Team: HvZ App

Members: Kyra Clark, Santiago Rodriguez, Alex Hadley, Matthew Waddell
Project 3C.1 — Design Plan

Github Link:
https://github.com/alexhad6/HvZ-app/blob/master/Administrative/Phase%203/Design_3c.txt
Primary author: Matthew

# Link to HVZ's UI Simulator::
https://marvelapp.com/prototype/6dj0dfj

# Specification plus Design

## Indiscriminate Pre-Game Path
(this part is the same for a player and a moderator)

Screen 0:
    Sign in:
        Input: String username, String password
        Output:void
        Description: API searches database for matching strings to verify existing
        username and password. The API then checks if the existing user is moderator
        and if so sends them to the moderator panel. Otherwise the API checks if the
        user is human or zombie and sends them to the corresponding player panel.
    Register:
        Input: String username, String password
        Output: None
        Description: API stores input username and password in database.

Screen 1:
    Create game:
        Input: Touch
        Output: None
        Description:API is periodically called to check the database for whether players
        have joined the user's game and display incoming players.
    Join Game:
        Input:Touch
        Output:None

Description: User is sent to a lobby screen which displays a list. The API is periodically called to check the database for new games. When Player calls the join game method, the API will check the database to ensure the banned boolean associated with player id is false.

Sign Out:

Input:Touch/None

Output: None

Description: User is sent to Screen 0. Authentication is removed.

------------------------------------------------------------------------------------------------------------------

# Pre-Game Moderator Path

Mod 0 (ie Pre-Game Lobby):

Start Game:

Input:Touch

Output: None

Description: Start Game checks that the lobby has a minimum number of players. If so then Game is instantiated and the caller is made moderator and sent to the Admin Panel.

Back:

Input:Touch

Output: None

Description: Sends User to Screen 1. API calls a database function to delete user's games and bans all players from the lobby.

------------------------------------------------------------------------------------------------------------------

# Moderator Paths During Game

Admin Panel:

Manage Players:

Input: Touch

Output: Mod1 Display

Description: Calls API.get current players() displays player names in a list of buttons with each index denoted by player name and an API. get player attributes embedded into each separately callable button. displays public method back and public method. Conjuring Mod1.

Manage Missions:

Input: Touch

Output: Mod2 Display

Description: Calls API to get current files and lists files currently in file storage. Share file method , Delete file method and upload file method as buttons are then Displayed configuring Mod2.

Make Announcement:

Input: Touch

Output: Mod3 DIsplay

Description: displays text box for user input strings and announce method.
Configuring Mod3

End Game:
Input: Touch
Output: None
Description: API first calls Ban method on all users,then stores game results in the database then Deletes Game Object.

Mod 1:
Select Player:
Input: Touch
Output: Mod1A Display
Description: API calls give player attributes with input of the player id returning player info to be displayed as a separately callable list of attributes with edit player attribute () embedded in them. The kick/ban method player and edit attribute.

Back:
Input: Touch
Output: None
Description: Reconfigures Admin Panel

Mod 1A:
Edit Player:
Input: Selected Attribute
Output: Modifiable String in textbox
Description: By selecting a listed attribute and and calling edit method
The selected attribute is passed to the edit method allowing the attribute to be modified. API is called to check if a modified string differs from corresponding database property and to replace if necessary.

Ban Player:
Input: Touch
Output: None
Description: Removes Player from game by using API to set banned boolean to true, a boolean that is ordinarily checked to be false before being allowed to join a game, in the database and then signing the user out of the app.

Kick Player:
Input: Touch
Output: None
Description: temporarily Removes Player from game by using API to set kicked boolean to true in the database and then signing the user out of the app. When a player calls a join game again, the kicked boolean is set back to false by the API. Serves as a forced reboot easily accessed by mod app framework.

Back:
Input: Touch

Output: None

Description: Reconfigures Display Mod1.

Mod 2:

Share File:

Input: Selected File

Output: None

Description: Adds File to Database using database API.

Delete File:

Input:Selected File

Output: None

Description: Removes selected mission files.

Upload File:

Input:File

Output: None

Description: Uploads File to mission files.

Back:

Input:Touch

Output: None

Description: Reconfigures Admin Panel.

Mod 3:

Announce:

Input:Text

Output: None

Description: Calls API to store text in announcements in the database.

Back:

Input: Touch

Output:None

Description: Reconfigures Admin Panel

Mod 4:

Termination Request Confirmed:

Input: Touch

Output: None

Description: Signs all users out. Stores relevant Game Information in the database using API.

Termination Request Denied:

Input: Touch

Output: None

Description: Reconfigures Admin Panel

---------------------------------------------------------------------------------------------------

(Pre-Game Player Path)

PrePlayer 0:

Join Game:

Input: Selected Game

Output: None

Description: Configures player lobby by adding player ID to incoming player list in database for selected game. Said list is Displayed along with method leave Lobby.

Player Lobby:

    Leave Lobby(ie Back):

        Inputs: Touch

        Outputs: None

        Description: Reconfigures PrePlayer 0 display and calls API to remove player ID From incoming players in the database.

--------------------------------------------------------------------------------------------------------------------

(Player Paths During Game)

Player Home:

    Missions:

        Input: Touch

        Output: None

        Description: Displays mission Files found by database API as separately selectable list.

    Stats:

        Input:Touch

        Output: None

        Description: Displays list of players and corresponding human or zombie attribute displays ShowAncestryTree method.

    Feedcode:

        Input:Touch

        Output: None

        Description: if player has attribute zombie, launches to feed code scan method opening the camera app. Otherwise the feed code generated at game instantiation for the player is represented as a scannable image.

    Announcements:

        Input:Touch

        Output: None

        Description: Displays lists of separately selectable announcements by calling database API to check if there are any recent announcements. Also displays the delete announcement method as a button.

Player 1:

    Open Filet:

        Input:Touch

        Output:Selected File

        Description:Selected File is displayed on user phone.

    Back:

        Input: Touch

        Output:None

        Description: Reconfigures User Home

Player 2:

    Show Ancestry Tree:

        Input: Touch

        Output: None

        Description: Ancestry kept accounted for via the API recording the ID
        Of the feed code scanner in the database and pairing it with the matched
        Feedcode number is displayed along with a back method in Player 2B display.

    Back:

        Input: Touch

        Output:None

        Description: Reconfigures User Home

Player 2B:

    Back:

        Input: Touch

        Output:None

        Description: Reconfigures Player 2.

Player 3Z:

    Scan Feed Code:

        Input: Image file

        Output: Notification on if scan was successful

        Description: API attempts to match FeedCode display to those stored in
        Database, if there is a match then zombie player is notified scan was successful
        And a kick boolean is stored in the database by the API associated with the ID of
        the scanned user as well as their human attribute being replaced with zombie.
        Then reconfigures User Home. If Notification is called by the player then the
        Show Ancestry Tree Method is called.

    Back:

        Input: Touch

        Output:None

        Description: Reconfigures User Home

Player 3H:

    Display Feed Code:

        Input:Touch

        Output: None

    Description: API prompted to recursively check whether the players attribute in the
    database has been changed to human for the next 3 minutes every minute until either
    The 3 minutes end or a change in the attribute has occurred(time selected malleable). If
    a change occurs the user is kicked from the game and their settings are reconfigured as
    zombie when they return.

    Back:

        Input: Touch

        Output:None

        Description: Reconfigures User Home

Player 4:

Read Announcement:
    Input:Touch
    Output:None
    Description: Displays announcement in a text box the majority of the display in Player 4a
Delete Announcement:
    Input: Selected Announcement
    Output: None
    Description: Removes Announcement from the list displayed to the user. This method of delete is compatible with the goal since the API only checks to add newer announcements.
Back:
    Input: Touch
    Output:None
    Description: Reconfigures User Home
Player4a:
Back:
    Input: Touch
    Output:None
    Description: Reconfigures User Home


# Feasibility for Specifications:

In order to be able to develop an app according to the above specifications, I require my language to be able to call and check the results from API on a loop, be able to both scan and produce QR codes as well as appropriately handle text, touch and image input which each have demonstrated feasibility in our project proposal. Additionally I have demonstrated feasibility of the User InterFace by creating each screen in a service which allows me to emulate each path I have described in order to show the interface paths are consistent.

Link to HVZ's UI Simulator: https://marvelapp.com/prototype/6dj0dfj
A couple a adaptations were made for my UI to fit the simulator, including a button to get from player lobby to user hub which would ordinarily have to have been initiated by the mod, but needed since the model is built for accounting the actions of one user at a time and to represent the path to both scanning and displaying feed code for the User the feedcode button was split into a left half going to the zombie scan screen and a right going to the human display screen for efficiency since this was the only point at which the human and zombie user interface paths diverged.