|  |
| --- |
|  |
| Trivial2a |
| Documentación |
|  |
|  |
| Alves De Araujo, Ruan Henrique - UO246819  Baragaño González, Samanta - UO231533  Fraile Muñiz, Ivana - UO216746  García Álvarez, Ángela - UO217608  González Fernández, Cristian - UO231602  Montero Hernández, Jose Antonio - UO179746  Piris García, María - UO230886  Velázquez Vico, Álvaro - UO232635 |

|  |
| --- |
|  |

# Índice

[Índice 1](#_Toc418777594)

[1. Descripción del proyecto 2](#_Toc418777595)

[2. Metodología utilizada 2](#_Toc418777596)

[3. Identificación de stakeholders (Descripción) 2](#_Toc418777597)

[4. Identificación inicial de los atributos de calidad 4](#_Toc418777598)

[5. Primer acercamiento a la solución y riesgos 5](#_Toc418777599)

[5.1 Riesgos relacionados con la solución 5](#_Toc418777600)

[6. Lista actualizada de stakeholders 5](#_Toc418777601)

[Definición (comunes) 5](#_Toc418777602)

[Definición (propias) 7](#_Toc418777603)

[7. Lista de atributos de calidad 8](#_Toc418777604)

[8. Atributos de calidad e interesados 10](#_Toc418777605)

[9. Descripción del negocio de la solución. 10](#_Toc418777606)

[9.1.1. Solución 10](#_Toc418777607)

[9.1.2 Tecnologías 11](#_Toc418777608)

[10. Escenarios de calidad 12](#_Toc418777609)

[11. Vistas 14](#_Toc418777610)

[12. Manual de usuario 14](#_Toc418777611)

[13. Manual del sistema 14](#_Toc418777612)

## 1. Descripción del proyecto

La empresa NoGame ha decidido comenzar el desarrollo de una plataforma de juegos Trivial en la que los usuarios puedan conectarse y jugar entre ellos, puesto que el juego Trivial que ahora posee en aplicación de escritorio tendría mayor potencial de poder usarse a través de la web.

Para su creación se reutilizará la lógica desarrollada para la aplicación de escritorio. A ésta se añadirá la una nueva capa encargada de permitir a los jugadores conectarse a través de una página Web, crear una nueva partida y jugar.

La empresa desea que los usuarios se registren en la aplicación de forma sencilla. Un usuario podrá acceder a su histórico y ver si existen otros usuarios conectados a los que invitar a jugar.

Pese a que el juego se desarrolla de la misma manera que en la aplicación de escritorio debe contemplarse que, al tratarse de un juego online, un usuario abandone la partida, ya sea voluntaria o involuntariamente, en tal caso el sistema debe permitir que el resto de usuarios pueda continuar con la partida.

La arquitectura de la aplicación web debe ser escalable y ofrecer buenos tiempos de respuesta a las acciones de los diferentes usuarios puesto que la empresa desea que permita manejar gran cantidad de usuarios.

La empresa desea que lógica e interfaz puedan separarse para experimentar con diferentes tipos de visualización. También está interesada en que la solución cumpla con los principios y estándares Web de diseño responsable poder utilizar la aplicación en diferentes tipos de dispositivos.

## 2. Metodología utilizada

Siguiendo las pautas vistas en clases de teoría de la asignatura, emplearemos el método de desarrollo ADD (Atribute-Driven Design).

## 3. Identificación de stakeholders (Descripción)

* **Empresa (NoGame):**

Son los dueños del proyecto y responsables de las decisiones finales, tanto a nivel de requisitos como de presupuesto.

Entre sus objetivos están:

1. Aplicación funcional y atractiva, que resulte interesante a los usuarios.
2. Bajo coste de desarrollo, el proyecto debe ser tener un desarrollo corto y de coste reducido.
3. Buenos tiempos de respuesta, que la aplicación tenga tiempos bajos de respuesta a las acciones de los usuarios independientemente de la cantidad de usuarios activos.
4. Aplicación apta para diferentes dispositivos, que cumpla los principios y estándares Web de diseño responsable.

* **Desarrolladores (Equipo 2a)**

Son los encargados de la construcción del software. Su responsabilidad es codificar, probar y mantener el código del sistema desarrollado.

Entre sus objetivos están:

1. Aplicación modular y de alta mantenibilidad, permitiendo así que sea posible ampliar su funcionalidad y que se haga de forma sencilla.
2. Aplicación con el mejor tiempo de respuesta posible, el sistema debe funcionar con rapidez aunque gestione un gran volumen de usuarios.

* **Arquitectos del sistema (Equipo 2a)**

Son los encargados de documentar, estructurar, diseñar y mantener el sistema que da solución al proyecto encargado por la empresa.

Entre sus objetivos están:

1. Aplicación escalable y fácil de mantener, debe estar preparado para ser usado por un gran número de usuarios, así como para que su funcionalidad sea ampliada.
2. Proyecto rentable, que tenga un grado de rentabilidad que haga atractivo su desarrollo por el precio establecido.

* **Gestores de bases de datos (Equipo 2a)**

Son los encargados de diseñar y mantener las bases de datos. También deben encargarse de la seguridad e integridad de los datos.

Entre sus objetivos están:

1. Garantizar la seguridad en el acceso a los datos
2. Garantizar la integridad de los datos almacenados

* **Cliente (Usuario jugador)**

Son aquellas personas que utilizarán la aplicación.

Entre sus objetivos están:

1. Registro sencillo en la aplicación, que se realice de forma simple y clara
2. Interfaz intuitiva, permitiendo que el uso de la aplicación sea sencillo
3. Confiabilidad en la aplicación, que esta no produzca errores.
4. Posibilidad y facilidad de visualización de su histórico de partidas, que el usuario pueda consultar sus resultados en partidas pasadas así como sus estadísticas de respuestas acertadas.

## 4. Identificación inicial de los atributos de calidad

La aplicación que se debe desarrollar debe cumplir con los siguientes atributos de calidad:

* **Disponibilidad:** La aplicación debe poder ser utilizada en cualquier momento.
* **Usabilidad:** Es importante que la curva de aprendizaje sea la mínima posible ya que es imprescindible que la aplicación pueda ser utilizada por el usuario sin ninguna dificultad.
* **Accesibilidad:** La aplicación debe ser utilizable por cualquier usuario independientemente de sus limitaciones individuales o derivadas del contexto de acceso.
* **Seguridad:** La aplicación gestiona información sensible de los usuarios e información importante para la empresa. Por ello se deben tomar todas las medidas de protección necesarias para garantizar la seguridad tanto del sistema como de sus usuarios.
* **Modificabilidad:**
* Facilidad de modificación o extensión de código, ya que en el futuro podría desearse, por ejemplo, cambiar las reglas del juego.
* Escalabilidad del producto, ya que se desea que en un futuro el volumen de usuarios que puedan utilizar la aplicación sea mayor.
* **Rendimiento:** 
  + Se debe garantizar un tiempo correcto del procesamiento y ejecución de los algoritmos involucrados en la operación de juego. La aplicación debe correr fluida sin ningún tipo de lag o delay.
  + Se debe garantizar que el tiempo de respuesta será independiente del número de jugadores activos.

## 5. Primer acercamiento a la solución y riesgos

**n-capas con MVC en una capa**

Se opta por desarrollar el juego siguiendo un patrón n-capas, diferenciando entre interfaz, lógica de negocio y persistencia. La capa de interfaz es figurada, puesto que combinamos el patrón mencionado con MVC por lo que se diferencias las capas de modelo, vista y controlador.

Tanto la capa de negocio; que contiene la lógica que permite que el juego funcione correctamente; como la capa de persistencia; que permite gestionar los datos de los diferentes usuarios así como sus históricos, comunicándose con nuestro SGBD; son reutilizadas de la aplicación de escritorio.

La capa de modelo es la representación de dominio específico de la información con la que opera la aplicación.

La capa de vista permite que la aplicación pueda ser utilizada mediante interacción con el usuario. Pueden existir diferentes vistas para un mismo modelo, para diferentes propósitos.

El controlador es el encargado de responder y procesar los eventos, peticiones HTTP. Además, también puede invocar a los cambios en el modelo. Esta capa se comunica con la vista siguiendo el modelo REST.

La arquitectura escogida permite crear una aplicación con bajo acoplamiento entre la lógica y la presentación, aunque el acoplamiento entre vista y controlador es elevado. Al existir la separación de responsabilidades, se puede adaptar el juego a nuevas plataformas sin modificar la capa de negocio ni la de persistencia.

**MVC con n-capas en M**

Se opta por desarrollar el juego siguiendo el patrón MVC aplicado a la arquitectura web. Este patrón divide la aplicación en capas bien diferenciadas modelo, vista y controlador.

La capa de modelo es la representación de dominio específico de la información con la que opera la aplicación. Esta capa a su vez, sigue una arquitectura n-capas, ya que contiene la lógica que permite que el juego funcione correctamente; y, además, encapsula la capa de persistencia, que permite gestionar los datos de los diferentes usuarios así como sus históricos, comunicándose con nuestro SGBD.

La vista, se trata de la interfaz de usuario, que permite que el modelo pueda ser utilizado mediante interacción con el usuario. Pueden existir diferentes vistas para un mismo modelo, lo que permite que la aplicación sea multiplataforma sin necesidad de modificar el modelo.

El controlador es el encargado de responder y procesar los eventos, peticiones HTTP. Además, también puede invocar a los cambios en el modelo. Se comunica con la vista siguiendo el modelo REST.

Esta arquitectura permite una aplicación con bajo acoplamiento entre la lógica y la presentación, aunque el acoplamiento entre vista y controlador es elevado. Al existir la separación de responsabilidades, se puede adaptar el juego a nuevas plataformas sin modificar la lógica.

### 5.1 Riesgos relacionados con la solución

La solución escogida se enfrenta a algunos riesgos que es necesario identificar:

* **Extensión a nuevas plataformas.** Relacionado con su reusabilidad y mantenibilidad, el hecho de que exista una división entre capas bien definidas nos permite una extensión fácil en el futuro.
* **Posibilidad de cambios de interfaz.** Relacionado con el punto anterior, una de las principales características buscadas es el poder cambiar la representación visual sin tener que modificar ninguna de las demás capas.

## 6. Lista actualizada de stakeholders

### Definición (comunes)

#### Empresa NoGame

* Son los propietarios y responsables principales de la financiación de la aplicación.
* Objetivos:
  + Obtener una aplicación funcional y atractiva, que resulte interesante a los usuarios.
  + Bajo coste de desarrollo, el proyecto debe ser tener un desarrollo corto y de coste reducido.
  + Asegurar buenos tiempos de respuesta, que la aplicación tenga tiempos bajos de respuesta a las acciones de los usuarios independientemente de la cantidad de usuarios activos.
  + Obtener una aplicación apta para diferentes tipos de dispositivos, que cumpla los principios y estándares Web de diseño responsable.

#### Desarrolladores juego Trivial

* Equipo responsable del desarrollo del funcionamiento de la aplicación.
* Objetivos:
  + Desarrollar una aplicación modular y de alta mantenibilidad, permitiendo así que sea posible ampliar su funcionalidad y que se haga de forma sencilla.
  + Desarrollar una aplicación con el mejor tiempo de respuesta posible, el sistema debe funcionar con rapidez aunque gestione un gran volumen de usuarios.

#### Arquitectos de la aplicación

* Equipo responsable de todo lo que tenga que ver con el análisis del proyecto
* Objetivos:
  + Diseñar una aplicación escalable y fácil de mantener, debe estar preparado para ser usado por un gran número de usuarios, así como para que su funcionalidad sea ampliada.
  + Presentar un proyecto rentable, que tenga un grado de rentabilidad que haga atractivo su desarrollo por el precio establecido.

#### Usuarios jugadores

* Son las personas que ejecutaran la aplicación
* Objetivos:
  + Poder realizar un registro sencillo en la aplicación, que sea de forma simple y clara.
  + Encontrar una interfaz intuitiva, permitiendo que el uso de la aplicación sea sencillo.
  + Interactuar con una aplicación confiable, que esta no produzca errores.
  + Tener la posibilidad y facilidad de visualización de su histórico, que el usuario pueda consultar sus resultados en partidas pasadas así como sus estadísticas de respuestas acertadas.

#### Usuario administrador

* Son los usuarios con privilegios suficientes para ver las estadísticas de juegos.
* Objetivos:
  + Trabajar con una interfaz de fácil de uso para la consulta de estadísticas generales.

### Definición (propias)

#### Gestores de bases de datos

* Son los responsables de mantener tanto el correcto funcionamiento del sistema de gestión de bases de datos, como de la integridad y consistencia de la bases de datos.

|  |  |  |
| --- | --- | --- |
| Código | Stakeholder | Intereses |
| ST-01 | Empresa NoGame | 1. Aplicación funcional y atractiva, que resulte interesante a los usuarios. 2. Bajo coste de desarrollo, el proyecto debe ser tener un desarrollo corto y de coste reducido. 3. Buenos tiempos de respuesta, que la aplicación tenga tiempos bajos de respuesta a las acciones de los usuarios independientemente de la cantidad de usuarios activos. 4. Aplicación apta para diferentes dispositivos, que cumpla los principios y estándares Web de diseño responsable. |
| ST-02 | Desarrolladores juego Trivial | 1. Aplicación modular y de alta mantenibilidad, permitiendo así que sea posible ampliar su funcionalidad y que se haga de forma sencilla. 2. Aplicación con el mejor tiempo de respuesta posible, el sistema debe funcionar con rapidez aunque gestione un gran volumen de usuarios. |
| ST-03 | Arquitectos de la aplicación | 1. Aplicación escalable y fácil de mantener, debe estar preparado para ser usado por un gran número de usuarios, así como para que su funcionalidad sea ampliada. 2. Proyecto rentable, que tenga un grado de rentabilidad que haga atractivo su desarrollo por el precio establecido. |
| ST-04 | Usuarios jugadores | 1. Registro sencillo en la aplicación, que se realice de forma simple y clara. 2. Interfaz intuitiva, permitiendo que el uso de la aplicación sea sencillo. 3. Confiabilidad en la aplicación, que esta no produzca errores. 4. Posibilidad y facilidad de visualización de su histórico, que el usuario pueda consultar sus resultados en partidas pasadas así como sus estadísticas de respuestas acertadas. |
| ST-05 | Usuario administrador | 1. Interfaz de fácil de uso para la consulta de estadísticas generales. |
| ST-06 | Gestores de bases de datos | 1. Garantizar la seguridad en el acceso a los datos. 2. Garantizar la integridad de los datos almacenados. |

## 7. Lista de atributos de calidad

|  |  |  |
| --- | --- | --- |
| Código | Descripción | Tipo de Atributo |
| AT001 | El juego debe proporcionarle al usuario una interfaz que involucre la interacción con los jugadores de forma sencilla. | Usabilidad y Accesibilidad |
| AT002 | Las representaciones visuales del juego deben ser atractivas. | Usabilidad y Accesibilidad |
| AT003 | Las acciones de tirar el dado y mover casillas deben ser sencillas e intuitivas. | Usabilidad y Accesibilidad |
| AT004 | Se debe asegurar los datos almacenados e introducidos por los usuarios en la aplicación. | Seguridad |
| AT005 | Consistencia e integridad de la bases de datos de la compañía. | Seguridad |
| AT006 | Seguridad de acceso a la base de datos. | Seguridad |
| AT007 | Denegar el acceso a usuarios ilegítimos de la aplicación. | Seguridad |
| AT008 | La aplicación debe estar separada en capas para evitar el acoplamiento. | Mantenibilidad |
| AT009 | Las modificaciones que se lleven a cabo en las implementaciones concretas no deberían implicar al total de la aplicación. | Mantenibilidad |
| AT010 | El jugador debe ver de forma clara a qué casillas puede desplazarse. | Usabilidad y Accesibilidad |
| AT011 | La lógica de negocio empleada debe ser totalmente independiente de las vistas que se utilicen. | Reusable |
| AT012 | Los algoritmos que calculan las casillas posibles deben garantizar un tiempo de ejecución bajo. | Rendimiento |
| AT013 | El tiempo de respuesta debe ser bajo independientemente del número de jugadores activos. | Rendimiento |
| AT014 | Un jugador puede acceder en todo momento independientemente del volumen de usuarios activos | Disponibilidad |
| AT015 | La aplicación debe estar preparada para añadir nuevas funcionalidades. | Modificabilidad |

## 8. Atributos de calidad e interesados

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Atributos  Stakeholders | ST-01 | ST-02 | ST-03 | ST-04 | ST-05 | ST-06 |
| AT001 |  |  |  | √ | √ |  |
| AT002 |  |  |  | √ |  |  |
| AT003 |  |  |  | √ |  |  |
| AT004 | √ | √ | √ | √ | √ | √ |
| AT005 | √ |  |  |  |  | √ |
| AT006 | √ |  |  |  |  | √ |
| AT007 |  |  |  |  |  | √ |
| AT008 | √ | √ | √ |  |  |  |
| AT009 |  | √ | √ |  |  |  |
| AT010 |  |  |  | √ |  |  |
| AT011 | √ | √ | √ |  |  |  |
| AT012 |  | √ |  | √ |  |  |
| AT013 | √ | √ |  | √ |  |  |
| AT014 | √ | √ |  | √ |  |  |
| AT015 | √ | √ | √ |  |  |  |

## 9. Descripción del negocio de la solución.

En este apartado se describe el modelo de negocio del programa.

El modelo de negocio de la aplicación está constituido por una aplicación web con interfaz gráfico.

### 9.1.1. Solución

Todos los niveles de la aplicación se encuentran separados entre capas, siguiendo las características básicas del patrón arquitectónico Modelo-Vista-Controlador (MVC) con el fin de lograr múltiples vistas del mismo modelo, tal y como se requiere:

* + Modelo: Esta capa debe contener la lógica de negocio involucrada en el funcionamiento del juego Trivial, así como las operaciones y procesos relacionados con tareas de bases de datos (CRUD) mediante DAOs. Es totalmente independiente de controladores y vistas. En la aplicación la lógica y la persistencia están perfectamente separadas en capas
  + Vista: Posee la abstracción visual del modelo y se comunica, únicamente con de controlador.
  + Controlador: Recibe las interacciones del usuario y las coordina con las acciones a realizar por el modelo. No se comunica directamente con la persistencia, pues es el modelo el encardo de esta tarea.

A fin de mejorar el mantenimiento de la aplicación y lograr el mínimo acoplamiento posible entre capas, la implementación de factorías y fachadas se hacen imprescindibles.

### 9.1.2 Tecnologías

En este apartado se exponen las distintas opciones tecnológicas elegidas.

* Lenguaje de programación: Como lenguaje principal hemos utilizado Java, debido a que es el lenguaje con el que el equipo está más familiarizado, permitiendo así un desarrollo ágil. Para el desarrollo de la aplicación web se ha utilizado el framework “Play”, que incluye una utilidad que nos facilita la creación de las vistas mediante lenguaje HTML5 con bloques de Scala (scala-based template engine). Además, se ha utilizado JavaScript para añadir funcionalidad a la web.
* Bases de Datos: La gestión de usuarios y partidas se realizará con el SGBD de bases relacionales HSQLDB , ya que nos brinda la sencillez que necesitamos para esta ocasión.
* Testabilidad: para realizar las pruebas automáticas que permiten verificar el correcto funcionamiento de la aplicación se han usado dos tecnologías: Travis y JUnit.

## 10. Escenarios de calidad

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Nº | Fuente | Estímulo | Entorno | Artefacto | Respuesta | Medición de respuesta | Atributo de calidad |
| 1 | Usuario jugador | Contestar a una pregunta | Producción. | Controlador | Mostrar las alternativas posibles. | La selección se debe hacer con un solo clic. | AT001 |
| 2 | Usuario  jugador | Mover a una casilla. | Producción. | Controlador | Moverse hasta una casilla. | El movimiento debe ser animado y menor a 1 segundo. | AT002  AT001 |
| 3 | Usuario  jugador | Lanzar el dado | Producción. | Controlador | Sacar un número del dado. | Se obtiene un número del dado  Con apenas un clic y después de una animación en menos de 1 segundo. | AT003  AT002 |
| 4 | Desarrolladores | Cambiar la implementación de una vista. | Desarrollo | Vista | Aceptación e integración de la nueva vista. | Ningún cambio en el modelo | AT008 |
| 5 | Usuario jugador | Ver informaciones de las partidas, sin privilegios. | Producción. | Modelo (Lógica) | Acceso a la información. | Acceso denegado por no tener suficientes permisos. | AT004 |
| 6 | Usuario administrador. | Modificar un valor estadístico. | Producción. | Modelo (Lógica) | Datos intactos. | 100% de los datos siguen intactos. | AT005 |
| 7 | Intento no autorizado de acceso a BBDD | Intento acceso a BBDD | Producción. | Modelo (Persistencia) | Acceso no autorizado rechazado | No se ha realizado ningún cambio en la base de datos | AT007  AT006 |
| 8 | Desarrolladores | Modificación en la lógica. | Desarrollo | Modelo (Lógica) | Compilar el código | Capa de persistencia y GUI no necesitan ser compiladas. | AT009 |
| 9 | Aplicación | Terminar partida. | Producción. | Modelo (Persistencia) | No perder información de las partidas y usuarios. | Se mantiene el 100% de la información. | AT005 |
| 10 | Usuario jugador | Mover a una casilla. | Producción. | Vista | Ver las opciones posibles. | Los destinos se destacan visiblemente. | AT010 |
| 11 | Desarrolladores | Añadir una nueva vista. | Desarrollo. | Modelo (Lógica) | Funcionamiento correcto. | No es necesario modificar ningún aspecto de la lógica. | AT011 |
| 12 | Usuario jugador | Hacer un movimiento. | Producción. | Modelo (Lógica) | Devolver los posibles movimientos. | El algoritmo debe tardar menos de 2 segundos. | AT012 |

## 11. Vistas

## 12. Manual de usuario

## 13. Manual del sistema