Team 22, Froogal Gamification Engine, Alapan Sau, Dhruv Kapur Pavani Babburi, Sidharth Giri

Design Overview

Architectural design

We decided to have two subsystems in our project according to the requirements of our client. The two subsystems are **Admin Dashboard** and **User Game Application.**

The **Admin Dashboard** is a webapp from where an Admin (of a particular store) can operate on the following features:

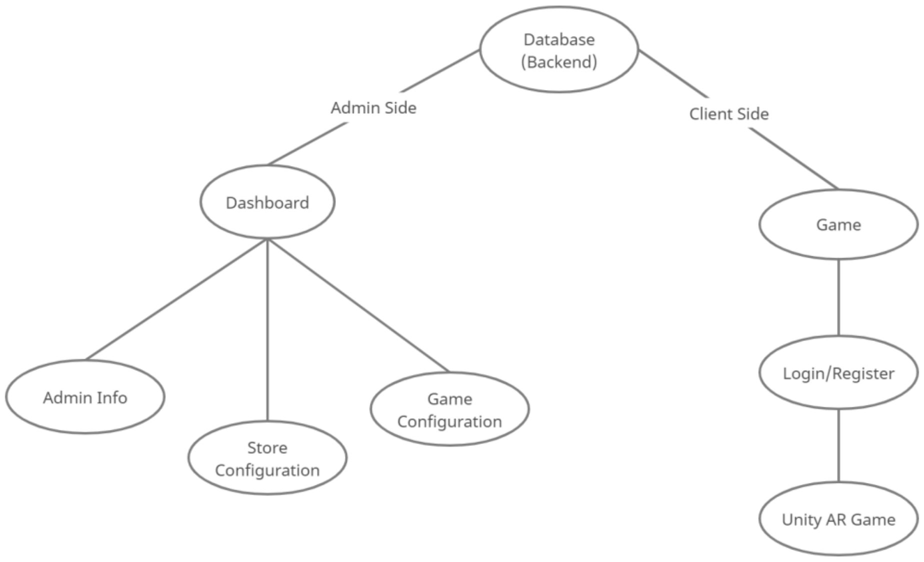
- Adding/Editing new store locations
- Adding/Editing game configurations

The Dashboard has two different tabs to redirect you to the individual modules that handle game configurations and store configuration.

The **User Game Application** is a native mobile application where users can provide their credentials and play an AR game and obtain coupons when the game ends. Here, the game presented to the user is the one whose game configuration has been setup by the administrator in the dashboard.

Upon opening the Application, the user is welcomed with the Login/Register page, wherein upon login the user gets access to play the game and win rewards.

The two subsystems collaborate through a single non-SQL backend to store the configurations that are specified by the company Admin, which the Game application later fetches from the backend to configure the game that needs to be rendered on the user's device.



System interfaces

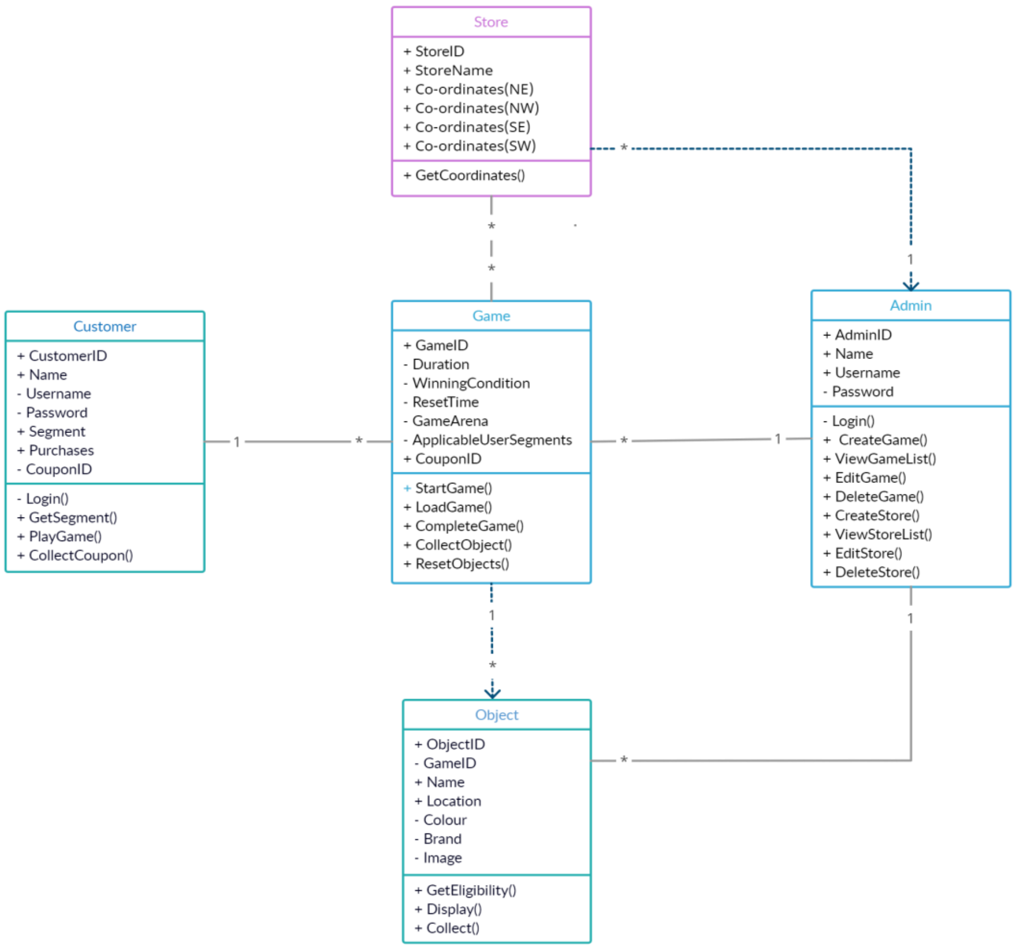
User Interface

The system caters to two categories of users, the businesses creating games for their customers and the customers playing the games for various offers.

The businesses can interact with the system through the Admin Dashboard, which allows them to create games a web app, which allows them to create, edit and delete games and view the relevant details.

The customers play an AR game in a mobile app, which allows them to collect various objects placed throughout the store. Upon collecting the specified items within the time constraint, the customer gets a coupon which can be cashed in the store.

Model

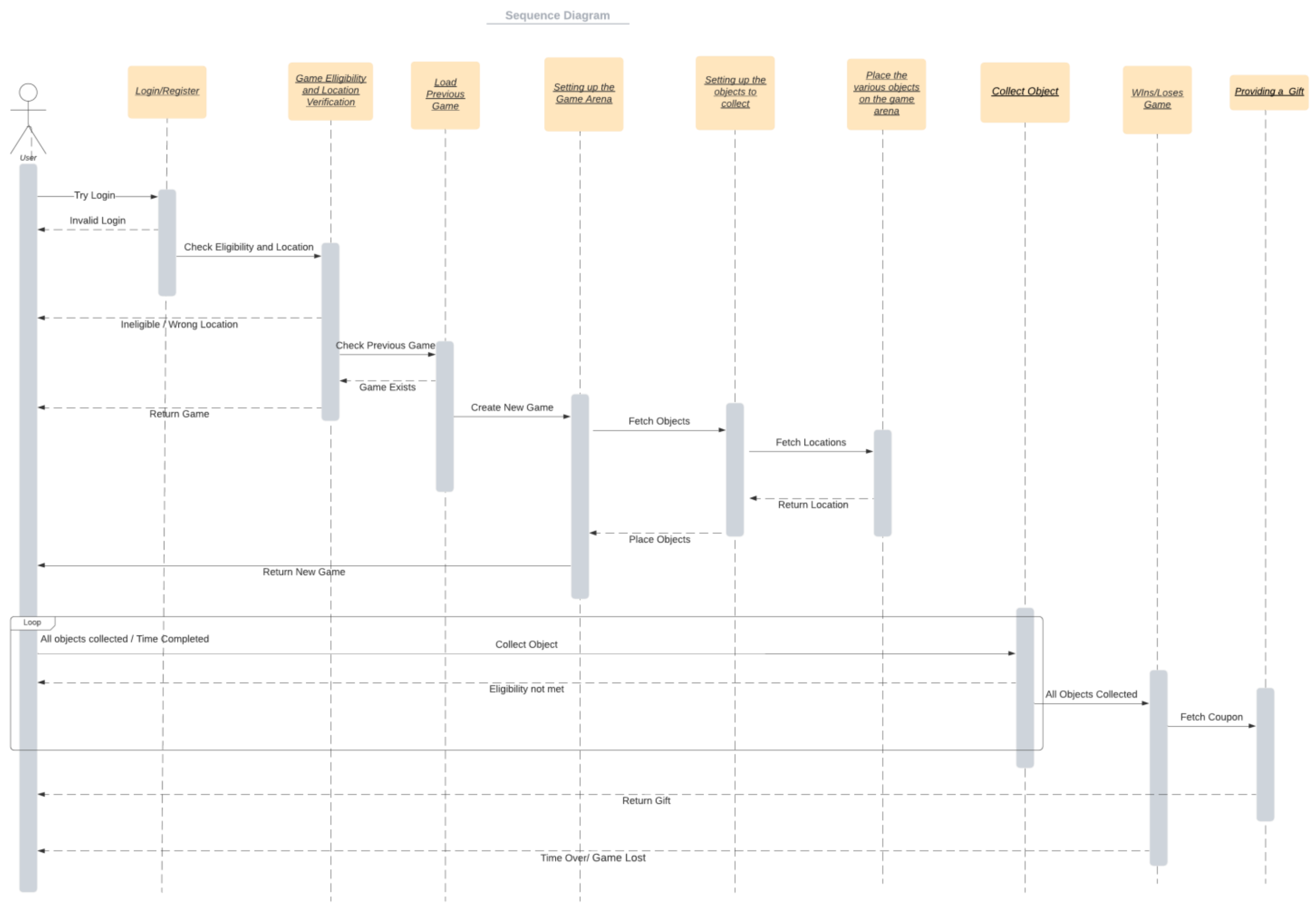


Customer	Class state • Name • Username • User Segment • Purchases Class behavior • Login () • GetSegment() • PlayGame() • CollectCoupon()
Store	Class state • Store Name • Co-ordinates Class behavior • GetCoordinates()
Game	Class state • Duration • Winning Condition • Reset Time • Game Arena • Applicable User Segments Class behavior • StartGame() • LoadGame() • CompleteGame() • CollectObject() • ResetObjects() • SaveGame()
Object	Class state • Name • Location • Color • Brand • Image Class behavior • CollectObject() • DisplayObjcet()
Admin	Class state • Name • Username Class behavior • Login() • CreateGame() • ViewGames()

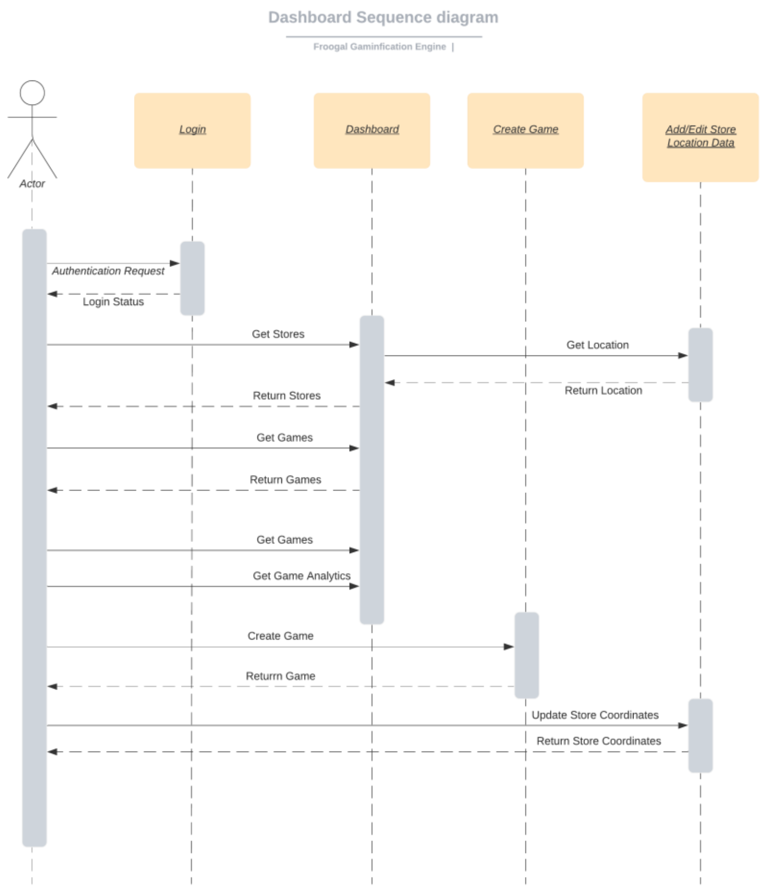
	• EditGame() • DeleteGame() • AddStore() • ViewStores() • EditStore() • DeleteStore()

Sequence Diagram(s)

1. The Game App Sequence



2. The Business Admin Sequence



Design Rationale

1. *Initially, the tech stack was decided to be Viro React (for AR Engine), React Native (game Application) and Node Js backend using express framework as the middleware. However, it was found Vero React was an extremely outdated framework and so dropped.*
2. *We decided on using Adonis as our backend middleware, due to its modularity and ease of use*
3. *We planned on using Unity Engine as our AR Engine as it can be exported to as both apk and web app.*
4. *We decided to create the game application interface in Java instead of React Native.*
5. *Initially the AR of the game included the height of the location (based on floors), however due to precision issues, the idea was dropped.*