, sed porttitor ipsum lacus vitae sem. Ut at arcu a nisi pretium imperdiet nec id augue. In a nibh sem. Sed laoreet ullamcorper tortor, ut convallis justo maximus quis. Etiam vestibulum orci nec tincidunt imperdiet. Maecenas accumsan ac odio sit amet imperdiet.

Type	Teams
Location	Heraklion, Greece
Difficulty	Normal

Figure 14 The Details Segment of the Hunt Screen

4.1.3.2 The Actions Segment

The actions segment enables the user to perform any actions available to them at the time they access it. The actions segment can display the following actions:

1) Join Hunt.
2) Leave Hunt.
3) Play Hunt.
4) View Members/Teams.
5) Invite to Hunt.

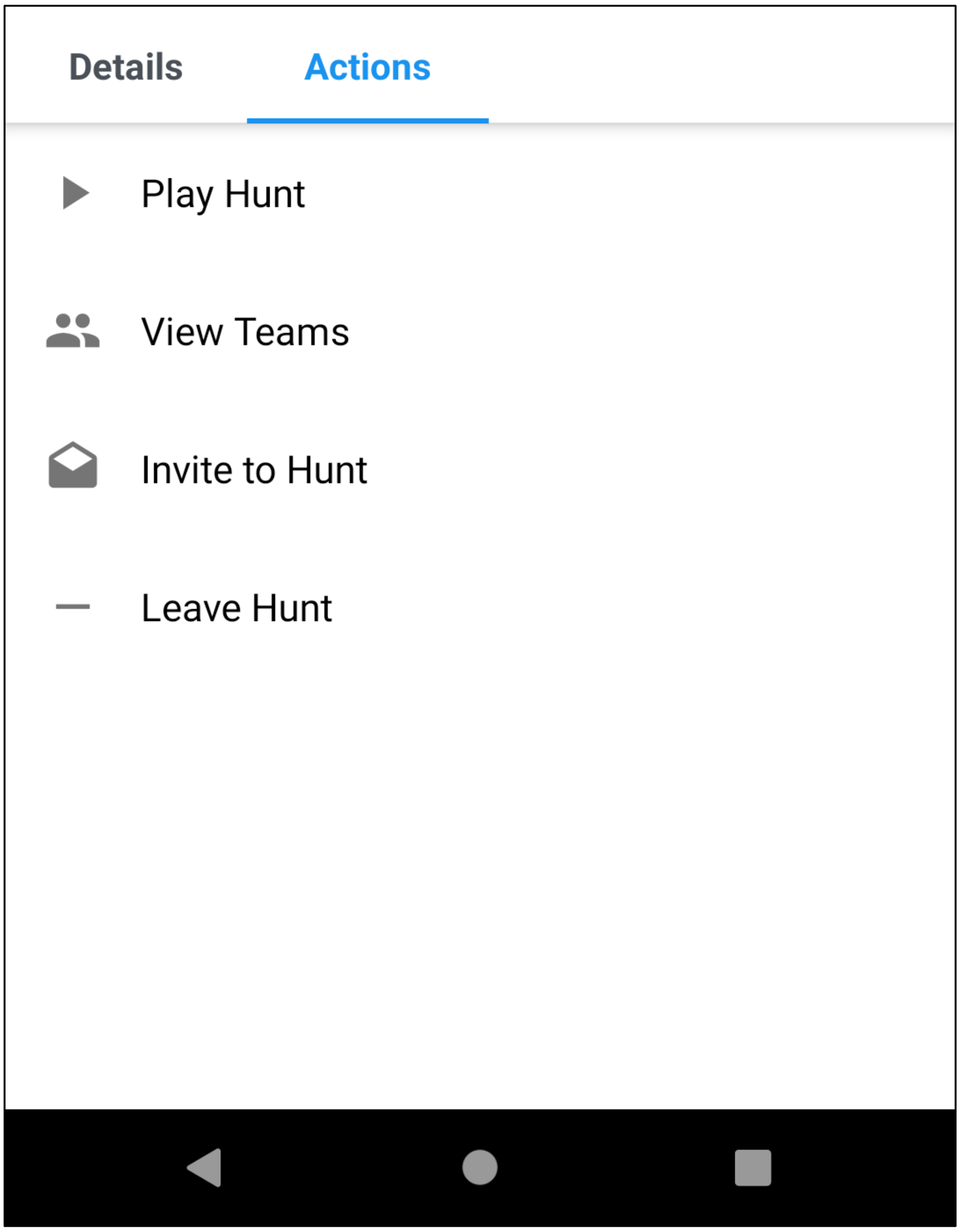


Figure 15 The Actions Segment of the Hunt Screen

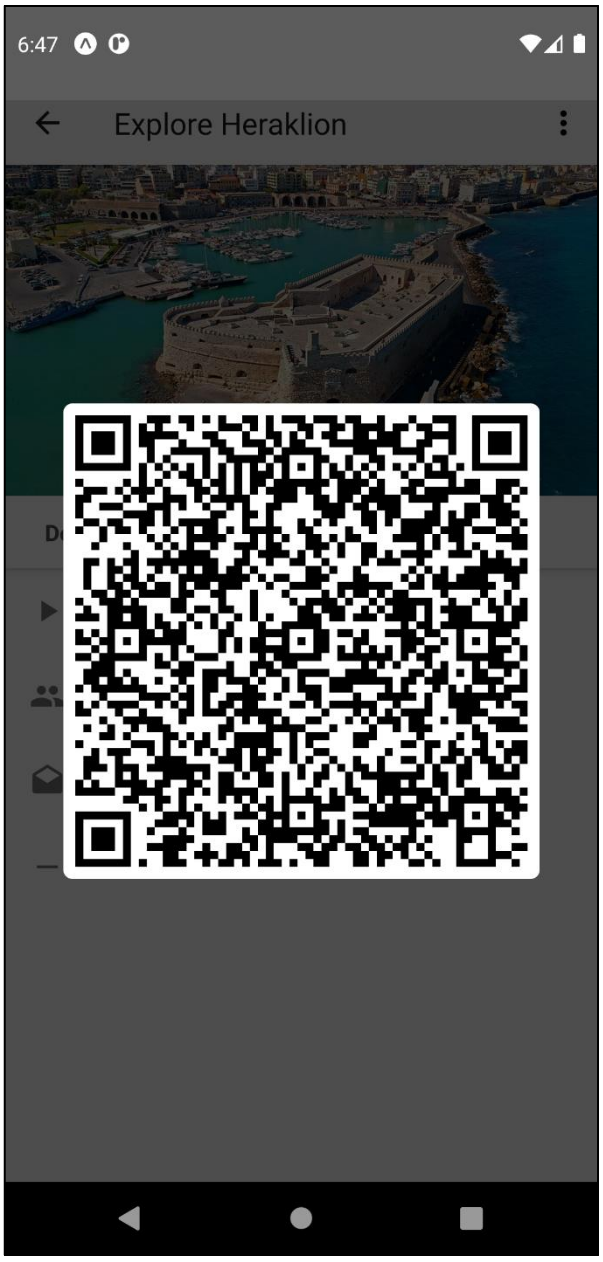


Figure 16 QR Code for the Invite to Hunt Action

4.1.3.3 The Admin Segment

The admin segment is only accessible to administrators. It displays a few administrative actions used to manage the hunt on the fly. The following actions are available through the admin segment:

1) Publish Hunt.
2) Start Hunt.
3) End Hunt.
4) Delete Hunt.

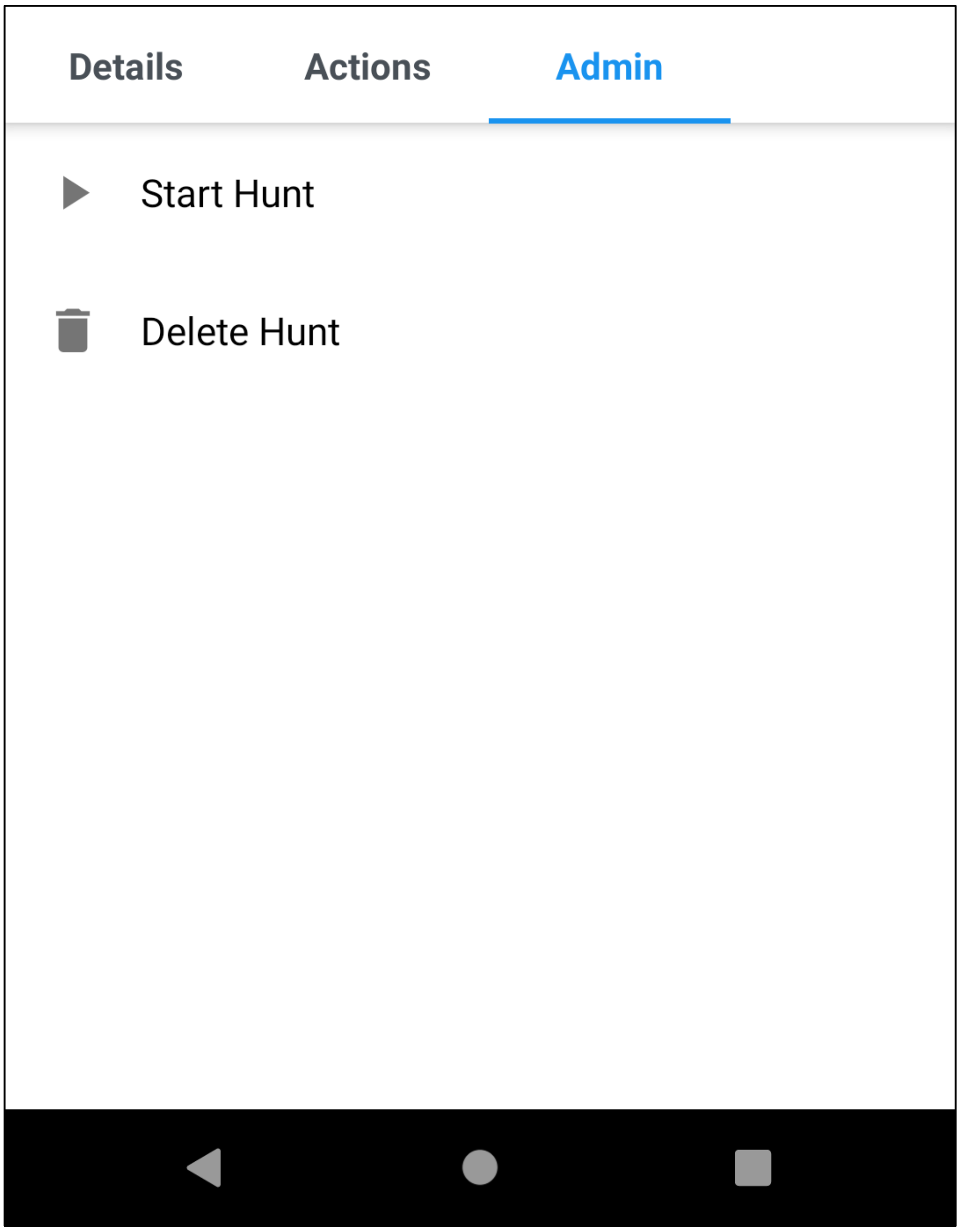


Figure 17 The Admin Segment of the Hunt Screen

4.1.3.4 The Teams/Members Screen

The teams/members screen provides a way for the user to view, manage, and interact with a hunt's users and teams. Showcased below are the layout of the screen, as well as the various features. Features such as creating and editing a team, as well as interacting with teams and users.

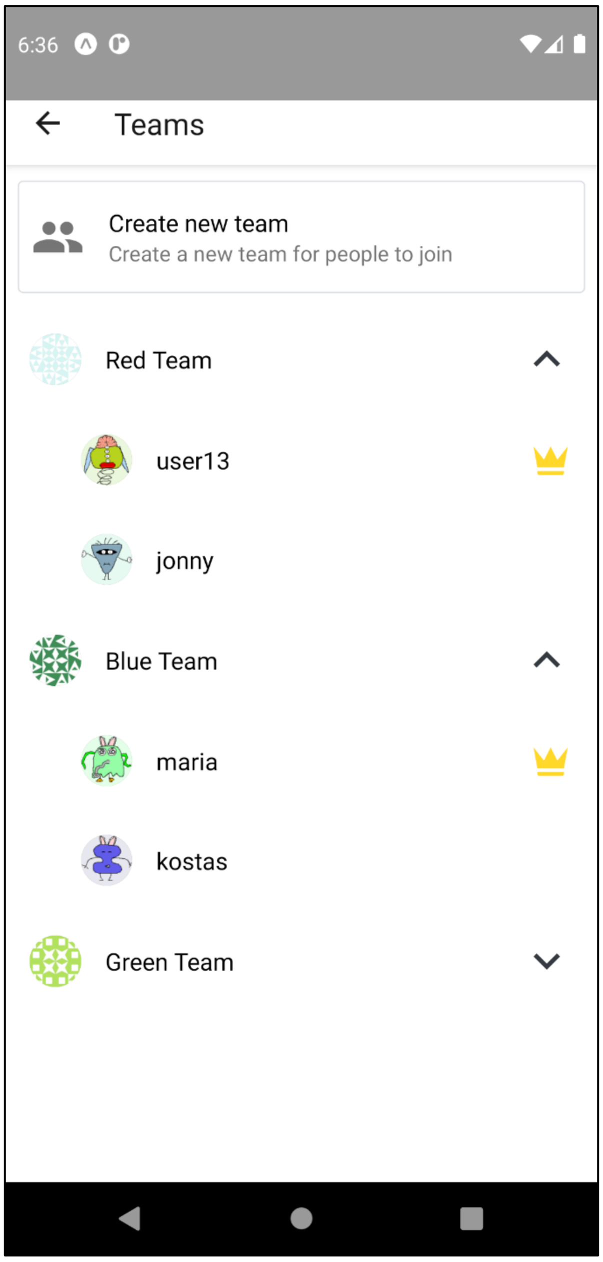
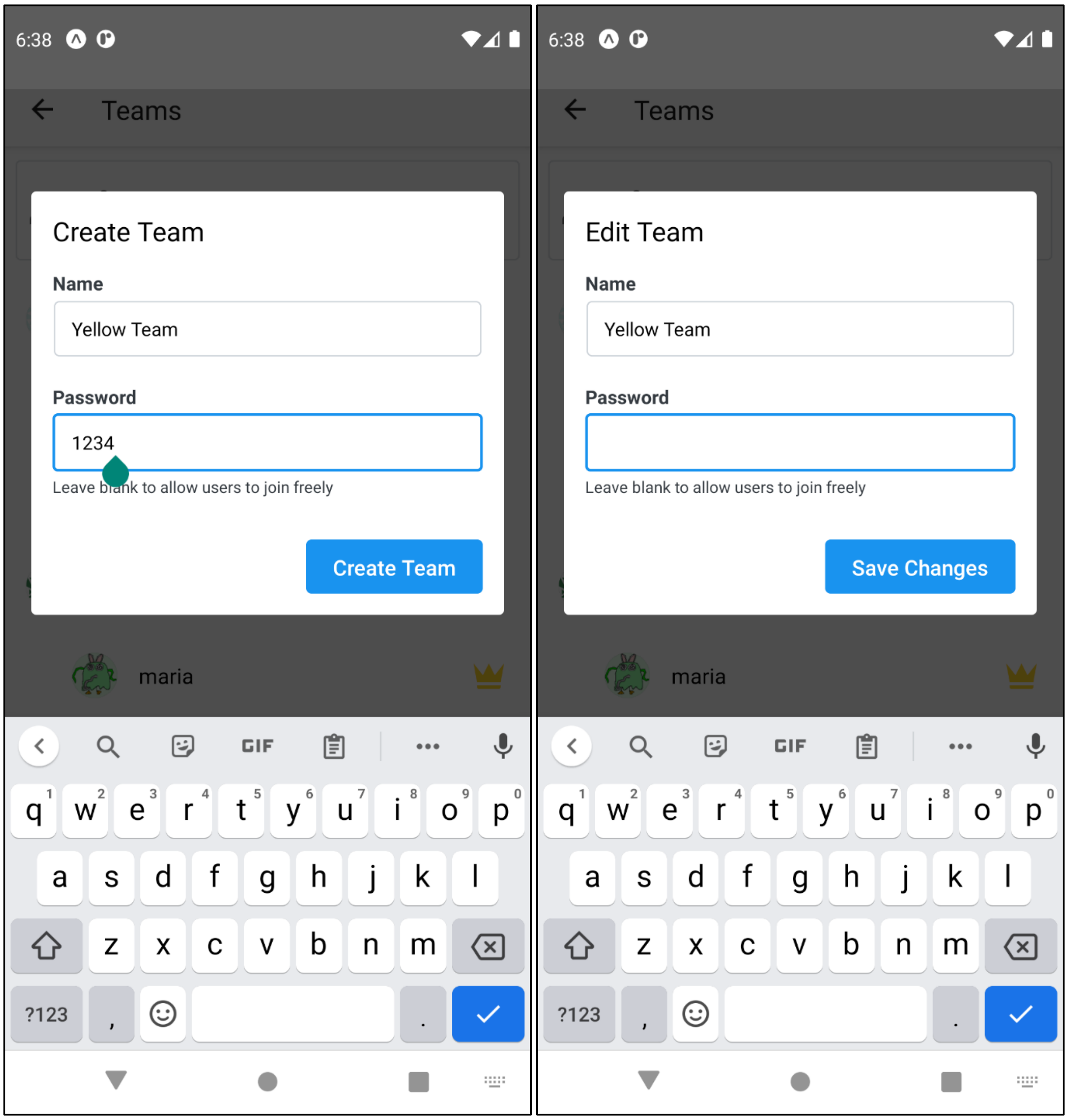


Figure 18 The Members/Teams Screen


Create Team

Name: Yellow Team

Password: 1234

Leave blank to allow users to join freely

Edit Team

Name: Yellow Team

Password:

Leave blank to allow users to join freely

Figure 19 The Create/Edit Team Dialog

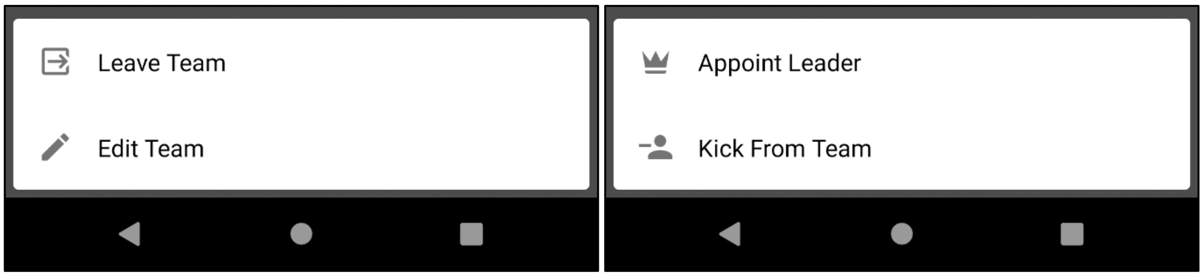


Figure 20 Team and Member Quick Options Menus

4.1.4 Game Screens

The game screens are how the players get to participate in the hunt. Through the various screens in this collection players can complete various tasks set by the hunt's creator as well as compete for the highest score. There are five screens in total providing the various necessary features required to participate in a hunt.

4.1.4.1 Navigation

Navigation between the screens is achieved via five bottom tabs, each representing each equivalent screen. Tapping on a tab opens the corresponding screen.

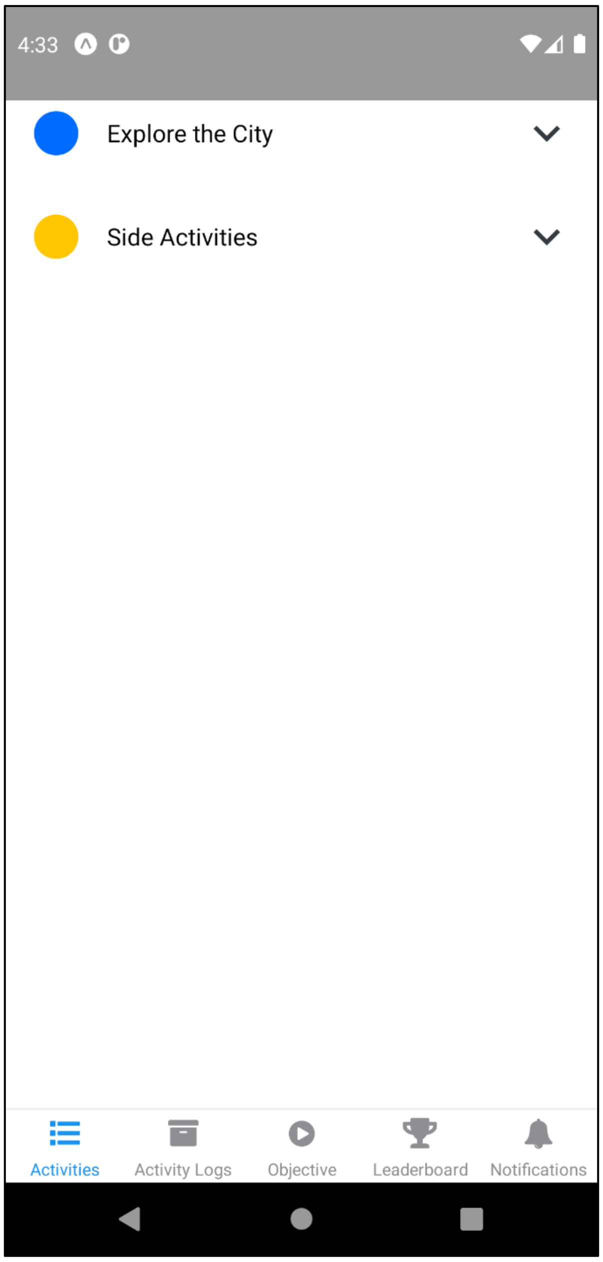


Figure 21 Game Screens Overview

4.1.4.2 Activities Screen

The activities screen offers an overview of the activities and their status. As showcased in (FIGURE), activity groups are represented by an icon of their color next to their name, as well as a collapse/expand button allowing quick navigation between them. Activities are represented by their type as an icon, followed by their title, and finally by a status indicator indicating whether they have been completed. Activities not yet accessible have their icon and title hidden, while optional activities have a subheading mentioning they are optional.

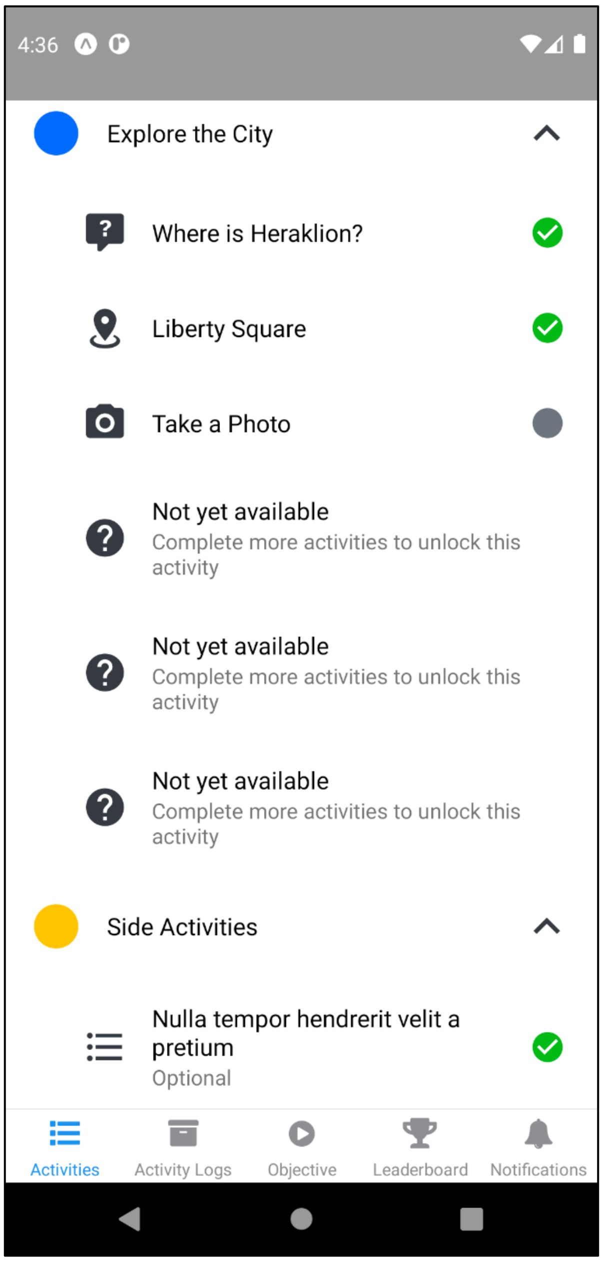
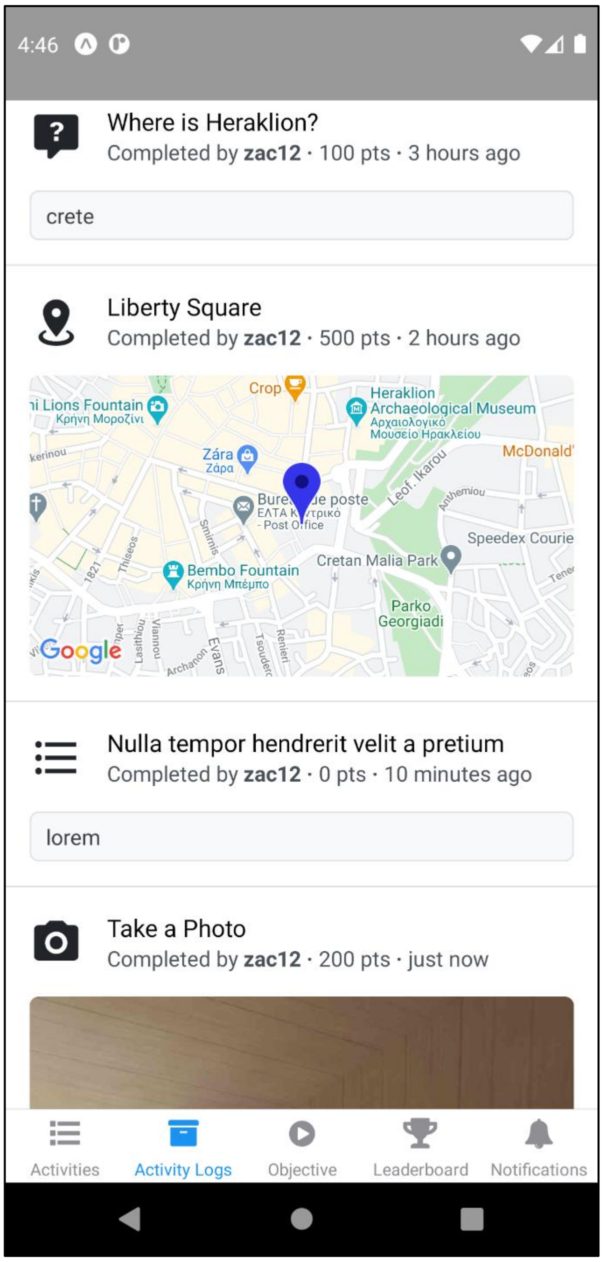


Figure 22 The Activities Screen

4.1.4.3 Activity Logs Screen

This screen displays a list of all activities a user, or their team has completed. Each item on the list displays a little information about the activity as well as the answer (when applicable). Tapping on an item will display all of the activity's information as well as the answer.


4:46

Where is Heraklion?
Completed by **zac12** · 100 pts · 3 hours ago

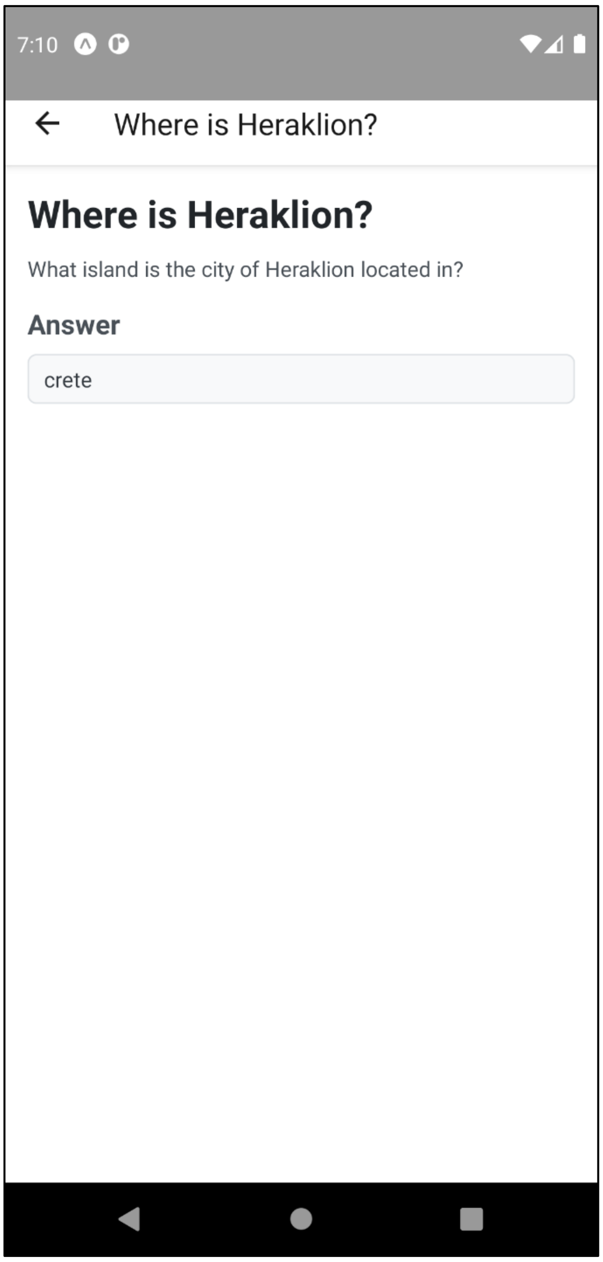
crete

Liberty Square
Completed by **zac12** · 500 pts · 2 hours ago

Nulla tempor hendrerit velit a pretium
Completed by **zac12** · 0 pts · 10 minutes ago

lorem

Take a Photo
Completed by **zac12** · 200 pts · just now


7:10

Where is Heraklion?

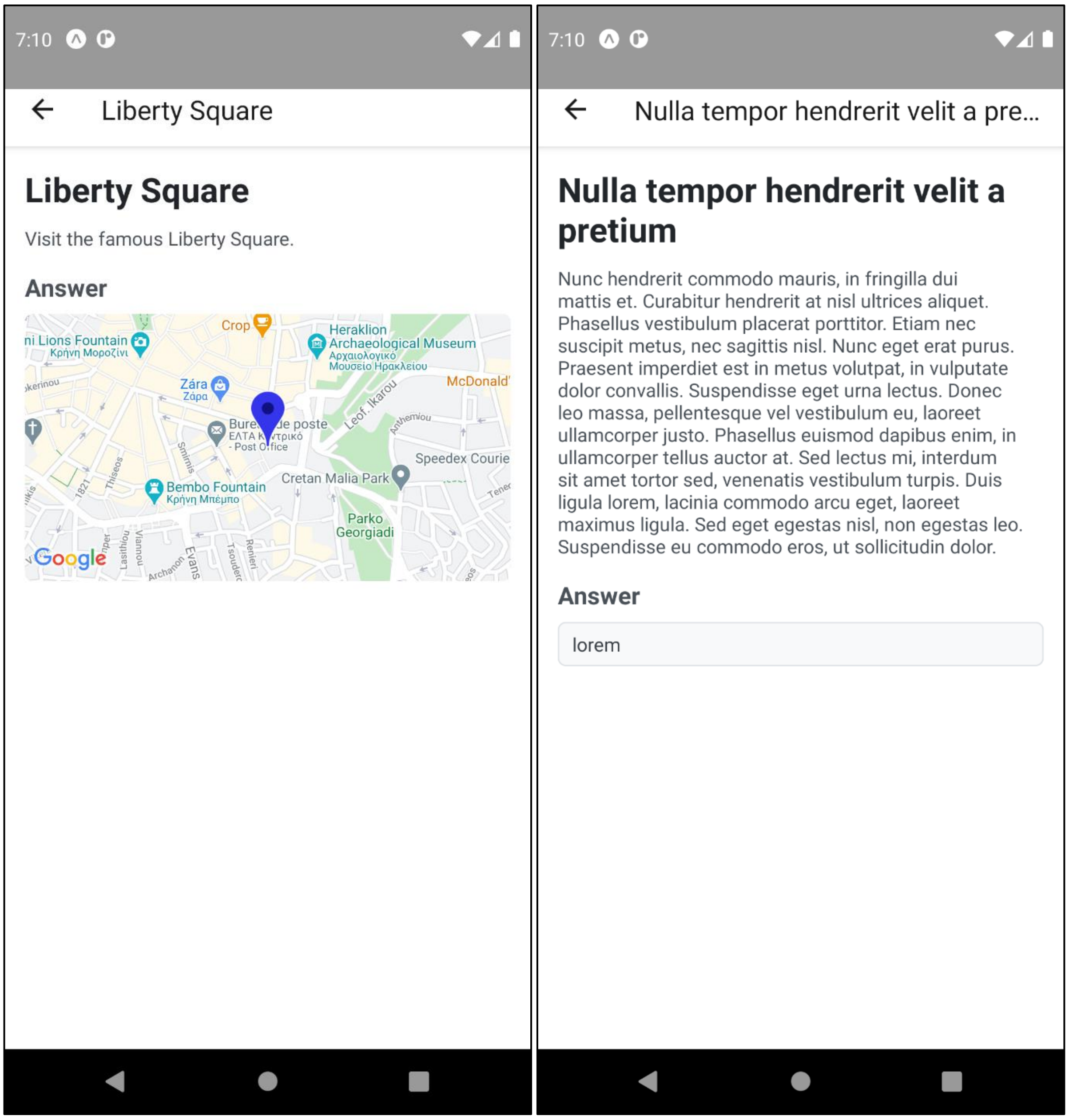
Where is Heraklion?

What island is the city of Heraklion located in?

Answer

crete

Figure 23 The Activity Logs Screen, and Completed Activity Details Screen for a Question Activity



Liberty Square

Visit the famous Liberty Square.

Answer

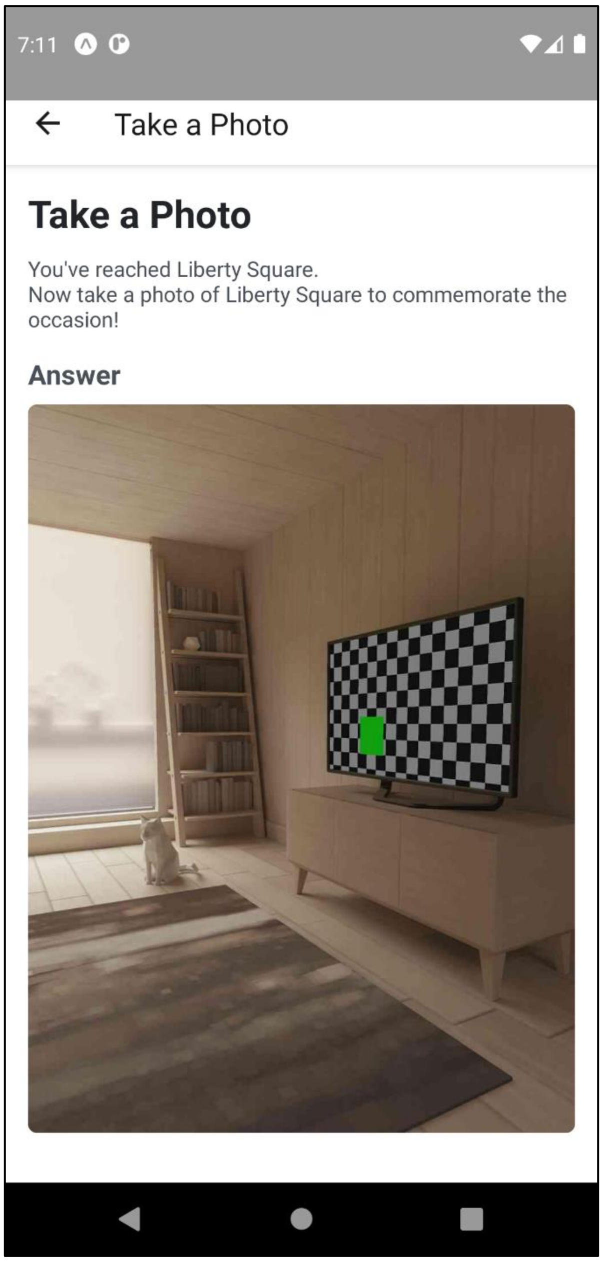
Nulla tempor hendrerit velit a pretium

Nunc hendrerit commodo mauris, in fringilla dui mattis et. Curabitur hendrerit at nisl ultrices aliquet. Phasellus vestibulum placerat porttitor. Etiam nec suscipit metus, nec sagittis nisl. Nunc eget erat purus. Praesent imperdiet est in metus volutpat, in vulputate dolor convallis. Suspendisse eget urna lectus. Donec leo massa, pellentesque vel vestibulum eu, laoreet ullamcorper justo. Phasellus euismod dapibus enim, in ullamcorper tellus auctor at. Sed lectus mi, interdum sit amet tortor sed, venenatis vestibulum turpis. Duis ligula lorem, lacinia commodo arcu eget, laoreet maximus ligula. Sed eget egestas nisl, non egestas leo. Suspendisse eu commodo eros, ut sollicitudin dolor.

Answer

lorem

Figure 24 The Completed Activity Details Screen for a GPS and a Multiple Choice Activities



Take a Photo

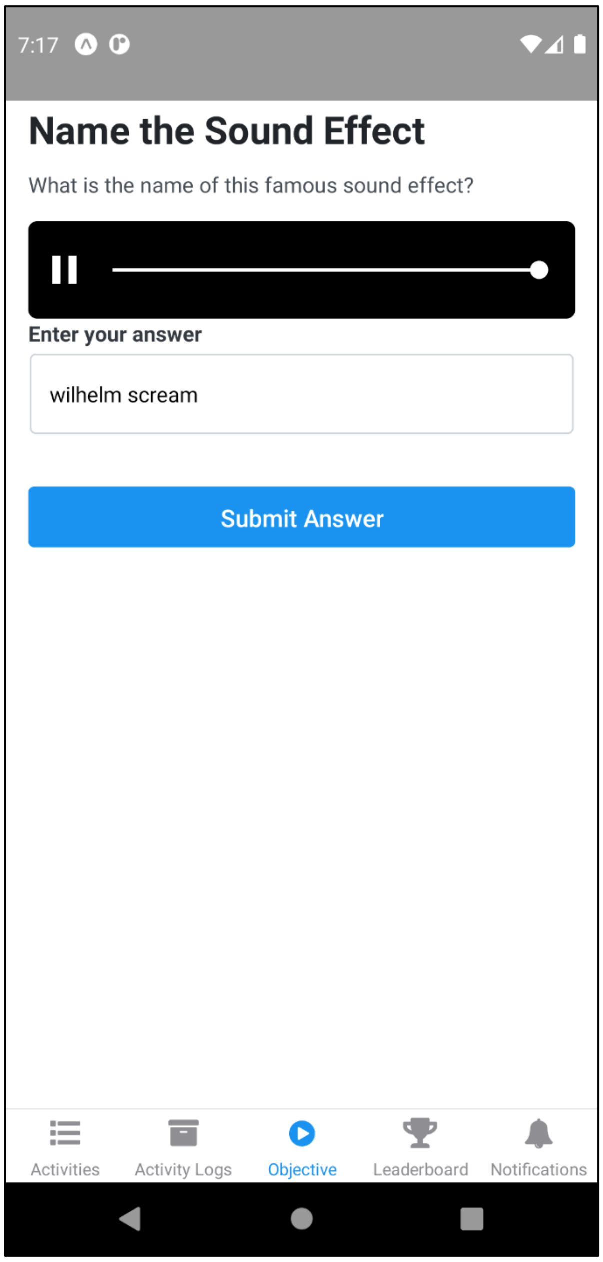
You've reached Liberty Square.
Now take a photo of Liberty Square to commemorate the occasion!

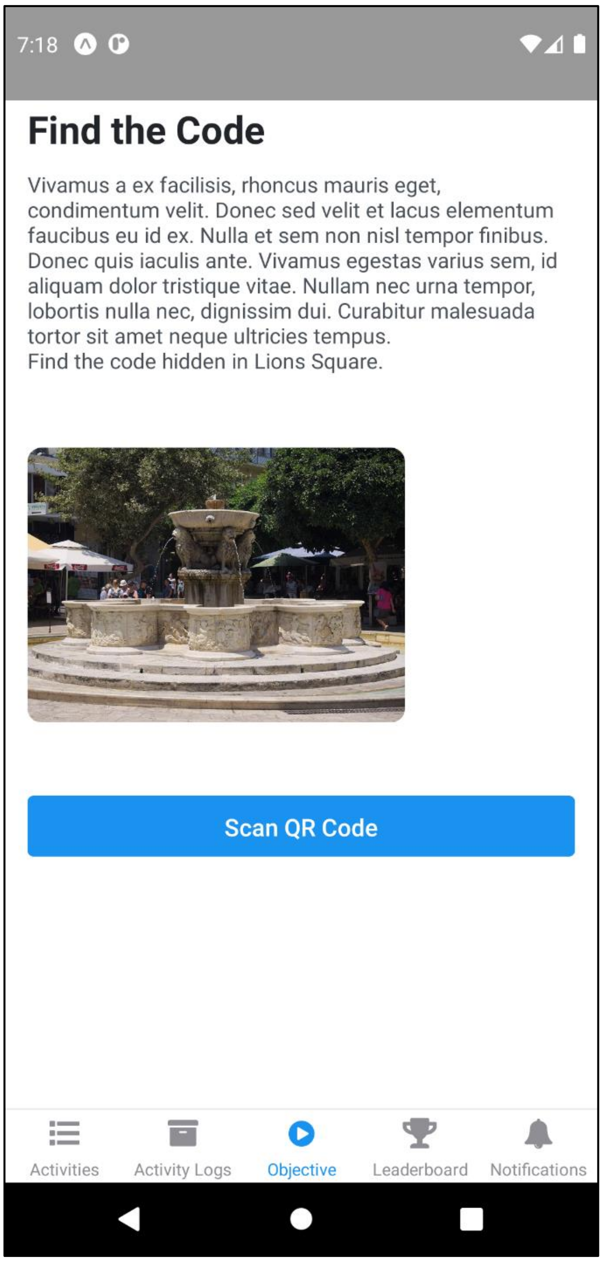
Answer

Figure 25 The Completed Activity Details Screen for a Photo Activity

4.1.4.4 Objective Screen

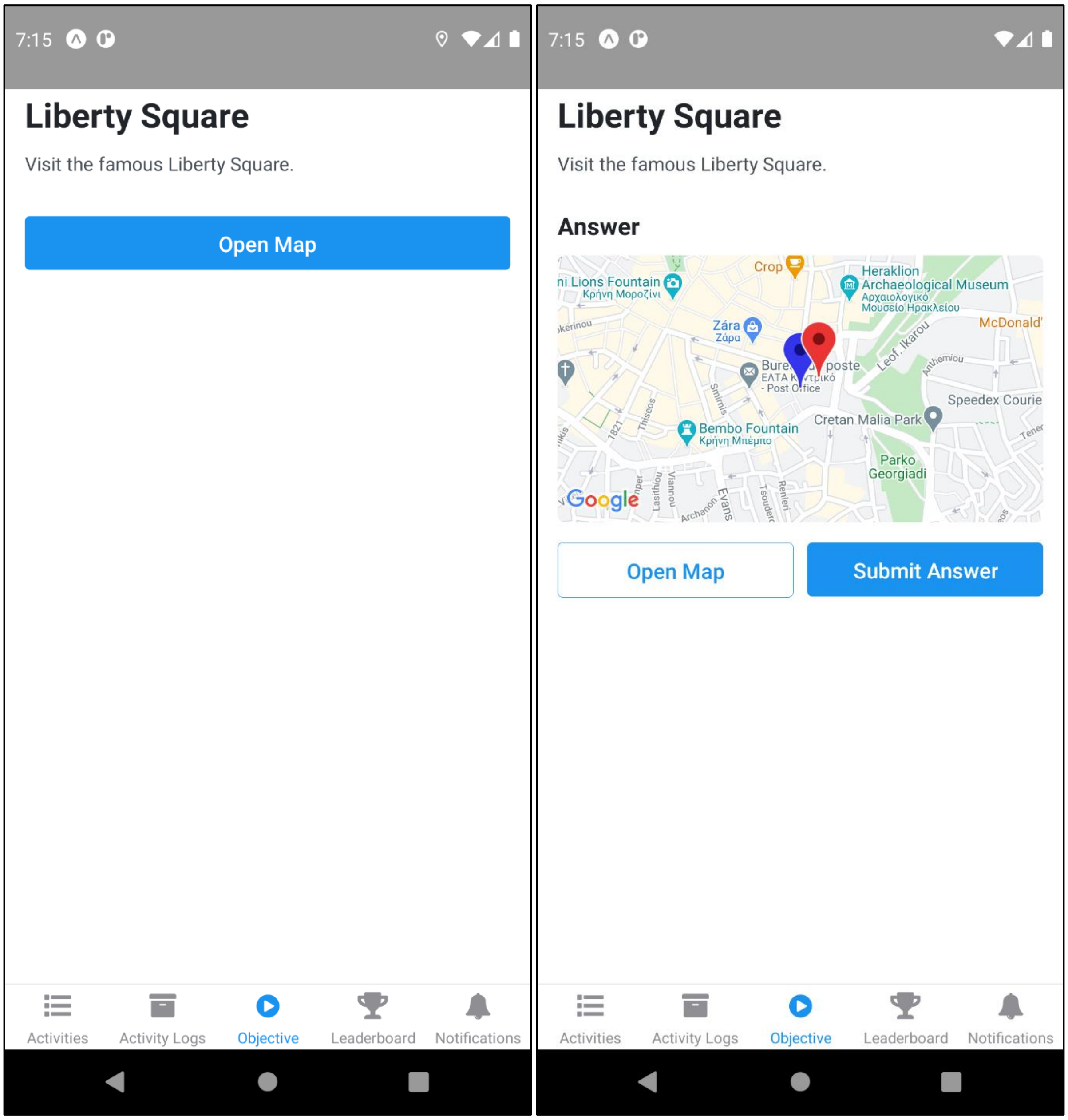
The objective screen is the most important of the game screens. It is through this screen that the players can complete activities. The objective screen features the details of the activity as well as a unique objective for each of the five activity types.





Vivamus a ex facilisis, rhoncus mauris eget, condimentum velit. Donec sed velit et lacus elementum faucibus eu id ex. Nulla et sem non nisl tempor finibus. Donec quis iaculis ante. Vivamus egestas varius sem, id aliquam dolor tristique vitae. Nullam nec urna tempor, lobortis nulla nec, dignissim dui. Curabitur malesuada tortor sit amet neque ultricies tempus.
Find the code hidden in Lions Square.

Figure 26 The Objective Screen for a Question and a QR Scan Activities


Liberty Square

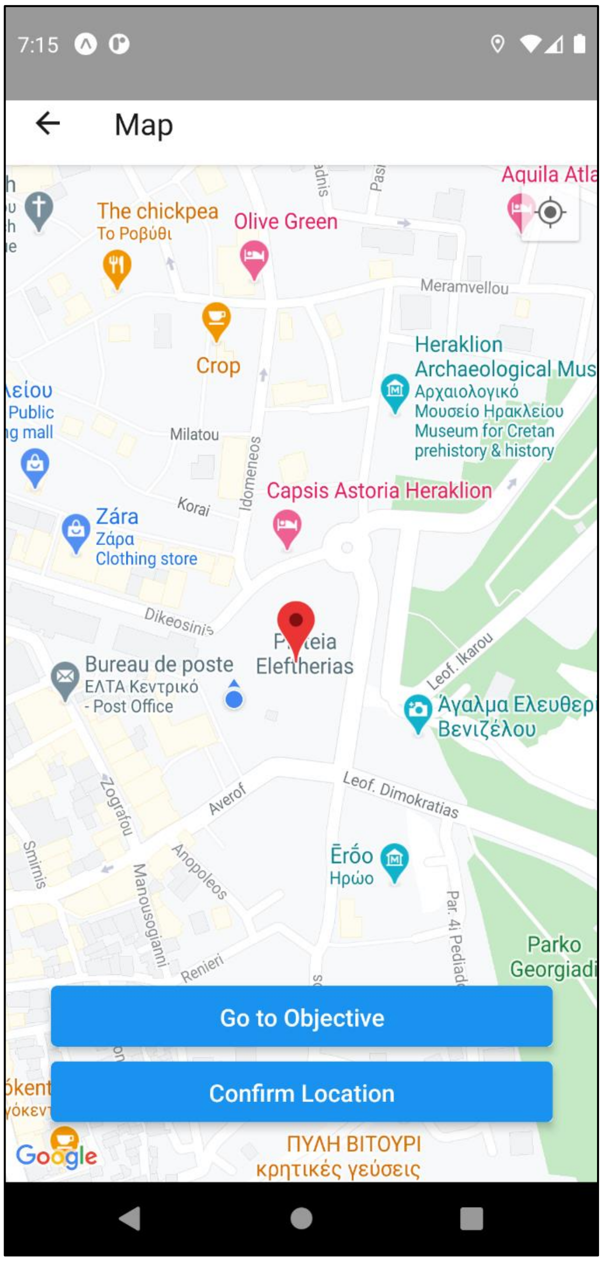
Visit the famous Liberty Square.

Liberty Square

Visit the famous Liberty Square.

Answer

Figure 27 The Objective Screen for a GPS Activity before and after coordinates have been selected



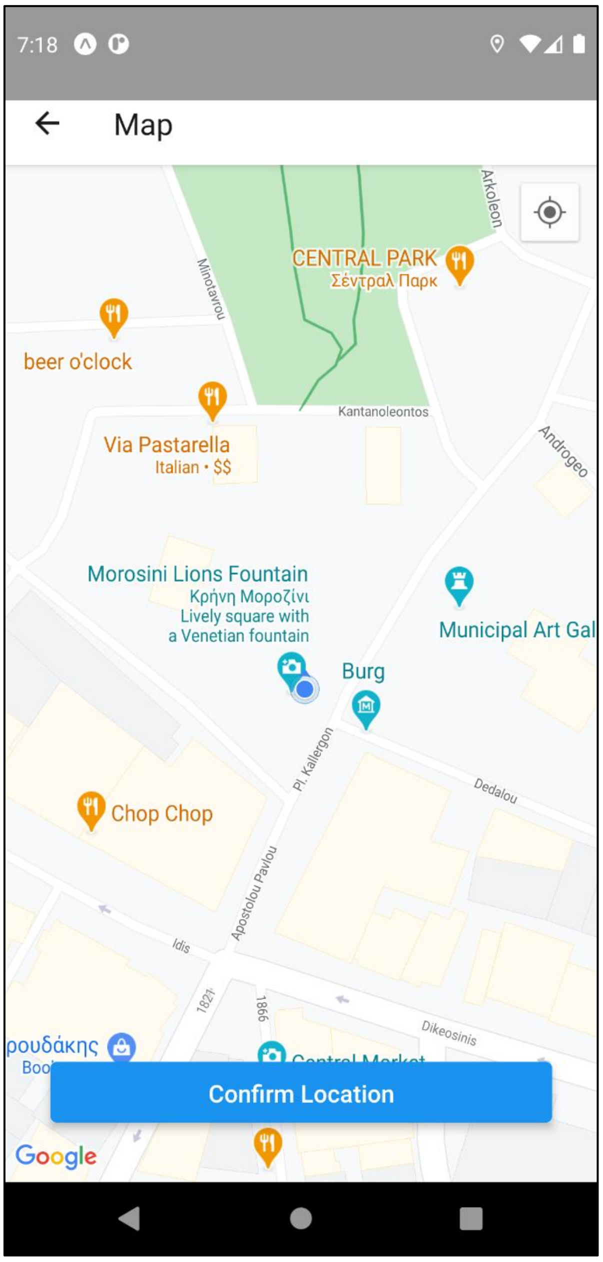
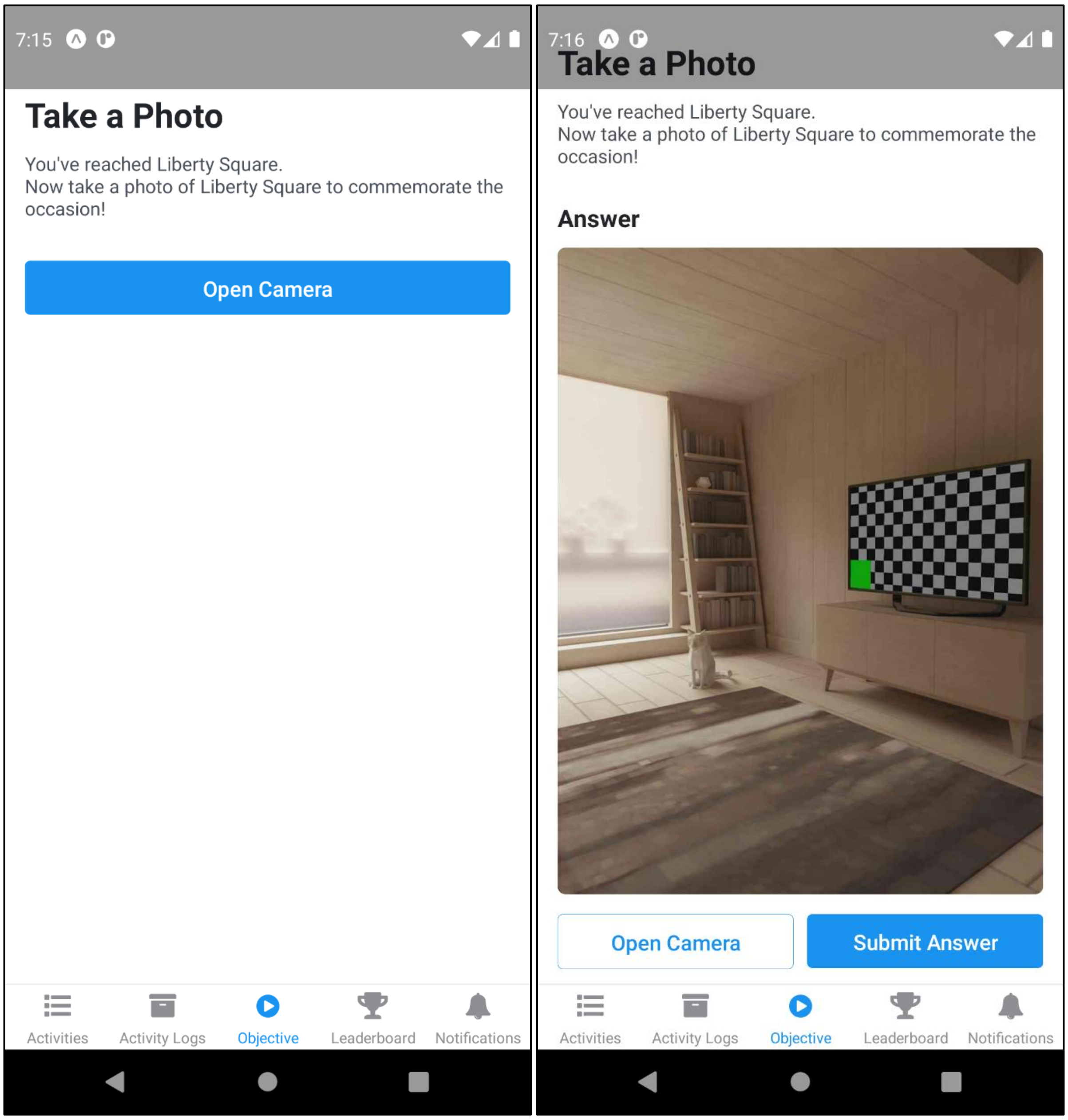


Figure 28 The Map Screen for a GPS Activity with the objective visible and hidden


7:15

Take a Photo

You've reached Liberty Square.
Now take a photo of Liberty Square to commemorate the occasion!

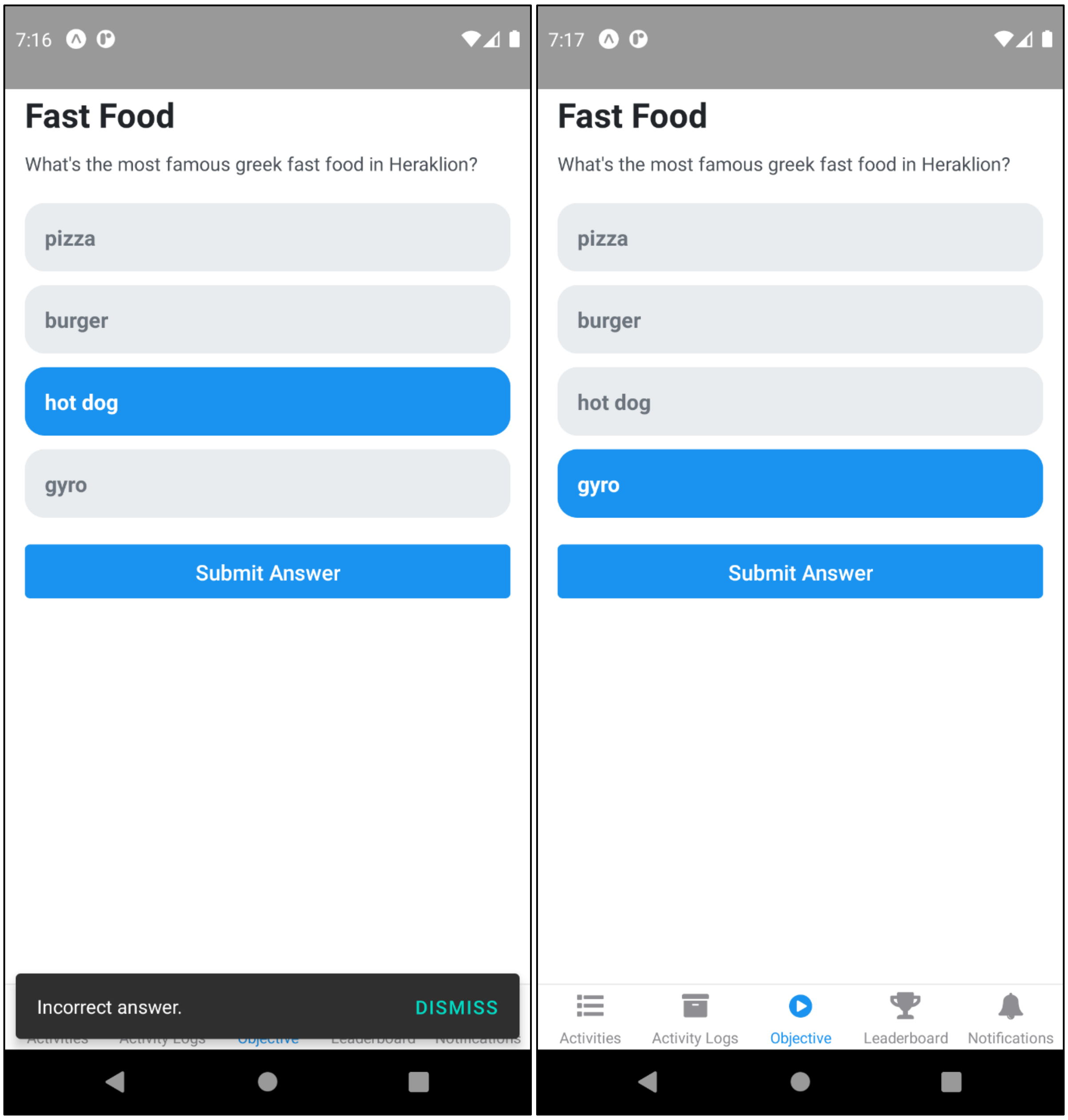
7:16

Take a Photo

You've reached Liberty Square.
Now take a photo of Liberty Square to commemorate the occasion!

Answer

Figure 29 The Objective Screen for a Photo Activity before and after the photo has been taken


7:16

Fast Food

What's the most famous greek fast food in Heraklion?

pizza

burger

hot dog

gyro

Submit Answer

Incorrect answer. DISMISS

7:17

Fast Food

What's the most famous greek fast food in Heraklion?

pizza

burger

hot dog

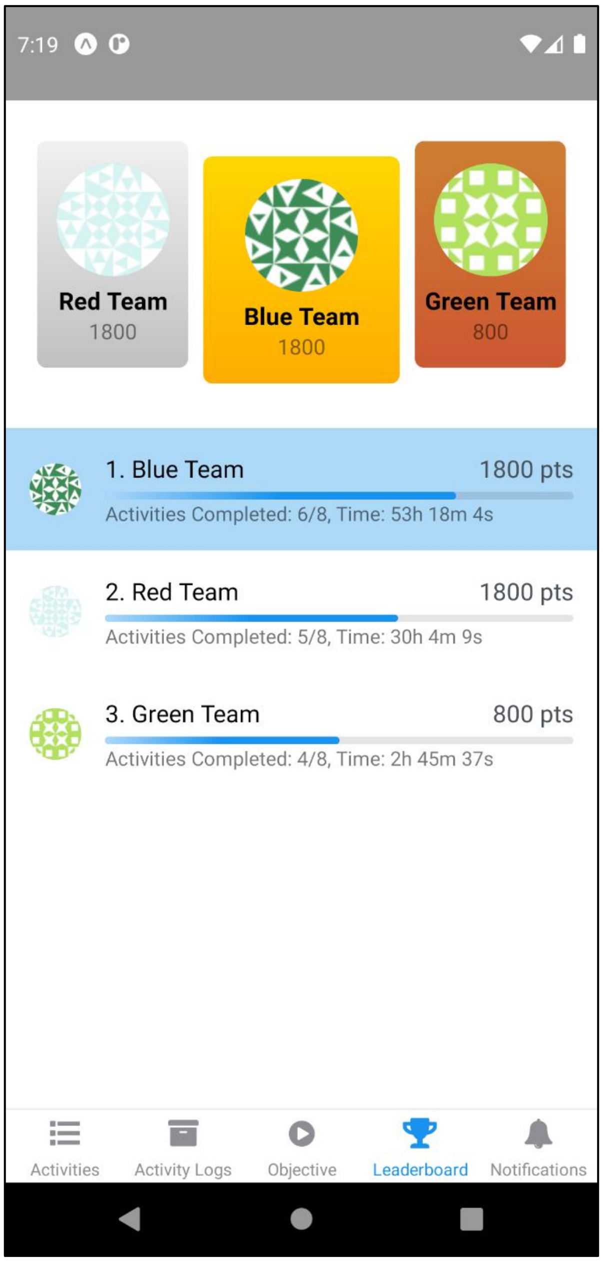
gyro

Submit Answer

Figure 30 The Objective Screen for a Multiple Choice Activity

4.1.4.5 Leaderboard Screen

The leaderboard screen is used to keep track of well each player/team is doing in the hunt. It features the top three players/teams at the top, as well as a more detailed view of each participant below.


Red Team
1800
Blue Team
1800
Green Team
800

1. Blue Team 1800 pts
Activities Completed: 6/8, Time: 53h 18m 4s

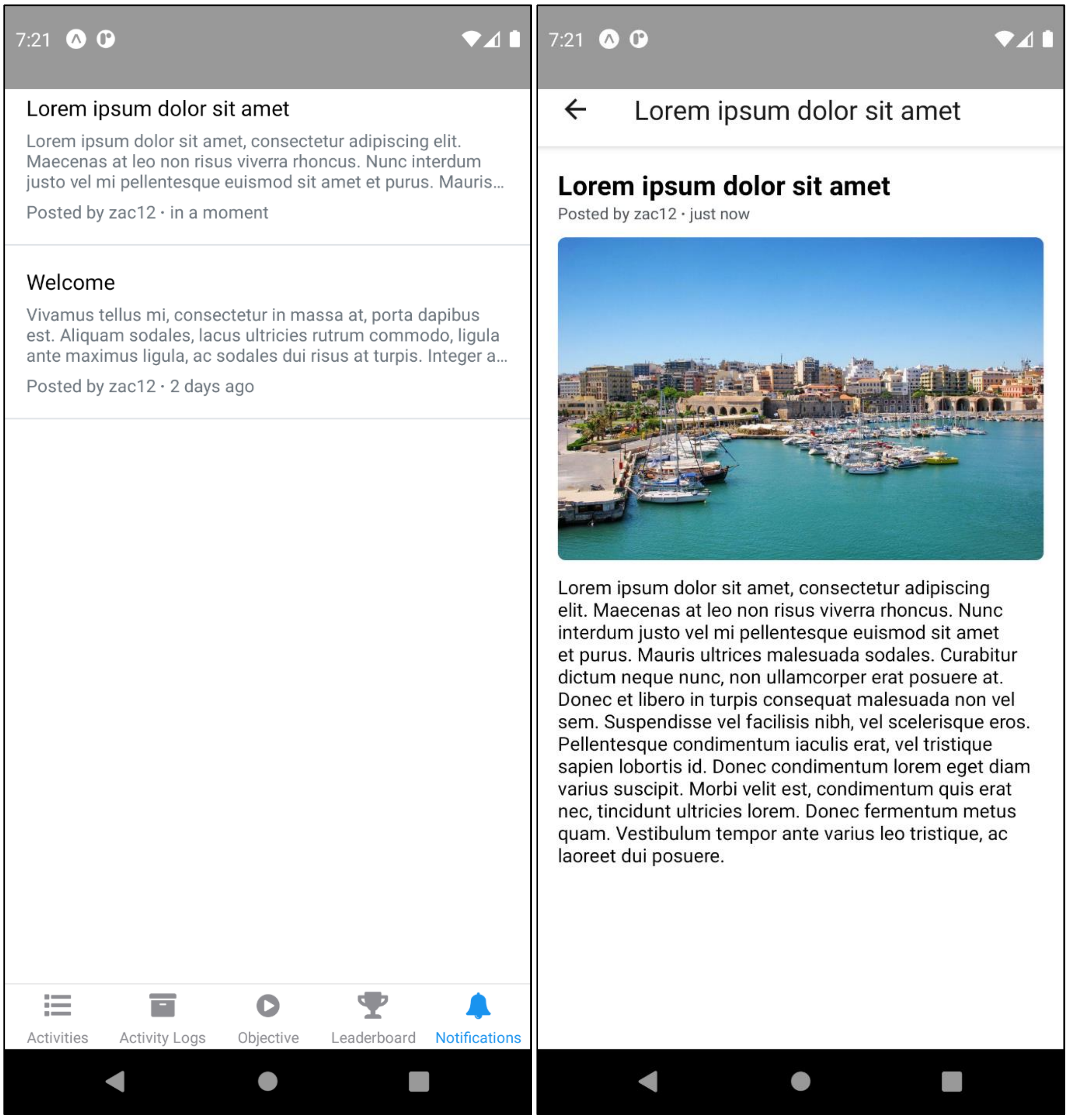
2. Red Team 1800 pts
Activities Completed: 5/8, Time: 30h 4m 9s

3. Green Team 800 pts
Activities Completed: 4/8, Time: 2h 45m 37s

Figure 31 The Leaderboard Screen

4.1.4.6 Notifications Screen

The notifications screen displays all of a hunt's notifications. When opened it features a list of all notifications sorted by time. When a notification is tapped it displays the full message along with its image (if available).


7:21

Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas at leo non risus viverra rhoncus. Nunc interdum justo vel mi pellentesque euismod sit amet et purus. Mauris…

Posted by zac12 • in a moment

Welcome

Vivamus tellus mi, consectetur in massa at, porta dapibus est. Aliquam sodales, lacus ultricies rutrum commodo, ligula ante maximus ligula, ac sodales dui risus at turpis. Integer a…

Posted by zac12 • 2 days ago

7:21

Lorem ipsum dolor sit amet

Lorem ipsum dolor sit amet

Posted by zac12 • just now

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas at leo non risus viverra rhoncus. Nunc interdum justo vel mi pellentesque euismod sit amet et purus. Mauris ultrices malesuada sodales. Curabitur dictum neque nunc, non ullamcorper erat posuere at. Donec et libero in turpis consequat malesuada non vel sem. Suspendisse vel facilisis nibh, vel scelerisque eros. Pellentesque condimentum iaculis erat, vel tristique sapien lobortis id. Donec condimentum lorem eget diam varius suscipit. Morbi velit est, condimentum quis erat nec, tincidunt ultricies lorem. Donec fermentum metus quam. Vestibulum tempor ante varius leo tristique, ac laoreet dui posuere.

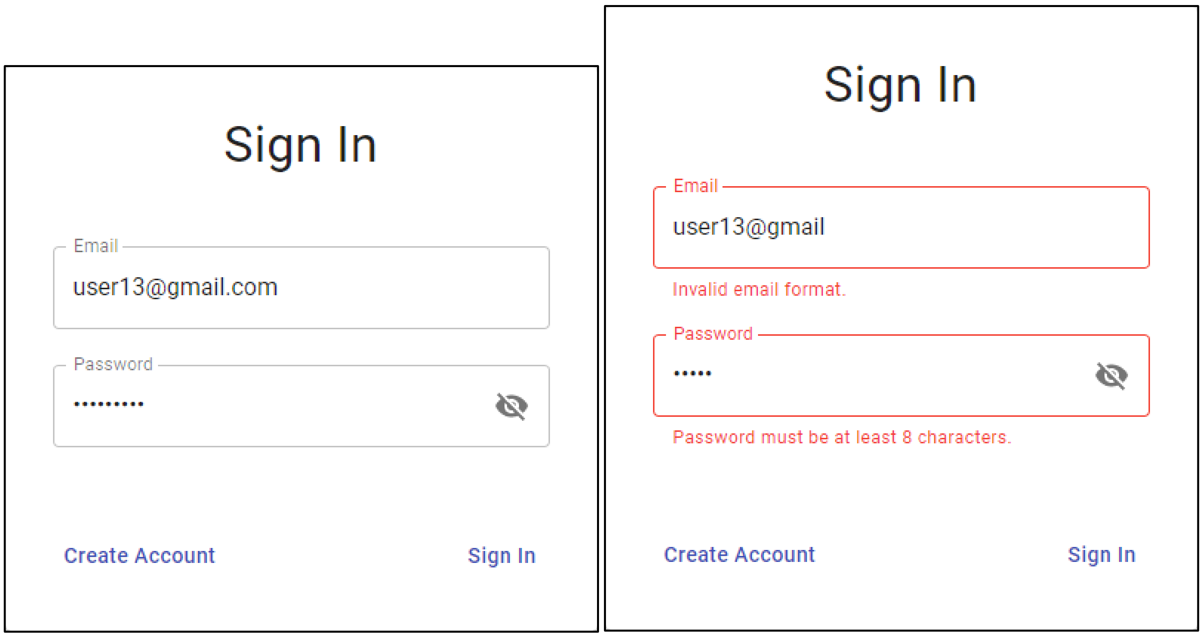
Figure 32 The Notifications and Notification Details Screens

4.2 Web Application

The web application was designed around hunt creation and management. As such we designed a robust hunt management system, giving the user all the necessary tools to bring their ideal hunt to life. The user interface for the website is focused on utility, following the latest user experience trends.

4.1.1 Authentication Pages

The web application has three authentication pages. The sign-up, sign-in, profile pages. Similar forms were used for all three pages. Each form features error checking and feedback as well as a button to show the password input. When first adding and editing a password, the user is asked to confirm it.



Sign In

Email: user13@gmail.com

Password: •••••••••

Create Account   Sign In

Sign In

Email: user13@gmail

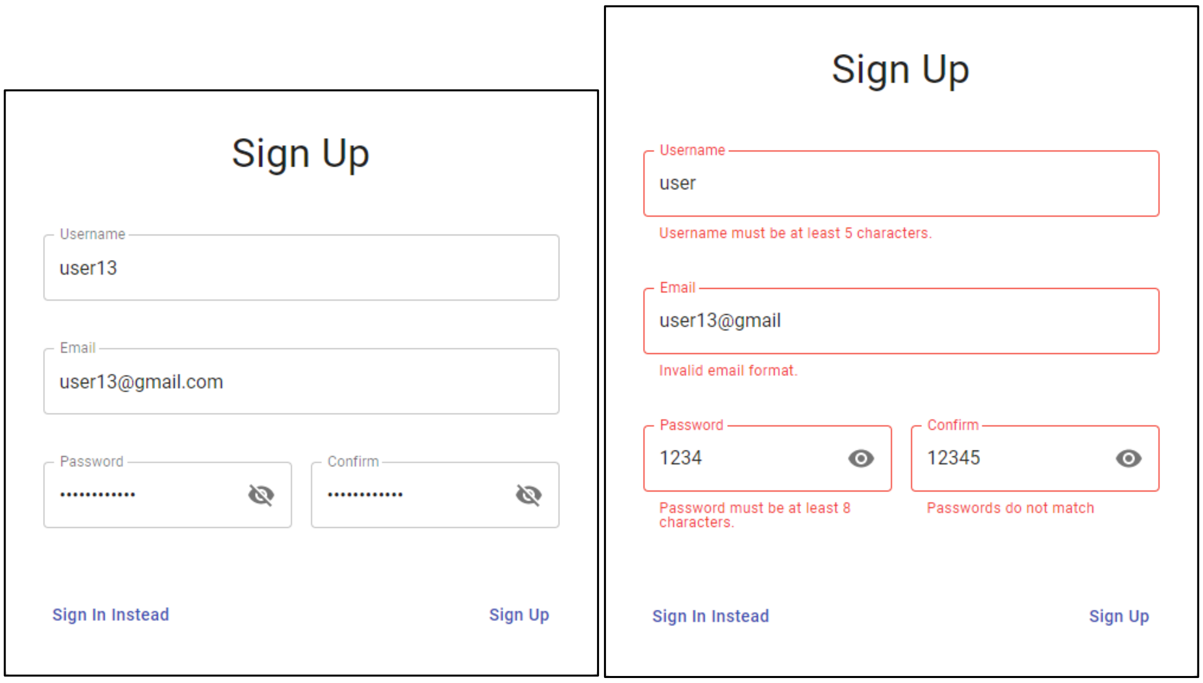
Invalid email format.

Password: •••••

Password must be at least 8 characters.

Create Account   Sign In

Figure 33 The Sign In Dialog



Sign Up

Username: user13

Email: user13@gmail.com

Password: ••••••••••••   Confirm: ••••••••••••

Sign In Instead   Sign Up

Sign Up

Username: user

Username must be at least 5 characters.

Email: user13@gmail

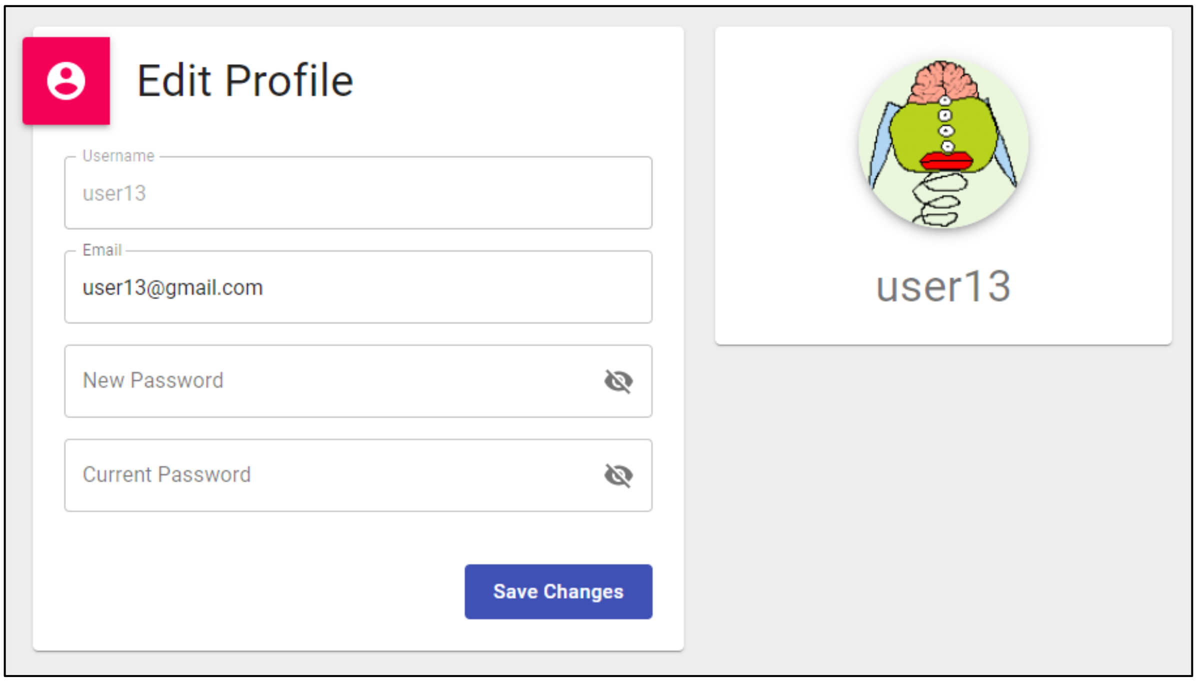
Invalid email format.

Password: 1234   Confirm: 12345

Password must be at least 8 characters.   Passwords do not match

Sign In Instead   Sign Up

Figure 34 The Sign-Up Dialog



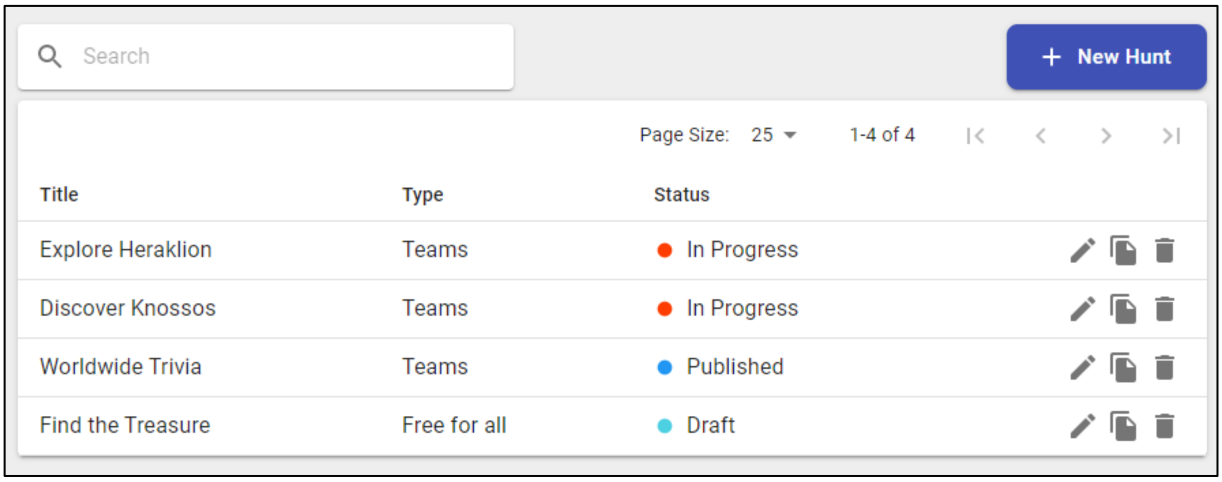
Edit Profile

Username	user13
Email	user13@gmail.com
New Password	
Current Password	

Figure 35 The User Profile Page

4.1.2 Hunts List Page

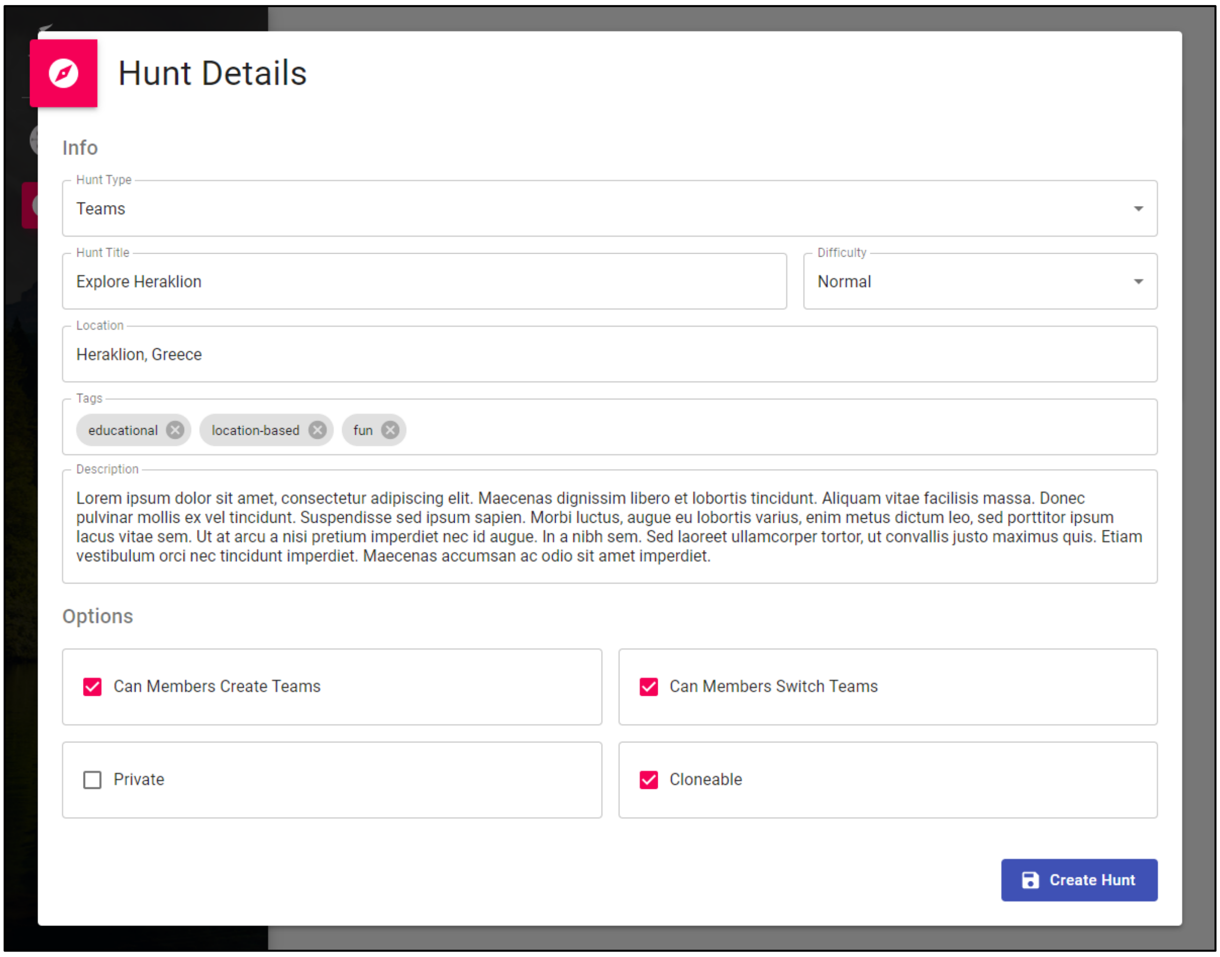
The hunts list page displays all the hunts a user has created, as well as a button allowing for the creation of new hunts. The list features search and pagination allowing for quick and seamless navigation. The list items feature basic information about the hunts, as well as three quick action buttons for editing, cloning, and deleting a hunt. When creating a new hunt a modal with a form pops up asking for the hunt's details.



Page Size: 25 1-4 of 4

Title	Type	Status
Explore Heraklion	Teams	● In Progress
Discover Knossos	Teams	● In Progress
Worldwide Trivia	Teams	● Published
Find the Treasure	Free for all	● Draft

Figure 36 The Hunts List Page


Hunt Details

Info

Hunt Type: Teams

Hunt Title: Explore Heraklion

Difficulty: Normal

Location: Heraklion, Greece

Tags: educational, location-based, fun

Description: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas dignissim libero et lobortis tincidunt. Aliquam vitae facilisis massa. Donec pulvinar mollis ex vel tincidunt. Suspendisse sed ipsum sapien. Morbi luctus, augue eu lobortis varius, enim metus dictum leo, sed porttitor ipsum lacus vitae sem. Ut at arcu a nisi pretium imperdiet nec id augue. In a nibh sem. Sed laoreet ullamcorper tortor, ut convallis justo maximus quis. Etiam vestibulum orci nec tincidunt imperdiet. Maecenas accumsan ac odio sit amet imperdiet.

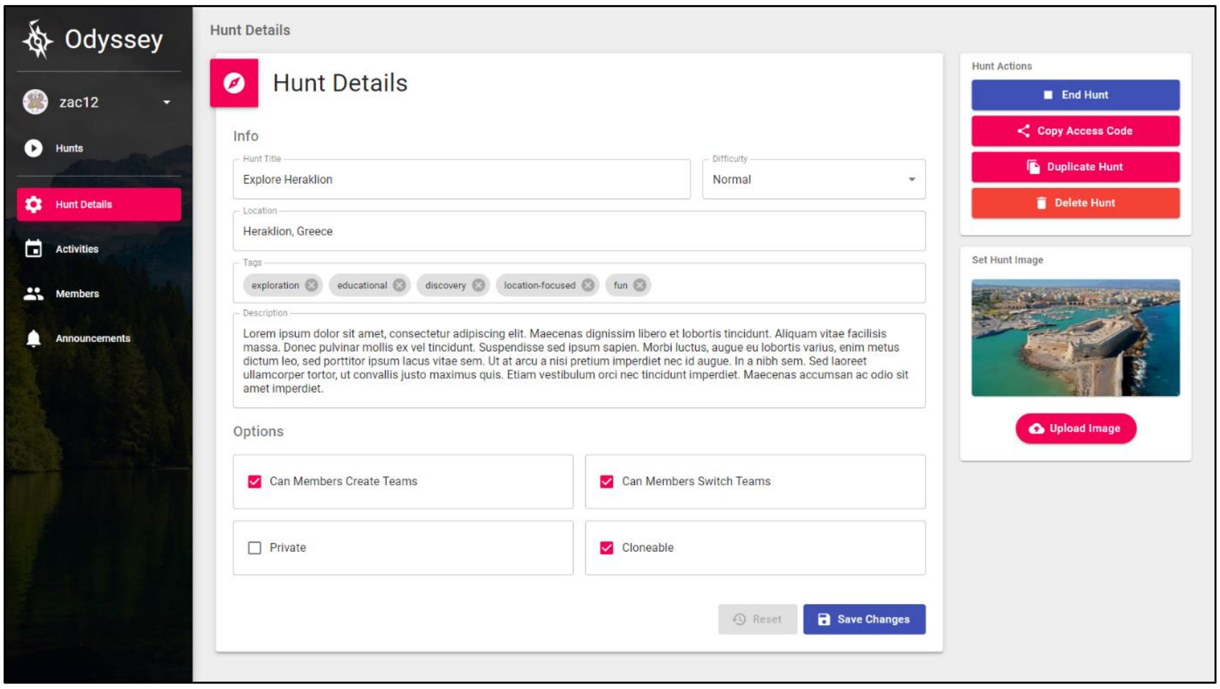
Options

- ✓ Can Members Create Teams
- ✓ Can Members Switch Teams
- ☐ Private
- ✓ Cloneable

Figure 37 The Hunt Creation Dialog

4.1.3 Hunt Management System

The hunt management system or HMS for short is how a user can create hunts. It is designed to be seamless, robust, and intuitive so that the user can build their ideal hunt without having to deal with a clunky user interface. It features four pages with which the user can create and manage a hunt. The HMS itself consists of two segments: the navigation sidebar on the left and the active page on the right.



Hunt Details

Hunt Details

Info

Hunt Title: Explore Heraklion

Difficulty: Normal

Location: Heraklion, Greece

Tags: exploration, educational, discovery, location-focused, fun

Description: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas dignissim libero et lobortis tincidunt. Aliquam vitae facilisis massa. Donec pulvinar mollis ex vel tincidunt. Suspendisse sed ipsum sapien. Morbi luctus, augue eu lobortis varius, enim metus dictum leo, sed porttitor ipsum lacus vitae sem. Ut at arcu a nisi pretium imperdiet nec id augue. In a nibh sem. Sed laoreet ullamcorper tortor, ut convallis justo maximus quis. Etiam vestibulum orci nec tincidunt imperdiet. Maecenas accumsan ac odio sit amet imperdiet.

Options

- ☑ Can Members Create Teams
- ☑ Can Members Switch Teams
- ☐ Private
- ☑ Cloneable

Figure 38 The Hunt Management Page

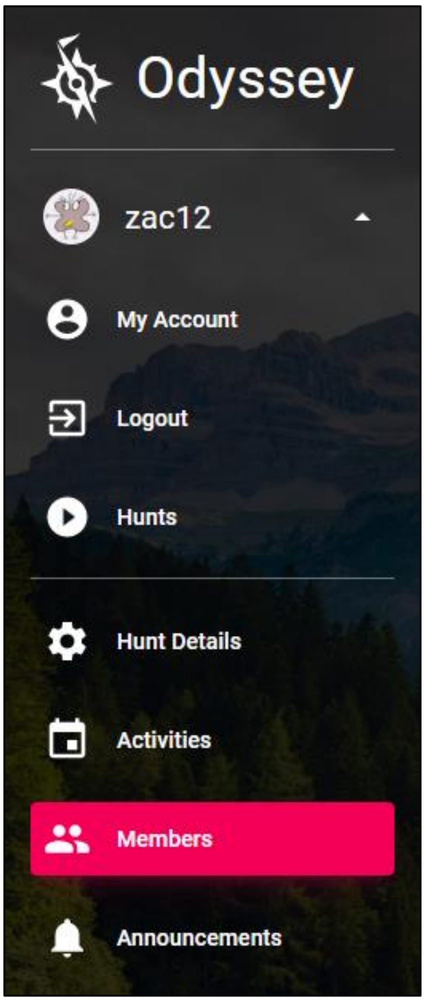
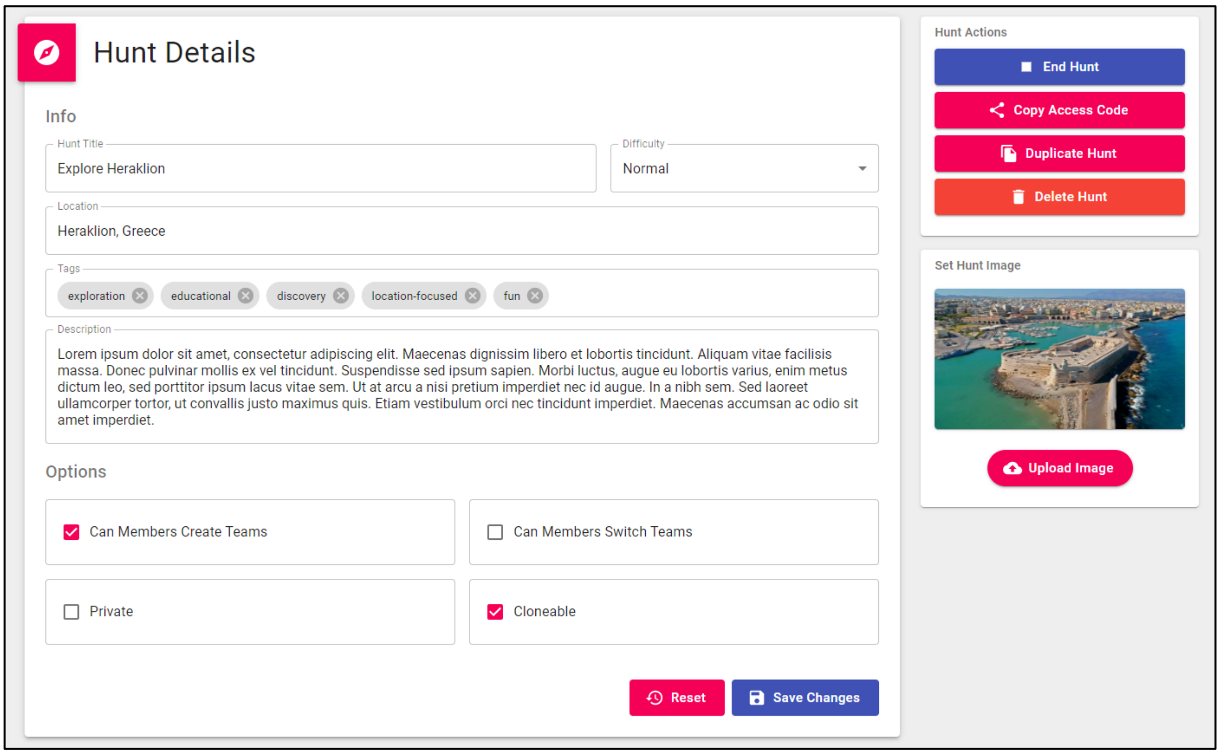


Figure 39 The Navigation Sidebar

4.1.3.1 Hunt Details Page

The hunt details page consists of three elements: The details form, which is used to edit the hunt's details and options, the image form, which enables the user to upload a hunt's featured image, and the hunt action buttons, which are used to perform various administrative actions to the hunt.


Hunt Details

Info

Hunt Title: Explore Heraklion

Difficulty: Normal

Location: Heraklion, Greece

Tags: exploration, educational, discovery, location-focused, fun

Description: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas dignissim libero et lobortis tincidunt. Aliquam vitae facilisis massa. Donec pulvinar mollis ex vel tincidunt. Suspendisse sed ipsum sapien. Morbi luctus, augue eu lobortis varius, enim metus dictum leo, sed porttitor ipsum lacus vitae sem. Ut at arcu a nisi pretium imperdiet nec id augue. In a nibh sem. Sed laoreet ullamcorper tortor, ut convallis justo maximus quis. Etiam vestibulum orci nec tincidunt imperdiet. Maecenas accumsan ac odio sit amet imperdiet.

Options

- [x] Can Members Create Teams
- [ ] Can Members Switch Teams
- [ ] Private
- [x] Cloneable

Figure 40 The Hunt Details Page

4.1.3.2 Activities Page

The activities page features a drag-and-drop interface that allows the user to create and manage activity groups and activities. Utilizing the drag-and-drop interface the user can arrange the order of the activity groups and the activities in them.

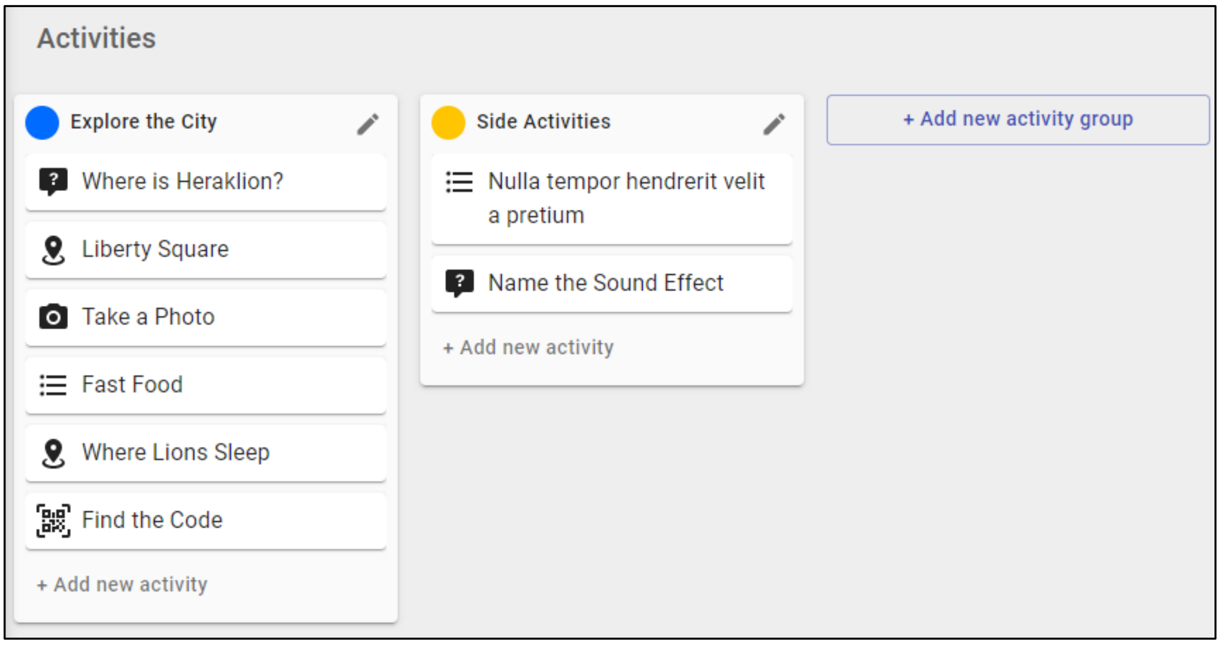
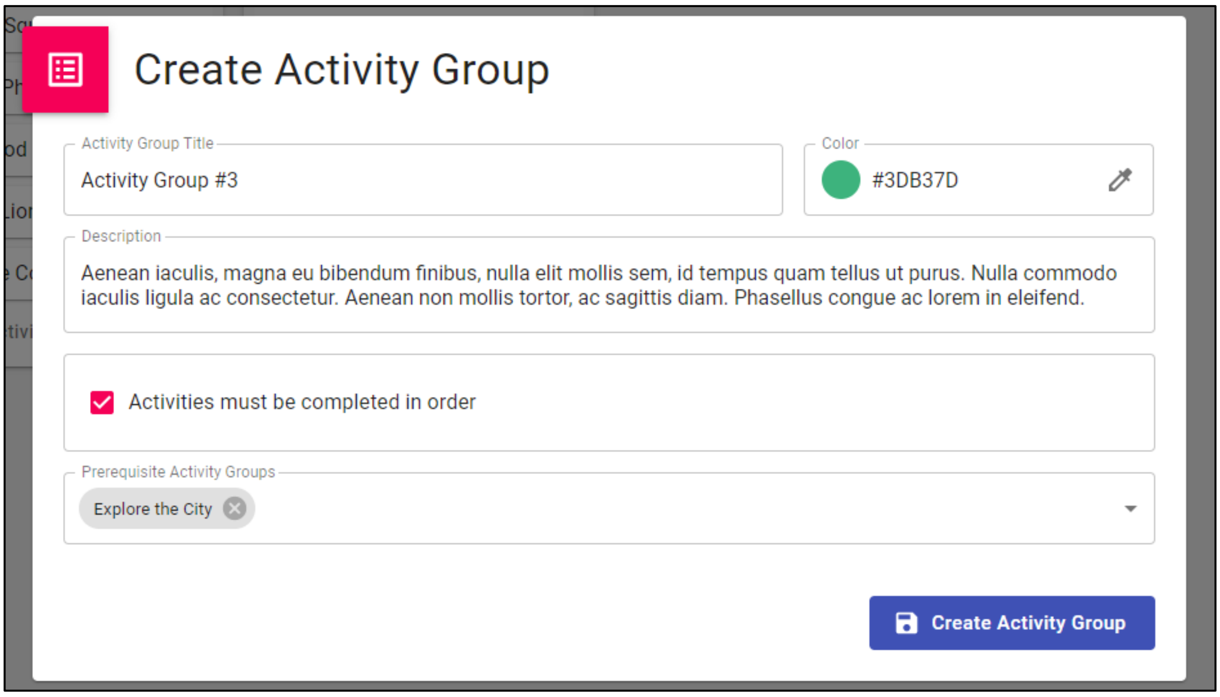


Figure 41 The Activities Page

4.1.3.2.1 Activity Group Management

When creating an activity group, the user is called to fill a form which asks for the group's title, color, description, whether the activities in that group will need to be completed in the order they are placed, and lastly whether the group requires any other groups to be completed before being unlocked. A color picker is present to allow precise color selection.

Both activity groups and activities support drag-and-drop. Dragging activity groups allows the user to change the order they appear in the mobile app. It is purely cosmetic and does not change the order activity groups have to be completed in. That is achieved through setting prerequisites. In the activities' case, the user can change also change the order by dragging which changes the order they appear inside the activity group. If the user has chosen to enable the "activities must be completed in order" option in the activity group, the order the activities have been arranged in will also be the order they need to be complete in. Lastly, the user can drag activities in between groups which change the group they belong to.



Create Activity Group

Activity Group Title: Activity Group #3

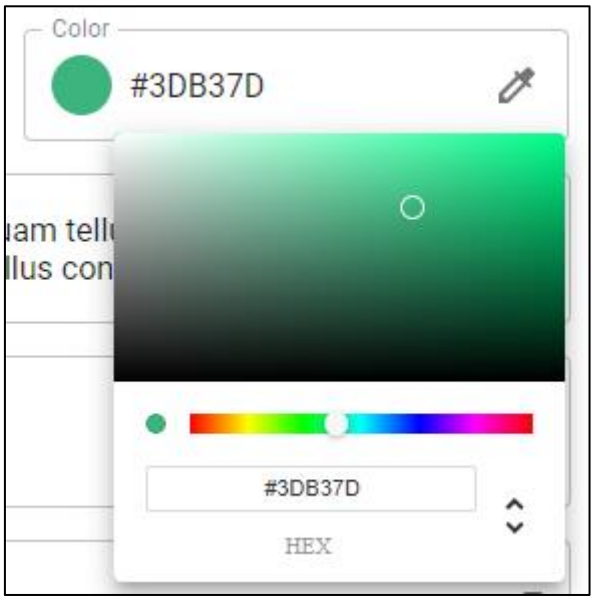
Color: #3DB37D

Description: Aenean iaculis, magna eu bibendum finibus, nulla elit mollis sem, id tempus quam tellus ut purus. Nulla commodo iaculis ligula ac consectetur. Aenean non mollis tortor, ac sagittis diam. Phasellus congue ac lorem in eleifend.

[✓] Activities must be completed in order

Prerequisite Activity Groups: Explore the City

Figure 42 The Activity Group Creation Dialog



Color: #3DB37D

#3DB37D
HEX

Figure 43 The Activity Group Color Picker

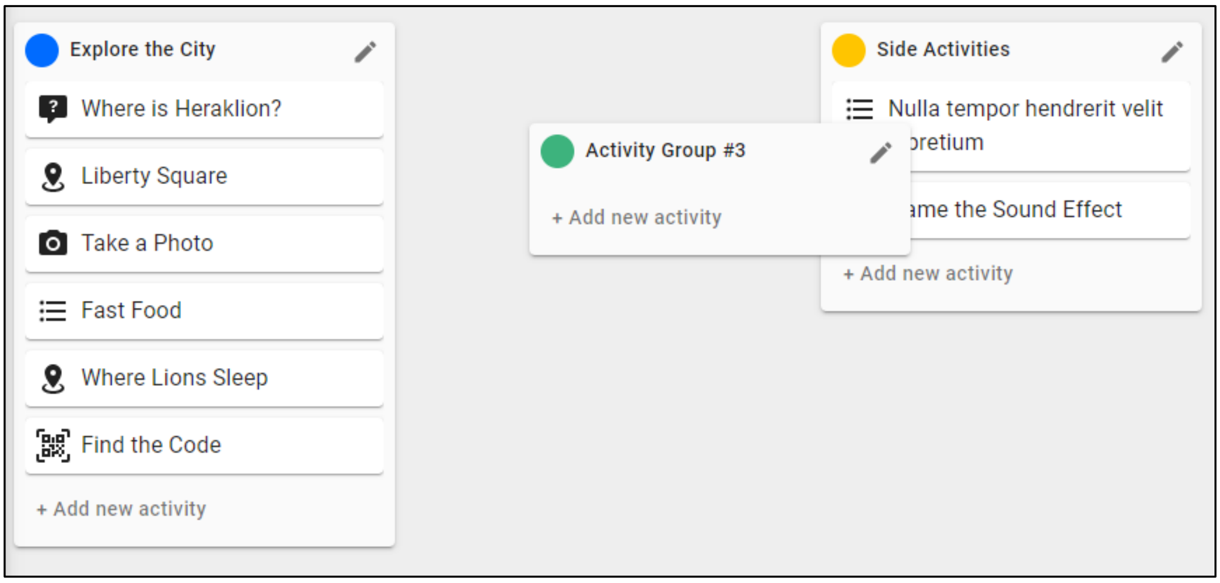


Figure 44 The Drag & Drop functionality for Activity Groups

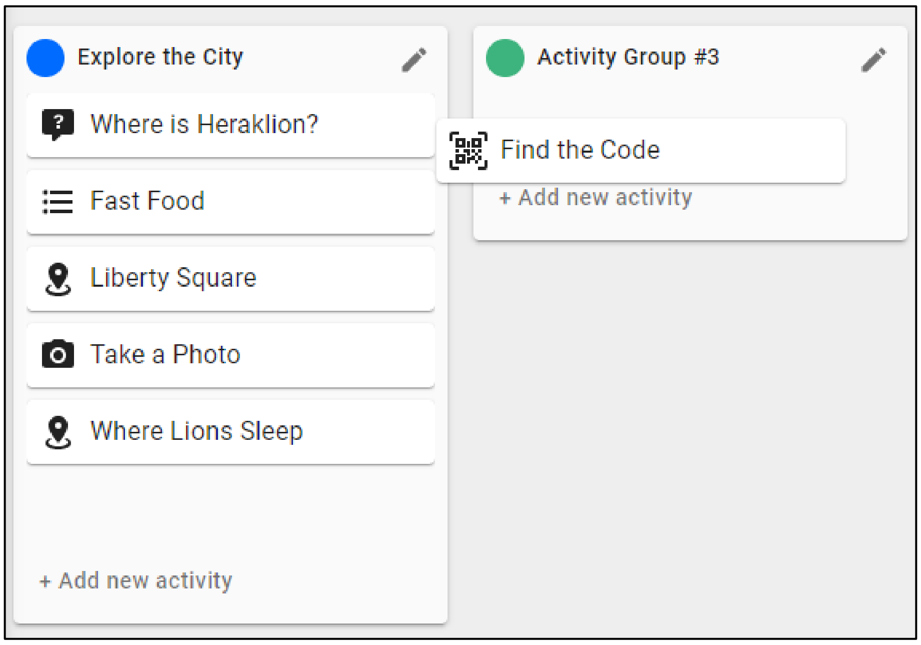
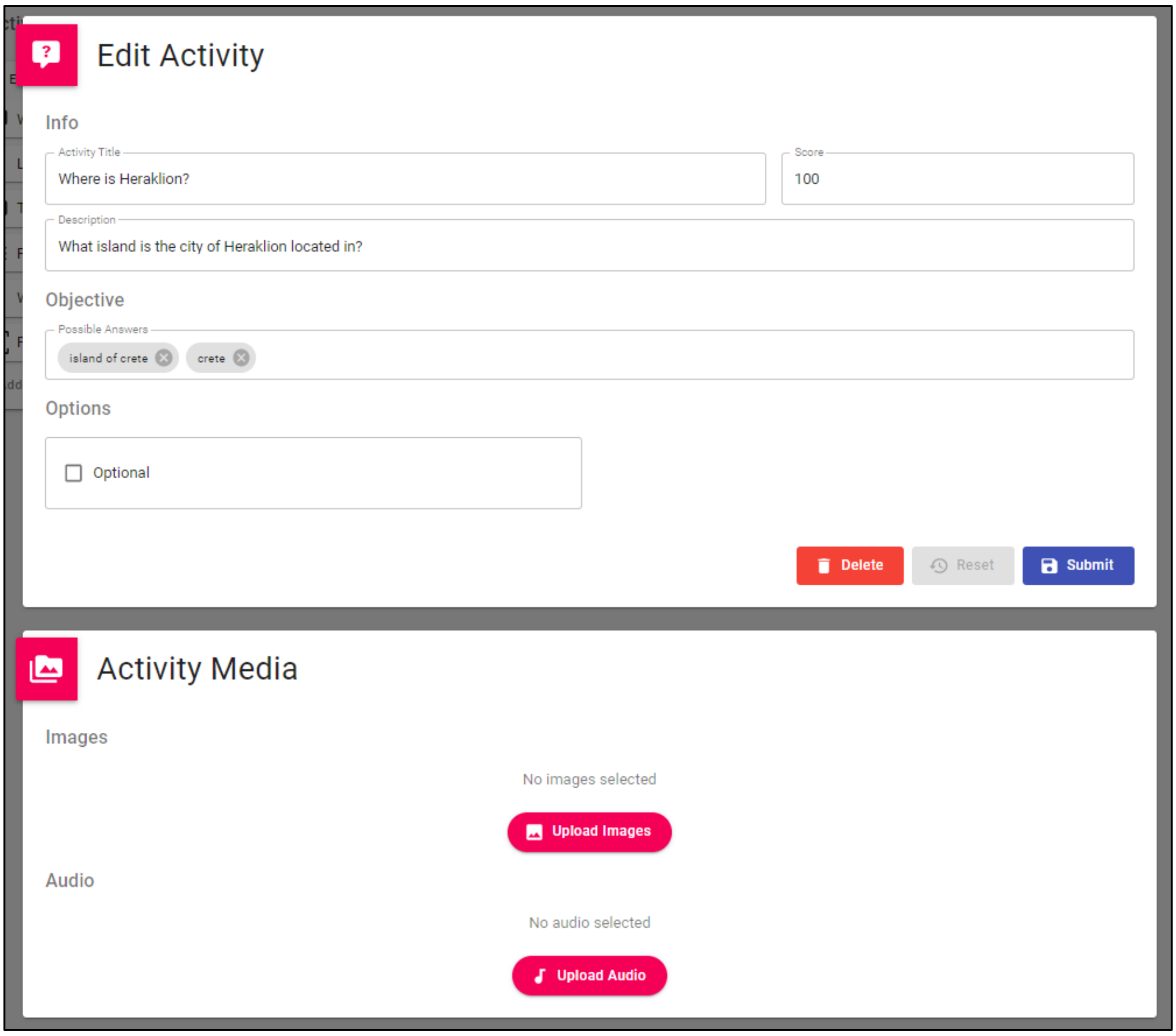


Figure 45 The Drag & Drop functionality for Activities

4.1.3.2.2 Activity Management

The creation and editing of activities, is done through a form. The form fields change depending on the type of activity the user has chosen to create/edit. The common fields of activities (title, score, description, media, and some options) stay the same while the objective changes.


Edit Activity

Info

Activity Title: Where is Heraklion?

Score: 100

Description: What island is the city of Heraklion located in?

Objective

Possible Answers: island of crete, crete

Options

Optional

Activity Media

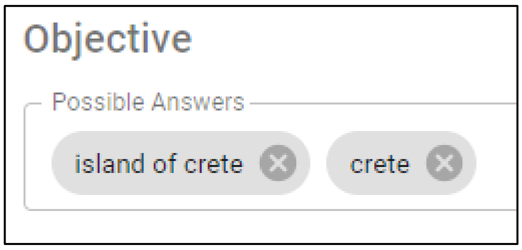
Images

No images selected

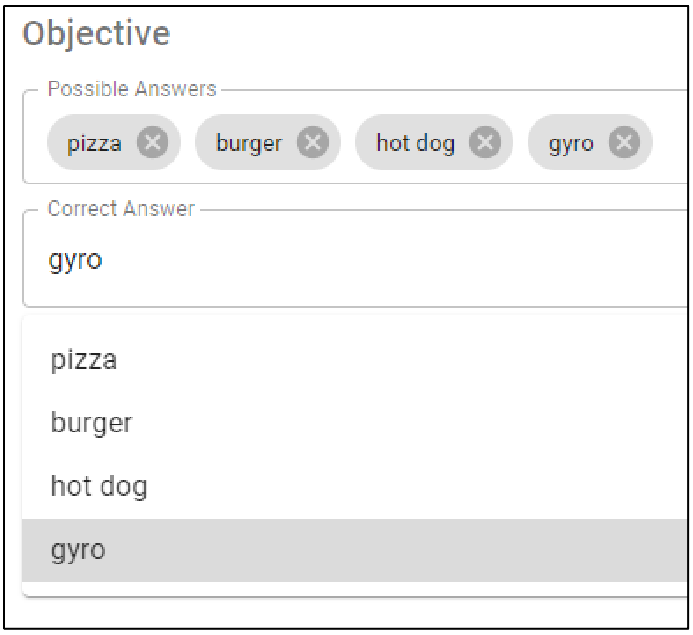
Audio

No audio selected

Figure 46 The Edit Activity Dialog


Objective

Possible Answers: island of crete, crete


Objective

Possible Answers: pizza, burger, hot dog, gyro

Correct Answer: gyro

Figure 47 The Objective Sections for Question and Multiple-Choice Activities

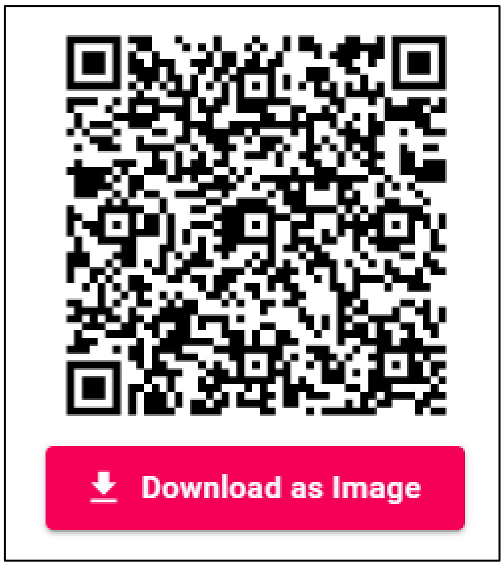
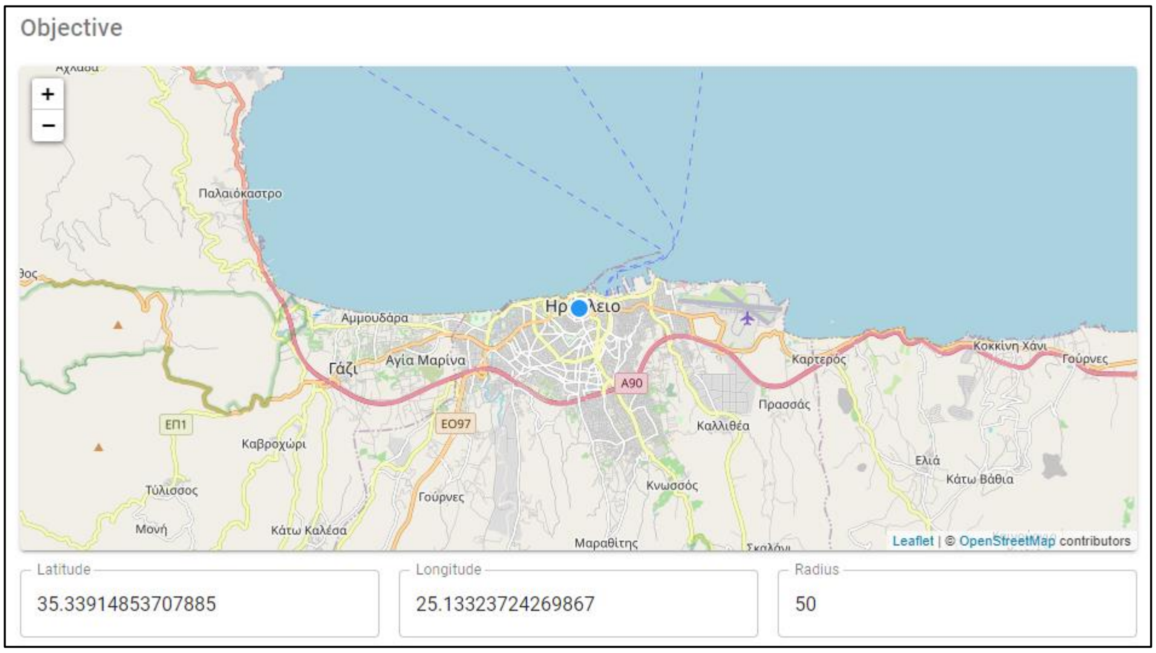


Figure 48 The Objective Section for QR Scan Activity



Latitude
35.33914853707885

Longitude
25.13323724269867

Radius
50

Figure 49 The Objective Section for GPS Activity

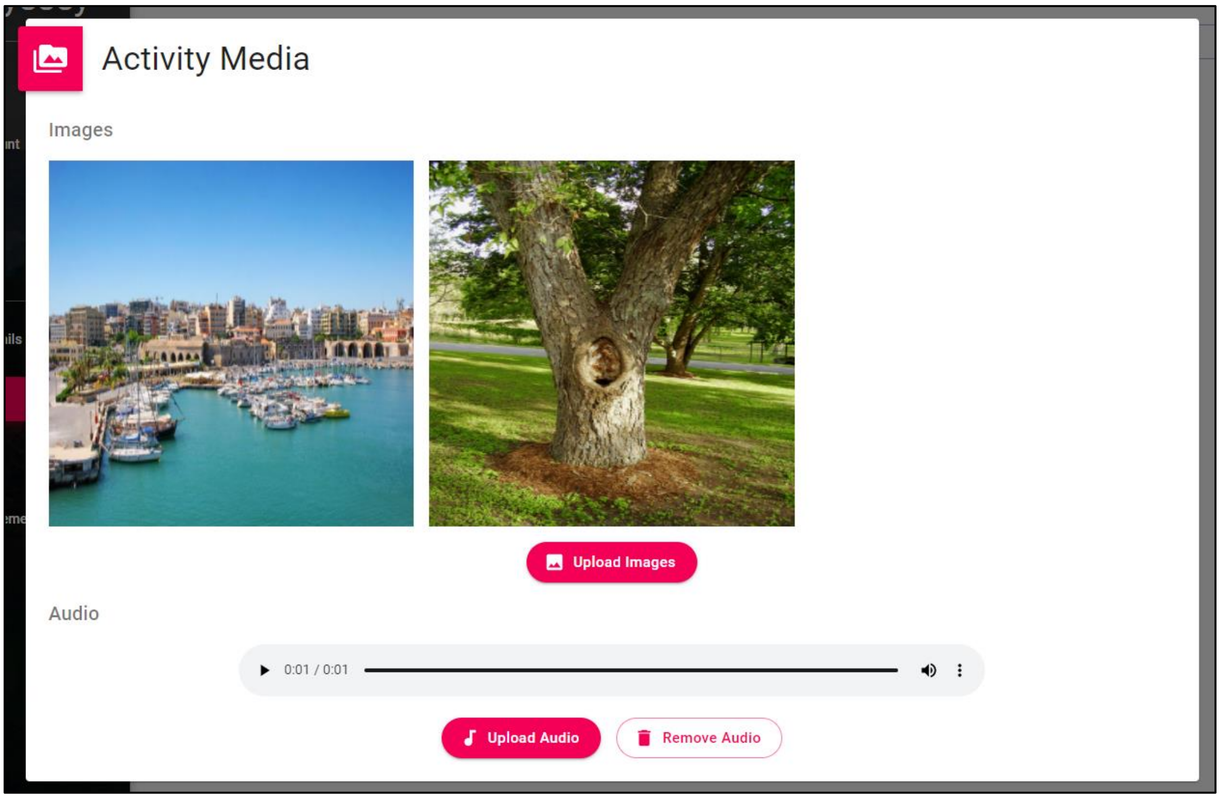


Figure 50 The Media Section for Activities

4.1.3.3 Members Page

The “members” page allows the user to manage a hunt’s members and teams. The user can create teams that can optionally require a password. A drag-and-drop interface was implemented to allow for seamless and intuitive management of members. More specifically, the user can add, remove, and move around members to and from teams simply by dragging members to the desired teams. The page also features member management, allowing the user to kick players from the hunt.

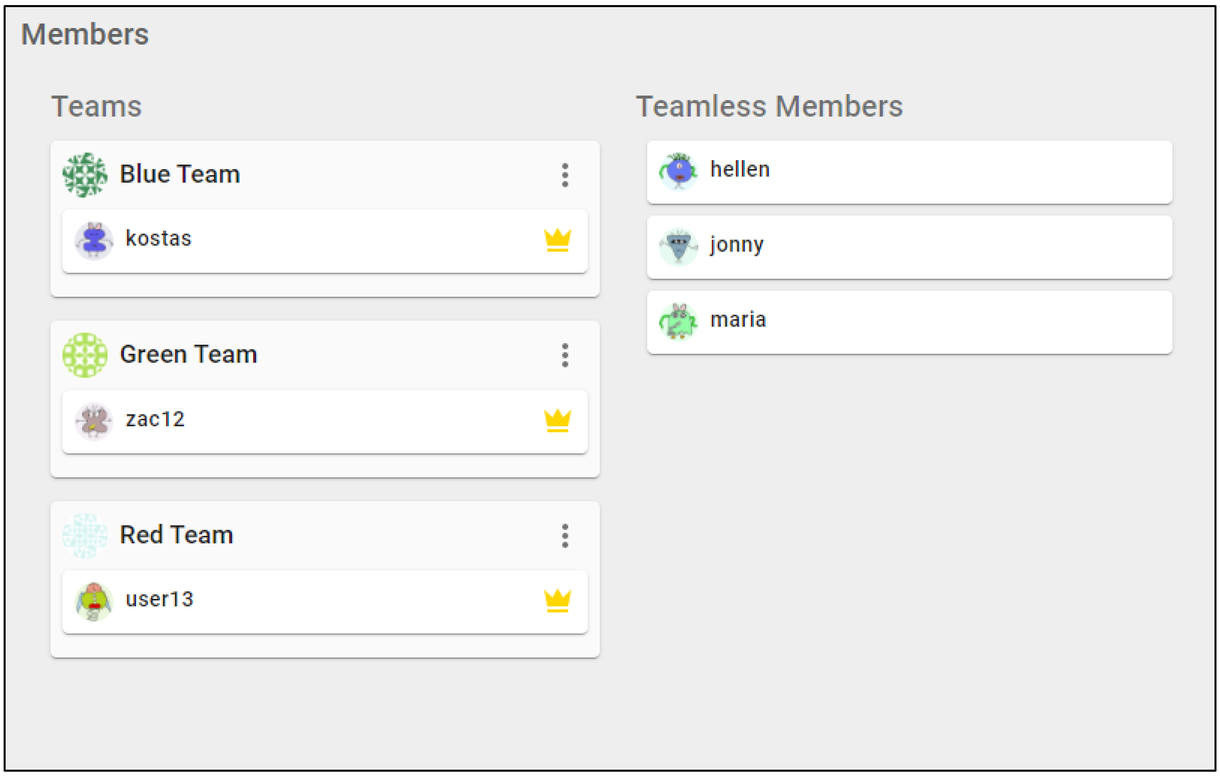


Figure 51 The Members Page

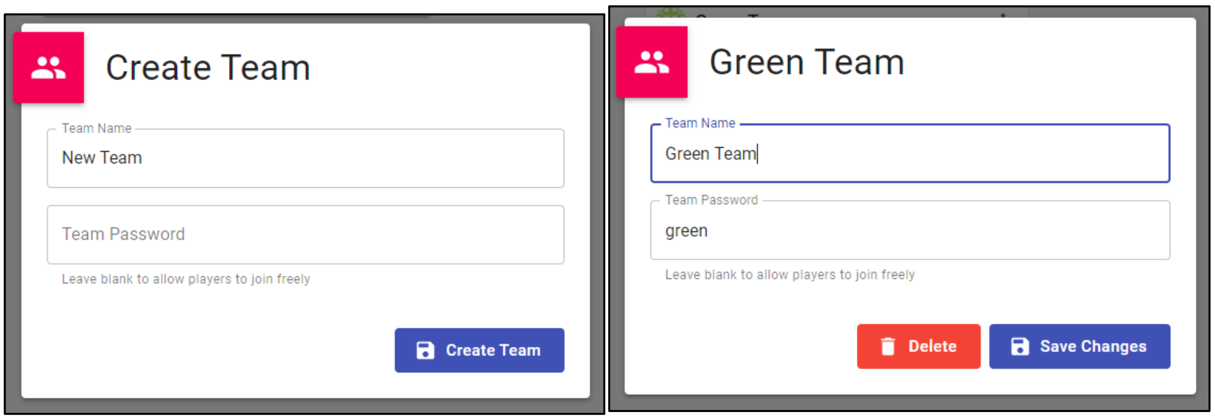


Figure 52 The Create and Edit Team Dialogs

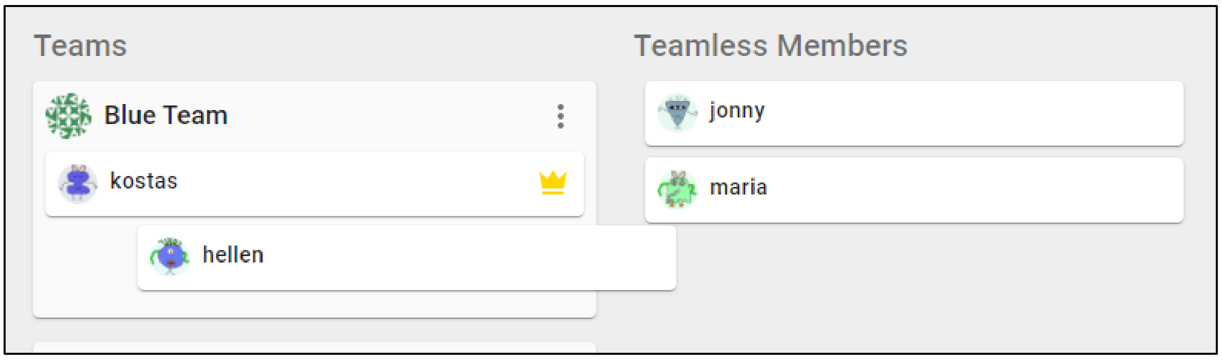


Figure 53 The Drag & Drop functionality for Members

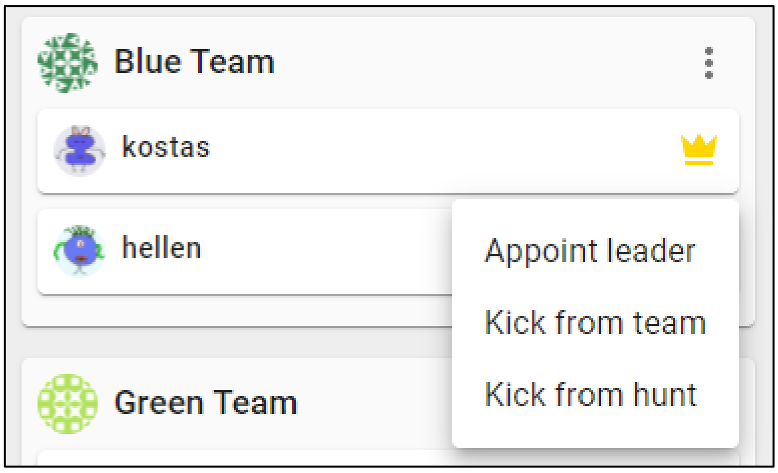
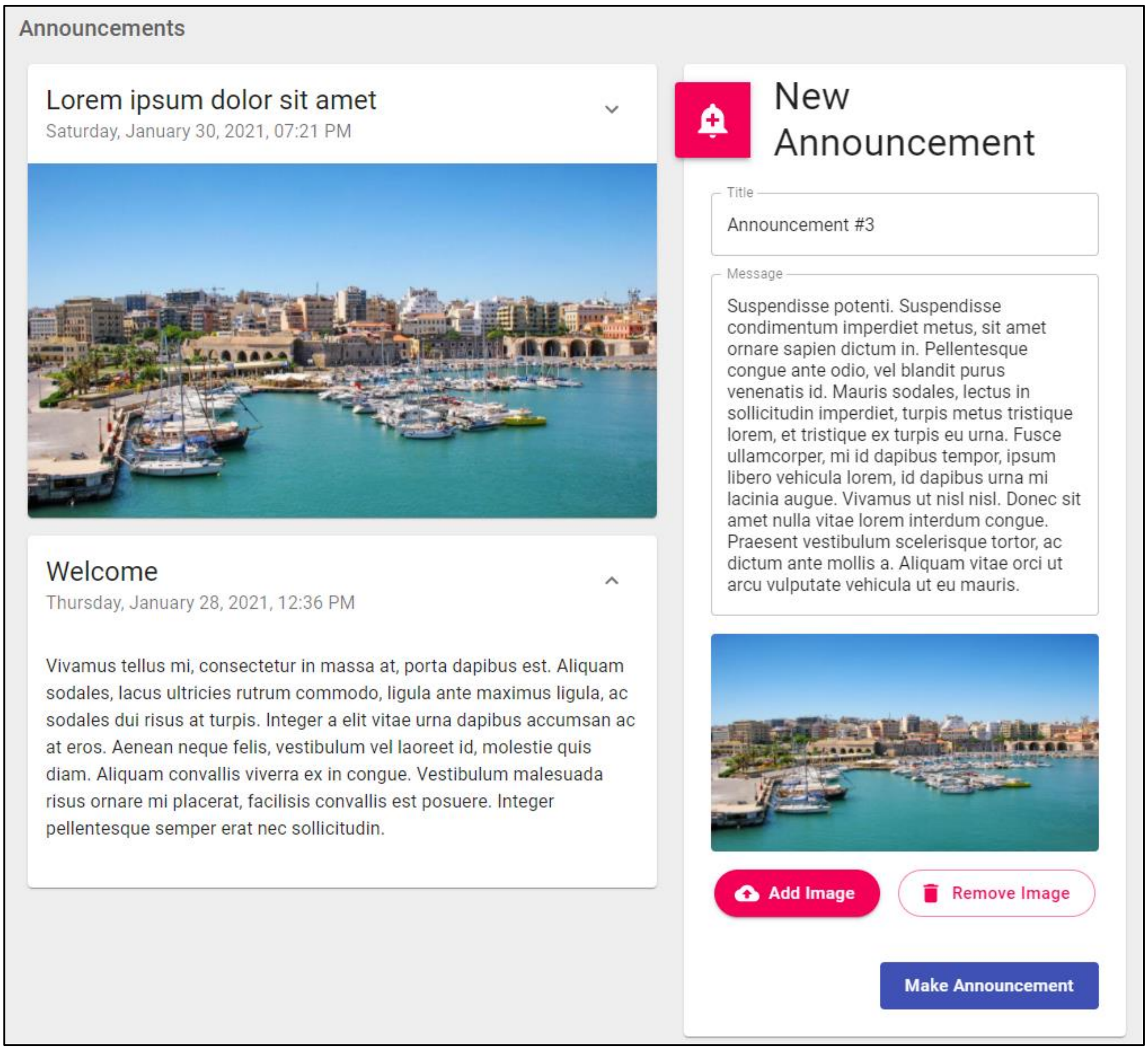


Figure 54 The Options for team members

4.1.3.4 Announcements Page

The announcements page, as its name implies is used to create announcements for the hunt. After creating an announcement through the form present, it is sent to all users that have joined the hunt. An announcement consists of three elements: a title, a message, and an optional image.


Announcements

Lorem ipsum dolor sit amet

Saturday, January 30, 2021, 07:21 PM

Welcome

Thursday, January 28, 2021, 12:36 PM

Vivamus tellus mi, consectetur in massa at, porta dapibus est. Aliquam sodales, lacus ultricies rutrum commodo, ligula ante maximus ligula, ac sodales dui risus at turpis. Integer a elit vitae urna dapibus accumsan ac at eros. Aenean neque felis, vestibulum vel laoreet id, molestie quis diam. Aliquam convallis viverra ex in congue. Vestibulum malesuada risus ornare mi placerat, facilisis convallis est posuere. Integer pellentesque semper erat nec sollicitudin.

New Announcement

Title: Announcement #3

Message: Suspendisse potenti. Suspendisse condimentum imperdiet metus, sit amet ornare sapien dictum in. Pellentesque congue ante odio, vel blandit purus venenatis id. Mauris sodales, lectus in sollicitudin imperdiet, turpis metus tristique lorem, et tristique ex turpis eu urna. Fusce ullamcorper, mi id dapibus tempor, ipsum libero vehicula lorem, id dapibus urna mi lacinia augue. Vivamus ut nisl nisl. Donec sit amet nulla vitae lorem interdum congue. Praesent vestibulum scelerisque tortor, ac dictum ante mollis a. Aliquam vitae orci ut arcu vulputate vehicula ut eu mauris.

Figure 55The Announcements Screen

5. Conclusion

In this thesis we presented a mobile game which allows any registered user to design a scavenger hunt and invite other players to participate in the hunt, either individually or in teams. When designing the scavenger hunt, the creator of the hunt can add as many activities as they like. Each activity can be one of the five following types: GPS activity, Question Activity, Multiple Choice Activity, Photo activity, and QR Scan Activity. Activities can be organized in activity groups to create more complex game scenarios. This mobile app also includes some other features such as a chat function between two individual users or between the members of a team, the ability for the hunt creator to make announcements for their hunts, and more.

Aside from the game features, we described all the different technologies we had to use together in order to develop this game.

5.1 Future Work

While the game in its current form can be enjoyable, there are certain aspects of the game which can be improved to provide a better user experience:

- Moderation
  Due to the nature of the game, there is a lot of user generated content such as chat messages, announcements, hunt descriptions, images, etc. To prevent users from uploading inappropriate content or harassing other players, a moderation system needs to be created to keep our game community safe and fun for everyone
- Push Notifications
  The ability for the game to send push notifications would be useful to notify the users of important game events even when they are not using the app
- Activities
  While the current activity types can cover the needs of most users, more activity types can be added to make the game more fun
- Many more…

•

References

[1] P. Christensson, "Programming Language Definition.," 23 September 2011. [Online]. Available: https://techterms.com. [Accessed 2 February 2021].

[2] W. contributors, "Java (programming language)," Wikipedia, The Free Encyclopedia., 16 February 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Java_(programming_language)&oldid=1007067957. [Accessed 16 February 2021].

[3] D. Flanagan, "JavaScript: The Definitive Guide, 6th Edition," in *JavaScript: The Definitive Guide, 6th Edition*, O'Reilly Media, Inc., 2011, p. 1.

[4] W. contributors, "JSON," Wikipedia, The Free Encyclopedia., 5 February 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=JSON&oldid=1005074494. [Accessed 15 February 2021].

[5] W. contributors, "HTML," Wikipedia, The Free Encyclopedia., 11 February 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=HTML&oldid=1006099685. [Accessed 15 February 2021].

[6] W. contributors, "Spring Framework," Wikipedia, The Free Encyclopedia., 13 January 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Spring_Framework&oldid=1000168330. [Accessed 16 February 2021].

[7] T. Occhino, "[JSConfUS 2013] Tom Occhino and Jordan Walke: JS Apps at Facebook," Youtube, 5 August 2013. [Online]. Available: https://www.youtube.com/watch?v=GW0rj4sNH2w. [Accessed 15 February 2021].

[8] "Release v17.0.1 facebook/react," Github, 22 October 2020. [Online]. Available: https://github.com/facebook/react/releases/tag/v17.0.1. [Accessed 15 February 2021].

[9] "Material-UI: A popular React UI Framework," 2021. [Online]. Available: https://material-ui.com/. [Accessed 15 February 2021].

[10] T. Occhino, "React Native: Bringing modern web techniques to mobile," Facebook, 26 March 2015. [Online]. Available: https://engineering.fb.com/2015/03/26/android/react-native-bringing-modern-web-techniques-to-mobile/. [Accessed 15 February 2021].

[11] W. contributors, "API," Wikipedia, The Free Encyclopedia., 12 February 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=API&oldid=1006385267. [Accessed 16 February 2021].

[12] W. contributors, "Hypertext Transfer Protocol," Wikipedia, The Free Encyclopedia., 13 February 2021. [Online]. Available:

https://en.wikipedia.org/w/index.php?title=Hypertext_Transfer_Protocol&oldid=1006560132. [Accessed 16 February 2021].

[13] W. contributors, "Representational state transfer," Wikipedia, The Free Encyclopedia., 14 February 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Representational_state_transfer&oldid=1006743979. [Accessed 16 February 2021].

•