Desarrollo de una aplicación que facilita la enseñanza y el estudio del póker en su variante Texas Hold'em

Josué De León Santana
Tutor: José Juan Hernández Cabrera
Tutor: José Évora Gómez

Las Palmas de Gran Canaria, Diciembre de 2014

Agradecimientos

De lo primero que me acuerdo al escribir estas líneas es de la persona que no solo me crió sino que también estaba siempre pendiente de que pudiera tener un futuro y una buena vida para que nunca tuviera que pasar por la pobreza con la que él creció. Gracias abuelo.

A mi madre y a mi abuela, que han hecho posible que llegara hasta este punto apoyándome en todo momento, siendo un ejemplo de trabajo constante y de amor incondicional.

A Gabriela, mi novia, que ha aguantado mis días de agobio, mis incontables cambios de parecer y mis interminables charlas de informática y poker sin quejas y siempre con cariño.

A toda mi familia, si hay algo que nunca se queda atrás en el camino eso es la familia y yo doy gracias porque con nuestros más y nuestros menos siempre hemos sido una familia pequeña pero muy unida.

A la familia Sobrado Delgado, por tratarme desde el principio como uno más. Ellos también son partícipes de este proyecto.

A mis amigos, que siempre son una vía de escape por donde desaparece el estrés.

A mis tutores, sin los cuales este proyecto sin duda no sería lo mismo y de los cuales he aprendido infinidad de lecciones.

A todas aquellas personas que me han ayudado a que este proyecto salga adelante.

GRACIAS.

Índice

CAPÍTULO 1. INTRODUCCIÓN

Al leer este capítulo entenderemos de donde viene la idea de crear una aplicación en el entorno del juego y más concretamente del póker. Además serán detalladas las herramientas que se han utilizado para ello y se explica que competencias se han cubierto.

Motivación y contextualización

En el año 2008 descubrí el póker y fue un juego que me fascinó. Tenía un componente azaroso pero, al adentrarte un poco más, descubres que está lleno de estrategia y estadística. Además vives en un entorno en el que es muy importante basar todas tus decisiones en las matemáticas, ya que cometer una acción irracional con tu banca te puede llevar a la bancarrota. Hay muchos ejemplos de jugadores que han ganado mucho dinero pero al querer avanzar más rápido de lo establecido jugando partidas que no son adecuadas para su banca lo han perdido todo.

El siguiente paso en mi vida relacionado con el poker fue la fundación de la Asociación Canaria de Póker Amateur (a partir de ahora ACPA) en 2009, que llegaría a presidir en el periodo de 2011-2013. ACPA nace con unos objetivos muy claros: obtener el reconocimiento del poker como juego mental; su promoción como actividad lúdica, social y de competición; y educar e informar a los aficionados sobre su desarrollo en Canarias. A medida que ACPA se da a conocer se crean unas relaciones con los casinos de Canarias para la creación de un proyecto en común. Nace así Canarias Póker Tour, una serie de torneos de póker que van rotando por los siete casinos del archipiélago (por ese entonces). Había pasado de ser un jugador a ser un gestor del juego ya que dentro de las principales responsabilidades se encontraban el diseño previo de los torneos y su dirección.

Como obra final presento este trabajo de fin de título, PokerTrainer, para la plataforma Android, en el cual he plasmado los conocimientos básicos que he ido adquiriendo a lo largo de los últimos años de carrera para crear una herramienta útil, portable y entretenida para los jugadores noveles de póker que quieran aprender lo que hay más allá del juego en sí. Concretamente se centra en la variante Texas hold'em.

Texas hold'em es actualmente la variante más popular. En el juego a cada jugador se le reparten dos cartas ocultas y, una vez ocurrido esto, comienza una ronda de apuestas. Si hay más de un jugador sin retirarse de la mano después de esta ronda se ponen sobre la mesa tres cartas descubiertas y comunitarias. Mientras siga habiendo más de un jugador sin retirarse de la mano se repite el proceso de rondas de apuestas y se enseña una carta comunitaria. El límite son cinco cartas comunitarias. El jugador que tenga la mejor mano formada entre sus cartas ocultas y las cartas comunitarias o el que sea el único en no retirarse gana la mano y consigue las fichas que estén en el bote.

El póker no es un juego de instinto y azar, contrario a lo que mucha gente piensa es un juego matemático sobre todo basado en la estadística y es por ello que existen jugadores **profesionales**. También existen jugadores

regulares que, aunque no viven del juego, tampoco pierden dinero apostando. Si los anteriores no pierden nunca, ¿quiénes ponen dinero en el sistema?. Los jugadores **recreacionales**, que son los que usan el sistema con motivo de ocio son los que de verdad mantienen el ecosistema vivo.

Según el informe del primer trimestre de 2013 que generó la DGOJ (Dirección general de ordenación del juego), en España hay un total de 1.420.256 jugadores. De esta cantidad, la mayoría de ellos son jugadores recreacionales. Uno de los problemas con los que se encuentra la industria es la caída de nuevos jugadores (véase la Figura 1.1). Esto se debe, entre otras cosas por dos fenómenos: la aparición de las escuelas y la práctica del "bumhunting".

Figura 1.1: Análisis de la dirección general de juego sobre la evolución del poker en España en el último trimestre de 2014

Durante la época en la que explotó el boom del póker online entre 2003 y 2006 se empezaron a crear unas alianzas entre las salas de póker y algunos jugadores. Los jugadores atraían a nuevos jugadores a las salas y las salas a su vez concedían una parte de los beneficios que generan esos nuevos jugadores a la persona que los llevaba al sistema.

A raíz de esto nacen las escuelas de póker. La mejor forma de atraer a alguien a una sala era dotando de conocimientos a esa persona para que ésta no pierda su ingreso inicial y pueda seguir jugando y recaudando beneficios a la escuela y a la sala.

El término "bumhunting" aparece cuando cierto sector de jugadores se dedica a jugar exclusivamente contra jugadores recreacionales, buscándolos con ayuda de herramientas de datamining, y dejando de dar acción a los demás jugadores de la mesa en cuanto los jugadores más débiles se han quedado sin dinero.

La nueva irrupción de estos elementos cambió todo el ecosistema del póker online. La consecuencia ha sido el endurecimiento del juego en las mesas a lo largo de los años por dos motivos: los jugadores regulares son mejores gracias a las escuelas y cada vez hay menos jugadores recreacionales.

La conclusión ha sido que el jugador recreacional (el que realmente pone dinero en el sistema) que no dedica parte de su tiempo a estudiar el juego

pierde su dinero cada vez más rápido y, debido a ello, sus expectativas de ocio no se cumplen.

Dado este análisis y, aunque no se plantee durante este trabajo de fin de título, si la aplicación tiene buena acogida podría plantearse un modelo de negocio con un nicho de mercado con los jugadores recreacionales por objetivo.

La idea

Hemos hablado del por qué del contexto del proyecto y también de los deseos del desarrollo móvil. Además se introdujo un pequeño análisis del estado actual del mercado. Pero ¿cómo se llegó hasta esa idea? ¿fue siempre la primera opción?.

A decir verdad no. El proyecto se empapó desde un principio de los principios ágiles y, como bien dice uno de ellos aceptamos el cambio como parte del trabajo. La idea inicial era desarrollar una aplicación que analizara una base de datos con jugadas y devolviera al usuario un indicador sobre si la jugada era o no correcta estadísticamente hablando pero, tras un estudio previo del mercado, los tutores y yo mismo decidimos que esa idea no encajaba bien con el nicho que previamente habíamos descubierto (los jugadores **recreacionales**) ya que ellos no utilizan software destinado a guardar sus jugadas de póker.

Al final tras decidir que ese era el mercado objetivo, nos decidimos por un software para los jugadores recreacionales, que, como si se tratara de un juego pudiesen elevar su nivel de póker. Además nos decidimos por la plataforma Android.

Objetivos

De acuerdo con las motivaciones detalladas anteriormente se ha creado PokerTrainer, una aplicación móvil desarrollada para la plataforma Android que expone a los jugadores distintos aspectos importantes que mejorarán su juego. Para llegar a los jugadores recreacionales se desarrollan los temas en forma de fases con distintos quiz que permiten aprender de manera divertida conceptos que los jugadores regulares han tenido que estudiar mediante tablas, artículos y vídeos.

En segundo término se encuentra el objetivo académico. La ilusión desde que empezó a gestarse el proyecto siempre fue desarrollar una aplicación móvil (tecnología que no había sido aprendida durante la carrera) y crear un modelo relacionado con el mundo del póker aplicando los conocimientos que he aprendido en la intensificación de ingeniería del software.

Marco teórico

Para el desarrollo del proyecto se ha considerado el uso de determinados conocimientos adquiridos durante la carrera así como otros elementos que han sido esenciales para la mejora en la implementación. Todos estos elementos/conceptos serán explicados en este apartado.

Patrones de diseño

Base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su **efectividad** resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser **reutilizable**, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

Los patrones de diseño pretenden:

- Proporcionar catálogos de elementos reusables en el diseño de sistemas software.
- Evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente.
- Formalizar un vocabulario común entre diseñadores.
- Estandarizar el modo en que se realiza el diseño.
- Facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente.

Asimismo, no pretenden:

- Imponer ciertas alternativas de diseño frente a otras.
- Eliminar la creatividad inherente al proceso de diseño.

No es obligatorio utilizar los patrones, sólo es aconsejable en el caso de tener el mismo problema o similar que soluciona el patrón, siempre teniendo en cuenta que en un caso particular puede no ser aplicable. "Abusar o forzar el uso de los patrones puede ser un error".

Código limpio

Resulta difícil crear una definición precisa de código limpio, y seguramente existan tantas definiciones como desarrolladores. Sin embargo, existen algunos principios que llevan a lograr un nivel básico de código limpio. Las 9 prácticas más relevantes para lograr código limpio a continuación.

1. El código malo hace demasiadas cosas - el código limpio es enfocado

2. El lenguaje con el que se escribió el código debería parecer que fue hecho para el problema
3. No debe ser redundante
4. Debe ser placentero leer el código
5. Puede ser extendido fácilmente por cualquier otro desarrollador
6. Debe tener dependencias mínimas
7. Más chico, mejor
8. Debe tener pruebas unitarias y de aceptación
9. Debe ser expresivo

Uso de herramientas profesionales

El uso de las herramientas adecuadas puede ser un factor importante para la consecución de un gran proyecto. Algunas de las herramientas usadas a lo largo del desarrollo son:

Hockeyapp

Distribución de aplicaciones: Una vez subas tus aplicaciones a HockeyApp los testers podrán descargarlas al instante e instalarlas. Además podrás tener total control sobre quienes pueden acceder a estas. Fue clave durante el proyecto para que los testers pudiesen descargar el ejecutable de la aplicación, instalarlo en sus dispositivos Android y devolver feedback.

Bitbucket

Servicio de alojamiento basado en web, para los proyectos que utilizan el sistema de control de versiones Mercurial y Git.
Se llama **control de versiones** a la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Una versión, revisión o edición de un producto, es el estado en el que se encuentra el mismo en un momento dado de su desarrollo o modificación.

SourceTree

Cliente de escritorio para manejar repositorios Git o Mercurial. Herramienta gracias a la cual dejaremos de temer los cambios en el código ya que permite cargar versiones anteriores del proyecto, subir versiones nuevas al repositorio y hacer anotaciones en cada una de ellas.

Android Studio

Entorno de desarrollo (IDE) para la plataforma Android basado en IntelliJ IDEA. Provee de nuevas funcionalidades y mejoras sobre el Android Development Tools de Eclipse y será el IDE oficial una vez que esté totalmente listo.

Principios SOLID

Es un acrónimo mnemónico introducido por Robert C. Martin a comienzos de la década del 2000 que representa cinco principios básicos de la programación orientada a objetos y el diseño.

Principio de responsabilidad única (Single responsability principle): Cada clase debe tener una única responsabilidad, y que esta debe estar contenida únicamente en la clase.

Principio abierto/cerrado (Open/Close principle): Las entidades de software deben estar abiertas para su extensión, pero cerradas para su modificación.

Principio de sustitución de Liskov (Liskov substitution principle): Los objetos de un programa deberían ser reemplazables por instancias de sus subtipos sin alterar el correcto funcionamiento del programa.

Principio de segregación de la interface (Interface segregation principle): Muchas interfaces cliente específicas son mejores que una interfaz de propósito general.

Principio de inversión de dependencia (Dependency inversion principle): Depender de abstracciones. No depender de concreciones.

Filosofía Ágil

El desarrollo ágil de software hace referencia a un conjunto de métodos de ingeniería del software basados en el desarrollo iterativo e incremental, donde los requerimientos y soluciones evolucionan mediante la colaboración de grupos auto organizados y multidisciplinarios. Existen muchos métodos de desarrollo ágil pero todos están basados en los principios del manifiesto ágil.

Principios del Manifiesto Ágil

- Individuos e interacciones sobre procesos y herramientas
- Software funcionando sobre documentación extensiva
- Colaboración con el cliente sobre negociación contractual
- Respuesta ante el cambio sobre seguir un plan.

Esto es, aunque valoramos los elementos de la derecha, valoramos más los de la izquierda.

Historias de usuario

Una de las principales innovaciones que representa el desarrollo ágil frente a los enfoques tradicionales, es en la forma de levantar los requerimientos del usuario. A diferencia del enfoque tradicional, en el cual el "Diseño de Sistema" contiene documentación detallada del comportamiento y representa el final de las conversaciones, en las metodologías ágiles se hace uso de las "**Historias de usuario**", las cuales se enfocan en definir lo que el usuario necesita hacer, sin describir el cómo, por lo que representa el inicio y no el final de las conversaciones.

1.- ¿Qué son las historias de usuario?

Las historias de usuario, son descripciones cortas de una necesidad de un cliente del software que estemos desarrollando. Su utilización es común cuando se aplican marcos de trabajo ágiles, tales como **Scrum** o el **Extreme Programming (XP)**.

2.- ¿Cómo se redactan las historias de usuario?

Al redactar una historia de usuario deben tenerse en cuenta describir el Rol, la funcionalidad y el resultado esperado de la aplicación en una frase corta. Adicionalmente, debe venir acompañada de los criterios de aceptación, no más de 4 por historia, redactado también en una frase que incluya el contexto, el evento y el comportamiento esperado ante ese evento.

3.- ¿Qué características deben tener las historias de usuario?

Algunas características deseables de las historias de usuario son:

- Que sean escritas por el usuario.
- Frase corta que encaje en una tarjeta de 3 por 5 pulgadas.
- Debe describir el rol desempeñado por el usuario en el sistema, descrito de forma explícita.
- Debe describir el beneficio para el área de negocio que representa esta funcionalidad.

4.- ¿En qué se diferencian de otros instrumentos de levantamiento de requerimientos?

No debemos confundirlas con Casos de Uso o Escenarios de uso, la gran diferencia, es que son más cortas y no deben describir la interfaz con el usuario, los pasos de navegación o flujo de procesos de la aplicación.

5.- ¿Para qué se utilizan?

Su propósito principal de esta herramienta es la estimación del esfuerzo necesario para implementar una nueva funcionalidad (Feature), en un software, siguiendo la definición de "Hecho" (DONE) que defina el equipo.

Adicionalmente, se utilizan como iniciadoras de conversaciones entre desarrolladores de software y usuarios del área de negocio, las cuales servirán para identificar los requerimientos del negocio, requerimientos técnicos y para encontrar los supuestos (premisas) no visibles en una primera aproximación. Su propósito, no es describir en detalle la funcionalidad.
Para estas estimaciones, se le asignan unidades de medida a las historias que representen magnitud y no días reales de duración, tales como: puntos de historia, días ideales u otra unidad de medida.

Diseño de la arquitectura

El diseño es una actividad en la que se toman decisiones importantes, frecuentemente de naturaleza estructural. Comparte con la programación un interés por la abstracción de la representación de la información y de las secuencias de procesamiento, pero el nivel de detalle es muy diferente en ambos casos. El diseño construye representaciones coherentes y bien planificadas de los programas, concentrándose en las interrelaciones de los componentes de mayor nivel y en las operaciones lógicas implicadas en los niveles inferiores.

En el diseño se modela el sistema y se encuentra su forma (incluida la arquitectura) para que soporte todos los requisitos —incluyendo los requisitos no funcionales y otras restricciones— que se le suponen.

Dentro del diseño podemos encontrar, entre otras cosas, los **diagramas de clase**. Un diagrama de clases es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargarán del funcionamiento y la relación entre uno y otro.

Otro de los elementos importantes dentro del diseño son los diagramas de paquetes. En el Lenguaje Unificado de Modelado, un **diagrama de paquetes** muestra cómo un sistema está dividido en agrupaciones lógicas mostrando las dependencias entre esas agrupaciones.

Dado que normalmente un paquete está pensado como un directorio, los diagramas de paquetes suministran una descomposición de la jerarquía lógica de un sistema.

Los paquetes están normalmente organizados para maximizar la coherencia interna dentro de cada paquete y minimizar el acoplamiento externo entre los paquetes. Con estas líneas maestras sobre la mesa, los paquetes son buenos elementos de gestión. Cada paquete puede asignarse a un individuo o a un equipo, y las dependencias entre ellos pueden indicar el orden de desarrollo requerido.

Gamificación

Gamificación (gamification en el ámbito anglosajón) es el empleo de mecánicas de juego en entornos y aplicaciones no lúdicas con el fin de potenciar la motivación, la concentración, el esfuerzo, la fidelización y otros valores positivos comunes a todos los juegos. Se trata de una nueva y poderosa estrategia para influir y motivar a grupos de personas.

La eclosión de la web 2.0 ha acelerado la creación de comunidades en torno a todo tipo de redes sociales, medios digitales o webs corporativas. Pero no

siempre es fácil estimular la actividad dinámica y frecuente entre los miembros de una comunidad.

Una correcta implementación de estrategias de gamificación permite pasar de la mera conectividad al engagement (o compromiso), logrando que los miembros de una comunidad, los trabajadores de una empresa, los estudiantes de un instituto, los habitantes de una ciudad –prácticamente cualquier colectivo o individuo- participen de manera dinámica y proactiva en acciones que generalmente requieren un esfuerzo de la voluntad.

La integración de dinámicas de juego en entornos no lúdicos no es un fenómeno nuevo, pero el crecimiento exponencial del uso del videojuego en los últimos años ha despertado el interés de expertos en comunicación, psicología, educación, salud, productividad –y casi cualquiera área humana- por descifrar las claves que hacen del videojuego un medio tan eficaz. En estos últimos años ha comenzado también la expansión en el estudio de su aplicación a otros ámbitos no necesariamente lúdicos. Gamificación es el término escogido para definir esta tendencia.

Justificación de las competencias cubiertas

CII01. Capacidad para diseñar, desarrollar, seleccionar y evaluar aplicaciones y sistemas informáticos, asegurando su fiabilidad, seguridad y calidad, conforme a principios éticos y a la legislación y normativa vigente.

PokerTrainer se ha desarrollado como una aplicación móvil para dispositivos con el sistema operativo Android. Su implementación se ha desarrollado en cuatro etapas:

- Análisis de requisitos: En el cual se han definido las historias de usuarios basadas en el problema a resolver y donde se han tomado las decisiones base sobre cómo abordar dicho problema.
- Diseño y arquitectura: Donde se define la arquitectura del software y se realiza un diseño detallado de los datos e interfaces.
- Desarrollo: Donde se ha implementado la solución a las historias de usuario mediante el diseño previo.
- Pruebas: Donde se definen las pruebas necesarias para validar el modelo de la aplicación.

CII02. Capacidad para planificar, concebir, desplegar y dirigir proyectos, servicios y sistemas informáticos en todos los ámbitos, liderando su puesta en marcha y su mejora continua y valorando su impacto económico y social.

Este proyecto ha sido concebido con la idea de agrupar principios importantes del desarrollo de ingeniería del software y, como tal, ha sufrido distintos cambios a medida que se implementaba, consiguiendo así un código más limpio y una mejora continua en las prestaciones. Además también ha sufrido cambios importantes en etapas avanzadas debido al feedback recogido por distintos betatesters.

CII18. Conocimiento de la normativa y la regulación de la informática en los ámbitos nacional, europeo e internacional.

Un análisis de las legislaciones que se han promulgado en diversos países arroja que las normas jurídicas que se han puesto en vigor están dirigidas a proteger la utilización abusiva de la información reunida y procesada mediante el uso de computadoras, e incluso en algunas de ellas se ha previsto formar órganos especializados que protejan los derechos de los ciudadanos amenazados por los ordenadores.

En cuanto a la regulación en el ámbito del juego es interesante comprobar como han habido cambios recientes provocados desde la Dirección General de la Ordenación del Juego en España con la inclusión de tres nuevas resoluciones que han entrado en vigor en Octubre de 2014 y que desarrollan

y marcan las pautas a seguir en relación a los requisitos técnicos de las actividades del juego online. Con estas tres resoluciones se ha terminado de desarrollar el Real Decreto 1613/2011 con instrucciones claras y precisas de cómo se debe recopilar y aportar la información necesaria para el control de la actividad en una sala online y cómo se debe comportar el software de los operadores a la hora de identificar y dar servicio al usuario.

TFG01. Ejercicio original a realizar individualmente y presentar y defender ante un tribunal universitario, consistente en un proyecto en el ámbito de las tecnologías específicas de la Ingeniería en Informática de naturaleza profesional en el que se sinteticen e integren las competencias adquiridas en las enseñanzas

PokerTrainer es una aplicación para dispositivos móviles que soporten el sistema operativo Android. En el desarrollo se han integrado muchos de los conocimientos adquiridos a lo largo de los años de enseñanza. Ha sido desarrollado individualmente.

T1. Capacidad para concebir, redactar, organizar, planificar, desarrollar y firmar proyectos en el ámbito de la ingeniería en informática que tengan por objeto, de acuerdo con los conocimientos adquiridos según lo establecido en apartado 5 de la resolución indicada, la concepción, el desarrollo o la explotación de sistemas, servicios y aplicaciones informáticas. (G1, G2)

Este primer punto se pone de manifiesto con la elaboración de la memoria.

T5. Capacidad para concebir, desarrollar y mantener sistemas, servicios y aplicaciones informáticas empleando los métodos de la ingeniería del software como instrumento para el aseguramiento de su calidad, de acuerdo con los conocimientos adquiridos según lo establecido en apartado 5 de la resolución indicada. (G1, G2)

Uno de los aspectos importantes del desarrollo de la aplicación ha sido plasmar los conocimientos adquiridos en la intensificación de Ingeniería del Software para promover y desarrollar software limpio, ágil, legible, modulable, abierto a cambios, etc.

T8. Conocimiento de las materias básicas y tecnologías, que capaciten para el aprendizaje y desarrollo de nuevos métodos y tecnologías, así como las que les doten de una gran versatilidad para adaptarse a nuevas situaciones.

Se demuestra habiendo realizado una aplicación con una tecnología no aprendida directamente en la docencia del Grado en Ingeniería Informática.

T11. Capacidad para analizar y valorar el impacto social y medioambiental de las soluciones técnicas, comprendiendo la responsabilidad ética y profesional de la actividad del Ingeniero Técnico en Informática.

Uno de los aspectos a valorar fue que valor podría añadir y qué impacto podría tener la aplicación en el ámbito para el que se desarrolla.

Capítulo 2. El producto

En este capítulo se mostrará cuál es la ficha del proyecto, el análisis final y la arquitectura de la aplicación.

Ficha de PokerTrainer

Título: PokerTrainer
Descripción breve: Aplicación que te enseñará conceptos básicos sobre Texas Hold'em.
Descripción completa: PokerTrainer es una aplicación dirigida a aquellos jugadores de poker en su variante Texas Hold'em que quieran aprender ciertos conceptos básicos del juego que les ayudará a mejorar. Se basa en dar un cierto concepto y generar un quiz posterior donde el usuario tendrá que validar que realmente ha aprendido la lección para poder pasar al siguiente nivel.
Tipo de aplicación: Juegos
Categoría: Cartas
Clasificación de contenidos: Nivel de madurez medio

Elementos gráficos:

Análisis

Para el análisis del problema se han utilizado las historias de usuario (véase marco teórico). La recopilación de historias es la que sigue:

1. Yo, como un jugador, necesito poder ver la lista de niveles con la finalidad de ver los niveles disponibles.
   Criterios de aceptación:
   Debe haber uno o más niveles disponibles.

2. Yo, como un jugador, necesito poder ver la introducción a la lección de "calles" con la finalidad de conocer la introducción a dicho nivel.
   Criterios de aceptación:
   El nivel está bloqueado
   El nivel está desbloqueado

3. Yo, como un jugador, necesito poder ver la introducción a la lección de "categorías" con la finalidad de conocer la introducción a dicho nivel.
   Criterios de aceptación:
   El nivel está bloqueado
   El nivel está desbloqueado

4. Yo, como un jugador, necesito poder ver la introducción a la lección de "fuerza de la mano" con la finalidad de conocer la introducción a dicho nivel.
   Criterios de aceptación:
   El nivel está bloqueado
   El nivel está desbloqueado

5. Yo, como un jugador, necesito poder ver la introducción a la lección de "tipos de mesa" con la finalidad de conocer la introducción a dicho nivel.
   Criterios de aceptación:
   El nivel está bloqueado
   El nivel está desbloqueado

6. Yo, como un jugador, necesito poder acceder al tutorial de un nivel con la finalidad de comenzar a aprender la lección.
   Criterios de aceptación:
   Se ha accedido al tutorial

7. Yo, como un jugador, necesito poder navegar por el tutorial de un nivel con la finalidad de acceder a todos los contenidos de dicho nivel.
   Criterios de aceptación:
   Se ha accedido al tutorial

8. Yo, como un jugador, necesito poder saltarme el tutorial de un nivel con la finalidad de desbloquear el quiz sin leerme todos los contenidos de dicho nivel.
    Criterios de aceptación:
        Se ha accedido al tutorial

9. Yo, como un jugador, necesito poder acceder al quiz de un nivel con la finalidad de conocer el quiz para pasar el nivel.
    Criterios de aceptación:
        El quiz está bloqueado
        El quiz está desbloqueado

10. Yo, como un jugador, necesito poder contestar a las preguntas del quiz de un nivel con la finalidad de pasar el nivel y saber mi resultado.
    Criterios de aceptación:
        Se ha accedido al quiz

Diseño de la aplicación

Como se ha explicado en el marco teórico en este apartado se detallarán el dominio del problema, los diagramas de clase y los diagramas de paquetes. Además también se mostrarán las interfaces de usuario.

Dominio del problema

Para poder entender mejor los diagramas, en esta sección se explicarán los conceptos más importantes que se han utilizado.

Card: hace referencia al objeto carta.
Deck: hace referencia al objeto baraja. Se utiliza la referencia de la baraja inglesa, que es la utilizada en el ámbito del póker.
CardSuit: hace referencia al palo de una carta.
Board: hace referencia a las cartas comunitarias que están a la vista.
Street: nombre que recibe el Board según el número de cartas comunitarias a la vista que haya.
Table: hace referencia al objeto mesa de póker.
Hand: interfaz que se usa para referirse a cualquier objeto llamado "mano" en el póker.
PokerHand: hace referencia a una mano de póker formada por cinco cartas.
TexasHand: hace referencia a las dos cartas ocultas que son repartidas a cada jugador en la variante Texas Hold'em.
Player: hace referencia a un jugador de póker.
Scene: hace referencia a un determinado momento dentro de una partida de póker en el que hay una mesa con un board concreto y unos jugadores.
Indicator: hace referencia al un término relacionado con algunos de los elementos explicados antes. Por ejemplo, el nombre de la categoría de un mano de póker será un indicador.

Diagramas de clase

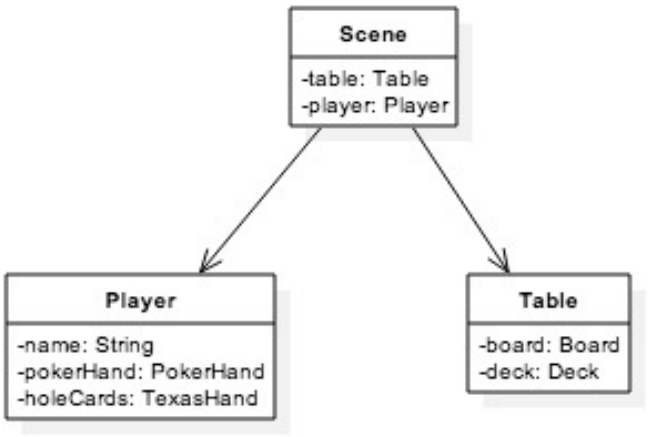

Figura 2.1: Escena.

Los diagramas a continuación forman parte del paquete de modelo.

Una escena (figura 2.1) está compuesta de un jugador con unas determinadas cartas ocultas y una mesa de póker que tiene unas determinadas cartas comunitarias a la vista.

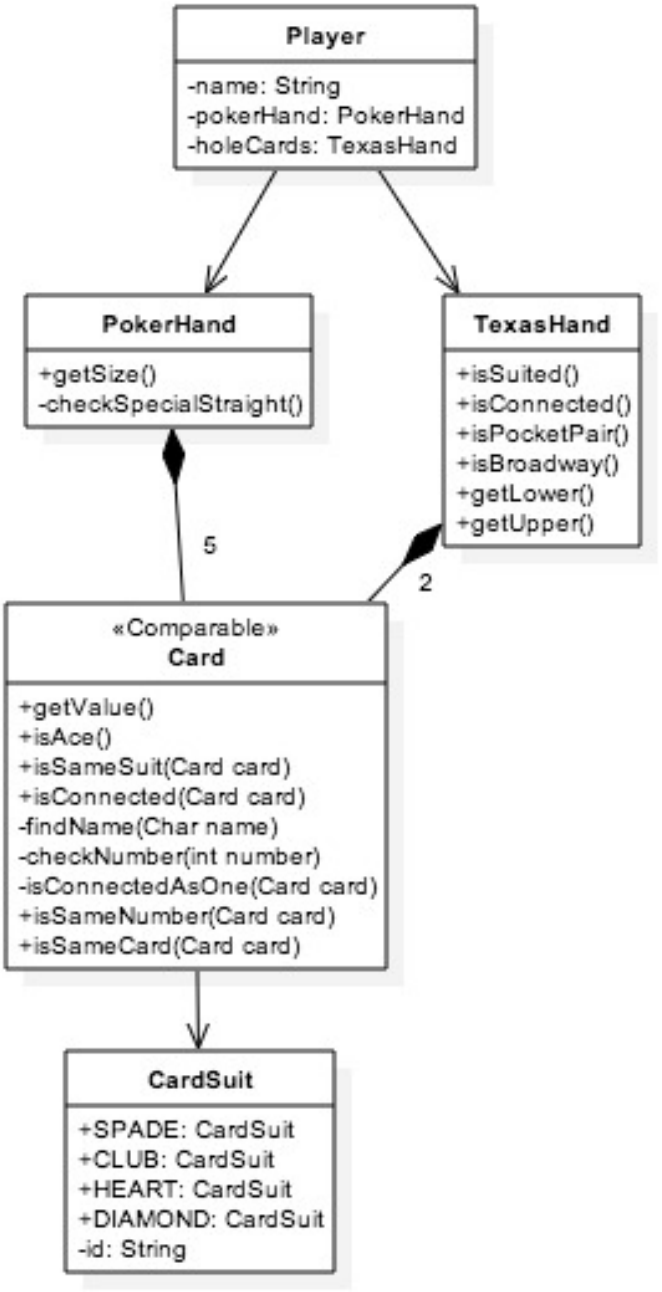

Figura 2.2: Jugador.

Un jugador (figura 2.2) tiene unas cartas ocultas y una mano de póker. Los dos atributos están compuestos de cartas.

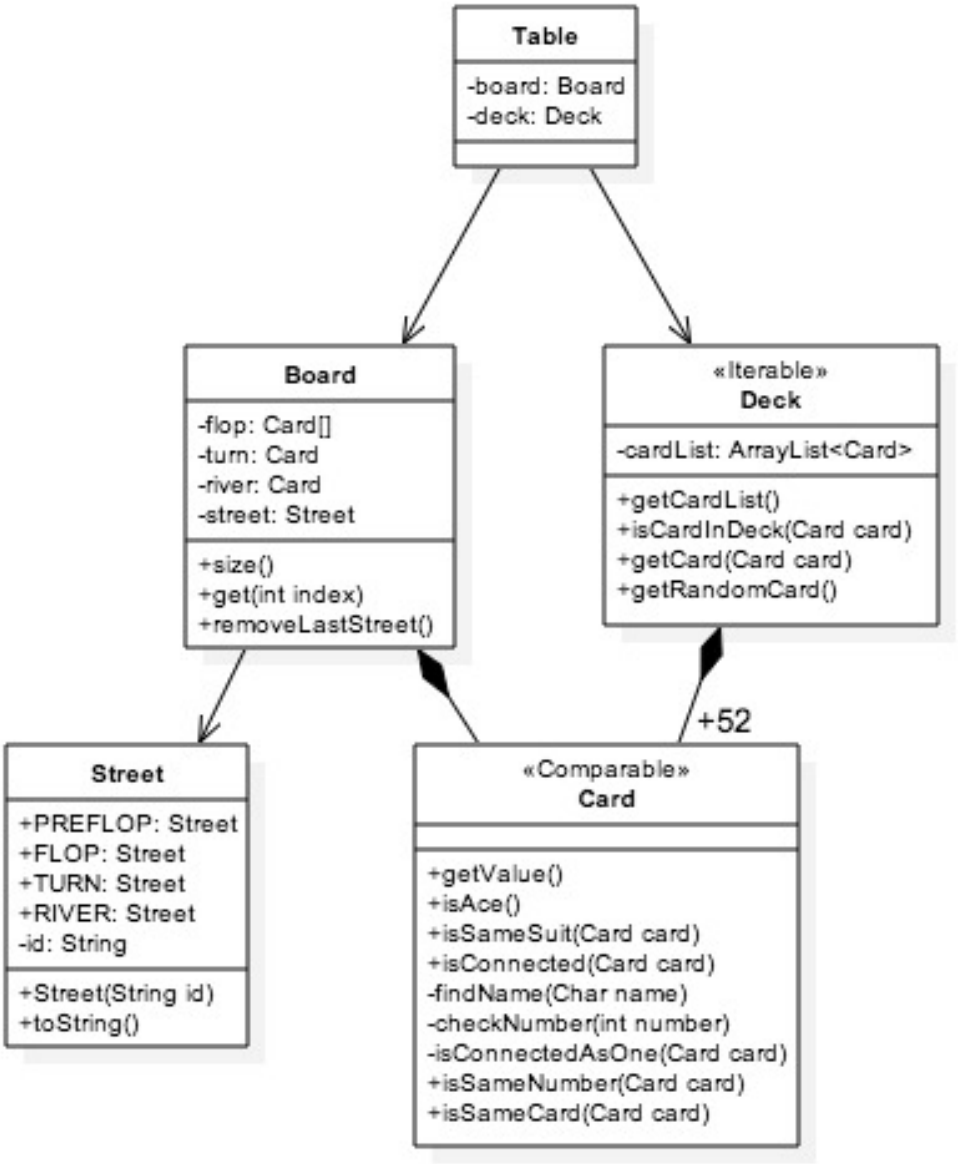

Figura 2.3: Mesa de poker.

Una mesa de póker (figura 2.3) está definida por unas cartas comunitarias y contiene una baraja única.

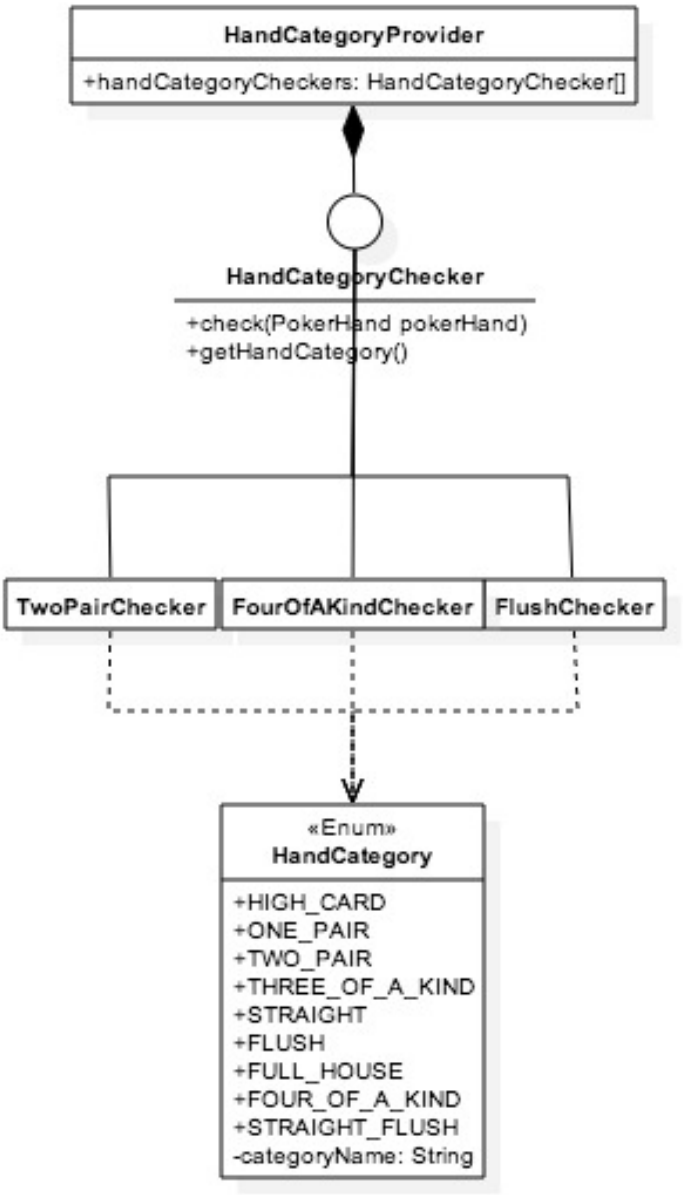

Figura 2.4: Comprobador de categorías

El comprobador de categorías (figura 2.4) se encarga de devolver a que categoría pertenece una determinada mano de póker. El proveedor (la clase HandCategoryProvider) será el que interactúe con el cliente haciendo así a la interfaz HandCategoryChecker transparente.

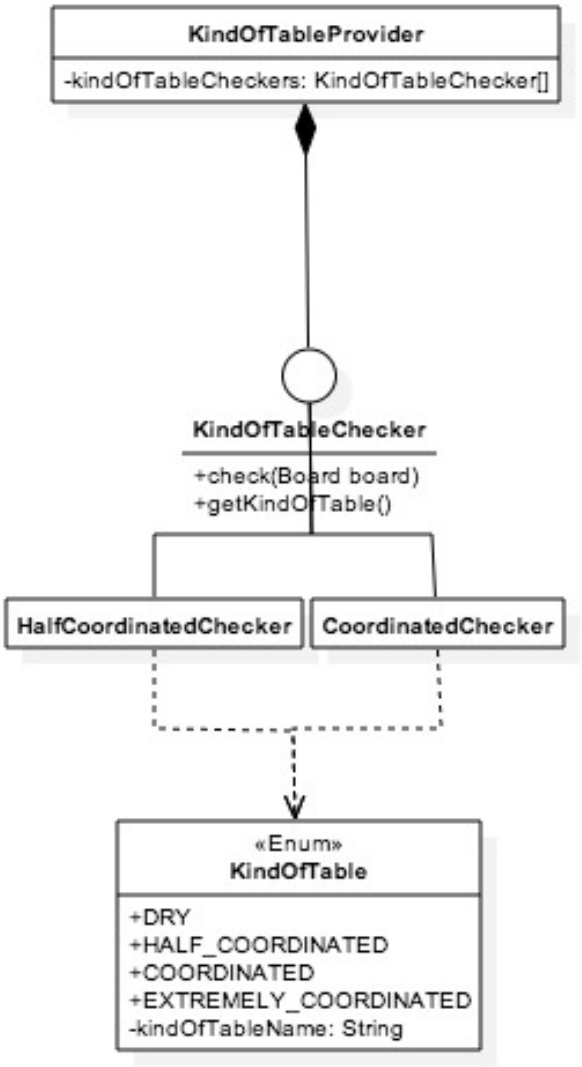

Figura 2.5: Comprobador de tipos de mesa

El comprobador de tipos de mesa (figura 2.5) utiliza el mismo patrón que el comprobador de categorías (figura 2.4) pero esta vez se encarga de devolver el tipo de mesa según las cartas comunitarias que se encuentren a la vista en ese momento.

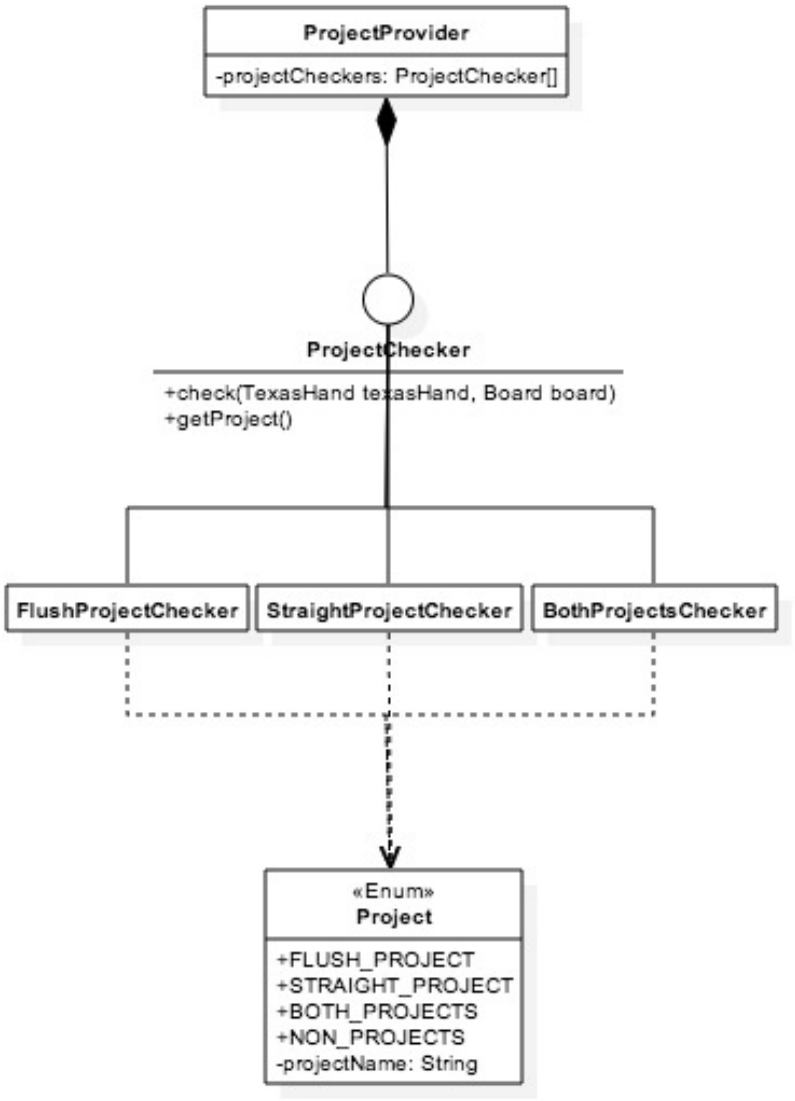

Figura 2.6: Comprobador de proyectos

El comprobador de proyectos (figura 2.6) funciona igual que los anteriores comprobadores, sólo que ahora se encarga de devolver el tipo de proyecto/s que tiene un jugador según sus cartas ocultas y las cartas comunitarias que hay a la vista.

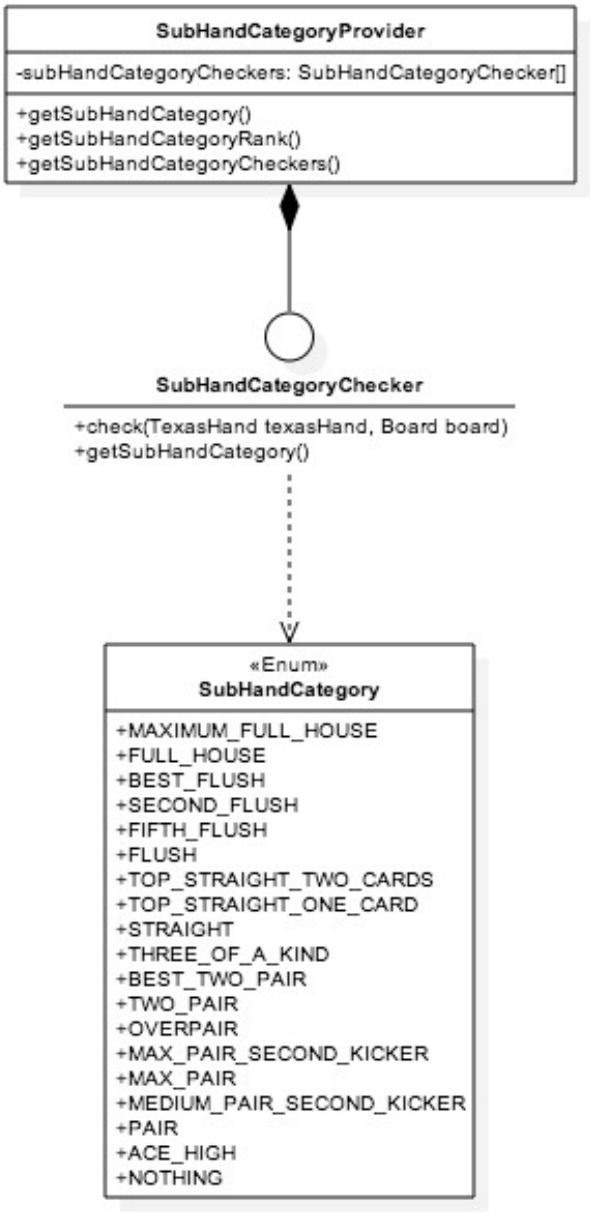

Figura 2.7: Comprobador de subtipos de categorías

El comprobador de subtipos de categorías (figura 2.7) se encarga de categorizar las manos en subtipos para ser más preciso a la hora de analizar la fuerza de la mano.

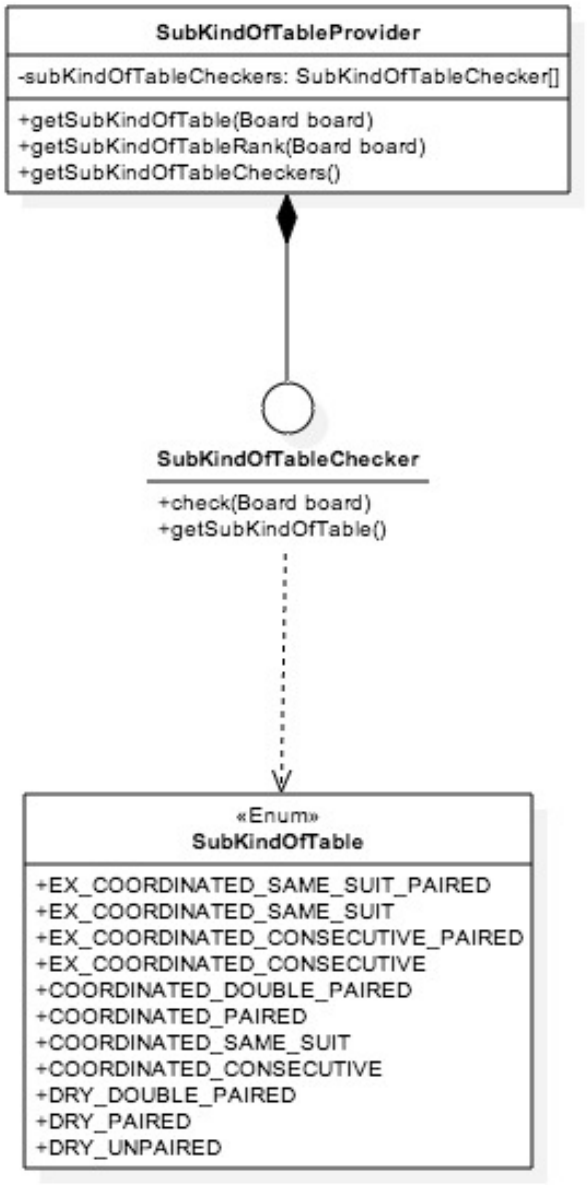

Figura 2.8: Comprobador de subtipos de mesa

El comprobador de subtipos de mesa (figura 2.8) al igual que el comprobador de subtipos de categorías (figura 2.7) categoriza los tipos de mesa en subtipos con el mismo fin.

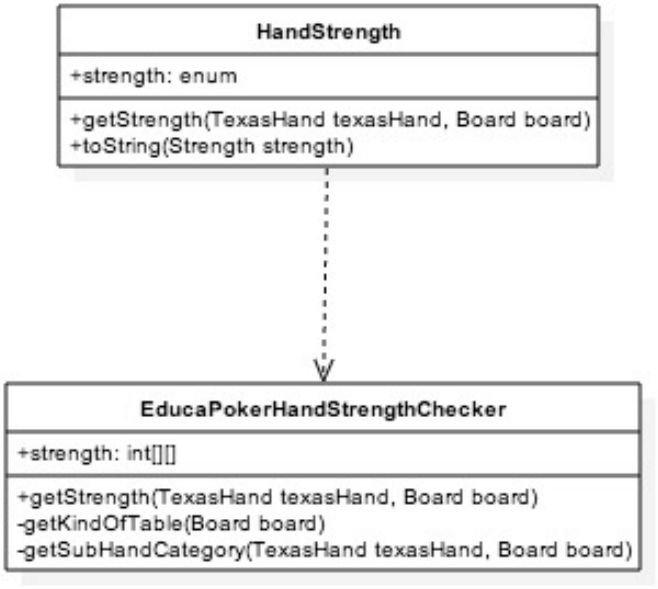

Figura 2.9: Comprobador de fuerza de la mano

El comprobador de fuerza de la mano (figura 2.9) se ayuda de los comprobadores de subtipos de categorías y mesas para calcular la fuerza de la mano que tiene un determinador jugador con unas determinadas cartas comunitarias.

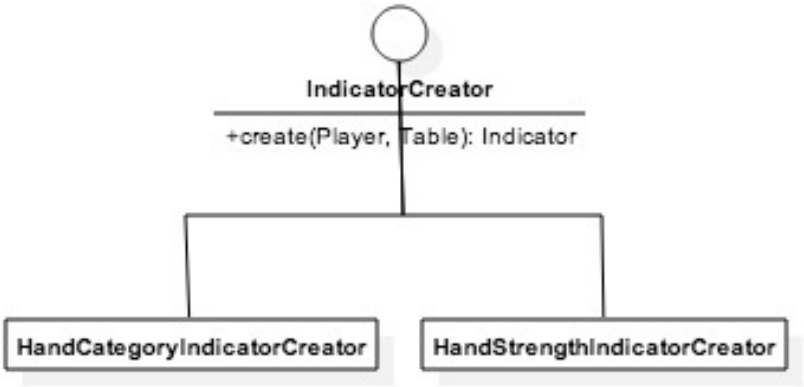

Figura 2.10: Creador de indicadores

La interfaz IndicatorCreator (figura 2.10) se encarga de crear los indicadores que queremos enseñar al usuario. Por tanto coinciden con las fases de la aplicación. Para aumentar niveles habrá que crear más indicadores que implementan el método create.

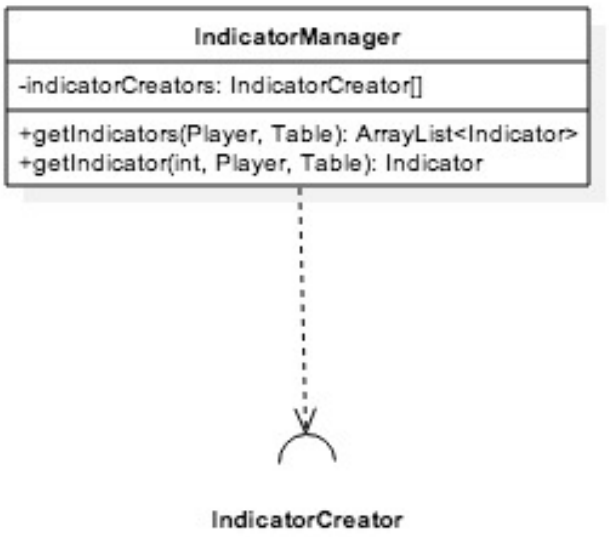

Figura 2.11: Manejador de indicadores

El controlador IndicatorManager (figura 2.11) es el encargado de generar que la interfaz IndicatorCreator haga su trabajo.

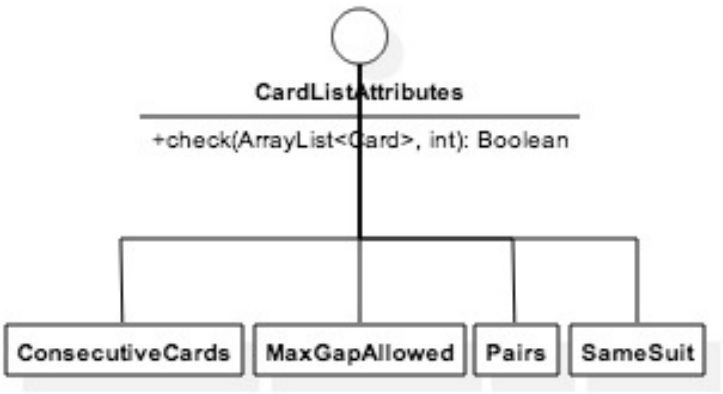

Figura 2.12: Atributos de un conjunto de cartas

Las diferentes clases que implementan la interfaz cardListAttributes (figura 2.12) permiten conocer aspectos que necesitan conocer los checkers para sus comprobaciones pero que son genéricos a cualquier conjunto de cartas y no solamente al póker. Por ejemplo, si un conjunto de cartas son del mismo palo, si hay cartas emparejadas etc.

Diagrama de paquetes

El diagrama de paquetes que se muestra en la figura 2.13 completa a los diagramas de clases de la sección anterior. Es uno de los últimos pasos para formalizar y entender la fase de diseño.

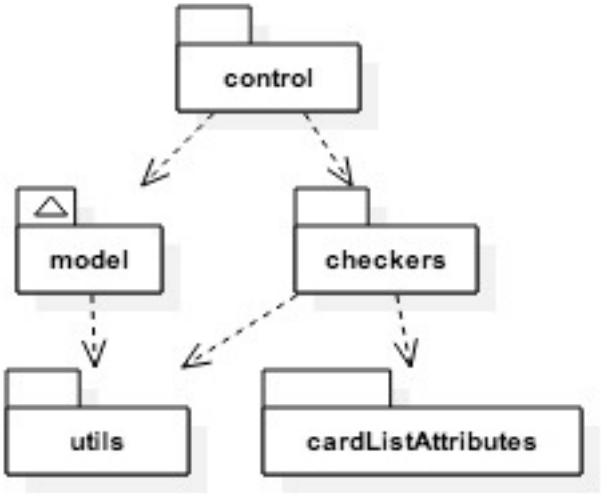

Figura 2.13: Diagrama de paquetes

El paquete de control es el encargado crear las clases de modelo y de hacer las comprobaciones mediante el paquete checkers para crear los indicadores para el usuario.

En el paquete de model se encuentran todas aquellas clases bases. Es decir, los objetos claves del póker y algunos otros como los indicadores.

Checkers se encarga de comprobar una serie de parámetros para decidir si es o no es un determinado indicador.

El paquete utils se refiere a todas aquellas clases que son necesarias para una mejora en la implementación pero que no son una parte base del juego del póker. En este paquete se realizan ordenamientos de cartas, comparadores, etc.

CardListAttributes nos proporciona una serie de atributos sobre una lista de cartas. Si son del mismo palo, están o no conectadas y algunos casos más. Es de vital importancia para los comprobadores pero se ha extraído el paquete porque las clases dentro de cardListAttributes son genéricas y por ello el paquete actúa como una librería.

Diseño de la interfaz de usuarios

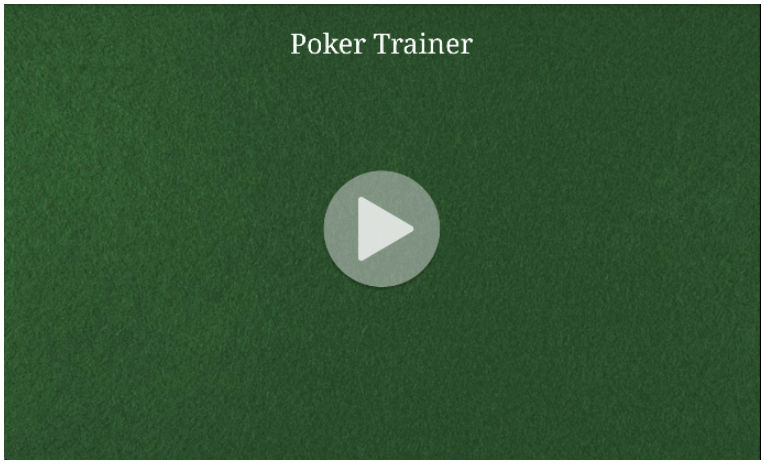

Figura 2.14: Interfaz inicial

La interfaz inicial (figura 2.14) de la aplicación es donde el usuario es introducido al juego ya que el título de la app es presentado y además se da cuenta de que la aplicación será en landscape. La única interacción que el usuario puede hacer es darle al play para continuar.

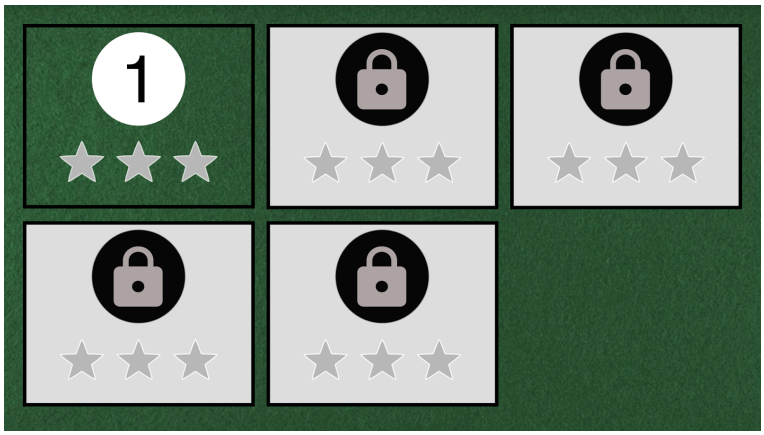

Figura 2.15: Interfaz del menú de fases

La interfaz de introducción a las fases. (figura 2.15) es donde el usuario puede ver todas las fases existentes, así como saber cuales tiene bloqueadas y cuales desbloqueadas además de las puntuaciones de cada una de ellas mediante los elementos de rating.

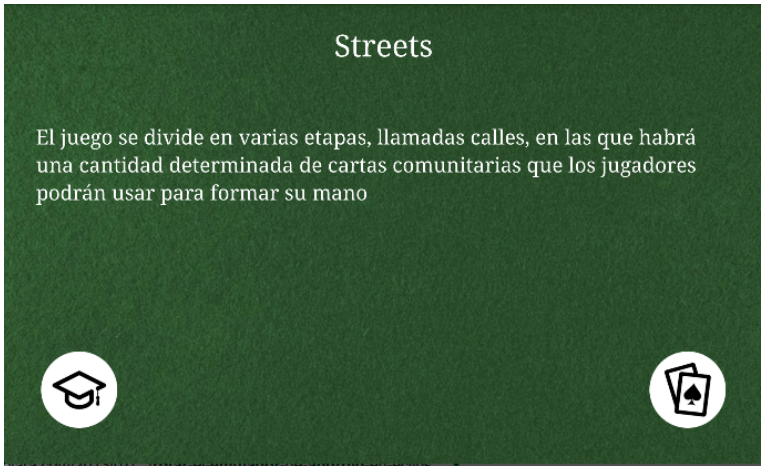

Figura 2.16: Interfaz de introducción a un nivel

La interfaz de introducción a una fase (figura 2.16) determinada donde el usuario conoce que va a aprender en esa fase. Además tiene los accesos al tutorial y puede tener el acceso al quiz bloqueado o desbloqueado.

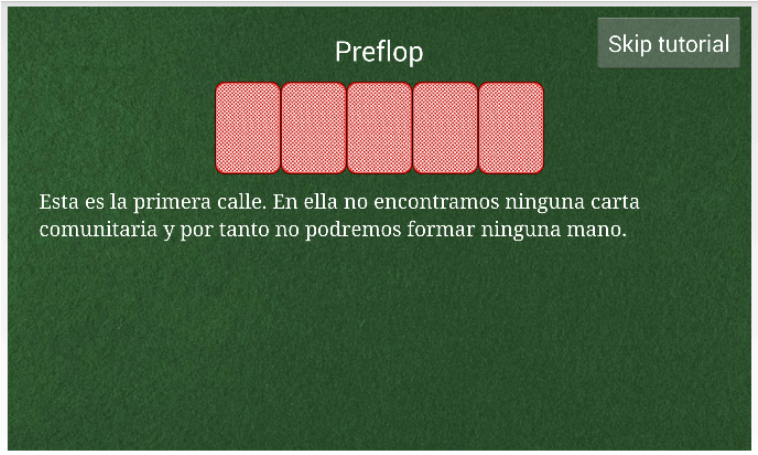

Figura 2.17: Interfaz del tutorial

La interfaz del tutorial de una fase (figura 2.17) es donde el usuario puede navegar por las distintas páginas explicativas del nivel al que le corresponde. Cuando el usuario termine el tutorial o haga click en el botón de skip tutorial se le desbloqueará el acceso al quiz de la interfaz anterior.

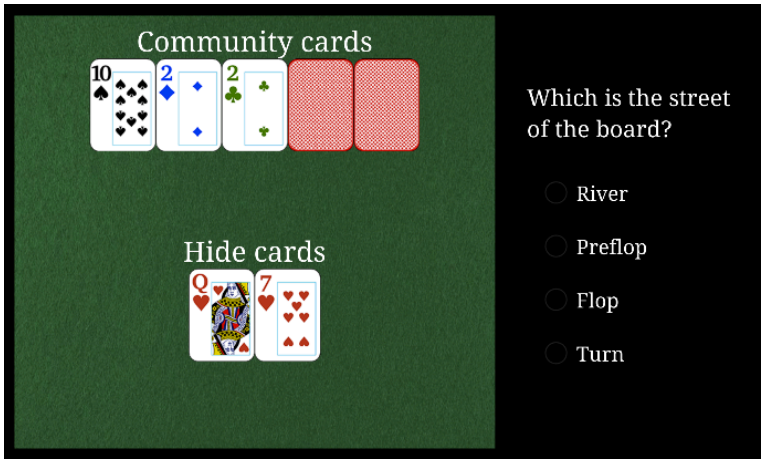

Figura 2.18: Interfaz del quiz

La interfaz de un quiz (figura 2.18) es donde el usuario podrá contestar preguntas relacionadas con la fase en la que se encuentra mediante los pertinentes radiobuttons.

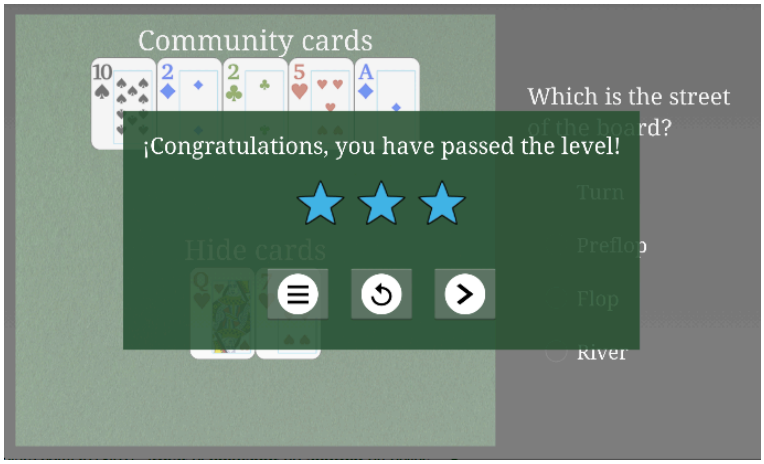

Figura 2.19: Quiz terminado

La interfaz de un quiz terminado (figura 2.19) es donde el usuario conocerá si ha pasado o no el nivel y con qué puntuación. Además se le da acceso al usuario a volver al menú de fases, repetir el quiz o pasar al siguiente nivel sin pasar por el menú.

Capítulo 3. Desarrollo

Leyendo este capítulo podremos entender la evolución del proyecto desde su idea a la concepción del mismo. Además se hablará también de lo que aporta y puede aportar en un futuro.

Desarrollo del proyecto

El desarrollo de la aplicación se pudo diferenciar en seis fases o iteraciones. Cada una de ellas aporta un valor al proyecto final.

Primera iteración:
En esta primera iteración se gestaron los requisitos, capturándolos a través del estudio de escuelas o con opiniones de diferentes interesados, ya fueran expertos en la materia o no. Además se planificaron una serie de temas a los que los usuarios finales pudieran acceder para obtener los conocimientos. Como última acción se llevó una investigación acerca de las distintas tecnologías con las que se podía llevar a cabo el proyecto. Terminó siendo elegida Android, entre otras cosas por su uso del lenguaje Java.

Segunda iteración:
En la segunda iteración se realizó el diseño del modelo y su implementación guiada por pruebas. Al terminar esta iteración se consigue una estructura a medida para el curso planificado.

Tercera iteración:
Al terminar la segunda iteración había algunos aspectos que se podían mejorar previendo poder añadir distintos cursos y quitando muchas dependencias innecesarias. Así que hubo una reestructuración y refactorización a nivel global. Como resultado conseguimos que los outputs del modelo sean únicamente unos indicadores que indican al jugador diferentes aspectos como la categoría de su mano, fuerza de su mano, etc.

Cuarta iteración:
En la cuarta iteración se procede a integrar el modelo con el entorno Android y a realizar un estudio de la tecnología.

Quinta iteración:
Diseño e implementación de la interfaz de usuario. Al finalizar esta iteración nos encontramos con una aplicación terminada pero no se ajustaba correctamente a los objetivos previstos, por lo que se realiza una iteración extra.

Sexta iteración:
Refactorización de las interfaces de usuario para hacer una gamificación (ver marco teórico) de los objetivos didácticos e incentivar al usuario a aprender.

Desarrollo en Android

Conseguir un buen diseño para aplicaciones Android no es un asunto trivial. Para conseguirlo hay que tener en cuenta ciertas características que se relatan a continuación.

Uno de los aspectos más importantes a tener en cuenta al diseñar una aplicación para el sistema operativo Android es la gran cantidad de dispositivos disponibles en el mercado. Por suerte, Android nos provee de la carpeta recursos, gracias a la cual el sistema operativo decidirá, en función del dispositivo, qué recurso le corresponde.

Soportando distintos tamaños de pantalla

Android categoriza las pantallas de los dispositivos usando dos propiedades: tamaño y densidad.

- Hay cuatro tamaños generales: small, normal, large, xlarge
- Hay cuatro densidades generales: ldpi, mdpi, hdpi, xhdpi

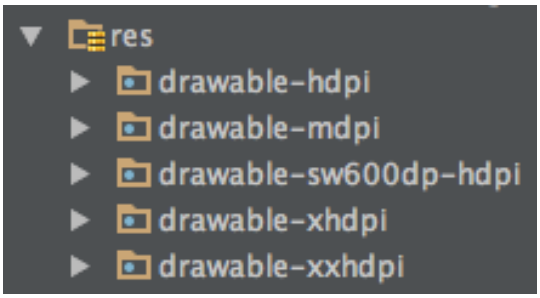

Figura 3.1: Directorio de recursos

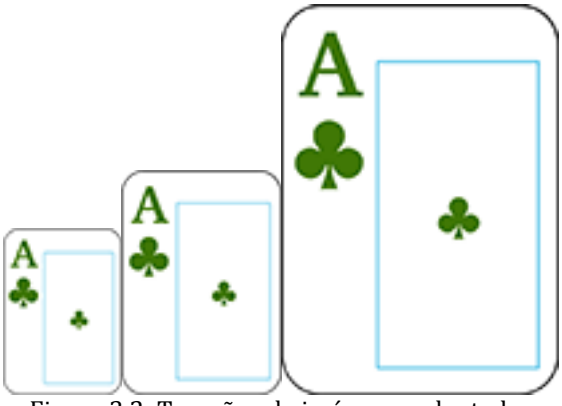

Figura 3.2: Tamaños de imágenes adaptados

Como podemos comprobar en las figuras 3.1 y 3.2, en nuestro proyecto se ha tenido en cuenta este aspecto, incluyendo imágenes de distintas resoluciones e incluyéndose en la correspondiente carpeta de recursos.

Soportando distintos lenguajes

Otro aspecto importante si nuestra aplicación se quiere mover más allá del mercado del lenguaje español es el idioma. De manera sencilla y parecida a la anterior Android provee dentro de la carpeta recursos un directorio llamado values, donde podemos incluir un fichero llamado strings que contendrá todas las strings asociadas a la aplicación.

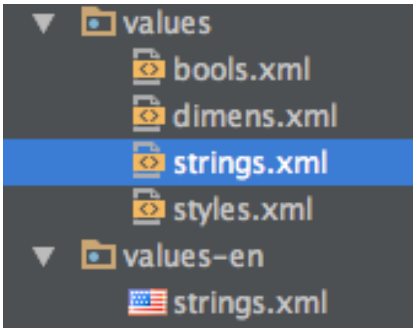

Figura 3.3: Distintos ficheros de strings

En esta aplicación se han construido dos ficheros strings (figura 3.3). Uno por defecto en español, y otro en inglés. La aplicación, al conocer que lenguaje está asociado por el dispositivo cargará un fichero strings u otro.

Opciones de almacenamiento

Android provee distintas opciones para la persistencia de datos. La solución que elijas depende de las necesidades específicas.

Las opciones de almacenamiento son las siguientes:

- Shared Preferences: Almacena tipos primitivos en pares clave-valor.
- Internal Storage: Almacena datos privados en el dispositivo
- External Storage: Almacena datos públicos en un almacenamiento externo compartido
- SQLite Databases: Almacena datos en una base de datos privada
- Network Connection: Almacena datos en la red con tu propio servidor.

En esta aplicación se ha optado por el uso de Shared Preferences ya que los datos que se guardan están relacionados con el bloqueo/desbloqueo de niveles y con la puntuación de cada nivel, aspectos que son perfectamente adaptables a pares clave-valor.

Aportaciones y trabajo futuro

Como trabajo futuro, la aplicación está diseñada para poder incluir más niveles de aprendizaje. Además si tiene aceptación entre el público objetivo se puede incluir un modelo de negocio que habría que estudiar. Otro de los aspectos a mejorar sería la accesibilidad al otro sistema operativo que junto con Android más mercado demanda, iOS.

Además, el software evolucionará añadiendo distintos estilos de quiz. En cuanto al modelo, habrá que añadir otros indicadores no contemplados en esta iteración.

Concluyendo personalmente opino que los objetivos académicos inicialmente previstos han sido alcanzados.

En el desarrollo de la capa Android se ha demostrado que en el trabajo base sobre el modelo es realmente importante seguir las buenas prácticas de la ingeniería del software y el desarrollo orientado a pruebas. Al realizar los quiz (la parte de la aplicación que trabaja más con el modelo) no ha habido ni un solo error de integración con el modelo, funcionando a la perfección desde un primer momento.

En cuanto a los objetivos didácticos de la aplicación se ha probado con satisfacción que algunos jugadores noveles de póker (los testers), encuentran los contenidos enriquecedores a la par que cómodos de aprender con el sistema utilizado.

En el ámbito del juego online, y más concretamente en el mundo del póker, existen las escuelas. La principal diferenciación que tiene esta aplicación de dichas escuelas es que gamifica (ver marco teórico) el acto de aprendizaje, haciendo que aprender un cierto concepto pase de ser leer un texto a pasar un nivel, demostrando efectivamente que has adquirido el concepto. Todo ello se complementa con el hecho de poder hacerlo desde un smartphone o una tablet, llegando así a muchos jugadores que antes no conocían de la existencia de las escuelas.

Como aportaciones personales, el desarrollo de este trabajo de fin de grado me ha permitido elevar mis habilidades de planificación y dirección de proyectos, análisis del mercado y redacción de la documentación pertinente. Además, me ha aportado nuevos conocimientos sobre el desarrollo de nuevas tecnologías que están en auge como el desarrollo mobile, más específicamente en el desarrollo para dispositivos que usen el sistema operativo Android. También me ha permitido conocer nuevas herramientas profesionales ya mencionadas en el marco teórico (ver capítulo 1) y trabajar

con estas para una mejora en la productividad y la seguridad entre otras cosas.

Fuentes de información

- Clean Code. A Handbook of agile software crafsmanship.
- Dirección General de Ordenación del Juego.
http://www.dgojuego.minhap.gob.es/
- Android developers. http://developer.android.com.
- http://agilemanifesto.org/iso/es/
- https://bitbucket.org/
- https://hockeyapp.net/
- www.sourcetreeapp.com
-www.gamificacion.com
-repositorio - https://bitbucket.org/kohrong/pokertrainertfg