



Rubén Tortosa Zamora

ESAT

Índice

Introducción	02
Recopilación de información	04
Árbol de contenidos	07
Wireframes	09
Cartas de color	11
Tipografías	12
Diseños	13
Elección de tecnología	20
Ejemplos de código	22
About	24

Introducción

Rebellion es una web app basada en uno de los muchos juegos de roles ocultos que existen.

En mi caso elegí "La Resistencia" por ser un juego que sin ser muy complicado presentaba bastantes retos a la hora de ser llevado a lo digital.

Esos retos son principalmente la interactividad en tiempo real y la presencia de un chat que permita la comunicación entre los distintos jugadores.



La Resistencia.

Resumen Juego

La resistencia es un juego de roles ocultos en el cual los jugadores no saben que rol está jugado su compañero.

Los jugadores se dividen en dos bandos., los agentes que intentarán que las misiones sean un éxito y los espías que harán todo lo contrario.

El juego consiste en cinco rondas con esta estructura:

- Nuevo líder
- Líder propone un equipo para la misión.
- El grupo discute y vota (en abierto) la valía de ese equipo.
- El equipo es enviado a la misión y los integrantes votan (en secreto) si la misión es un éxito o no.
- Nueva ronda.

La partida termina cuando se consiguen dos éxitos o dos fracasos en las misiones.

La esencia del juego está en crear las dudas y la discordia a través del debate, exponiendo argumentos tanto a favor como en contra.



La Resistencia.

Recopilación de información

¿Qué es Rebellion?

Un juego interactivo online de 5 a 10 personas.

¿Cómo es?

Un juego de roles ocultos dónde los jugadores deben llevar a cabo sus objetivos teniendo en cuenta el rol que se les haya asignado.

Características

- Aplicación web orientada a dispositivos móviles
- Chat en tiempo real
- Sistema de votaciones abiertas / cerradas
- Perfil de usuario

Target

Grupo de amigos que buscan pasar un buen rato discutiendo entre ellos e intentando averiguar quienes son los espías.

Objetivo de la App

El objetivo principal de la app es conseguir que el usuario disfrute de una experiencia online lo más parecida posible a una partida con los amigos de manera física.

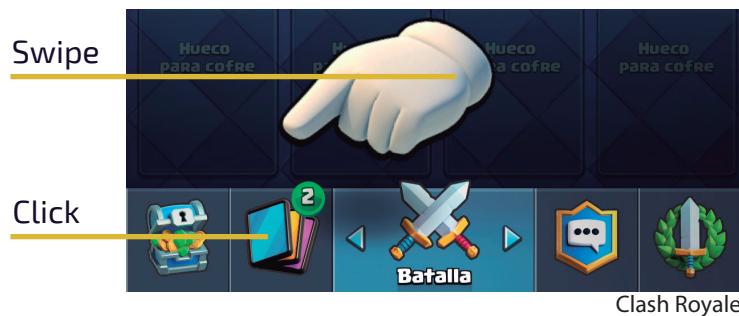
Además de servir como apoyo en las partidas físicas al poder servir de sustituto de los elementos del juego.

Recopilación de información

Diseño

La aplicación estará basada en una navegación horizontal mediante swipe o click en los elementos del menú que permita moverse entre las diferentes secciones. Como se puede ver por ejemplo en Clash Royale o el propio WhatsApp.

Además para el chat tomaremos como referencia las aplicaciones de chat WhatsApp y Telegram.



Log Conversación

Mensaje Sistema

April 25

Mensaje Usuario

Hi 4:10 PM ✓

This is a secret! 4:12 PM ✓

Mensaje Jugador

Ok 4:13 PM

Cuadro Texto

Type message



Telegram

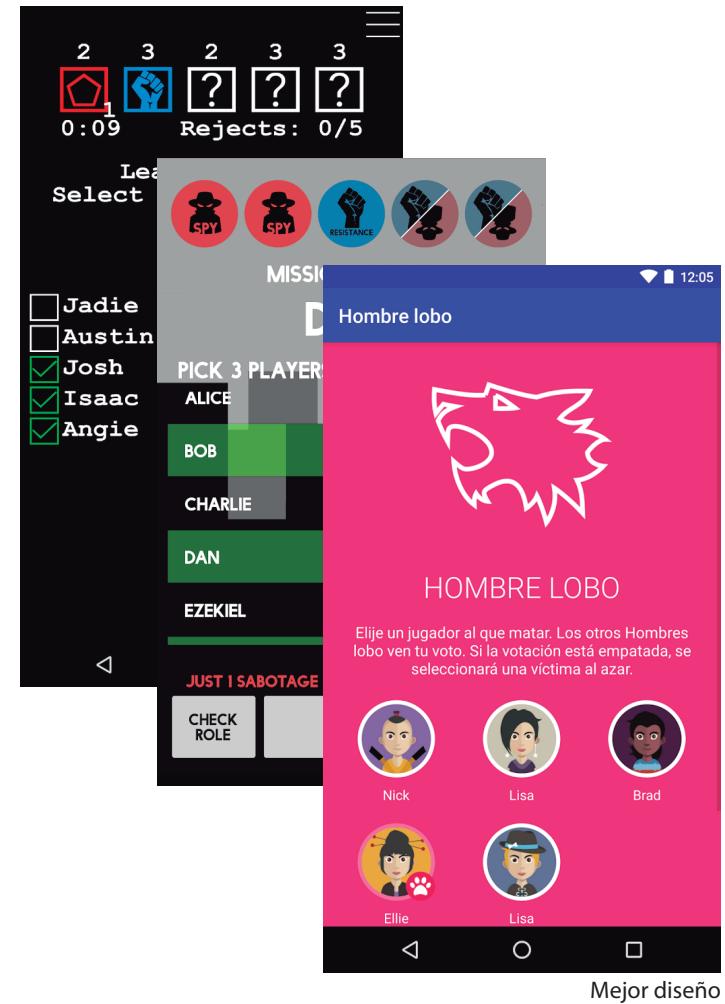
Recopilación de información

Competencia

Las búsquedas realizadas en Google Play nos muestran diversos juegos de roles oculto. Todas ellas tienen características comunes la más destacada es que ninguna de ellas permite el juego online. Todas se basan en el modo Pass & Play.

También destaca que casi todas ellas carecen de un diseño más o menos agradable como se puede apreciar en los ejemplos.

Ejemplos



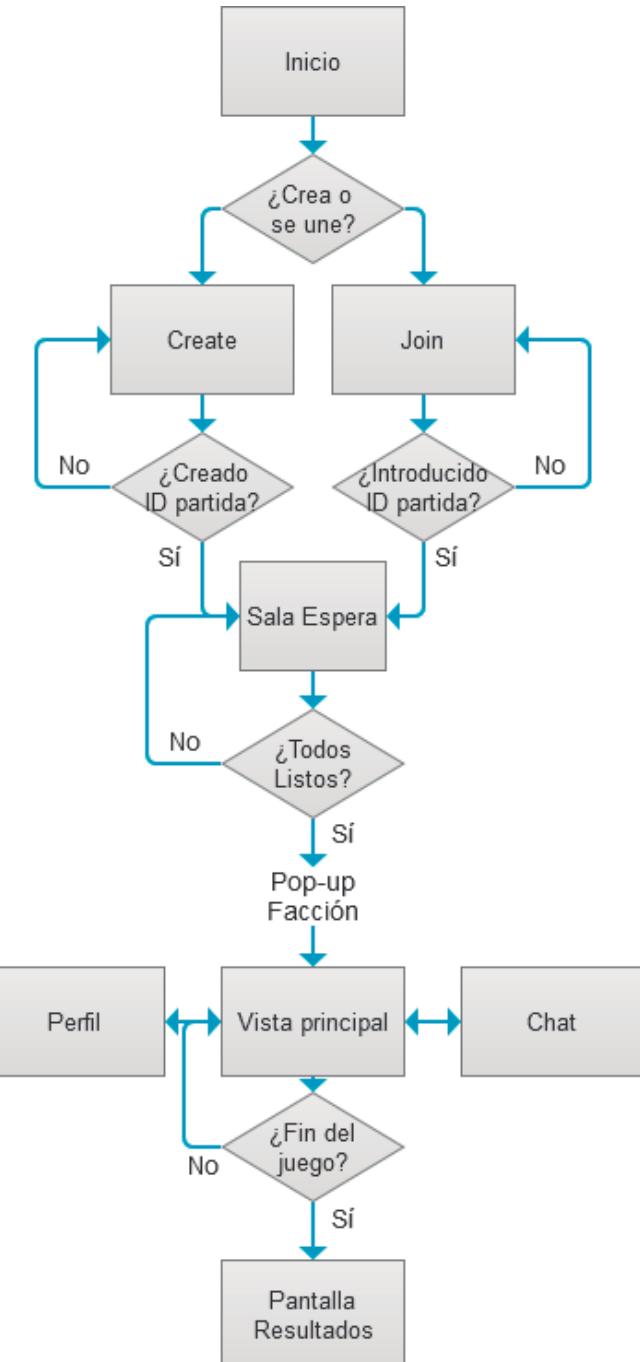
Árbol de contenidos

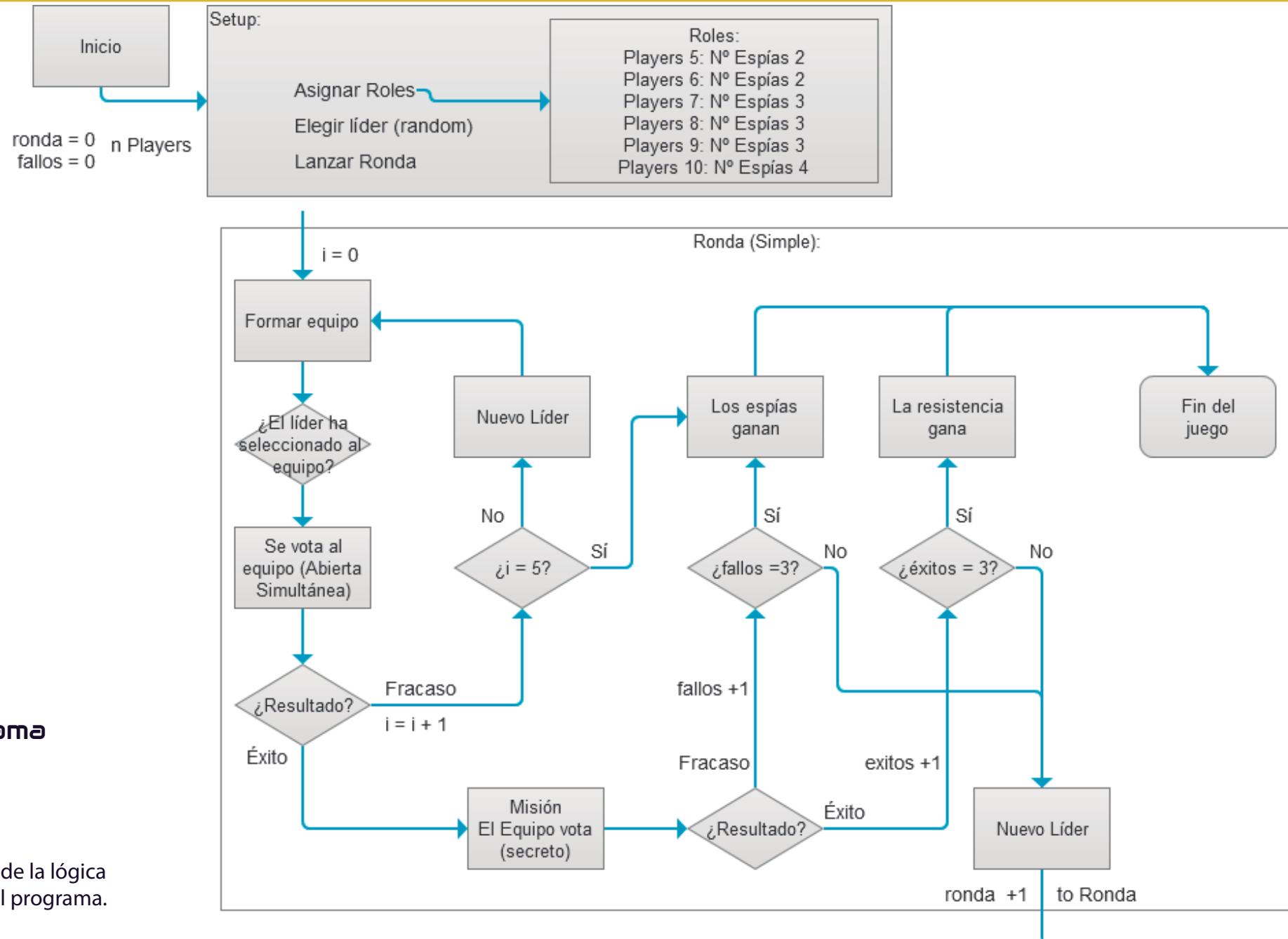
Como prácticamente la totalidad del las aplicaciones. Rebellion empieza en la pantalla de inicio dónde el jugador podrá crear una partida nueva o unirse a una ya existente.

Después se pasará a una sala de espera donde podrá introducir su nombre y añadir una foto. Cuando todos jugadores estén listo el creador de la partida podrá iniciar el juego.

Una vez iniciado el juego saltará un Pop-Up en el cual se informa al jugador de cuál es su facción.

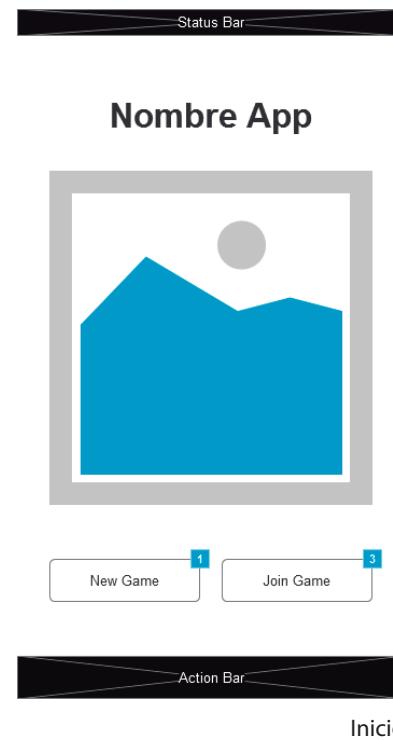
El jugador puede navegar libremente por los diferentes apartados hasta que se alcance el final de la partida donde será conducido a la pantalla de resultado.





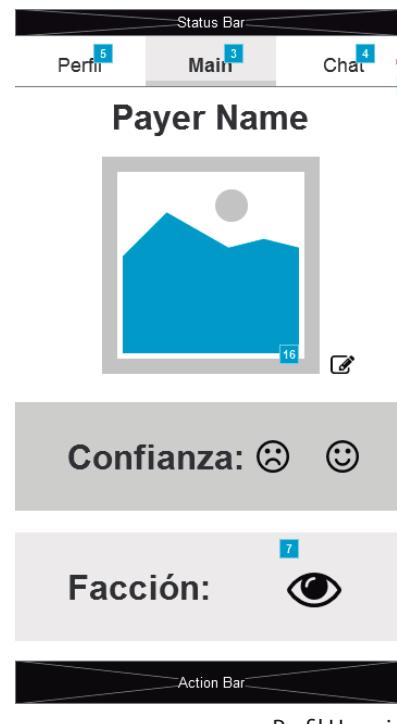
Wireframes

Diseños preliminares donde se muestra la composición de las diferentes pantallas del juego mediante componentes genéricos.

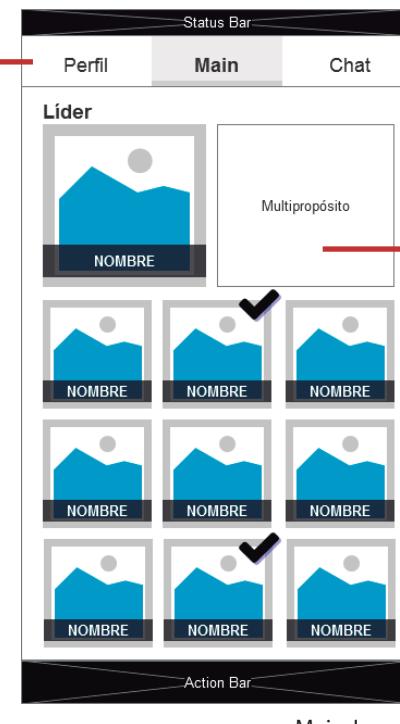


Wireframes

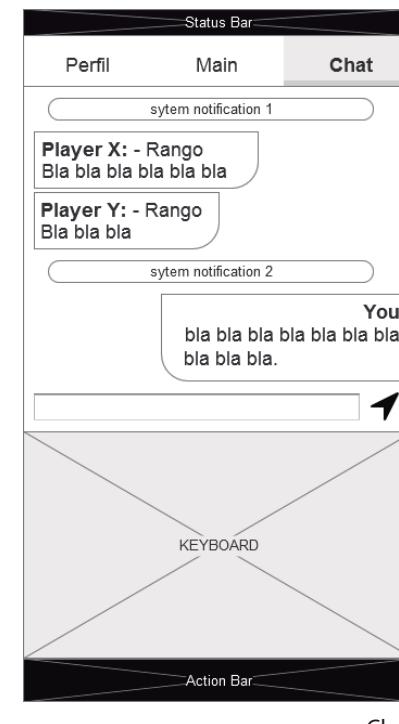
Aquí se puede apreciar la composición de que será el juego principal. Perfil, Juego y Chat.



Perfil Usuario



Main Juego



Chat

Cartas de color

Se empezo trabajando con tres líneas diferentes de diseño.

Future
Vintage
Art-Decó

Cada una de esta líneas requería de una paleta de colores propia que transmitiese lo mejor posible las sensaciones de cada estilo.

Por ejemplo, para una línea futurista y con la ayuda de herramientas como coolors, se llegó a la conclusión de que colores oscuros y fríos transmitían mejor la sensación de impersonalidad tecnológica que se buscaba.

Al igual que los colores marrones se adaptaban mejor a un posible tema vintage de espías clásicos.

Future Sci-Fi



Vintage



Art Decó



Gama Básica

Colores Resalto

Tipografías

Cada uno de los temas requería de una tipografía representativa que transmitiese lo mejor posible las sensaciones de cada temática.

Por ejemplo, para una línea futurista y con la ayuda de Google fonts, se buscó una tipografía principal algo agresiva y moderna que evocase la tecnología acompañada de una secundaria que no perdiese la esencia y fuese legible.

De la misma manera una fuente con serifa se adaptaba mejor a la sensación y aires clásicos de nuestro tema vintage.

Future - Audiowide

Cuerpo de texto - Exo 2.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex

Vintage - Zilla Slab

Cuerpo de texto - Special Elite

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex

Art Decó - Limelight

Cuerpo de texto - Poiret One

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex

Diseños

En la parte de abajo se muestran las tres líneas de diseño principales con los elementos más importantes. Al final la línea Future Sci-fi fué la elegida para desarrollar el diseño completamente quedando las otras descartadas. Si en un futuro se implementan nuevos temas se podrían rescatar.



Future - Sci-Fi

Una vez elegido como definitivo el tema future se continuó con el diseño de los elementos restantes y de los componentes necesarios.

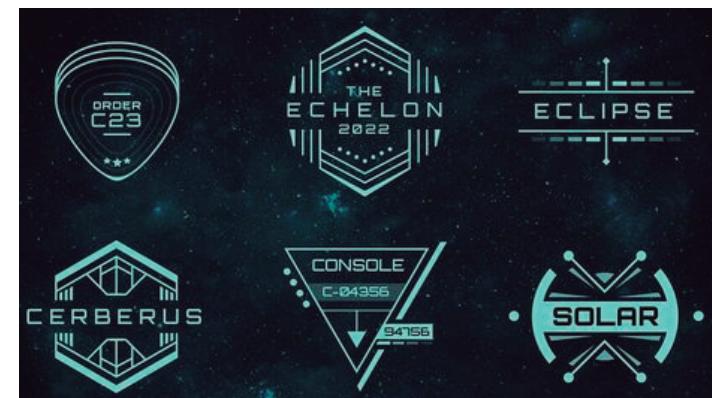
Menú

Se pensó en hacer un menú con texto o con iconos, al final se optó por utilizar iconos. El uso de iconos hacia del menú una parte más interesante visualmente.

	Activo	Inactivo
Perfil		
Pantalla principal		
Chat		

Logotipos

Los logotipos están basados en múltiples referencias encontradas en la web, las referencias son de otros logotipos sci-fi. Muchos de ellos compartían como característica principal el uso de formas geométricas básicas.



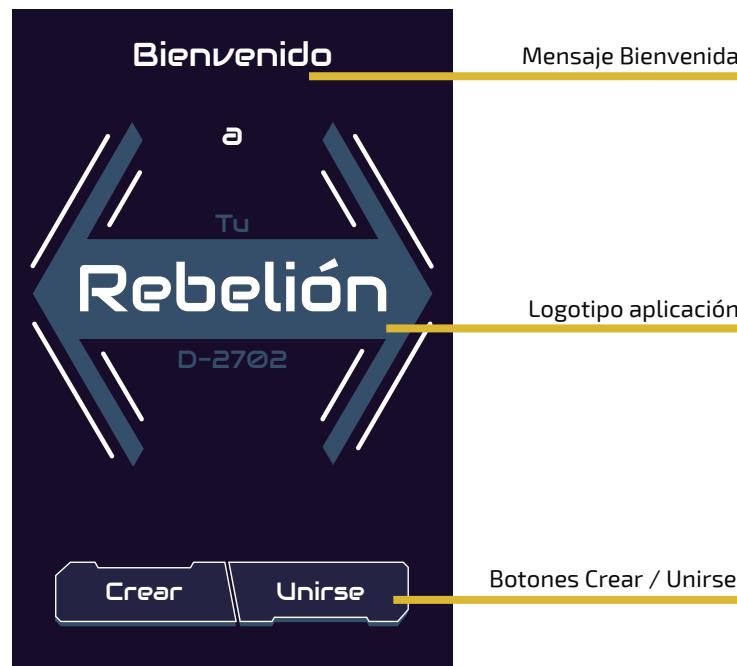
Ejemplos



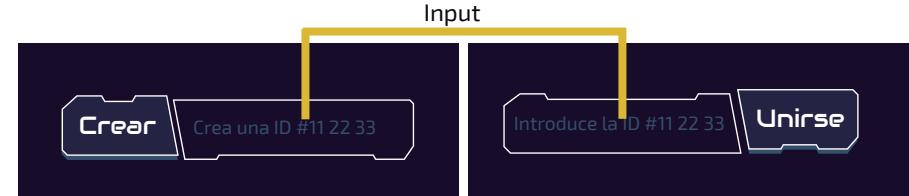
Pantallas

En esta y sucesivas páginas se mostrarán los diferentes diseños de cada una de las pantallas de la aplicación y alguno de los diferentes estados que éstas pudiesen adoptar.

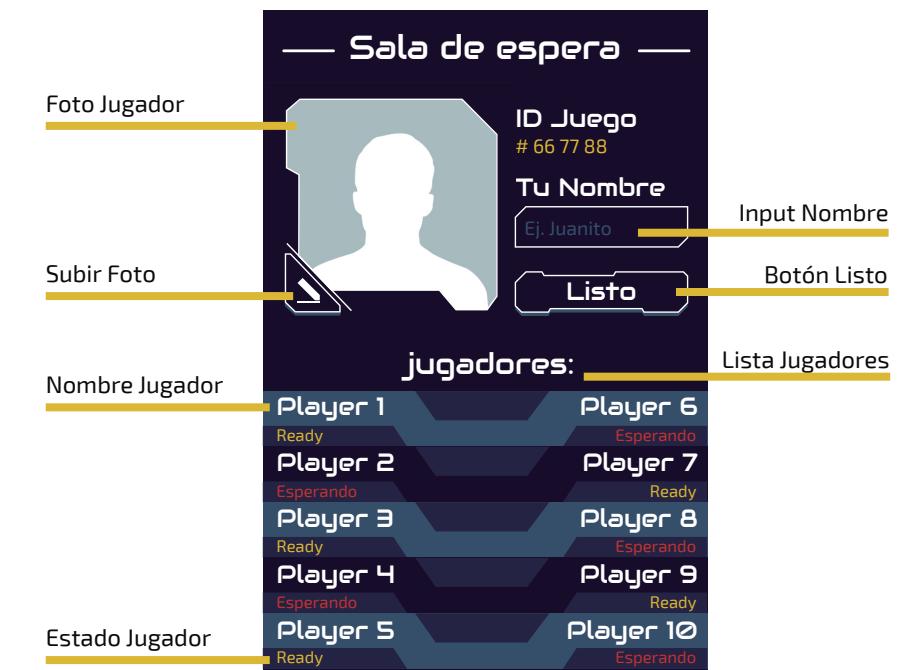
Pantalla de Inicio



Botones activados

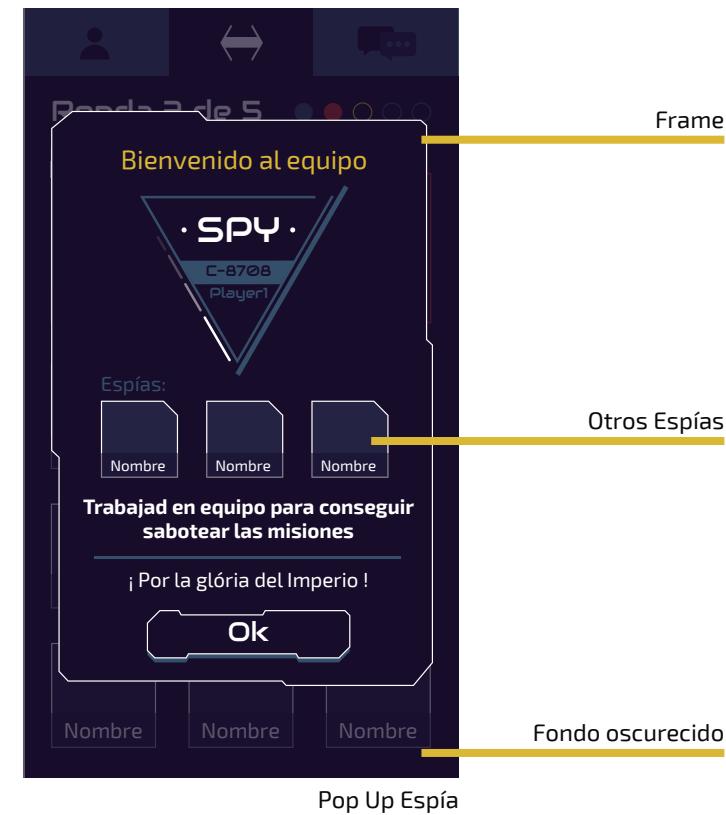


Pantalla de espera



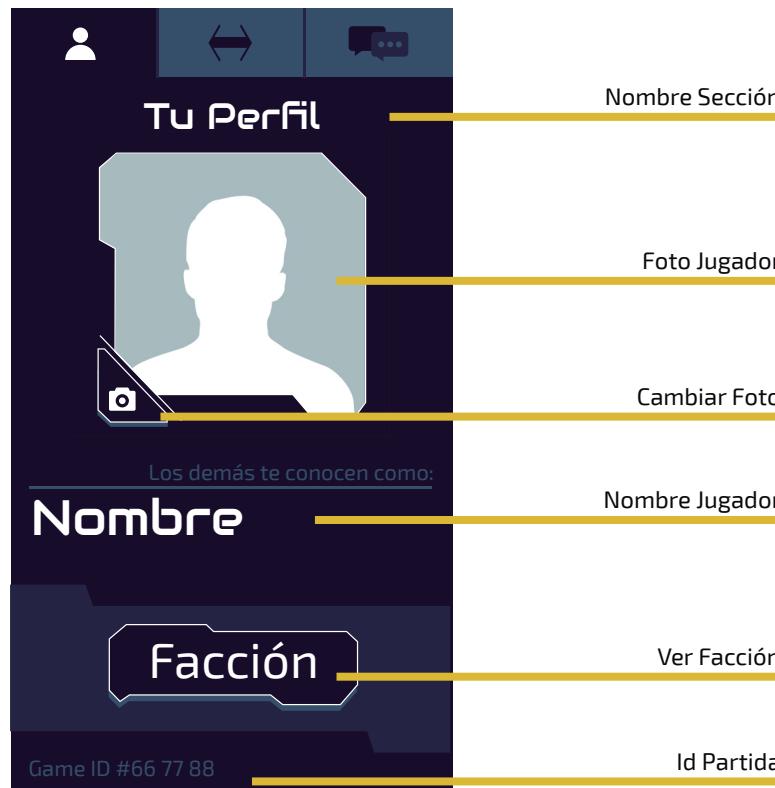
Pop Up

Una vez comenzada la partida al jugador se le muestra un pop-up que le comunica a que facción pertenece. El diseño continúa en la línea sci-fi mostrando bordes irregulares de líneas rectas.

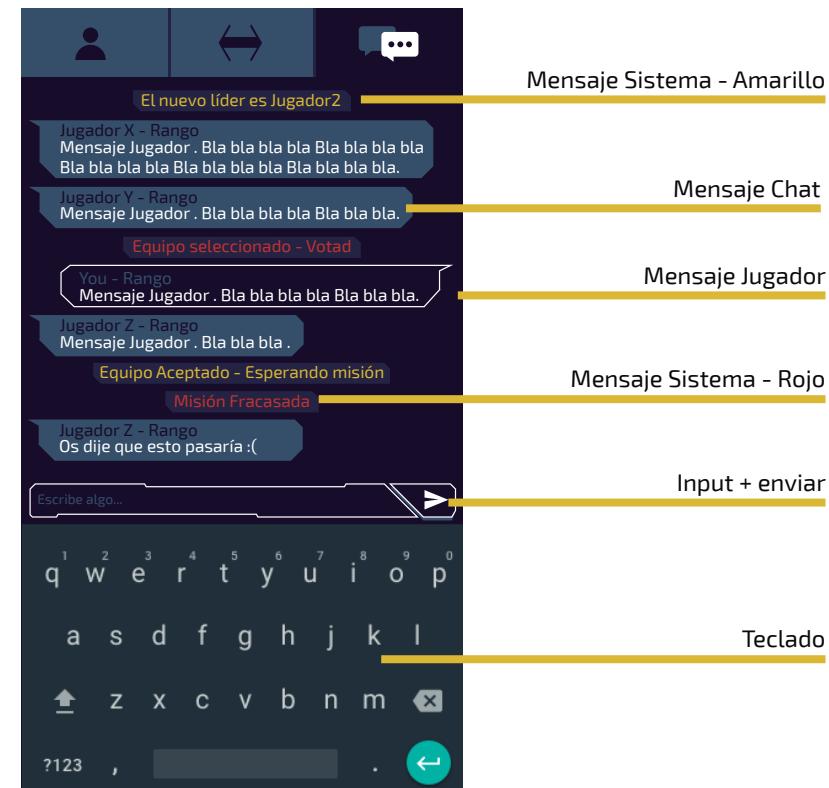


Juego principal

Perfil

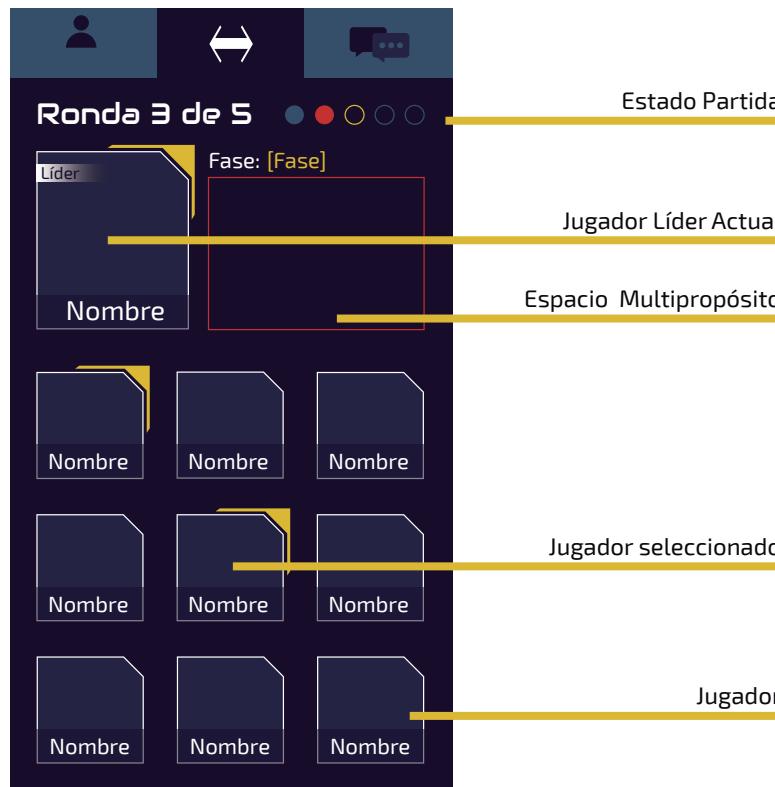


Chat



Juego principal

Pantalla Principal



Estado Partida

- Ronda - Éxito
- Ronda - Actual
- Ronda - Fallo
- Ronda - Restantes

Espacio Multipropósito

Este espacio adopta un aspecto según las necesidades de la fase actual del juego. Como se puede apreciar en los ejemplos.



Fin del Juego

Alcanzadas las condiciones de fin de juego la aplicación pasa a mostrar una de las siguientes pantallas dependiendo de la facción del jugador



Elección de tecnología

Rebellion es una Web App por lo que estará hecha en HTML, CSS y JS

Sin embargo usar la versión básica de Js no es la mejor opción debido a que el código sería muy extenso y complejo de redactar.

Por eso y porque se necesitaba que las variables estuviesen centralizadas se eligió Vue con Vuex.

Vue permite escribir cada componente por separado juntando en un mismo archivo el HTML, CSS y la lógica del mismo. Además permite la reutilización de componentes y el renderizado condicional que facilita mucho el flujo de trabajo.



Vue Js



Firebase

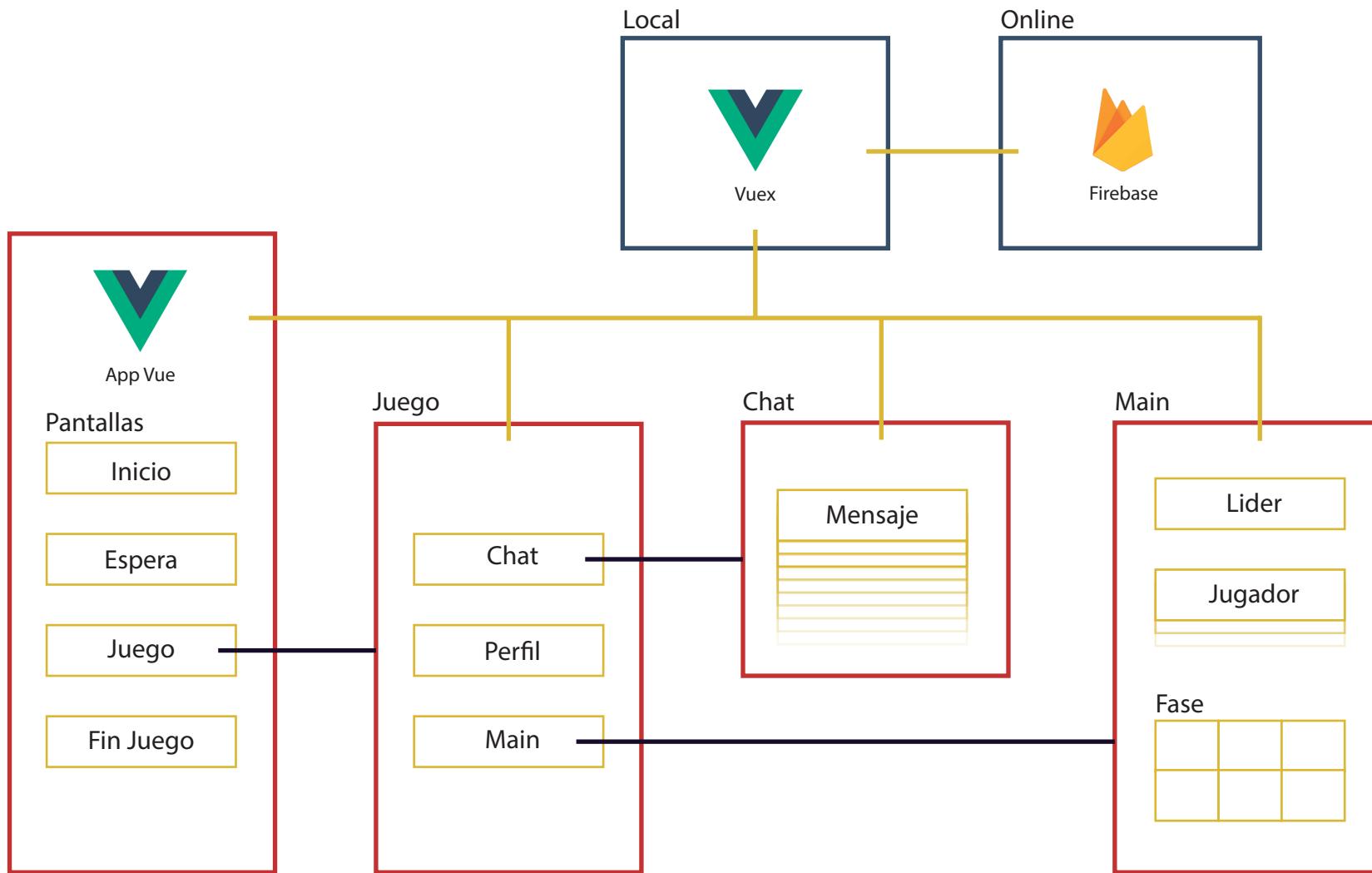
Para la interactividad entre jugadores era necesario que los clientes accediesen al jugador que crea la partida o bien, mantener los datos de manera centralizada en una base de datos.

Firebase contaba con una serie de características muy convenientes para este proyecto. Tales como ser en tiempo real y no requerir de ningún servidor específico.

Al ser en tiempo real la creación del chat es muy sencilla ya que los jugadores reciben los mensajes en muy poco tiempo permitiendo una comunicación fluida entre ellos.

Por otra parte la integración de Firebase con Vue es muy agradable y permite el desarrollo de una lógica avanzada que posibilita que todos los clientes estén sincronizados con los mismos datos.

Estructura App



Leyenda

- A Subcomponente
- Dependencia Datos
- Subcomponente
- Componente
- Datos

Ejemplos de código

Ejemplo del store Vuex con el estado inicial de las variables y una mutación.

```
export const store = new Vuex.Store({
  state: {
    // Datos Locales

    estados: ['inicio', 'espera', 'juego', 'fin'],
    punteroEstado: 0,
    pantallaActual: 'inicio',
    popUpEstado: true, // PopUp abierto/Cerrado

    idJugador: null,
    idPartida: null,

    db: db,

    // Datos on-line
    partidasCurso: {}, //objeto con las key partidas en curso
    datos: {
      iniciada:false,
      liderRechazadoVeces: 0,
      exitosMision: 0,
      faseActual: 0,
    },
    jugadores: {},
  },
  chat: { // Chat Partida Actual
  },
},
watch: {
  // meter watch por si la partida desaparece hacer algo
},
mutations: {
  irPantalla (state, pantalla) {
    state.punteroEstado = pantalla;
    state.pantallaActual = state.estados[state.punteroEstado];
  },
}
```

```
methods: {
  clickListo() {
    //Si el botón pone Empezar y hacemos click empezamos
    la partida
    if (this.estadoBoton === 'Empezar') {
      this.iniciarPartida(); // Set flag Partida Inicia-
      da
      this.siguientePantalla();
    }else {
      this.setListo();
      this.setNombre(this.nombreTemp);
    }
  },
}
```

Componente Espera.vue haciendo uso de la mutación mostrada en el apartado anterior.

Ejemplo del renderizado condicional de Vue en el componente Espera.js. Se puede apreciar como renderiza tantos jugadores como hay sin necesidad de complejas sentencias Js.

```
<h3>Jugadores:</h3>
<div id="salaEspera__contJugadores">
  <template v-for=" jugador in jugadores">
    <div class="jugador">
      <h5 class="nombre">{{jugador.nombre}}</h5>
      <h6 v-if="jugador.listo" class="estado
      amarillo">Listo</h6>
        <h6 v-else class="estado rojo">Esperando</h6>
      </div>
    </template>
  </div>
```

Ejemplos de código

Ejemplo de como la acción del Store.js CrearPartida hace el setup inicial y se prepara para escuchar los cambios que se efectuen en la base de datos

```
crearPartida(store, idPartida) {
    // Set id partida local con el introducido por el usuario
    store.state.idPartida = idPartida;

    // Creamos la partida en firebase
    store.state.db.ref(idPartida).set({
        datos: { iniciada:false, liderRechazadoVeces: 0,
        exitosMision: 0, faseActual: 0, },
        jugadores: false,
        chat: {
            '-LDXpb5nE0Y7ryqyUHw': {
                mensaje: 'Bienvenidos al chat de Rebellion',
                origen: 'sistema2',
            },
        },
    });
    // Vinculamos datos locales con db
    store.state.db.ref(store.state.idPartida
    +'/datos/').on("value", snapshot => {
        store.state.datos = snapshot.val();
    });
    // datos app
    store.state.db.ref(store.state.idPartida
    +'/jugadores/').on("value", snapshot => {
        store.state.jugadores = snapshot.val();
    });
    // jugadores
    store.state.db.ref(store.state.idPartida
    +'/chat/').on("child_added", snapshot => {
        store.state.chat[snapshot.key]= snapshot.val();
    });
}, // chat
},
```

Ejemplo de como el componente Chat.vue utiliza la función enviarMensaje para llamar a la acción que hace el push del mensaje a la base de datos.

Chat.vue - Methods

```
methods: {
    enviarMensaje() {
        this.pushMensaje({
            origen: this.jugadores[this.idJugador].nombre,
            mensaje: this.mensajeTemp,
            lider: this.jugadores[this.idJugador].lider
        });
        this.mensajeTemp = '';
    },
}
```

Store.js - vuex - actions

```
pushMensaje(store, {mensaje, origen, lider}){
    store.state.db
        .ref('/' + store.state.idPartida + '/chat/')
        .push({
            'origen': origen,
            'mensaje': mensaje,
            'lider': lider,
        });
},
```

About

Rubén Tortosa Zamora
33 años - Agosto 1984
Valencia

Ingeniero de Telecomunicaciones en Imagen y sonido
Diseñador y Desarrollador web
Mente inquieta

Rebellion

Web App
Juego interactivo social
Fomenta la discusión y el debate

Hace uso de unas de las tecnologías modernas e interesantes que existen ahora mismo en el mundo del desarrollo Web.

Como autor, empezar desde cero con estas tecnologías en el desarrollo del proyecto me ha permitido aprender muchos conceptos nuevos e interesantes.

