qwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnm

|  |
| --- |
| **Trivial 4b**  Arquitectura del software  04/03/2015  Grupo 4b |

Tabla de contenido

[1. Introducción 3](#_Toc413145635)

[1.1. Propósito 3](#_Toc413145636)

[1.2. Alcance 3](#_Toc413145637)

[1.3. Referencias 3](#_Toc413145638)

[1.4. Visión general 3](#_Toc413145639)

[2. Stakeholders 3](#_Toc413145640)

[2.1. Dueño de la empresa contratante 4](#_Toc413145641)

[2.2. Equipo de desarrollado 4](#_Toc413145642)

[2.3. Administrador de la base de datos 4](#_Toc413145643)

[3. Requisitos del proyecto 4](#_Toc413145644)

[3.1. Requisitos funcionales 4](#_Toc413145645)

[3.2. Requisitos no-funcionales 4](#_Toc413145646)

[4. Atributos de calidad 4](#_Toc413145647)

[5. Escenarios de calidad 4](#_Toc413145648)

[6. Vistas del Modelo de Diseño 5](#_Toc413145649)

[6.1. Descomposición en Subsistemas 5](#_Toc413145650)

[6.1.1. Subsistema 1 5](#_Toc413145651)

[6.1.2. Subsistema 2 5](#_Toc413145652)

[6.1.3. Subsistema 3 5](#_Toc413145653)

[6.2. Diseño de clases 5](#_Toc413145654)

[7. Manual de usuario 5](#_Toc413145655)

# Introducción

En este proyecto se intenta dar solución a las necesidades planteadas por la empresa NoGame. Dicha empresa se dedica a la creación de videojuegos de pregunta/respuestas y tiene como proyecto el desarrollo de una línea de juegos basados en una variante del Trivial.

Previamente al desarrollo del videojuego la empresa quiere hacerse con una base de preguntas y respuestas utilizando para ello un banco de preguntas ya existente. Dicho banco de preguntas distribuye las preguntas en ficheros con el formato GIFT. Debido a ello la empresa NoGame quiere poseer una aplicación que le permita convertir este formato de ficheros en otro formato.

La aplicación debe ejecutarse en dos etapas. Una primera etapa que analiza los ficheros con las preguntas y otra etapa en la que exportar las preguntas a un fichero de salida y las almacena en la base de datos. El formato que se desea obtener es JSON, el cual podrá ser usado más adelante para insertar su contenido en una base de datos, la cual en principio podría ser MongoDB aunque aún no ha sido confirmado.

La empresa NoGame quiere que la aplicación sea lo más autosuficiente posible, es decir, que se pueda automatizar en lo máximo posible el proceso de conversión entre formatos. No obstante esto no debe impedir que pueda llevar a cabo un proceso de revisión y depurado el cual muestre los errores que hayan podido ocurrir.

## Propósito

Este documento proporciona una apreciación global y comprensible de la arquitectura del sistema usando diferentes puntos de vista para mostrar distintos aspectos del sistema. Intenta capturar y llegar a las decisiones de arquitectura críticas que han sido hechas en el sistema.

## Alcance

En este documento se analizarán los requisitos de nuestra arquitectura tanto funcional como no funcional. Además se analizarán detalladamente cada uno de los stakeholders y sus respectivos intereses dentro del proyecto. Así como también la distribución arquitectónica del proyecto mediante los pertinentes diagramas. Y por último un manual de usuario, que servirá como ayuda para la implementación y uso del programa final.  
Todo ello será plasmado en el documento de manera clara y sencilla para su correcta comprensión evitando toda posible mal interpretación.

## Referencias

Para la elaboración de este documento se usará como referencia el documento de la guía académica de la asignatura Arquitectura del Software.

## Visión general

La organización de documento consistirá en diferentes apartados en los que se analizarán los Stakeholder y sus necesidades respecto al proyecto, las vistas del diseño arquitectónico, los requisitos de calidad y sus diferentes escenarios.

# Stakeholders

En esta sección se analizaran los diferentes Stakeholders que tienen relación con nuestro proyecto.

## Dueño de la empresa contratante

Esta persona exigirá una producto que sea estable que sea desarrollado en el menor tiempo posible ya que quiere poder lanzar al mercado cuanto antes su nueva línea de juegos, basados en la temática trivial, que constará de diferentes preguntas con multirespuesta.

## Equipo de desarrollado

Este grupo de personas quieren tener un margen de tiempo amplio para poder desarrollar el programa de manera que sea fácilmente mantenible y extensible a nuevas funcionalidades.

## Administrador de la base de datos

Esta persona quiere que nuestro programa disponga de configuración que permita modificar las opciones de conexión a la base de datos de una manera fácil y sencilla. También exigirá que nuestro programa sea capaz de conectarse y trabajar con cualquier tipo de base de datos que pueda necesitar en el futuro.

# Requisitos del proyecto

## Requisitos funcionales

* El programa debe generar un fichero de salida en formato JSON con las preguntas y respuestas.
* Las preguntas deben ser obtenidas de una fuente externa, un fichero de formato GIFT.
* El programa debe ser lo menos interactivo posible. Tan solo debe solicitar unos simples datos, tales como, el nombre del fichero de entrada, el nombre del fichero de salida y la extensión del fichero en ambos casos.
* Se debe almacenar las preguntas y respuestas en una base de datos.
* El proceso debe poder ser automatizado.

## Requisitos no-funcionales

* Poder agregar preguntas desde diferentes formatos de fichero.
* La base de datos puede ser SQL o NOQSL como por ejemplo MongoDB.
* Generar un log que permita analizar errores en el proceso de conversión

# Atributos de calidad

* **Adaptabilidad.**

Capacidad de procesar ficheros con distintas extensiones.

* **Usabilidad**.

Sencilla comprensión y uso de la aplicación.

* **Testabilidad**.

Facilidad para verificar el producto.

* **Conectividad**.

Posibilidad de conectarse a una base de datos externa al propio sistema.

# Escenarios de calidad

**[en desarrollo aún]**

# Vistas del Modelo de Diseño

## Descomposición en Subsistemas

[Diagrama mostrando la descomposición del Modelo de Diseño en Subsistemas, sus interfaces y las dependencias entre ellos, mostrando para cada Subsistema la Capa de la Arquitectura en la que se encuentra.]

### Subsistema 1

[Breve descripción del Subsistema, incluyendo un diagrama conteniendo las clases y los subsistemas que lo componen y para cada uno de ellos una breve descripción, esto se encuentra con mayor detalle en el Modelo de Diseño.]

### Subsistema 2

[Breve descripción del Subsistema, incluyendo un diagrama conteniendo las clases y los subsistemas que lo componen y para cada uno de ellos una breve descripción, esto se encuentra con mayor detalle en el Modelo de Diseño.]

### Subsistema 3

[Breve descripción del Subsistema, incluyendo un diagrama conteniendo las clases y los subsistemas que lo componen y para cada uno de ellos una breve descripción, esto se encuentra con mayor detalle en el Modelo de Diseño.]

**[en desarrollo aún así que podría haber mas subsistemas]**

## Diseño de clases

[Diagrama UML con las diferentes clases]

# Manual de usuario

[Descripción de como se ha de usar el programa. Y como actuar antes los posibles contratiempos que puedan surgir]