Índice

Sistema de gestión universitaria.

Descripción General.

El sistema de gestión universitaria es un proyecto python, basado en el SIA (sistema de información académica) de la Universidad Nacional de Colombia. Que surge de los inconvenientes que suelen experimentar los estudiantes con el exceso de materias, profesores, trabajos, sistemas e información en general. Este posee un diseño cómodo para que el usuario acceda a la información depositada con facilidad, además ofrece al usuario varias opciones para mejorar su experiencia universitaria, dónde podrá crear su propio horario académico, ver si este tiene fallas; y en caso de que si la aplicación le dará un horario recomendado, mirar su acceso a becas, calificar a sus docentes y realizar cálculos pertinentes para de la nota necesaria para pasar una materia.

Encapsulamiento

Paquetes

El proyecto consta de 5 paquetes, los cuales son:

El paquete ***"baseDatos"*** el cual contiene los módulos:

- Deserializador.py
- Serializador.py
- Una carpeta Temp que contiene lo que guarda el serializador y lee el deserializador todos los archivos de esta son .pkl

El paquete ***"Calendario"*** el cual contiene los módulos:

- Beca.py
- Facultad.py
- gestionDatos.py
- Facultad.py
- Horario.py
- Materia.py
- Tarea.py
- TareaEstudiante.py

El paquete ***"Images"*** el cual contiene:

- Este paquete contiene las imágenes que se muestran a lo largo del código, imágenes que son invocadas a través de su ruta dentro de la interfaz.

El paquete ***"personas"*** el cual contiene los módulos:

- Estudiante.py
- Persona.py (contiene Clase abstracta)
- Profesor.py

El paquete ***"UI_main"*** el cual contiene el módulo:

- Main.py

(Aclaración)

Cada paquete contiene su respectivo archivo __init__.py, esto con el fin de poder importar los módulos que se encuentren dentro de dichos paquetes.

Descripción del contenido de los módulos

El modulo ***"Persona.py"*** contiene una clase abstracta y posee los atributos:

- nombre : Str
- ID : Int
- Email : Str

Todos estos datos serán registrados para usarse en las clases ***"Estudiante.py"*** *y* **"*Persona.py"***

La clase "***Estudiante.py"*** posee los atributos:

- **super** .__init__(nombre, ID, email)
  Estos son los atributos que la clase hereda de su padre persona
- ***Private*** fallaHorario: Boolean
  Este atributo también será usado en la primera funcionalidad detectar problemas con franja horaria, el cual tomará un valor inicial de false, y si la funcionalidad detecta un problema con el horario, el atributo tomará un valor de true, y dará a lugar a la segunda funcionalidad, Sugerir Horario y materia.

- **Private** calificacion_asignada**:**Boolean

- ***Private*** fue_becado: Boolean
  Este atributo será usado en la clase ***Beca*** en el método asignarEstudiantesBeca, para comprobar que el objeto Estudiante puede aplicar o no a una de los 3 tipos de beca.

- **Public** promedio: Int
  Este atributo el cual es inicializado en 0.0, almacenará el promedio del estudiante dependiendo de de las notas de las tareas y la calificación final de las materias y el número de estas.

- **Private List** materias_inscritas: Materia
  Este atributo será usado en la primera funcionalidad detectar problemas con franja horaria, puesto a que cada materia inscrita por el estudiante tendrá su propio horario y dos materias no pueden chocar en la misma franja horaria. también se usará en el método de estudiante retirarMateria().

- ***Public List*** materias_cursadas: Materia
  Esta lista será usada en la segunda funcionalidad Sugerir Horario y materia, en el método sugerirMaterias(), en donde si el estudiante ya ha tomado esa materia antes, no será recomendado en el nuevo horario que da a lugar si el horario creado por el estudiante anteriormente presentó fallas. También esta lista se usará en el cálculo del metodo de estudiante calcularPorcetajeAvance, en donde se toma el tamaño de la lista.

- ***Public*** porcentajeDeAvance: double
  Se utilizará en el método de estudiante calcularPorcetajeAvance().
- **Public List** intento_materias
  Esta lista trabajara dentro del método estudiante inscribirMateria(), y tendrá la función de comprobar que la materia a añadir al horario por el estudiante si cumpla con lo requerido, es decir el prerrequisito y como mínimo una materia de tipo fundamentación.

- ***Private Materia*** profesores inscritos
  Esta lista inscribe a los profesores según la materia y el horario elegido por el estudiante.

La clase ***"Estudiante.py"*** posee los siguientes métodos:

- inscribir_materia:
  El método toma en primer lugar dos argumentos, los cuales son nombre_materia: Str y la lista de tipo Materia: materias_disponibles, las cuales son aptas para la inscripción.
  El método verifica si la materia con el nombre dado está en la lista de las materias disponibles y su estado de estar inscrita o no. En caso de que no esté inscrita, verifica si tiene una materia de prerrequisito y si esta ya ha sido cursada. Si no se presenta un prerrequisito o este ya ha sido cursado, se agrega la materia a la lista de materias inscritas.
  Por último el método calcula el número total de créditos de las materias inscritas, el cual debe ser mayor o igual a 10, y verifica si hay al menos una materia de fundamentación, si se cumplen estos dos requisitos se inscribe al estudiante en la materia y se devuelve True, de lo contrario, se devuelve False.

- retirar_materia(self, Materia)
  Se retira el objeto materia que se añadió a materias_inscritas.

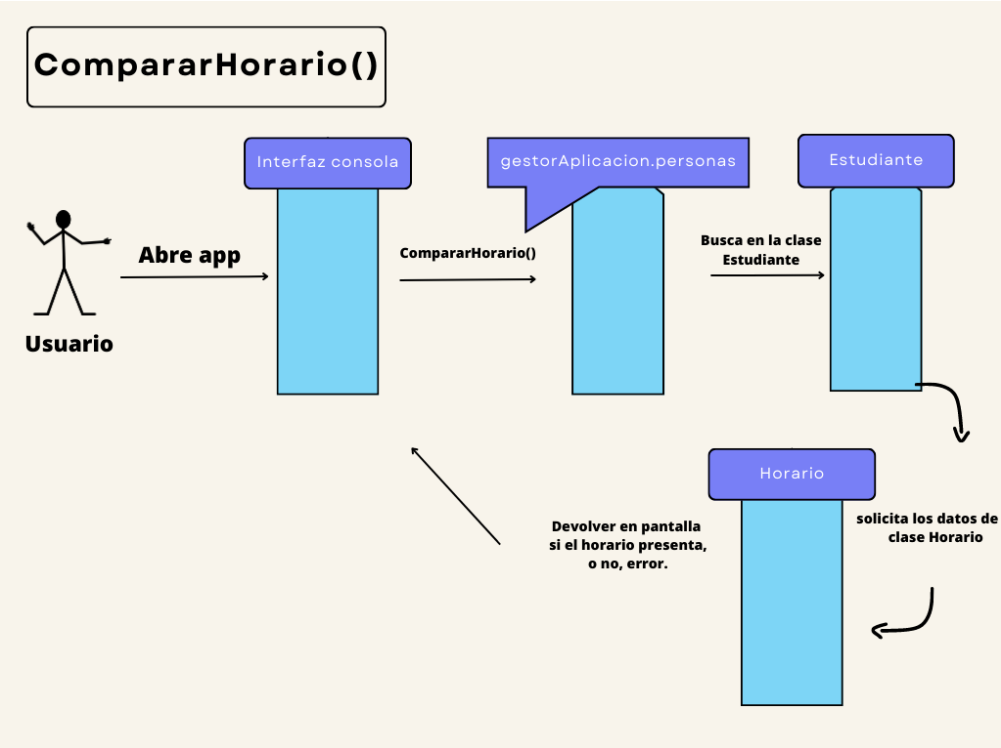
- aplicar_beca
  Se añaden todos los estudiantes de manera automática a la lista estudiantes de la clase ***Beca.py***, donde luego serán catalogados.

- calcular_promedio
  La función calcula el promedio de las calificaciones de las materias inscritas por el estudiante. Primero, se inicializa la variable final_score en 0.0 y se obtiene el número de materias inscritas. luego, para cada materia inscrita, se calcula el promedio de las calificaciones de las tareas de esa materia y se

suma al final_score. Finalmente, si hay al menos una materia inscrita, se divide final_score entre los números de materias inscritas y se redondea a dos decimales antes de devolverlo.

Funcionalidad 1: Detectar problemas con franja horaria

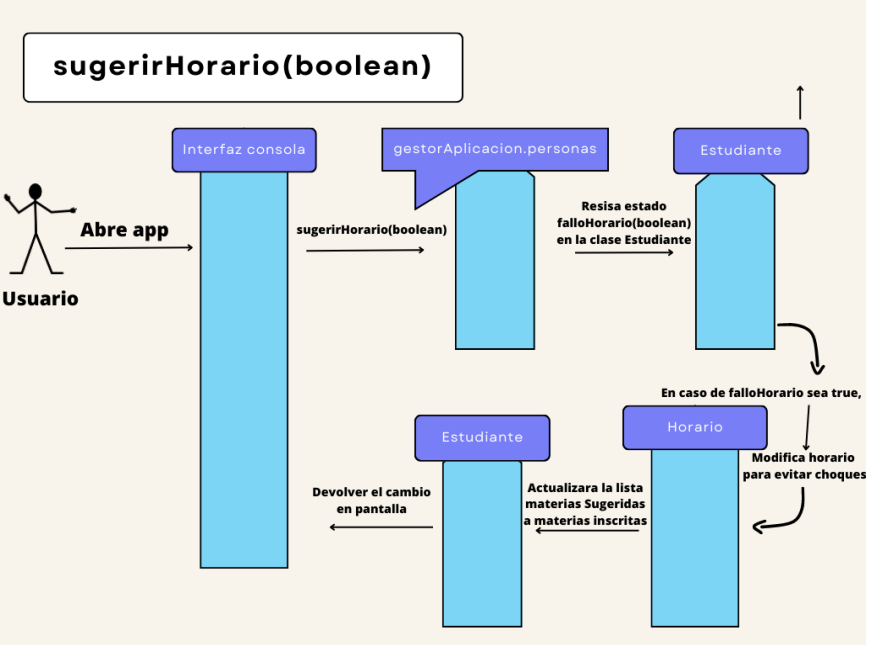
- comparar_horario(self, materias_inscritas):
  El método toma como argumento a materias_inscritas, que es la lista de tipo ***Materia.py*** inscrita por el estudiante. El método verifica si hay conflictos de horario entre las materias inscritas. primero inicializando una lista vacía llamada materias_error y se establece el atributo falla_horario en False.
  Posteriormente para cada par de materias inscritas, se obtienen sus horarios y se verifica si hay algun dia en el en el que ambas materias choquen en el horario. si hay algun dia en común y la hora de inicio o fin de una materia coincide con la hora de inicio o fin de otra materia, se establece el atributo falla_horario en True y se agregan ambas materias a la lista materias_error.
  Finalmente, el método devuelve el valor del atributo falla_horario, que indica si hay o no conflictos en el horario de las materias inscritas.



Funcionalidad 2: Sugerir Horario y materia

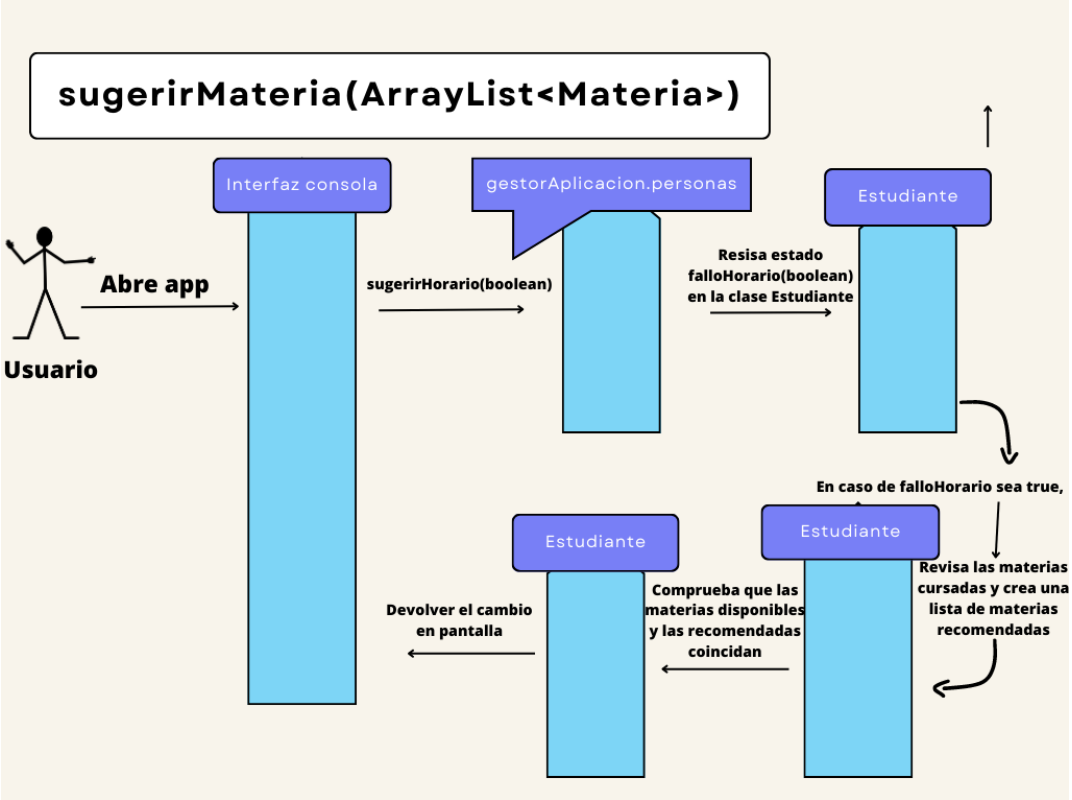
- sugerir_horario (self, falla_horario)

El método toma como argumento falla_horario, de tipo booleano, cuyo valor fue establecido en el método comparar_horario, e indica si hay conflicto en el horario con las materias inscritas. el método recomienda un horario para el estudiante que no tenga estos conflictos, siendo de esta manera, inicializando la lista vacia materias_sugeridas en primer lugar.
si falla_horario retorna true, el método verifica si cada materia si cada materia inscrita tiene conflictos de horario con las materias ya agregadas a la lista materias_sugeridas. en caso de que no haya conflicto, la materia se agrega a dicha lista.
Por último, se establece el atributo privado materias_inscritas en la lista materias_sugeridas.



- sugerir_materias (self, materias_disponibles: List[“Materia”])
  El método toma como argumento la lista de tipo ***Materia.py*** llamada materias_disponibles, las cuales almacena los objetos disponibles para la inscripción.
  El método recomienda las materias basándose en su historial académico, es decir, revisando la lista de materias_cursadas y aparte las materias disponibles. primero se inicializa una lista vacia materias_recomendadas. posterior a esto el método recorre la lista de materias disponibles y verifica si son de tipo fundamentación o disciplinar. En caso de que lo sean, se verifica si el estudiante ya ha cursado esta materia y ya cumple con el prerrequisito. y en caso de que no haya cursado la materia y se cumpla el prerrequisito, se verifica si hay conflictos con el horario con las materias ya agregadas a la lista materias_recomendadas. si no hay conflictos, la materia se agrega a la lista materias_recomendadas y se inscribe al estudiante.

En caso de que no haya materias de tipo fundamentación o disciplinar disponibles o si el estudiante no pueda tomarlas, el método considera materias de libre elección y sigue el proceso anterior para verificar si el estudiante puede tomarlas.
finalmente, se establece el atributo privado materias_inscritas en la lista materias_recomendadas.



- calcular_porcentaje_avance
  Este método lo que hará es calcular el porcentaje de avance, dividiendo el tamaño de la lista materias_cursadas por 9 y multiplicando el resultado por 1.0, luego redondea el resultado y lo multiplica por 100, finalmente actualiza este valor en la variable de PorcentajeDeAvance.

La clase "***Profesor.py"*** posee los atributos:

- **super** .__init__(nombre, ID, email)
  Estos son los atributos que la clase hereda de su padre persona
- **Public** List calificaciones_docente
  Es la lista en donde se almacenan las calificaciones de los docentes realizadas por los estudiantes.
- **Private** List materias_asignadas
  Es la materia que se le asigna a cada profesor, así como su horario.

- calificacion_docente
  Se almacena el valor arrojado después de la cuarta funcionalidad Evaluación docente.

La clase ***"Profesor.py"*** posee los siguientes métodos:

- asignar_materia()
  Se le asigna una materia a un profesor y se almacena en materias_asignadas.

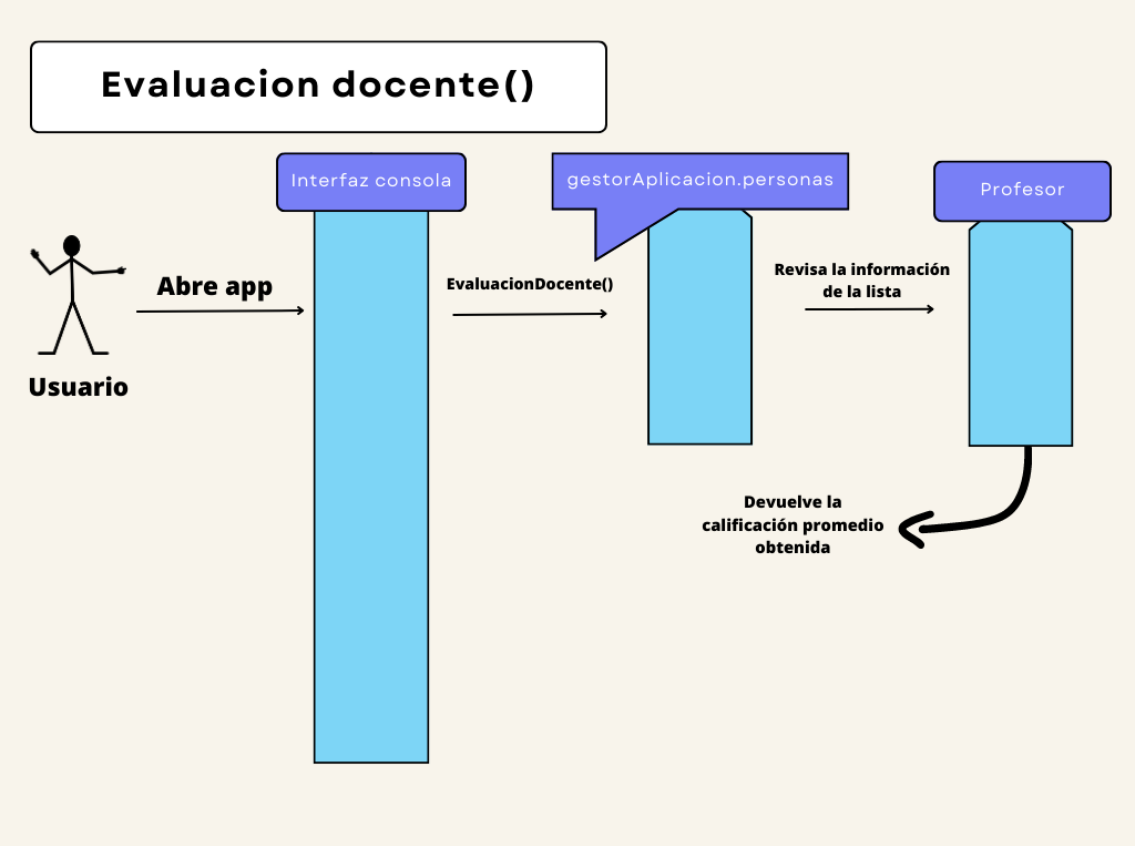
- retirar_materia()
  Se le retira una materia asignada al profesor.

- ingresar_calificacion()
  Se asigna la calificación del docente realizada en la cuarta funcionalidad Evaluación docente.

- retirar_calificacion()
  Se le retira una calificación ya asignada a profesor, de la cuarta funcionalidad Evaluación docente.

<u>Funcionalidad 4: Sugerir Horario y materia</u>

- evaluacion_docente
  Primero se inicializa una variable llamada total_calificaciones en 0. Luego se utiliza el ciclo for para iterar las calificaciones en calificaciones_docente y suma cada una en la variable total_calificaiones. Después se calcula el promedio dividiendo el total de calificaciones por la cantidad de calificaciones y se redondea al décimo más cercano. finalmente el resultado se almacena en calificacion_docente.



El modulo ***"Facultad.py"*** contiene la clase facultad que contiene los atributos:

- ***Private*** __nombre
  Se asigna como: Minas.

- ***Private*** carrera
  Se asigna como: IngenieriaSistema.

- **Public** Materias
  Esta lista contiene todas las materias que se usarán dentro de la aplicación.

- ***Private*** __profesores
  Esta lista contiene los profesores que se usarán dentro de la aplicación.

La clase ***"Facultad.py"*** posee el siguiente metodo para asignar materia a los profesores:

- Empezamos con un ciclo for en el rango de la "longitud" de la lista "Materias", después de esto el parámetro profesor lo igualamos al profesor que contenga la materia en ese momento del ciclo, después tenemos un condicional el cual nos dice que si profesor no está en el atributo __ profesores, este se agregue a profesor. Por último se le asigna la materia.

El módulo ***"Beca.py"*** contiene la clase **Beca** que posee los siguiente:

- Una subclase llamada tipos que es de tipo "Enum", esta subclase contiene los 3 tipos de becas disponibles en nuestro sistema

- Un parámetro que es una lista llamada **estudiante**, lo que hace esta lista es registrar automáticamente los estudiantes registrados al sistema de becas, donde posteriormente se irán evaluando según los requisitos si aplica para alguna o para ninguna beca.

- Un atributo llamado (**Public) nombre.**

- Un atributo de tipo lista llamado (**Private) __estudiantes_aptos_inicial:** Para el primer tipo de beca, los estudiantes aptos se guardarán en esta lista, los requisitos para aplicar a esta beca son: que el estudiante tenga un porcentaje de avance mayor o igual a 20 y menor a 40, el promedio sea mayor o igual a 4.5 y que no haya sido becado anteriormente.

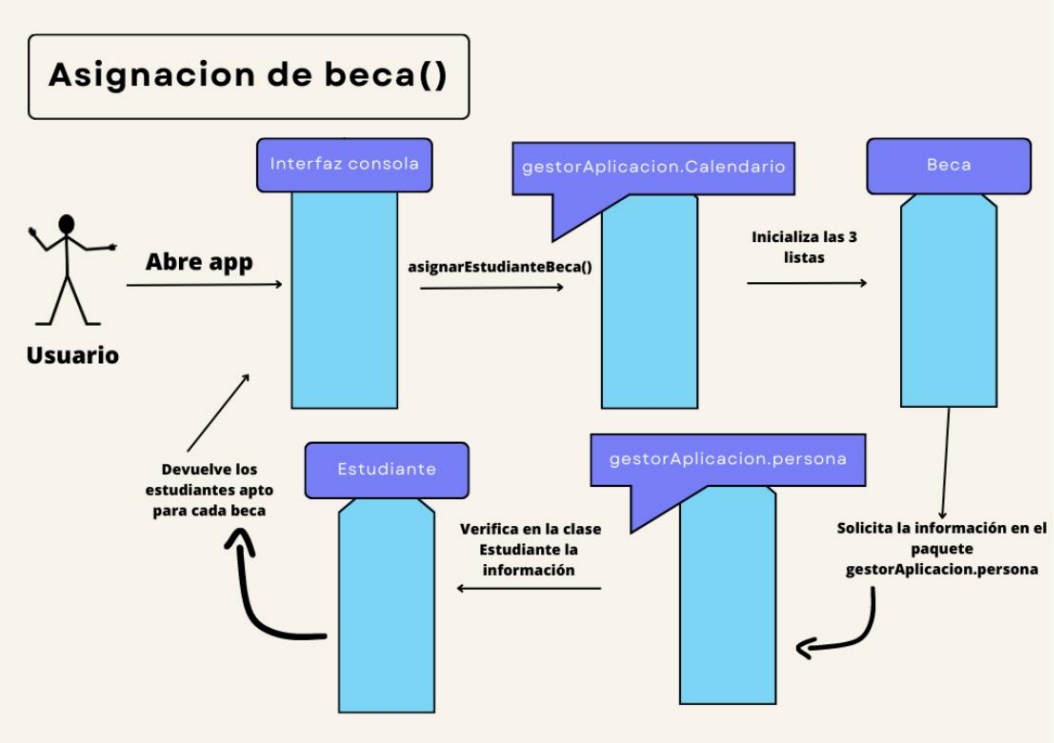
- Un atributo de tipo lista llamado (**Private) __estudiantes_aptos_normal:** Para el segundo tipo de beca, los estudiantes aptos se guardarán en esta lista, los requisitos para aplicar a esta beca son: que el estudiante tenga un porcentaje de avance mayor o igual a 40 y menor a 60, el promedio sea mayor o igual a 4.0 y que no haya sido becado anteriormente.
- Un atributo de tipo lista llamado (**Private) __estudiantes_aptos_avanzada:** Para el tercer tipo de beca, los estudiantes aptos se guardarán en esta lista, los requisitos para aplicar a esta beca son: que el estudiante tenga un porcentaje de avance mayor o igual a 60, el promedio sea mayor o igual a 3.5 y que no haya sido becado anteriormente.

La clase ***"Beca.java"*** contiene el siguiente método:

<u>Funcionalidad 3: Asignación de becas:</u>

- **def asignar_estudiantes_beca(self):**
  En esta funcionalidad interactuan las clases "Beca.py" y la clase "Estudiante.py" Primero se inicializan las 3 listas de estudiantes aptos estudiantes_Aptos_Inicial2,estudiantes_Aptos_Normal2, estudiantes_Aptos_Avanzada2 para cada una de las becas, se utiliza el ciclo for para iterar sobre la lista estudiantes, la cual contiene a todos los estudiantes y así poder catalogarlos según corresponda la beca a la que puede aplicar, verificando si cumple con los requisitos mencionados anteriormente para ser elegible, si el estudiante cumple con los requisitos, se agrega a la lista correspondiente de estudiantes aptos para beca. Después del ciclo for, el método ordena cada una de las listas de estudiantes aptos para becas en orden descendente según su promedio calculado en el método de "Estudiante.py" calcular_Promedio(), después se verifica si cada lista tiene

al menos dos estudiantes, y en caso de ser un solo estudiante se agrega a la lista correspondiente**.**



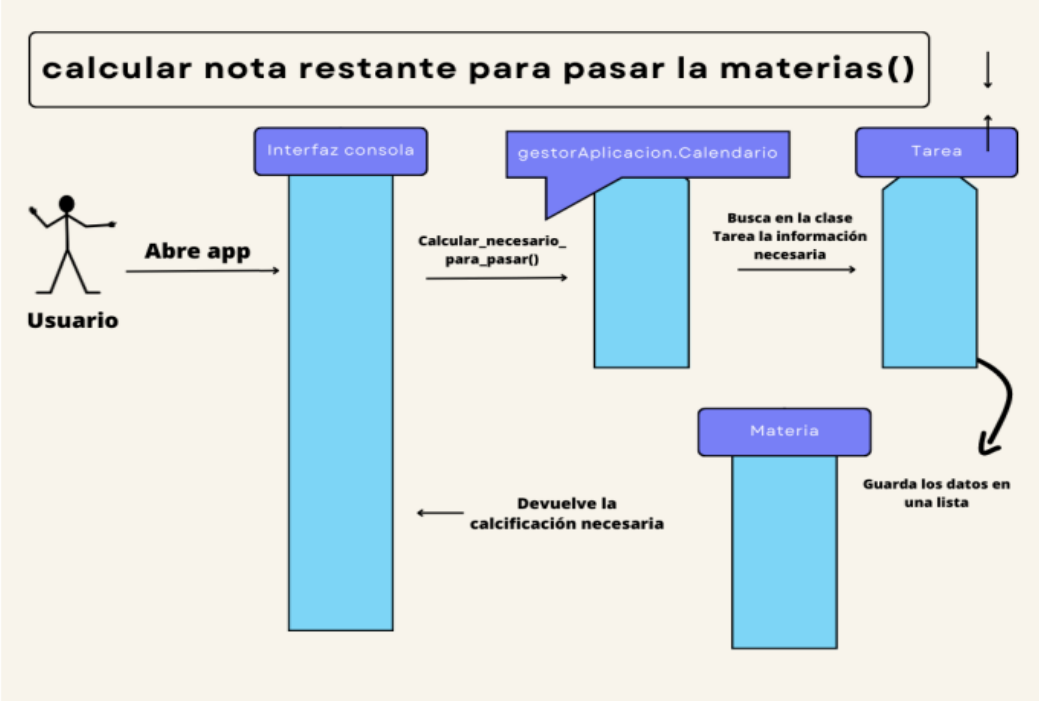
El modulo ***"Materia.py"*** contiene a la clase materia la cual posee los siguientes metodos:

- (***Public***) **inscribir_estudiante(self, nuevo_estudiante)**
- (***Public***) **retirar_estudiante(self,estudiante)**
- (***Public***) **inscribir_tarea(self,tarea)**
- (***Public***) **retirar_tarea(self,tarea)**
- (***Public***) **calcular_promedio(self,estudiante):**
  El método toma como parámetro un objeto estudiante y empieza a iterar con un ciclo for para empezar a iterar sobre __tareas_de_materia. Dentro del ciclo, se utiliza el método get_Grade de tarea para obtener la calificación del estudiante en esa tarea y la suma a una variable llamada total_Score. después del ciclo for, el método calcula el promedio dividiendo el total de calificaciones por el número de tareas y redondea el resultado

Funcionalidad 5: Calcular nota restante para pasar la materia.

- Calcular_necesario_para_pasar(self.Estudiante). Esta funcionalidad relaciona la clase "Materia.py" con la clase "Estudiante.py", este método toma como parámetro un objeto estudiante y usa un ciclo for para iterar sobre tareas_de_materia que es de tipo tareas. Dentro del ciclo el método utiliza

get_Grade() de tarea para obtener la calificación del estudiante en esa tarea y la suma a la variable total_Score. También se lleva un contador de cuántas tareas hay. posterior al ciclo for, el método calcula el promedio dividiendo el total de calificaciones por el número de tareas y redondea el resultado. Luego se verifica si el promedio es menor a 3, en caso de serlo, calcula la nota necesaria para pasar y este resultado se lo asigna a la variable llamada Nota_necesaria y devuelve este valor como resultado. Si el promedio es mayor o igual a 3, el método devuelve 0 como resultado.



Explicación interfaz gráfica.

Ahora que todas las funcionalidades complejas y contexto de algunas de las clases fueron explicadas anteriormente, ahora pasaremos con el tema de la interfaz gráfica y cómo funciona.

Pantalla Inicio

Al ejecutar el programa lo primero que vemos es una pantalla que nos da la bienvenida, arriba a la izquierda si interactuamos con el encuadre mediante un click pasan las “Hojas de vida” de cada uno de los desarrolladores programa y en la parte inferior izquierda tendremos el botón “ingresar” con el cual le daremos inicio al programa.

Ventana principal

Después de haberle dado al botón “ingresar” en la pestaña anterior, nos enviará a una ventana principal, en la cual tendremos la opción de iniciar sesión con un ID ya creado o registrarse. En caso de que el ID no esté registrado el programa nos arroja un error. la cual contiene la información básica, como el nombre, el ID y el correo electrónico (datos los cuales son heredados de la clase persona), el promedio actual del estudiante y el porcentaje de avance que es calculado en base a las materias puestas como cursadas en la sección de registro.

Ventana de registro

Al haberle dado click al botón “registrarse” nos enviará a una pestaña en la cual debemos ingresar Nombre, ID, Email, clarificar si fue becado anteriormente y además marcar las materias que ya haya visto anteriormente. Posterior a eso debemos darle registro para poder guardar el usuario correspondiente.

Menú principal estudiante.

Si en vez de haberle dado click “registrarse” iniciaste sesión con tu ID en la pestaña anterior, este te enviará a una nueva pestaña en la cual a primera vista tendrás tú información personal que agregaste en el registro. la cual contiene la información básica, como el nombre, el ID y el correo electrónico (datos los cuales son heredados de la clase persona), el promedio actual del estudiante y el porcentaje de avance que es calculado en base a las materias puestas como cursadas en la sección de registro. en un menú desplegable en la parte superior izquierda, llamado deseo, podremos ver diferentes opciones entre las que se tiene:

- inscribir asignatura
- ver materias inscritas
- realizar calificación docente
- ver horario
- materias cursadas anteriormente
- ver calificación docente
- visualizar becados
- volver al menú principal

Inscribir asignaturas.

En esta pestaña podrás inscribir asignaturas, teniendo en cuenta las materias vistas anteriormente y los prerrequisitos para materias más avanzadas.
las materias disponibles en el programa son:

- Cálculo diferencial (4 créditos)
- Cálculo integral (4 créditos)
- Calculo varias variables (4 créditos)
- Fundamentos de programación (3 créditos)
- Programación orientada a objetos (3 créditos)
- Estructura de datos (3 créditos)
- Cátedra antioquia (3 créditos)
- Cátedra Apun (3 créditos)
- Cátedra felicidad (3 créditos)

Se debe tener en cuenta que ciertas materias funcionan como prerrequisitos de otras y esto se evalúa en el momento donde el usuario registra las materias ya vistas, en caso de que el usuario quiera registrar una materia sin haber cursado un prerrequisito, saldrá error.
Las materias que funcionan como prerrequisito son: Cálculo diferencial como prerrequisito de cálculo integral, e integrado funciona como prerrequisito de cálculo en varias variables, estas tres materias hacen parte de la categoría obligatorio. Fundamentos de programación es prerrequisito de programación orientada a objetos y esta es prerrequisito de Estructura de datos, estas tres materias hacen parte de la categoría fundamentación. Por último está la categoría de libre elección, las cuales no son y no necesitan requisitos previos.
finalmente se exige que el horario registrado por el estudiante posea al menos 10 créditos y una materia de tipo obligatorio y fundamentación.

Ver materias Inscritas

En esta pestaña podrás visualizar tus asignaturas inscritas, en donde se despliega la lista que guarda las materias inscritas, que es usada en la primera funcionalidad de la clase estudiante. a su vez también se aplica la segunda funcionalidad, en donde se presente el caso de que el estudiante no haya inscrito materias aun, o las que haya inscrito hayan presentado choques en el horario, el sistema le recomendará un horario.

Realizar calificación docente

En esta pestaña podrás realizar la evaluación docente, tendrás que acceder a la materia, cuando se acceda a esta ventana, se mostrará todos los profesores que el estudiante tenga inscritos, en base a las materias seleccionadas anteriormente y en base a esto podrás calificarlo, es una calificación de 1 a 5, accediendo en el botón donde este el nombre del profesor, y accediendo a la ventana donde se tendrá el campo para calificarlo.

Ver horario

En esta pestaña se podrá ver el horario inscrito por el estudiante, donde se mostrará los días, las horas de inicio y la hora de finalización de las clases de cada materia.

Materias cursadas anteriormente

En esta pestaña están las materias que has cursado anteriormente, dichas materias se registraran con anterioridad en la sección de registro del menú, donde el usuario señala que materias ya ha cursado, en caso de que no se haya cursado ninguna asignatura no

aparecerá nada. se debe tener en cuenta el sistema de los prerrequisitos y que materias habilitan otras materias al ser señaladas como cursadas.

Ver calificación docente

En esta pestaña podrás observar la calificación que tienen los docentes, de esta manera podrás hacerte una idea de que docente es “recomendado” y cual no.

Visualizar becados

En esta pestaña se podrán ver los nombres de los estudiantes que son aptos para becas, siendo estos, 3 tipos de becas dependiendo de ciertos requerimientos, por ejemplo, el primer tipo de beca, llamada beca inicial, es apta para los estudiantes con un porcentaje de avance entre el 20 y el 40 de la carrera, y con un promedio de 4.5 en adelante. El segundo tipo de beca, llamada beca normal, es apta para los estudiantes con un porcentaje de avance entre el 40 y el 60, y con un promedio de 4.0 en adelante. Por último, el tercer tipo de beca, llamada beca avanzada, es apta para los estudiantes con un porcentaje de avance entre el 60 y el 100, y con un promedio mayor a 3.5.

Volver al menú principal

Con esta opción puedes volver al menú principal, la cual contiene la información básica, como el nombre, el ID y el correo electrónico (datos los cuales son heredados de la clase persona)