# Base de Datos Continental – TI2 CyED

## Especificación del Problema

|  |  |
| --- | --- |
| **Cliente** | Universidad Icesi |
| **Usuario** | * Profesores |
| **Requerimientos Funcionales** | 1. Generar registros a partir de las bases de datos 2. Persistencia de los datos generados 3. Agregar una nueva persona 4. Buscar una persona 5. Actualizar una persona existente |
| **Contexto del problema** | Se requiere almacenar una cantidad muy grande de datos. Estos datos serán personas, y cada persona contara con nombre, apellido, y código. Se desea que la búsqueda se realice de una forma muy eficiente, por lo que tenemos que implementar un código para volver esto una búsqueda rapida |
| **Requerimientos no Funcionales** | 1. Se debe contar con una interfaz de usuario. 2. Se deben guardar las bases de datos desde donde se generarán los registros en un directorio (/data). 3. Obtener información de los nombres y apellidos para la simulación de creación de registros a partir de los datos de la página “data.world”[[1]](#footnote-2). 4. Generar los datos de edad para los registros con una distribución que esté basada en la distribución dada en la página “index mundi”[[2]](#footnote-3). 5. Generar la estatura aleatoriamente en un intervalo que tenga sentido. 6. Asignar nacionalidad de tal forma que se mantengan los porcentajes dados en el post “Population by Country - 2020” en la página “kaggle”[[3]](#footnote-4). 7. Se deben implementar pruebas unitarias automáticas. |

## Requerimiento funcional 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre o identificador** | R1: Generar registros a partir de las bases de datos | | |
| **Resumen** | El sistema debe poder simular la creación un registro de máximo un millón de personas. Cada registro debe tener un código, nombre, apellido, sexo, fecha de nacimiento, estatura y nacionalidad. Se debe tener como opción empezar a generar los registros con un campo de texto, en donde se especificará cuántos registros se generarán. Por defecto se genera el máximo posible. Una barra de progreso debe indicar cuánto se está demorando el progreso y al final se muestra cuánto se demoró. | | |
| **Entradas** | **Nombre entrada** | **Tipo de dato** | **Condición de selección o repetición** |
| N/A | N/A | N/A |
| **Actividades generales necesarias para obtener los resultados** | 1. Generar de formar aleatoria todos los datos 2. Verificar que el código autogenerado no este siendo usado por otro usuario | | |
| **Resultado o Postcondición** | Un millón de usuarios creados (O la cantidad escrita) y cada uno con un código único | | |
| **Salidas** | **Nombre salida** | **Tipo de dato** | **Condición de selección o repetición** |
| N/A | N/A | N/A |

## Requerimiento funcional 2

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre o identificador** | R2: Persistencia de los datos generados | | |
| **Resumen** | Una vez los datos se generan, debe haber una opción para guardarlos en la base de datos del programa, y así poderlos consultar posteriormente. Todos los datos del programa deben ser persistentes. | | |
| **Entradas** | **Nombre entrada** | **Tipo de dato** | **Condición de selección o repetición** |
| N/A | N/A | N/A |
| **Actividades generales necesarias para obtener los resultados** | 1. Tener al menos un usuario creado | | |
| **Resultado o Postcondición** | Un archivo json con los datos de los usuarios generados | | |
| **Salidas** | **Nombre salida** | **Tipo de dato** | **Condición de selección o repetición** |
| N/A | N/A | N/A |

## Requerimiento funcional 3

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre o identificador** | R3: Agregar una nueva persona | | |
| **Resumen** | En el sistema se debe poder añadir una nueva persona con un formulario en la interfaz que reciba la información correspondiente. | | |
| **Entradas** | **Nombre entrada** | **Tipo de dato** | **Condición de selección o repetición** |
| Nombre completo | String | Se debe de componer de dos partes, un nombre y un apellido |
| Genero | String | El comboBox no puede estar vacío |
| Fecha de nacimiento | LocalDate | El LocalDate no puede estar vacío |
| Altura en metros | double | El campo debe tener al menos un número |
| Nacionalidad | String | El campo no puede estar vacío |
| **Actividades generales necesarias para obtener los resultados** | 1. Pedir todos los datos y verificar cada una de sus condiciones | | |
| **Resultado o Postcondición** | Un usuario nuevo con todos los datos ingresados y un código unico | | |
| **Salidas** | **Nombre salida** | **Tipo de dato** | **Condición de selección o repetición** |
| nuevoUsuario | Person |  |

## Requerimiento funcional 4

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre o identificador** | R4: Buscar una persona | | |
| **Resumen** | El sistema debe permitir buscar una persona por los criterios de   * Nombre * Apellido * Nombre completo * Código   El programa debe permitir que, a medida que se digiten los caracteres de búsqueda, vayan apareciendo en una lista emergente debajo del campo, máximo 100 nombres de la base de datos que empiecen con los caracteres digitados hasta el momento.  Al lado del campo desde donde se digita la cadena, debe aparecer el número de elementos coincidentes. | | |
| **Entradas** | **Nombre entrada** | **Tipo de dato** | **Condición de selección o repetición** |
| Tipo de búsqueda | String | ComboBox seleccionado |
| Condición de busqueda | String | N/A |
| **Actividades generales necesarias para obtener los resultados** | 1. Empezar a comparar en el árbol avl hasta obtener una coincidencia para después hacer un preorden y obtener las personas con la condición de busqueda | | |
| **Resultado o Postcondición** | Usuarios que tengan coincidencia en la condición de búsqueda (máximo 100) | | |
| **Salidas** | **Nombre salida** | **Tipo de dato** | **Condición de selección o repetición** |
| coincidenciaDeUsuarios | ArrayList<Person> |  |

## Requerimiento funcional 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Nombre o identificador** | R5: Actualizar una persona existente | | |
| **Resumen** | Al haber 20 (parametrizable) elementos que coinciden con la búsqueda, debe desplegarse un listado de registros que coinciden y un botón al lado con la opción de editar. Este botón desplegará un formulario en el que se podrán editar todos los campos de la persona exceptuando su código, y una opción de eliminar este registro. | | |
| **Entradas** | **Nombre entrada** | **Tipo de dato** | **Condición de selección o repetición** |
| Nombre completo | String | Se debe de componer de dos partes, un nombre y un apellido |
| Genero | String | El comboBox no puede estar vacío |
| Fecha de nacimiento | LocalDate | El LocalDate no puede estar vacío |
| Altura en metros | double | El campo debe tener al menos un número |
| Nacionalidad | String | El campo no puede estar vacío |
| **Actividades generales necesarias para obtener los resultados** | 1. Seleccionar la opción de editar si todos los campos están llenos (Todo se puede editar menos el codigo). | | |
| **Resultado o Postcondición** | Un usuario con algunos datos cambiados menos el codigo | | |
| **Salidas** | **Nombre salida** | **Tipo de dato** | **Condición de selección o repetición** |
| usuario | Person |  |

1. https://data.world/alexandra/baby-names [↑](#footnote-ref-2)
2. https://www.indexmundi.com/es/estados\_unidos/distribucion\_por\_edad.html [↑](#footnote-ref-3)
3. https://www.kaggle.com/datasets/tanuprabhu/population-by-country-2020 [↑](#footnote-ref-4)