

## PRIMERA ENTREGA – PROGRAMA: SISTEMA DE EVALUACIÓN

### INTEGRANTES:

1. Alexander Sánchez Sánchez - A00368238
2. Santiago Arévalo Valencia - A00368494

### JUSTIFICACIÓN DEL TRABAJO EN EQUIPO

Según el contenido y los requerimientos del proyecto “Sistema de Evaluación”, consideramos pertinente y apropiada la propuesta de que el grupo de trabajo esté conformado por 2 integrantes. Esto, basados en la necesidad de tener una adecuada división de tareas y requerimientos durante el proceso de desarrollo del programa, teniendo en cuenta el tiempo destinado para la elaboración de este y la carga académica que como estudiantes tenemos con demás cursos. Consideramos que, en esta ocasión, el proyecto que se desea llevar a cabo abarca un número considerable de temas a implementarse, por lo que es oportuno tener una cooperación durante el desarrollo, no solamente por lo eficiente que puede ser la realización de diferentes puntos del proyecto de forma simultánea, sino también por el aporte de ideas que se puede dar y la innovación y creatividad que esto generaría para el proyecto.

Por lo anterior, reiteramos la necesidad de trabajar en equipo de dos integrantes, para el desarrollo del programa “Sistema de Evaluación”

## Proyecto: Sistema de Evaluación

### Enunciado

Actualmente la virtualidad ha traído muchos cambios prácticos a la educación, dentro de estos la forma en que se evalúan a los estudiantes en las diferentes temáticas de los cursos. Por esto, la I.E María Antonia Penagos (Palmira, Valle del Cauca) solicita el desarrollo de un programa que permita, a grandes rasgos, registrar, formular y administrar exámenes virtuales que desempeñaran, posteriormente, sus estudiantes durante los cursos.

Por consiguiente, se hace indispensable que este programa cumpla con unos requisitos planteados por el grupo de docentes de la institución para así poder llevar a cabo las actividades educativas con la menor cantidad de contratiempos posible.

De acuerdo con estos requerimientos solicitados, la institución educativa requiere que el programa cuente con una interfaz gráfica sencilla e intuitiva que permita al profesorado operar el programa fácilmente y, al tiempo, cumplir con sus labores evaluativas de manera habitual. Además, se requiere que el programa almacene su propia información en archivos serializados que permitan retomar el estado previo a un cierre de sesión.

Luego, el programa debe estar en la capacidad de gestionar (es decir, crear, modificar y borrar), en primera instancia, cualquier tipo de evaluación que sea pertinente realizar, es decir, evaluaciones como: exámenes, actividades, cuestionarios y talleres. Toda evaluación debe tener un nombre, las calificaciones, un porcentaje expresando la importancia de la evaluación en la nota definitiva del estudiante, la fecha en la que será realizado y un espacio de texto para redactar la evaluación. Además, cada evaluación debe almacenar los estudiantes que la realizan a través de un árbol binario. Todo cuestionario y taller es una actividad. En segunda instancia, el programa también debe estar en la capacidad de gestionar a los miembros de la institución, esto es aquellos que evalúan y son evaluados (profesores y estudiantes). Todo miembro debe tener un nombre, un apellido, un correo y un código. Y en tercera instancia, debe gestionar todos los cursos. Estos tienen como atributos el nombre del grado, una lista de estudiantes y una lista de evaluaciones.

Los exámenes deben tener un tiempo límite de realización. También, toda actividad tiene una fecha en la que se deja la misma y enlaces de ayuda. Así mismo, los cuestionarios deben tener un espacio para especificar la cantidad de intentos que tiene y método de calificación (opcional). Y, finalmente, los talleres deben tener un espacio para las respuestas.

Cada estudiante, además de los atributos como miembro, tiene una nota promedio definitiva. Y un profesor, aparte de sus atributos como miembro, tiene también un atributo que muestra si es profesor tiempo completo o medio tiempo, una lista de cursos, su profesión, un contador de exámenes hechos y una lista de las notas promedio de todos los estudiantes por cada curso.

Los estudiantes tienen que almacenarse en una colección o lista para así poder ser buscados de forma binaria a través de algún algoritmo propio usando como criterios el nombre y el código.

Además, los estudiantes deben poder ser ordenados de acuerdo con su nombre en orden lexicográfico ascendente y también con su nota en orden descendente. También, los mismos tendrán que guardarse en un árbol binario que represente las notas definitivas de todos los estudiantes en el curso.

Todos los árboles binarios usados deben almacenar los datos de forma recursiva ordenados por nota de manera descendente y mostrados en el mismo orden al recorrer la estructura en inorden.

Similarmente, los profesores y los cursos tienen que ser manejados en listas enlazadas propias las cuales agreguen y eliminen sus datos de manera recursiva. Lo mismo aplica para los árboles binarios usados en el modelo.

Al momento de manejar el programa, se deben manejar 4 excepciones propias del API de Java: `NumberFormatException`, para prevenir errores al momento de digitar las notas en un formato que no es numérico; `FileNotFoundException`, cuando no se encuentre el archivo serializado dentro de la carpeta de datos; `NullPointerException`, en este caso el programa podrá seguir funcionando en caso de que, al buscar a un estudiante, este no se encuentre y no maneje valores nulos; y finalmente, `InterruptedException`, en caso de un cierre inesperado en el programa que lance la excepción en un hilo. Además, se deben manejar 3 excepciones personalizadas: `ExistingCodeException`, para prevenir estudiantes duplicados, es decir, el registro de un estudiante ya existente; `EmptyEvaluationException`, en caso de que se intente crear una evaluación vacía; y, por último, `OutOfDateException`, para cuando se intenta establecer una fecha de inicio para una evaluación anterior a la fecha actual del sistema.

El programa debe estar en la capacidad de exportar e importar, por medio de archivos de texto plano (.txt), los registros de los estudiantes por curso y el listado de evaluaciones hechas en cada curso.

Todos los métodos usados en este programa deberán tener un diseño de casos de prueba y su respectiva implementación para asegurar el buen funcionamiento de este para diferentes situaciones que se presenten.

Finalmente, el programa debe manejar al menos 3 hilos adicionales al hilo principal. Estos hilos se usarán al momento de buscar de forma binaria un estudiante; como división de este proceso, al mostrar y actualizar permanentemente la hora en las ventanas del programa y para calcular y actualizar el promedio de un estudiante cuando un profesor agrega una evaluación.

## REQUERIMIENTOS FUNCIONALES

→ **Req1. Autenticar** el ingreso al sistema de un profesor (Login).

- ❖ **Req1.1 Verificar** que el código ingresado coincida con alguno de los códigos de profesores existentes en el sistema.
- ❖ **Req1.2 Verificar** que la contraseña ingresada coincida con la asociada al código anterior.

→ **Req2. Registrar** un nuevo profesor en el sistema, solicitando la información necesaria.

→ **Req3. Gestionar** los miembros de la institución.

- ❖ **Req3.1** Crear un nuevo miembro de la institución. Esto es crear un nuevo profesor o estudiante con sus respectivos atributos y agregarlo a la lista general de estudiantes o profesores de la institución.
- ❖ **Req3.2** Modificar la información de un miembro de la institución, es decir, de un profesor o un estudiante.
- ❖ **Req3.3** Borrar el registro de un miembro de la institución. Esto es, eliminar un profesor o un estudiante del sistema y, por lo tanto, removerlo de la lista general de profesores o estudiantes.

→ **Req4. Gestionar** los cursos de la institución.

- ❖ **Req4.1** Crear un nuevo curso en la institución y agregarlo a la lista general de cursos del sistema.
- ❖ **Req4.2** Modificar la información de un curso en la institución.
  - **Req4.2.1** Agregar un estudiante existente a la lista de estudiantes del curso actual.
  - **Req.4.2.2** Agregar una evaluación a la lista de evaluaciones del curso (Req5.1).
- ❖ **Req3.3** Borrar el registro de un curso de la institución. Esto es, eliminar un curso del sistema y, por lo tanto, removerlo de la lista general de cursos.

→ **Req5. Gestionar** las evaluaciones de un curso.

- ❖ **Req5.1** Crear una nueva evaluación para un curso. Esto es, crear un examen o una actividad, la cual a su vez también puede ser un cuestionario o un taller. Y posteriormente, agregar esta a la lista de evaluaciones del curso.

- ❖ **Req5.2** Modificar la información de una evaluación, es decir, modificar cualquier atributo de un examen, de una actividad, de un cuestionario o de un taller.
- ❖ **Req5.3** Borrar el registro de una evaluación en el curso. Esto es, eliminar una evaluación del curso y, por lo tanto, removerlo de la lista de evaluaciones del curso.
- **Req6. Almacenar** toda la información del sistema en archivos serializados (Serialización).
- **Req7. Importar/Exportar** parte de la información del sistema
  - ❖ **Req7.1 Importar y exportar** la lista de estudiantes y la lista de evaluaciones de un curso.
  - ❖ **Req7.2 Importar y exportar** la lista de evaluaciones de cada curso.
- **Req8. Buscar** un estudiante utilizando el algoritmo de búsqueda binaria con los criterios de nombre y código.
- **Req9. Ordenar** los estudiantes de acuerdo con su nombre en orden lexicográfico ascendente y también con su nota en orden descendente.
- **Req10. Almacenar** en dos árboles binarios: todos los estudiantes que presentan una evaluación y todos los estudiantes de un curso ordenados por nota definitiva de ese curso.
  - ❖ **Req10.1 Ordenar** los árboles binarios implementados por nota de manera descendente.
  - ❖ **Req10.2 Mostrar** los árboles binarios implementados en orden descendente al recorrer la estructura en inorden.

## Mockups del programa

- **Página de Registro**

The image shows a web browser window with the address bar displaying 'register.fxml'. The page has a dark gray background and a large white title 'Registrate' at the top. Below the title, there are five input fields arranged in three rows: 'Nombre' and 'Código' in the first row, 'Apellido' and 'Profesión' in the second row, and 'Correo' in the third row. To the right of the 'Correo' field, there is a label 'Disponibilidad:' followed by a green button labeled 'Tiempo Completo'. At the bottom of the form, there are two blue buttons: 'Atrás' on the left and 'Registrarme' on the right.

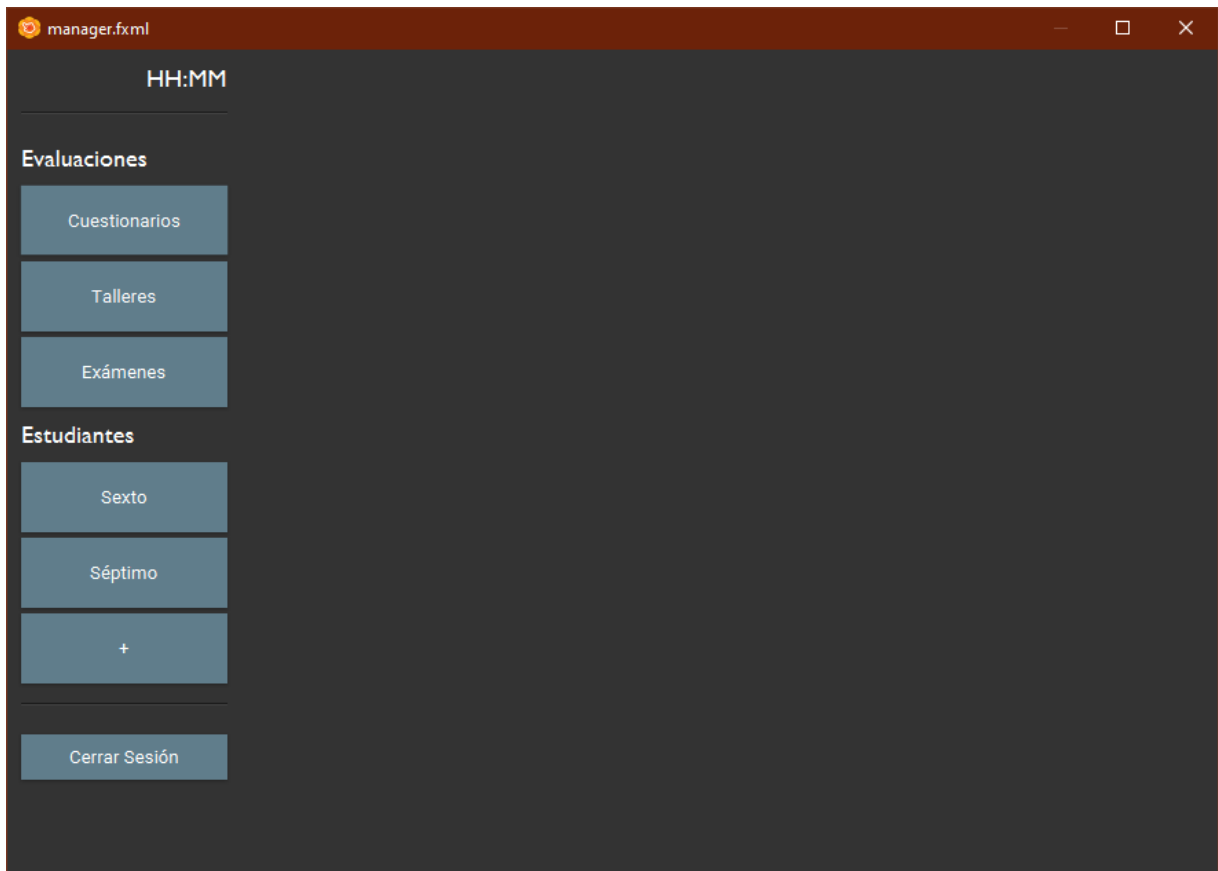
*¿Por qué no tener una cuenta en nuestro fabuloso programa? En esta pantalla puede crear una cuenta de profesor y destacar la gestión de todos sus cursos y evaluaciones.*

- **Panel de cuestionarios**

The screenshot shows a window titled 'questionnaire-pane.fxml'. The main heading is 'Cuestionarios'. On the left, there is a label 'Escoja un curso:' above a dropdown menu. Below this are two blue buttons: 'Importar Lista' and 'Exportar Lista'. The main area features a table with columns: 'Nombre', '%', 'Fecha Inicio', 'Fecha Fin', and 'Intentos'. Below the table is a section titled 'Información:' containing several input fields: 'Nombre', '%', 'Fecha Inicio' (with a calendar icon), and 'Fecha Fin' (with a calendar icon). To the right of these is a vertical spinner control. Below the 'Información:' section is a large white box labeled 'Contenido del cuestionario'. At the bottom right, there is a blue button labeled 'Crea un nuevo cuestionario'.

*El panel de cuestionarios ayuda al profesor a manejar la creación y el cambio de los cuestionarios, pero todo en uno.*

- **Pantalla de gestión**



*Desde aquí, el profesor puede gestionar todo el sistema.  
Puede acceder a todas las evaluaciones y a todos los  
alumnos.*



- **Pantalla de inicio de sesión**



*La pantalla de inicio de sesión proporciona una forma intuitiva de ingresar a la aplicación simplemente recordando el nombre y el código del maestro.*

- **Panel de exámenes**


The screenshot shows a software window titled "assessment-pane.fxml" with standard window controls. The main content area is titled "Exámenes". On the left, there is a label "Escoja un curso:" above a dropdown menu. To the right of this is a table with four columns: "Nombre", "%", "Fecha Fin", and "Tiempo Límite". Below the table is a section titled "Información:" containing four input fields: "Nombre", "%", "Tiempo Límite", and "Fecha Fin". The "Fecha Fin" field has a calendar icon. Below the "Información:" section is a large text area labeled "Contenido del examen". At the bottom of the window, there are three buttons: "Importar Lista", "Exportar Lista", and "Crea un nuevo taller".

*Desde lo más fácil hasta lo más difícil. Realice un seguimiento de todos sus próximos exámenes utilizando este panel: crearlos, modificarlos y eliminarlos. Todo lo que necesitas en un solo lugar.*

- **Panel de cursos**

course-pane.fxml

## Curso 03



**Nuevo Estudiante:**

Nombre

Apellido

Código

Correo

Crear

Importar Lista

Exportar Lista

Nombre	Apellido	Código	Correo
--------	----------	--------	--------

**Información:**

Nombre


Apellido

Código

Correo

Nota

Actualizar



*Este panel se explica por sí mismo. Desde aquí, el profesor puede visualizar a todos los alumnos de un curso, modificar su información y crear alumnos.*

- **Panel de talleres**

The screenshot shows a software window titled 'assessment-pane.fxml' with a dark theme. On the left, there's a sidebar with the title 'Talleres' and a dropdown menu labeled 'Escoja un curso:'. The main area contains a table with columns 'Nombre', '%', 'Fecha Inicio', and 'Fecha Fin'. Below the table is a section titled 'Información:' with input fields for 'Nombre', '%', 'Fecha Inicio', and 'Fecha Fin', each with a calendar icon. Underneath is a large text area labeled 'Contenido del taller'. At the bottom, there are four blue buttons: 'Importar Lista', 'Exportar Lista', 'Ver respuestas', and 'Crea un nuevo taller'.

Nombre	%	Fecha Inicio	Fecha Fin
--------	---	--------------	-----------

**Información:**

Nombre:  %:

Fecha Inicio:  Fecha Fin:

**Contenido del taller**

Importar Lista Exportar Lista Ver respuestas Crea un nuevo taller

*Las evaluaciones son una excelente manera de hacer que nuestros estudiantes progresen, así que, ¿por qué no hacer un seguimiento de ellos?*

# README

## School Evaluation System

**School Evaluation System** is a program for institutional use that allows a teacher to formulate evaluations in the different courses he/she is in charge of.

This program allows a teacher to enter the system and then view two sections, evaluations and students. In the evaluations section you can enter the questionnaires, workshops or exams that you have formulated for a course and then you can edit or delete an evaluation. You can also import or export evaluations. In the student's section, you can create a new course by entering its name or select from the list the course you want to enter and then you will have a visualization of the list of students of that course, as well as the options to edit the information of a selected student and create a new student. You will also be able to import or export the list of students.

💡 The idea of the **School Evaluation System** project was born from the need for a total transition to technology due to the change to virtuality in education. This project seeks to provide a tool to any teacher in their evaluation management with students.

## Creators

This program was created unique and exclusively by [Santiago Arévalo \(santiagoarevalo\)](#) and [Alexander Sánchez \(ALEXJR2002\)](#).

## Project Design

The program design (functional requirements and class diagram) is documented in the following [link](#).

## Technical Conditions

*Language:* Java 1.8

*Operative System:* Windows 10 x64

*Development Environment:* Eclipse IDE

## Requerimientos No Funcionales

- **Req1. Gestionar** los campos donde se puede modificar la información de los objetos (Estudiantes y evaluaciones)
  - ❖ **Req1.1 Habilitar** los campos una vez se selecciona un algún objeto.
- **Req2. Mostrar** una pantalla de alerta cuando, en la pantalla de login, no coincida el código y la contraseña.
- **Req3. Alertar** al usuario cuando el profesor que intenta registrar ya existe.
- **Req4. Crear** un nuevo curso a través de una pequeña pantalla de diálogo.
- **Req5. Guardar** la información del programa en segundo plano cada vez que el usuario modifique el sistema.
- **Req6. Permitir** registrar a un nuevo profesor en cualquier momento desde la pantalla de “login”. Es decir, el número de profesores puede ser mayor a 1 y seguir permitiendo registrar a otro más.
- **Req7. Exportar** la información del panel en el que esté en un archivo en formato CSV (.csv)
- **Req8. Importar** los archivos exportados en formato CSV (.csv) e insertarlos correctamente en el sistema.
- **Req9. Mostrar** la hora actualizada en pantalla principal de la aplicación.
- **Req10. Permitir** el registro de un estudiante con su foto correspondiente.

## DISEÑO DE PRUEBAS UNITARIAS

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenary1	EvaluationSystemTest	<pre> sequenceDiagram     participant ES as :EvaluationSystem     participant T as :Teacher     ES-&gt;&gt;T: firstTeacher           </pre>
setupScenary2	EvaluationSystemTest	<pre> sequenceDiagram     participant ES as :EvaluationSystem     participant T1 as :Teacher name = Carlos lastName = Martinez email = carlos@gmail.com code = 1005783 password = carlos123 fullTime = true     participant T2 as :Teacher name = Manuel lastName = Reyes email = seyerman@gmail.com code = 1006738 password = reyes123 fullTime = false     participant T3 as :Teacher name = Diana lastName = Carvajal email = diana@gmail.com code = 1005903 password = diana321 fullTime = true     ES-&gt;&gt;T1: firstTeacher     T1-&gt;&gt;T2: next     T2-&gt;&gt;T3: next           </pre>
setupScenary1	CourseTest	<pre> sequenceDiagram     participant C as :Course     participant S as 0..* students     C-&gt;&gt;S: students           </pre>
setupScenary2	CourseTest	<pre> sequenceDiagram     participant C1 as :Course     participant C2 as :Course name = 11-2     participant C3 as :Course name = 10     participant C4 as :Course name = 8     C1-&gt;&gt;C2: students     C1-&gt;&gt;C3: students     C1-&gt;&gt;C4: students           </pre>
setupScenary1	TeacherTest	<pre> sequenceDiagram     participant T as :Teacher     participant C as :Course     T-&gt;&gt;C: firstCourse           </pre>
setupScenary2	TeacherTest	<pre> sequenceDiagram     participant T as :Teacher     participant C1 as :Course name = 7-3     participant C2 as :Course name = 8-1     participant C3 as :Course name = 5-2     T-&gt;&gt;C1: firstCourse     C1-&gt;&gt;C2: next     C2-&gt;&gt;C3: next           </pre>

## Diseño de Casos de Prueba

Objetivo de la Prueba: Validar que se crea y se agrega correctamente un profesor al sistema.				
Clase	Método	Escenario	Valores de Entrada	Resultado
EvaluationSystem	addTeacher	setupScenary1	name = Carlos lastName = Martínez email = carlos@gmail.com code = 1005677 password = carlos123 fullTime = true	true Se creó y se agregó correctamente el nuevo profesor llamado Carlos a la lista de profesores. La lista de profesores en el sistema ahora tiene un tamaño igual a 1.
EvaluationSystem	addTeacher	setupScenary2	name = Pipe lastName = Martínez email = pipe@gmail.com code = 1005667 password = pipejdj fullTime = false	true Se creó y se agregó correctamente el nuevo profesor llamado Carlos a la lista de profesores. La lista de profesores en el sistema ahora tiene un tamaño igual a 4.

Objetivo de la Prueba: Validar que se crea y se agrega correctamente un curso al sistema.				
Clase	Método	Escenario	Valores de Entrada	Resultado
Teacher	addCourse	setupScenary1	name = 11-2	true Se creó y se agregó correctamente el nuevo curso llamado 11-2 a la lista de cursos. La lista de cursos en el sistema ahora tiene un tamaño igual a 1.
Teacher	addCourse	setupScenary2	name = 11-2	true Se creó y se agregó correctamente el nuevo curso llamado 11-2a la lista de cursos. La lista de profesores en el sistema ahora tiene un tamaño igual a 4.



**Objetivo de la Prueba:** Validar que se crea y se agrega correctamente un estudiante al sistema.

Clase	Método	Escenario	Valores de Entrada	Resultado
Course	addStudent	setupScenary1	name = Alex lastName = Martínez email = alex@gmail.com code = 1001277	true Se creó y se agregó correctamente el nuevo estudiante llamado Alex a la lista de estudiantes. La lista de estudiantes en el sistema ahora tiene un tamaño igual a 1.
Teacher	addStudent	setupScenary2	name = Santiago lastName = Perez email = santiago@gmail.com code = 100563738	true Se creó y se agregó correctamente el nuevo estudiante llamado Santiago a la lista de estudiantes. La lista de estudiantes en el sistema ahora tiene un tamaño igual a 4.

