**DOCUMENTO DE REQUERIMIENTOS**

**PRESENTADO POR:**

**GRUPO UMBRELLA**

**JOSE JAVIER VIRVIESCAS TOLEDO**

**ALVARO ANDRES SUAREZ ALFONSO**

**ERICA ESTEFANNY PRADO CASAÑEDA**

**OSCAR JAVIER MORENO REY**

**GABRIEL ALBERTO CASTILLO PRADA**

**PRESENTADO A:**

**LUIS DANIEL BENAVIDES**

**UNIVERSIDAD DE LOS ANDES**

**INGENIERÍA DE SISTEMAS Y COMPUTACIÓN**

**ESPECIALIZACIÓN EN CONSTRUCCIÓN DE SOFTWARE**

**BOGOTÁ D.C.**

**ABRIL 04 DE 2015**

TABLA DE CONTENIDO

[TABLA DE CONTENIDO 2](#_Toc415945532)

[1. INTRODUCCIÓN 3](#_Toc415945533)

[2. REQUIERIMIENTOS FUNCIONALES 4](#_Toc415945534)

[3. INTERFAZ DE USUARIO 5](#_Toc415945535)

[4. REFERENCIAS Y CONTEXTO 6](#_Toc415945536)

[5. RESTRICCIONES 7](#_Toc415945537)

[6. DISEÑO 8](#_Toc415945538)

[**6.1.** **Diagrama de clases** 8](#_Toc415945539)

[**6.2.** **Diagrama de despliegue** 8](#_Toc415945540)

[**6.3.** **Diagrama de casos de uso** 9](#_Toc415945541)

[**6.4.** **Diagrama de componentes** 10](#_Toc415945542)

1. INTRODUCCIÓN

Se pretende establecer un marco general de los requerimientos obtenidos en la etapa de elicitación del proyecto TSP con el fin de definir el pilar del desarrollo de las funciones analizadas para cada ciclo. Adicional a esto contar con una base común para los miembros del equipo y al final de cada ciclo poder verificar más fácilmente el cumplimiento de cada requerimiento.

Para dirigir el trabajo y el desarrollo de las funciones seguiremos los procesos estructurados por TSP con el fin de ejecutar un trabajo en equipo de forma organizada y exitosa mejorando los niveles de calidad y la productividad del proyecto.

Nuestro grupo de trabajo Umbrella: Álvaro Suarez (Líder del equipo), Javier Virviescas (Líder de desarrollo), Oscar Moreno (Líder de Soporte), Gabriel Castillo (Líder de Planeación) y Erica Prado (Líder de Calidad) presentamos el siguiente levantamiento de requerimientos de software.

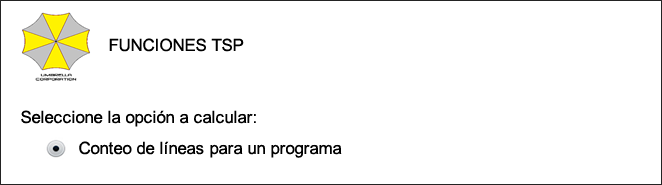
1. REQUIERIMIENTOS FUNCIONALES

A continuación se listarán los requerimientos funcionales encontrados para el ciclo 1 del proyecto de TSP

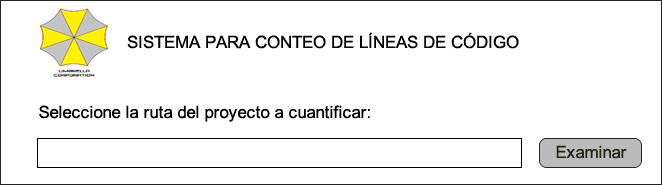
* Selección de ejercicio: Se debe permitir al usuario seleccionar de una lista de ejercicios disponibles el que desee ejecutar.
* Ejercicio No 1: Se deben contar las líneas de código de los archivos cargados desde una ruta en el navegador. El estándar de conteo con el cual se debe trabajar solo contará las líneas físicas de cada archivo sin tener en cuenta los comentarios y los espacios en blanco.

1. INTERFAZ DE USUARIO

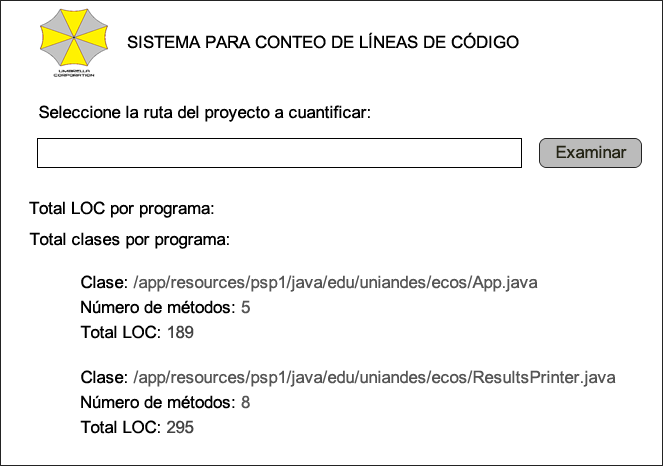
Como pantalla principal tenemos un menú con las funcionalidades disponibles para el usuario, el ciclo 1 cubre el conteo de líneas de un programa determinado por el usuario, para acceder a este debe seleccionar la opción de “Conteo de líneas para un programa”:



Luego se muestra una pantalla para que el usuario seleccione la ruta donde se encuentra la carpeta correspondiente al programa al cual se le hará el conteo:



Los resultados se visualizan de la siguiente forma:



1. REFERENCIAS Y CONTEXTO

**Desarrollo:**

Para el desarrollo nos basamos en el patrón MVC (Modelo Vista Controlador) como modelo para la construcción del software, separando los datos, la lógica de negocio y la interfaz de usuario. Esto con el fin de desacoplar la vista del modelo y así lograr mayor reusabilidad, menos trabajo y menor riesgo al momento de modificar o agregar código.

* Modelo: Contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.
* Vista o interfaz de usuario: Compone la información que se envía al cliente y los mecanismos interacción con éste.
* Controlador: Actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.

En cuanto al lenguaje de programación haremos uso de Java por su orientación a objetos ya que es un lenguaje basado en clases y capaz de manejar los conceptos básicos de Búsqueda Dinámica, Herencia y Subtipos. Esto nos ofrece facilidad de diseño, reusabilidad, facilidad de mantenimiento, sistemas más complejos, abstracción, etc.

**Tecnologías:**

Para el desarrollo de las funciones definidas en cada ciclo utilizaremos los siguientes componentes tecnológicos:

**Java:** Lenguaje orientado a objetos desarrollado por Sun Microsystems. Permite el desarrollo de aplicaciones bajo el esquema de Cliente Servidor, y aplicaciones distribuidas.

Es distribuido, multiplataforma, compilado, seguro y posee una arquitectura neutral. Proporciona un conjunto de clases potente y flexible lo hace más manejable desde el punto de vista del programador.

**Netbeans:** Entorno de desarrollo gratuito y de código abierto que permite el uso de un amplio rango de tecnologías de desarrollo para aplicaciones Web. Brinda soporte a múltiples tecnologías y puede instalarse en varios sistemas operativos.

**Heroku:** Plataforma como servicio en la nube con una interfaz avanzada de gestión de recursos y con soporte a múltiples lenguajes de programación.

**Git:** Es un sistema distribuido. No existe un repositorio central. Todas las copias que existen de un proyecto son un branch del proyecto. Cada usuario que tiene una copia puede hacer cambios y cualquier otra operación sin afectar a ningún otro usuario.

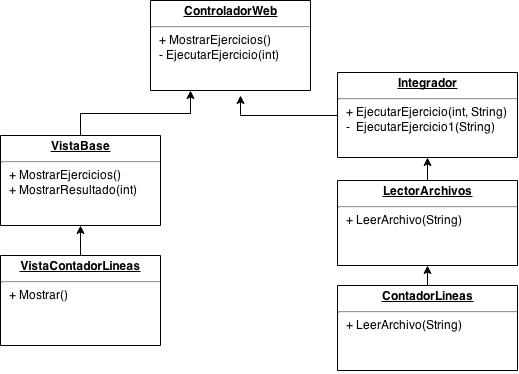
1. RESTRICCIONES

El siguiente listado corresponde a las restricciones encontradas en el proyecto para el ciclo 1.

* Todos los ejercicios desarrollados deben ser desarrollados en Java
* El aplicativo debe ser desplegado en el ambiente web de Heroku y debe ser posible ejecutarlo desde internet

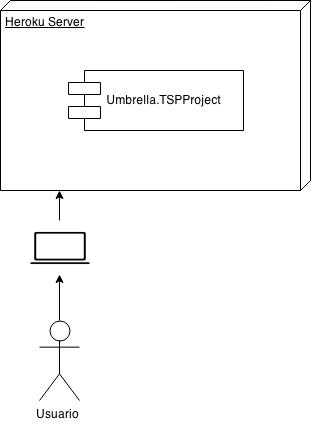
1. DISEÑO
   1. Diagrama de clases

El siguiente diagrama muestra las 6 clases principales utilizadas para desarrollar el ciclo 1 de desarrollo:

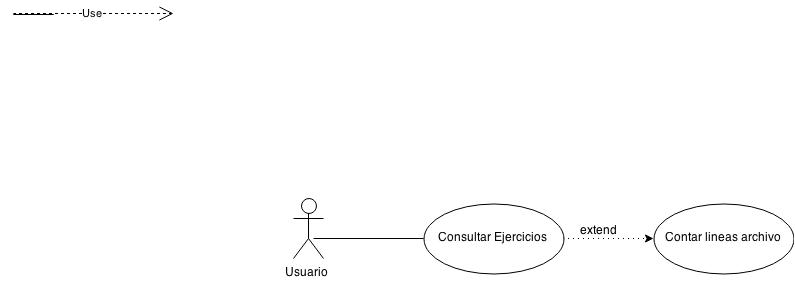
* ControladorWeb: Clase encargada de comunicarse con las vistas y el integrador para poder mostrar finalmente el contenido.
* Vista Base: Vista que muestra el listado de ejercicios
* VistaContadorLineas: Es la vista encargada de generar el HTML del ejercicio número 1 el cual debe contar las líneas de código un archivo
* Integrador: Clase que se encarga de agrupar todas las subclases que contienen la lógica de cada una de los ejercicios. Adicionalmente le pasa el modelo al controlador.
* Lector de Archivos: Clase genérica que lee archivos y retorna las líneas del mismo
* Contador Lineas: Clase encargada de ejecutar la lógica del ejercicio 1 en el cual se deben contar las líneas de código un archivo.

* 1. Diagrama de despliegue

Ya que el sistema no contiene suficiente interacción con sistemas externos o acceso a Base de datos, solo se contará con un servidor web el cual será el encargado de almacenar la aplicación y exponerla en internet. Los usuarios podrán consumir el aplicativo desde sus dispositivos por medio de un navegador web.



* 1. Diagrama de casos de uso
* Consultar Ejercicios (CU-01) : El usuario accede al aplicativo y consulta el listado de ejercicios disponibles (Para el ciclo uno solo se encontrará uno)
* Contar Líneas Archivo (CU-02): El usuario selecciona el ejercicio que contará las líneas de código de un archivo que el mismo cargue.



* 1. Diagrama de componentes

El proyecto cuenta con cuatro componentes principales:

* Controlador: Encargado de generar la comunicación entre la vista y el modelo.
* Vista: Son las encargadas de generar el HTML que se mostrará en los navegadores de los usuarios.
* Integrador: Encargado de ejecutar toda la lógica de negocio referente a cada uno de los ejercicios
* Utilidades: Clases que se encargan de hacer tareas generales del aplicativo como leer archivos físicos.
* 