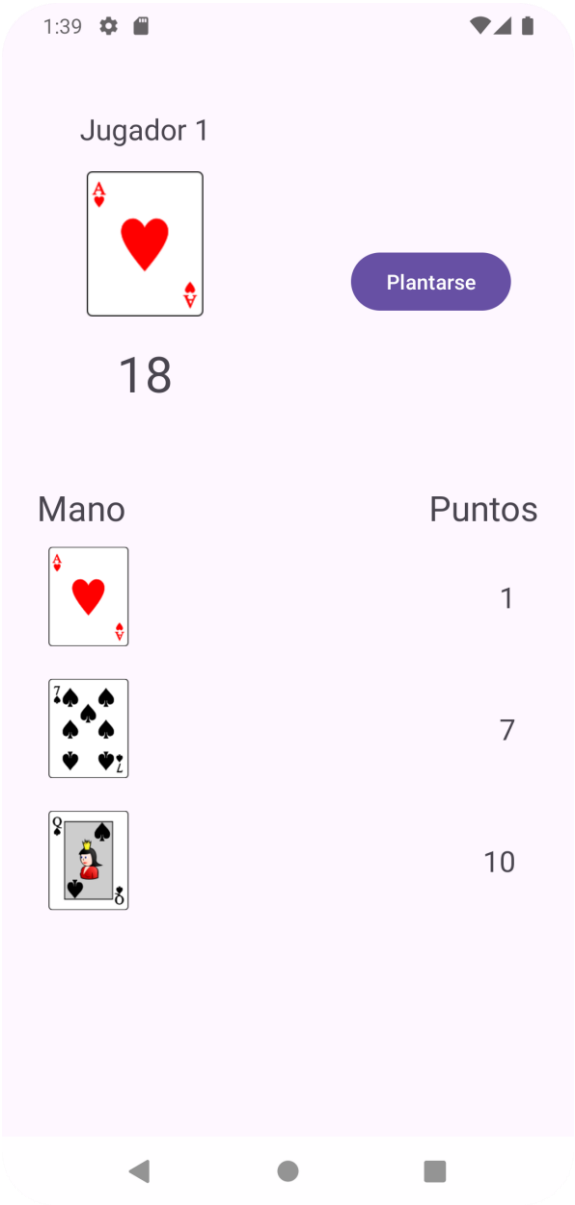
NOMBRE		ETAPA / CICLO	CURSO
		CFGS DAW/DAM	1º
APELLIDOS		ASIGNATURA/MÓDULO	CONVOCATORIA
		PROGRAMACIÓN	ORDINARIA
DNI	FECHA	NOTA	
	12-5-2025		

Realiza una aplicación Android para jugar al BlackJack con las siguientes especificaciones:

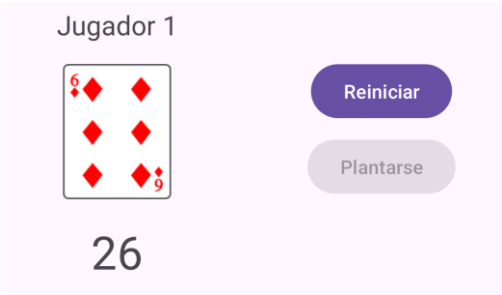
1. (1p) Crea una vista como la que se muestra en la siguiente imagen.



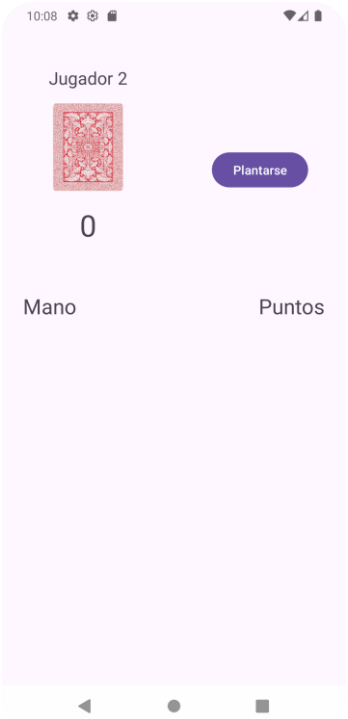
2. (1p) El jugador puede:
    - Pedir carta simplemente haciendo clic en la carta que está bajo el nombre del jugador.

    - Cuando el jugador hace clic en la carta, se debe obtener una carta de la baraja y añadir la carta a un recylerview, de tal manera que la última carta extraída se muestre en la primera posición.
    - Debajo de la carta debe mostrar la puntuación total del jugador como se muestra en la imagen anterior.
    - En el recyclerView, debe aparecer un listado de las cartas extraídas junto con la puntacion de cada carta. El as siempre vale 1 y las figuras 10. Resto de cartas su valor.

    - Si el jugador se pasa de 21, se deberá mostrar un Toast indicando que el jugador se ha pasado y deberá de impedir que el jugador pueda extraer más cartas, plantarse y deberá mostrar un botón para reiniciar la partida.



3. (1p) El jugador, siempre que no se haya pasado, podrá plantarse haciendo clic en el boton Plantarse. En este caso, se la App debera abrir una nueva Actividad donde jugara el Jugador2 con la misma baraja que se estaba jugando en la actividad anterior de tal manera que:



- Si el usuario cancela pulsando el botón atrás, se deberá retornar a la actividad1 y mostrar un Toast con el texto: "El otro jugador ha abandonado".
- Si el jugador2 se pasa, se deberá retornar a la actividad1 y mostrar un Toast con el texto: "El jugador2 se ha pasado"
- Si el jugador2 se planta, se deberá retornar a la actividad1 y mostrar con un Toast que jugador ha ganado o empatado si se da el caso.

En cualquier caso, cuando se vuelva a la actividad1 y mostrado el mensaje correspondiente, la app solo deberá dejar opción a reiniciar la partida.

4. (2p) Para poder realizar la app anterior se deberá tener las siguientes consideraciones:
   - Se debe disponer de una baraja con las siguientes propiedades:
     i. Un constructor que cree las 52 cartas. Para poder asignar cada imagen a la carta, se debe utilizar la variable images del Anexo III.
     ii. Un método getCarta() que obtenga la primera carta de la baraja.
     iii. Un método barajar() que desordene las cartas de la baraja.
   - Cada carta:
     i. Deberá tener un único valor y una imagen asociada. El valor de la carta es: 1 para los ases, 10 para las figuras, y resto el numero de la imagen.

   - Cada jugador debe tener una mano que:
     i. Debe almacenar las cartas que va extrayendo de la baraja obligatoriamente en un Set.
     ii. Debe poder mostrar las cartas según el orden en el que fueron extraídas.
     iii. Además, deberá tener las siguientes funciones:
       1. Poder añadir una carta en la primera posición.
       2. Poder saber cuantas cartas contiene la mano.

3. Obtener la carta que se encuentra en una determinada posición.
4. Reiniciar la mano y dejarla sin cartas.
5. Obtener la puntuación de la mano según las cartas que contenga.

Ayuda: Recuerda agregar un RecylerAdapter y un LinearLayoutManager al RecyclerView.
Ayuda2: Puedes crear cualquier clase que creas necesaria.

Annex 1 (Communication between activities)

```
ActivityResultLauncher<Intent> activityResultLauncher = registerForActivityResult(new
ActivityResultContracts.StartActivityForResult(),
    result -> {
        ...
            }
    );
```

Annex 2 (RecyclerView)

```
public class MyRecyclerViewAdapter extends RecyclerView.Adapter<MyRecyclerViewAdapter.MyViewHolder> {

    private LayoutInflater inflater;
    private Context context;

    public MyRecyclerViewAdapter(@NonNull Context context) {
        this.context = context;
        inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
    }

    @NonNull
    @Override
    public MyRecyclerViewAdapter.MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = inflater.inflate(R.layout.simple_element,parent,false);
        return new MyViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull MyRecyclerViewAdapter.MyViewHolder holder, int position) {

    }

    @Override
    public int getItemCount() {
        return 0;
    }

    public class MyViewHolder extends RecyclerView.ViewHolder{

        public MyViewHolder(@NonNull View itemView) {
            super(itemView);
        }
    }
}
```

Anexo III. Variable images

```
public static Map<String, List<Integer>> imagenes = Map.of(
    "club", List.of(R.mipmap.club_a, R.mipmap.club_2, R.mipmap.club_3, R.mipmap.club_4,
R.mipmap.club_5, R.mipmap.club_6, R.mipmap.club_7, R.mipmap.club_8, R.mipmap.club_9,
R.mipmap.club_10, R.mipmap.club_j, R.mipmap.club_q, R.mipmap.club_k),
    "diamond", List.of(R.mipmap.diamond_a, R.mipmap.diamond_2, R.mipmap.diamond_3,
R.mipmap.diamond_4, R.mipmap.diamond_5, R.mipmap.diamond_6, R.mipmap.diamond_7,
R.mipmap.diamond_8, R.mipmap.diamond_9, R.mipmap.diamond_10, R.mipmap.diamond_j,
R.mipmap.diamond_q, R.mipmap.diamond_k),
    "heart", List.of(R.mipmap.heart_a, R.mipmap.heart_2, R.mipmap.heart_3, R.mipmap.heart_4,
R.mipmap.heart_5, R.mipmap.heart_6, R.mipmap.heart_7, R.mipmap.heart_8, R.mipmap.heart_9,
R.mipmap.heart_10, R.mipmap.heart_j, R.mipmap.heart_q, R.mipmap.heart_k),
    "spade", List.of(R.mipmap.spade_a, R.mipmap.spade_2, R.mipmap.spade_3, R.mipmap.spade_4,
R.mipmap.spade_5, R.mipmap.spade_6, R.mipmap.spade_7, R.mipmap.spade_8, R.mipmap.spade_9,
R.mipmap.spade_10, R.mipmap.spade_j, R.mipmap.spade_q, R.mipmap.spade_k)
);
```