PLAN DE PROYECTO

Karen Dahiana Gómez Gómez CC. 1040873146

Descripción de las actividades

Generación de Nombres Aleatorios

- Definimos una función llamada *'generar_nombres_aleatorios(nombres, apellidos, cantidad)'* para generar nombres completos aleatorios para una cantidad específica de estudiantes.
- La función toma tres argumentos: *'nombres'* (una lista de nombres), *'apellidos'* (una lista de apellidos) y *'cantidad'* (la cantidad de nombres que se desean generar).
- Utilizamos dos bucles *'for'* anidados para combinar cada nombre con cada par de apellidos, generando así todas las posibles combinaciones de nombres completos.
- Luego, mezclamos aleatoriamente las combinaciones de nombres utilizando *'random.shuffle()'*.
- Creamos un conjunto ('set') llamado 'nombres_completos' para almacenar los nombres generados y garantizar que no haya duplicados.
- Iteramos sobre las combinaciones de nombres mezclados y los agregamos al conjunto 'nombres_completos', deteniéndonos una vez que hemos alcanzado la cantidad deseada de nombres únicos.
- Finalmente, convertimos el conjunto 'nombres_completos' en una lista y la devolvemos como el resultado de la función.

Este proceso asegura que cada estudiante tenga un nombre completo único y aleatorio, lo que es fundamental para la identificación individual de los estudiantes en el sistema. La generación aleatoria de nombres a partir de las listas de nombres y apellidos garantiza una distribución equitativa y diversa de nombres entre los estudiantes simulados.

Generación de Correos Electrónicos, Sedes y Documentos de Identidad

- Los correos electrónicos se generan concatenando el nombre del estudiante, un número aleatorio y un dominio predefinido.
- Los documentos de identidad se generan como una cadena de 10 dígitos aleatorios.

1. Generación de Correos Electrónicos:

- Definimos una función llamada *'generar_correos_electronicos(nombres)'* para generar correos electrónicos aleatorios para cada estudiante.
- Iteramos sobre la lista de nombres proporcionada como entrada.
- Para cada nombre, convertimos el nombre a minúsculas y reemplazamos los espacios con guiones bajos para crear el nombre de usuario del correo electrónico.
- Luego, agregamos un dominio predefinido *'(@universidad.edu.co)'* al nombre de usuario.
- Generamos un número aleatorio de 4 dígitos utilizando *'random.choices()'* y lo concatenamos al nombre de usuario para crear el correo electrónico completo.
- Agregamos cada correo electrónico generado a una lista de correos electrónicos.
- Finalmente, devolvemos la lista de correos electrónicos generados.

2. Generación de Documentos de Identidad:

- Definimos una función llamada *'generar_documentos_identidad(cantidad)'* para generar números de documento de identidad aleatorios para la cantidad especificada de estudiantes.
- Utilizamos un bucle *'for'* para generar la cantidad deseada de documentos de identidad.
- En cada iteración, generamos una cadena de 10 dígitos aleatorios utilizando *'random.choices()'.*
- Agregamos cada documento de identidad generado a una lista de documentos de identidad.

- Finalmente, devolvemos la lista de documentos de identidad generados.

Estos procesos aseguran que cada estudiante tenga un correo electrónico único y un número de documento de identidad único, lo que es esencial para mantener la integridad de los datos y facilitar la identificación de los estudiantes en el sistema.

3. Asignación de Sedes:

- Definimos una lista de sedes disponibles.
- Creamos una función llamada '*asignar_sedes(estudiantes, sedes)*' que toma como entrada la lista de estudiantes y la lista de sedes disponibles.
- En cada iteración sobre los estudiantes, seleccionamos aleatoriamente una sede de la lista de sedes disponibles utilizando '*random.choice()*'.
- Asignamos la sede seleccionada al estudiante y actualizamos el diccionario del estudiante con la nueva sede.
- Luego, llamamos a la función '*asignar_sedes()*' pasando la lista de estudiantes y la lista de sedes disponibles para asignar las sedes a los estudiantes.
- Finalmente, imprimimos los datos actualizados de los estudiantes con las sedes asignadas.

Asignación de Estudiantes a Clases

- Definimos una función llamada 'asignar_estudiantes_a_clases(estudiantes, clases)' para asignar aleatoriamente a los estudiantes a las diferentes clases disponibles.
- La función toma dos argumentos: 'estudiantes' (una lista de diccionarios que contienen los datos de los estudiantes) y 'clases' (una lista de clases disponibles).
- Iteramos sobre cada clase en la lista de clases disponibles.
- Para cada clase, seleccionamos aleatoriamente un número de estudiantes de la lista de estudiantes para llenar el cupo de esa clase.
- Utilizamos 'random.sample()' para seleccionar aleatoriamente estudiantes sin reemplazo, lo que garantiza que cada estudiante se asigne a una sola clase.

- Asignamos los estudiantes seleccionados a la clase actual.
- Repetimos este proceso para cada clase en la lista de clases disponibles.
- La función devuelve un diccionario que contiene la asignación de estudiantes a clases, donde las claves son los nombres de las clases y los valores son listas de estudiantes asignados a cada clase.

Este proceso garantiza una distribución equitativa y aleatoria de estudiantes en las clases disponibles, lo que es esencial para la diversidad y la representación equitativa de los estudiantes en el sistema educativo simulado.

Guardado de Listados de Clases:

- Después de haber asignado a los estudiantes a las diferentes clases disponibles, es fundamental almacenar esta información en archivos para su posterior referencia y utilización.
- La función 'guardar_listados_de_clases(asignaciones, ruta)' se encarga de realizar esta tarea, donde 'asignaciones' es un diccionario que contiene la asignación de estudiantes a clases y 'ruta' es la ubicación donde se guardarán los archivos.
- En primer lugar, iteramos sobre cada clase en el diccionario de asignaciones.
- Para cada clase, creamos un archivo, ya sea en formato Excel o CSV, según los requisitos del proyecto, para almacenar la lista de estudiantes asignados a esa clase.
- Luego, escribimos los datos de los estudiantes asignados a la clase en el archivo correspondiente. Esto puede implicar la creación de una tabla o listado con los estudiantes y sus respectivas información (nombre, correo electrónico, etc.).
- Finalmente, verificamos que los archivos se hayan guardado correctamente en la ubicación especificada.

En resumen, el proceso de Guardado de Listados de Clases convierte la asignación de estudiantes a clases en archivos de datos que pueden ser almacenados de forma estructurada y recuperados fácilmente en el futuro. Esto garantiza que la información esté disponible para su posterior análisis, seguimiento del progreso de los estudiantes, generación de informes, entre otros propósitos.

Implementación del Registro en el Archivo de Log:

- La implementación del registro en el archivo de log es crucial para el seguimiento y la depuración del programa durante su ejecución.
- Para implementar esta funcionalidad, primero definimos una función llamada ‘registrar_evento(evento, tiempo, accion, informacion_adicional)’ que se encarga de registrar eventos específicos en el archivo de log.
- Esta función toma como entrada los detalles del evento que se va a registrar, como el tipo de evento (‘evento’), la marca de tiempo (‘tiempo’), la acción realizada (‘accion’) y cualquier información adicional relevante (‘informacion_adicional’).
- Dentro de la función, abrimos el archivo de log en modo de escritura ('w' o 'a'), dependiendo de si se desea sobrescribir el archivo existente o agregar nuevos registros al final del archivo.
- Luego, formateamos el mensaje de registro utilizando la información proporcionada y lo escribimos en el archivo de log.
- Es importante tener en cuenta la estructura del registro, que generalmente incluye la fecha y hora del evento, seguidas de la descripción del evento y cualquier información adicional relevante.
- Una vez que se ha registrado el evento, cerramos el archivo de log para asegurarnos de que los cambios se guarden correctamente y se liberen los recursos del sistema.

En resumen, la implementación del registro en el archivo de log proporciona una manera efectiva de rastrear y documentar las acciones y eventos importantes durante la ejecución del programa, lo que facilita la depuración y el análisis posterior del funcionamiento del software.

Cronograma

1. Semana 1:

- Generación de Nombres Aleatorios:
    - Lectura de archivos CSV de nombres y apellidos.
    - Implementación de la generación aleatoria de nombres completos.
- Generación de Correos Electrónicos:
    - Desarrollo de la función para generar correos electrónicos aleatorios.
- Selección de Sedes:
    - Creación de una lista de sedes disponibles.
    - Implementación de la selección aleatoria de sedes para los estudiantes.
- Generación de Documentos de Identidad:
    - Creación de la función para generar números de documento de identidad aleatorios.

2. Semana 2:

- Asignación de Estudiantes a Clases:
    - Definición de la función para asignar estudiantes aleatoriamente a clases.
- Guardado de Listados de Clases:
    - Diseño del método para guardar los listados de clases en archivos.
    - Implementación de la escritura de los datos de los estudiantes en los archivos correspondientes.

3. Semana 3:

- Implementación del Registro en el Archivo de Log:
    - Diseño de la función para registrar eventos en el archivo de log.
    - Configuración de la estructura del archivo de log y los datos a registrar.
    - Pruebas de la función de registro y verificación de su funcionamiento.

4. Semana 4:

- Pruebas y Depuración:
    - Ejecución de pruebas exhaustivas para cada una de las funciones implementadas.
    - Identificación y corrección de errores o fallos encontrados durante las pruebas.
    - Ajustes necesarios en las funciones según los resultados de las pruebas.

Este cronograma propone un marco temporal para el desarrollo y la implementación de las tareas mencionadas, asegurando un progreso ordenado y la finalización oportuna del proyecto. Es importante ajustar las fechas y las actividades según las necesidades específicas del proyecto.

Presupuesto

Para calcular el presupuesto del proyecto tendremos en cuenta que el pago se realizará en tiempo de práctica de formación, donde 1 SMLV equivale a un total de 50 horas de práctica. Dado que el grupo está compuesto por una sola persona y el total de horas invertidas es de 50, calcularemos el valor total en términos de horas de práctica:

Total de horas invertidas por el grupo: 50 horas

Valor de 1 SMLV en horas de práctica: 50 horas

Valor total del proyecto en horas de práctica:

Valor_total = Total_horas_invertidas / Valor_SMLV
= 50 horas / 50 horas
= 1 SMLV

Por lo tanto, el presupuesto del proyecto sería equivalente a 1 SMLV en términos de horas de práctica de formación. Este valor representa la compensación que recibiría el grupo de estudiantes por su trabajo en el proyecto, de acuerdo con las condiciones establecidas.

NOTA

Es importante tener en cuenta que este documento representa únicamente un plan inicial para el desarrollo del proyecto. Proporciona una vista general detallada de las diferentes etapas y procesos involucrados en la implementación del sistema educativo simulado. Sin embargo, es importante reconocer que este plan está sujeto a cambios y ajustes a medida que avanza el desarrollo del proyecto. Se espera que surjan nuevas ideas, requisitos adicionales y desafíos durante la ejecución del proyecto, lo que podría requerir modificaciones en la estrategia y el enfoque delineados aquí. Por lo tanto, este documento sirve como una guía inicial y un punto de partida para el equipo de desarrollo, proporcionando una visión clara de los pasos a seguir y las consideraciones clave a tener en cuenta.