PRESENTACIÓN ASW

Diapositiva 2
En primer lugar, vamos a hacer un pequeño repaso de los dos primeros entregables antes de describir con más detalle el tercero

Diapositiva 3
Empezamos con el Parser:

Diapositiva 4
La estructura de la aplicación se divide en dos módulos distintos e independientes:
- Un “Parser” que convierte las preguntas (XML o GIFT) a un formato intermedio (JSON)
- El “Uploader” que guarda los ficheros JSON en una BD de tipo MongoDB en un servidor

Ambas se ejecutan desde la consola del administrador

Diapositiva 5
Para conseguir lo anterior, hemos utilizado dos estilos arquitectónicos:
- El estilo secuencia (bash) que permite la automatización de las tareas
- La descomposición modular para obtener alta cohesividad, bajo acoplamiento y fácil depuración de los módulos anteriores.

Diapositiva 6
Seguimos con el juego de ordenador:

Diapositiva 7
La estructura de la aplicación se divide en tres capas siguiendo el patrón MVC:
- La Interfaz Gráfica de Usuario, con la que interactuará el usuario
- Un controlador que atiende a las peticiones que le hace la GUI
- Un modelo que cumple las funciones de una capa de persistencia

Diapositiva 8
Para conseguirlo, se han utilizado los siguientes estilos:
- Descomposición modular, igual que en el primer entregable
- Datos compartidos, debido a que se tiene una base de datos centralizada de preguntas, que es la misma que la utilizada por el Parser
- Modelo-Vista-Controlador (MVC) que permite tener totalmente desacopladas las tres capas, pudiendo hacer cambios en la Interfaz sin afectar a la lógica de negocio

Diapositiva 9
Ahora vamos a mostrar un video demostrativo del segundo entregable:

Diapositiva 10
VIDEO (Improvisar)

Diapositiva 11
A partir de ahora, vamos a explicar con mayor detalle el tercer entregable, es decir, la página web

Diapositiva 12
Comenzamos describiendo las tecnologías y herramientas utilizadas para su desarrollo.

Hemos utilizado Java para la lógica de negocio y la persistencia y HTML5, CSS y Javascript para el desarrollo de las vistas de la Web
- Todo ello utilizando Eclipse como entorno de desarrollo

Para facilitar la implementación del patrón MVC, hemos utilizado el Framework de desarrollo rápido Play. Que además, nos da herramientas para:

- Crear una API REST
- Utilización de Websockets, utilizados para el modo multijugador
- Implementar seguridad y autenticación para acceder a determinadas vistas de la aplicación

Además, hemos utilizado Travis como herramienta de integración continua para que se ejecuten los test de forma automática

- Parte de dichos test han sido creados con Cucumber, que permite hacer pruebas de la lógica de negocio utilizando historias de usuario con un lenguaje casi natural

Por último, cabe destacar que las bases de datos utilizadas para almacenar la información de la aplicación son PostgreSQL (para jugadores y estadísticas) y MongoDB (para preguntas)

Diapositiva 13
Ahora vamos a describir los estilos arquitectónicos utilizados en este entregable:

Diapositiva 14
Al igual que antes, seguimos utilizando los estilos: datos compartidos y descomposición modular

Como se puede ver en el diagrama de contexto, utilizamos el estilo MVC para lograr la independencia entre las vistas y el controlador

- Esto nos permite cambiar el look&feel de la aplicación y tener varios tipos de tableros

Diapositiva 15
Además, se utilizan tres estilos más:

- Como se puede ver en el diagrama de despliegue, existe una separación entre cliente (que ejecuta los módulos de la Vista) y servidor (que ejecuta los controladores y los módulos del modelo) que permite mayor escalabilidad

- Se utilizan Web Services (REST) para obtener una interfaz uniforme de acceso a recursos (mediante su URI) que utilizará la Vista para llamar a métodos del Controlador

- Por último, se utiliza el estilo basado en eventos para conseguir asincronacidad y mejorar el rendimiento
    - Esto se utiliza en los websockets, que sólo actualizan la vista cuando el controlador genera un evento concreto

Diapositiva 16
Ahora, vamos a describir los atributos de calidad de dicho proyecto:

Diapositiva 17
Disponibilidad de la base de datos y del servidor de juego en todo momento

Facilidad de consulta de las estadísticas de juego y sencillez de uso del juego

- Implementados creando unas vistas con estética sencilla y sin demasiadas opciones

Diapositiva 18
Página web responsiva, es decir, que se adapte a múltiples dispositivos móviles, pantallas independientemente de su tamaño

- No se ha implementado porque habría aumentado el tiempo de desarrollo exponencialmente

Posiblidad de reutilización de algún modulo en variantes del juego futuras

- La lógica de negocio es totalmente independiente de las vistas y descompuesta en módulos

Prevención contra pérdidas de datos en la base de datos y sincronización de accesos a la misma

- El segundo sí está implementado, debido a que las BD que hemos utilizado ya lo tienen

Seguridad de la aplicación y autenticación por roles, es decir, que le usuario no puede acceder a las vistas del administrador

- Implementado gracias al Framework Play

Diapositiva 19

Probar la lógica de negocio sin necesidad de una interfaz gráfica de usuario o unas vistas

- La lógica es totalmente independiente de las vistas gracias al patrón MVC

Tiempos de respuesta cortos

- Implementado gracias a los websockets y la API REST

Cambio de tablero, jugadores, reglas…

- Independencia de la interfaz y lógica descompuesta en módulos cohesivos

Diapositiva 20

Por último, vamos a mostrar un vídeo sobre el tercer entregable:

Diapositiva 21

VIDEO (Improvisar)