Universidad ORT Uruguay
Facultad de Ingeniería

Primer obligatorio

Entregado como requisito para la materia Diseño de Aplicaciones 1

Federico Cetraro - 193221

Sebastián Ramallo – 207425

Tutor: Bruno Canepa – Ricardo Szyfer

2017

Declaración de autoría

Nosotros, Federico Cetraro y Sebastián Ramallo, declaramos que el trabajo que se presenta en esa obra es de nuestra propia mano. Podemos asegurar que:

- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad;
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra;
- En la obra, hemos acusado recibo de las ayudas recibidas;
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros;
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Abstract

En el siguiente documento se presentan los puntos requeridos por el obligatorio de Diseño de Aplicaciones 1, conteniendo la descripción general del trabajo, el análisis de requerimientos, una justificación del diseño implementado, los diagramas de clases y modelo conceptual del problema, casos de uso, informe sobre prácticas aplicadas de Clean Code y evidencia de las mismas, casos de pruebas del software y finalmente el resultado de la ejecución de las pruebas unitarias con su respectiva evidencia de cobertura y TDD.

Palabras clave

Clean Code, TDD, Diseño de Aplicaciones.

Índice

1. Introducción

A continuación se presenta la documentación de los puntos del obligatorio de Diseño de Aplicaciones 1 con sus justificaciones correspondientes y los requerimientos solicitados en la letra del mismo.

2. Cuerpo de la obra

1.1.Descripción general del trabajo

El trabajo realizado cuenta con todas las funcionalidades solicitadas en los requerimientos, incluyendo la Gestión de Materias (Alta, Baja y Modificación), Gestión de alumnos (Alta, Baja, Modificación y Consulta), gestión de Docentes (Alta, Baja, Modificación y Consulta) y Gestión de Camionetas (Alta, Baja, Modificación, Consulta y Calculo de Rutas).

Sobre el Cálculo de rutas, el software se encarga de dividir a los alumnos en las camionetas que haya en el sistema de forma equitativa, asignando en primer lugar a las camionetas de mayor capacidad. Este algoritmo proveerá las distintas rutas que hará cada camioneta desde la escuela.

En cuanto a la interfaz, la misma no fue autogenerada por temas de tiempo, aunque se logró cubrir los requerimientos mencionados anteriormente siendo conscientes de que haberlo hecho de esta manera aumenta el impacto de cambio a la hora de agregar, modificar o eliminar un módulo en el sistema.

1.2.Análisis de requerimientos

Requerimientos Funcionales

RF1 – Alta de Materia

Descripción: El sistema permite dar de alta una materia. Requiere de: Codigo de materia, Nombre y los alumnos que participan en ella.

Prioridad: Alta

RF2 – Modificar Materia

Descripción: El sistema permite modificar los datos de una materia ya registrada en el sistema.

Prioridad: Alta

RF3 – Baja Materia

Descripción: El sistema permite dar de baja una materia ya registrada en el sistema.

Prioridad: Alta

RF4 – Alta de Alumno

Descripción: El sistema permite dar de alta un estudiante. Requiere de: Numero de estudiante, Cédula de identidad, Nombre, Apellido y las materias que está inscripto.

Prioridad: Alta

RF5 – Modificar Alumno

Descripción: El sistema permite modificar los datos de un alumno ya registrado en el sistema.

Prioridad: Alta

RF6 – Baja Alumno

Descripción: El sistema permite dar de baja un alumno ya registrado en el sistema.

Prioridad: Alta

RF7 – Listar Alumno

Descripción: El sistema permite mostrar un listado de los alumnos, pudiendo visualizar de cada uno: Nombre, Apellido, Número de estudiante y las materias que está inscripto.

Prioridad: Alta

RF8 – Alta Docente

Descripción: El sistema permite dar de alta un Docente. Requiere de: Nombre, Apellido y las materias que dicta.

Prioridad: Alta

RF9 – Modificar Docente

Descripción: El sistema permite modificar los datos de un Docente ya registrado en el sistema.

Prioridad: Alta

RF10 – Baja Docente

Descripción: El sistema permite dar de baja un Docente ya registrado en el sistema.

Prioridad: Alta

RF11 – Listar Docentes

Descripción: El sistema permite mostrar un listado de los Docentes, pudiendo visualizar de cada uno: Nombre, Apellido, y las materias que dicta.

Prioridad: Alta

RF12 – Alta Camioneta

Descripción: El sistema permite dar de alta una Camioneta. Requiere de: Id de camioneta y Capacidad.

Prioridad: Alta

RF13 – Modificar Camioneta

Descripción: El sistema permite modificar los datos de una Camioneta ya registrada en el sistema.

Prioridad: Alta

RF14 – Baja Camioneta

Descripción: El sistema permite dar de baja una Camioneta ya registrado en el sistema.

Prioridad: Alta

RF15 – Ver Camionetas

Descripción: El sistema permite ver cuantas camionetas hay a disposición junto con su capacidad.

Prioridad: Alta

RF16 – Calcular Ruta

Descripción: El sistema permite calcular las rutas que cada camioneta debe hacer para completar su capacidad recorriendo la menor distancia posible.

Prioridad: Alta

Requerimientos No Funcionales

RNF1 – Sistema Operativo

Descripción: El sistema tiene que funcionar correctamente en Windows 7 o superior.

RNF2 – Lenguaje de desarrollo

Descripción: El sistema debe ser desarrollado en el lenguaje C# en Microsoft Visual Studio .NET 2015.

RNF3 – Clean Code

Descripción: El sistema debe cumplir con los lineamientos de Clean Code, utilizando técnicas y metodologías agiles en el diseño.

RNF4 – Usabilidad

Descripción: Debe ser simple de usar, intuitiva y atractiva.

RNF5 – Generar datos de prueba

Descripción: El sistema debe permitir generar datos de prueba de manera automática

1.3.Justificación de diseño

- ***Separado por Capas***

Para nuestra Solución utilizamos el diseño "Separado por Capas" (Layers). Elegimos este diseño ya que obteníamos los siguientes beneficios:

- Reutilización de niveles,
- Soporte para la estandarización.
- Los cambios en el Código son locales a cada nivel.

También como ventaja se puede destacar que, si se agrega una nueva funcionalidad a cualquier entidad del dominio, el impacto de cambio se reduce a dichos cambios y no necesariamente afectara a toda la solución.

En cambio, si se agrega un Módulo nuevo a la solución hay que modificar el dominio, lo cual impactara en toda la solución (salvo en la UI) ya que todos los paquetes dependen del dominio de las entidades obligando a la recopilación de varios paquetes de la solución.

- ***Singleton***

Varias clases precisan referenciar a un mismo elemento, queremos asegurarnos de que no hay más de una instancia de ese elemento. El patrón de diseño Singleton nos garantiza que una clase sólo tenga una instancia y proporciona un punto de acceso global a ella.

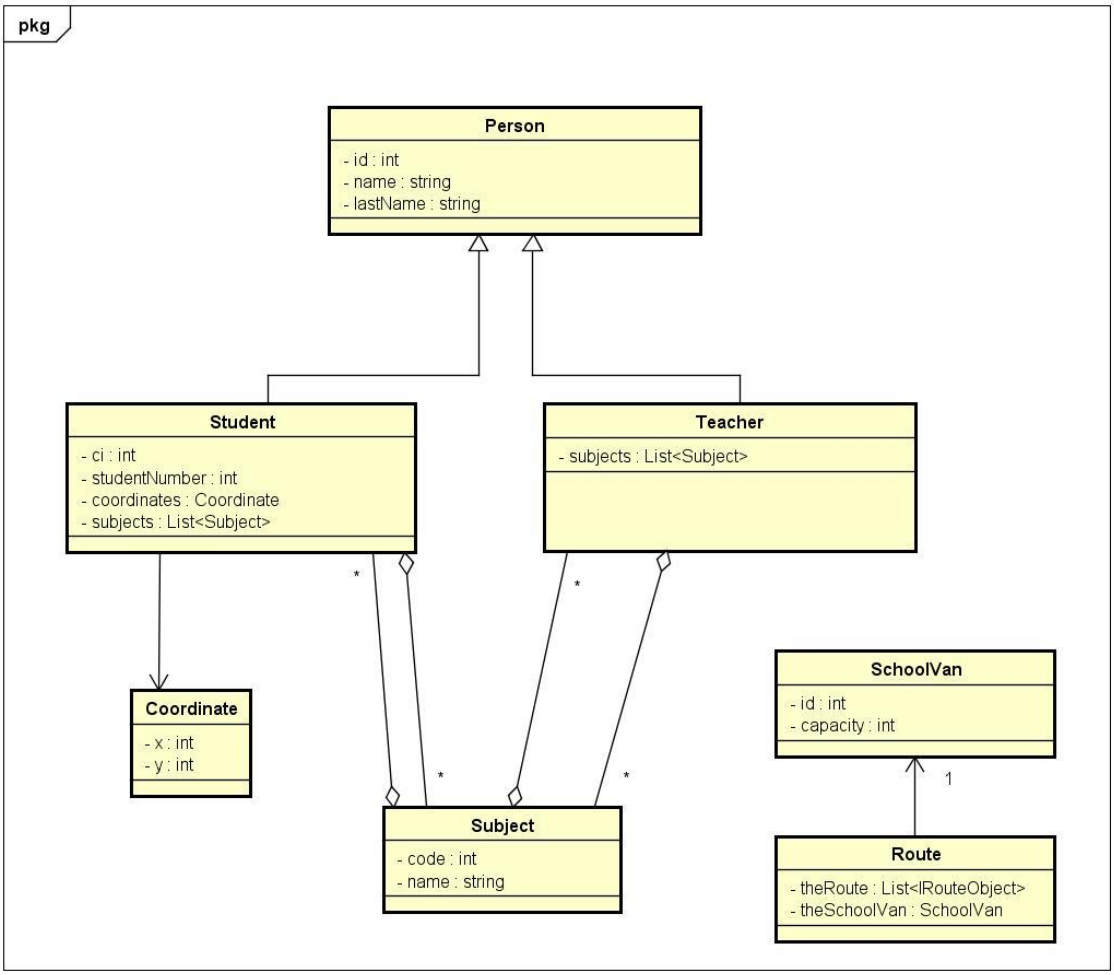
Utilizamos dicho patrón para el almacenamiento de nuestros datos, generando una única instancia cada vez que se utiliza el repositorio en la solución. Esto fue realizado en el namespace "Repository".

- ***Facade***

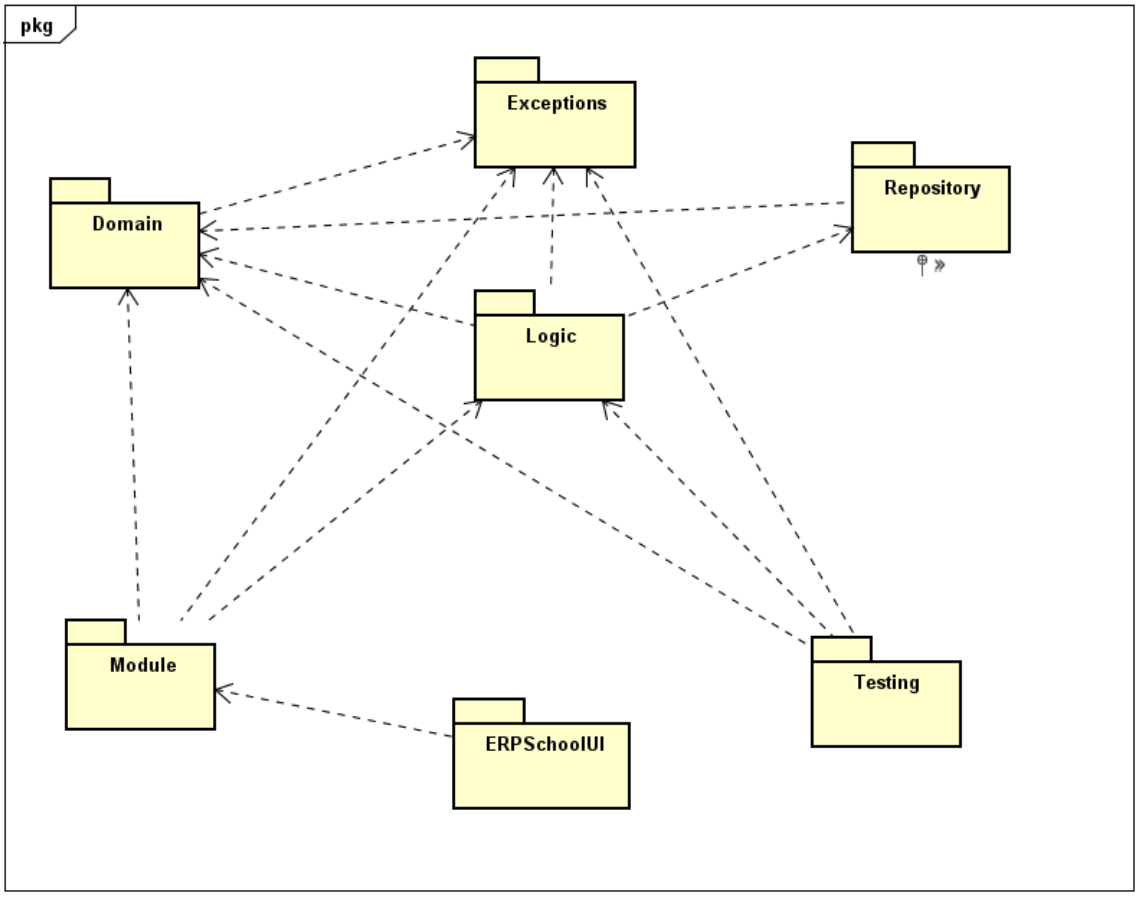
Facade es un tipo de patrón de diseño estructural. Es implementado por la necesidad de estructurar un entorno de programación y reducir su complejidad con la división en subsistemas, minimizando las comunicaciones y dependencias entre estos.

Utilizamos el patrón de diseño eliminar las dependencias entre la UI y la Lógica de nuestra solución.

1.4.Modelo Conceptual

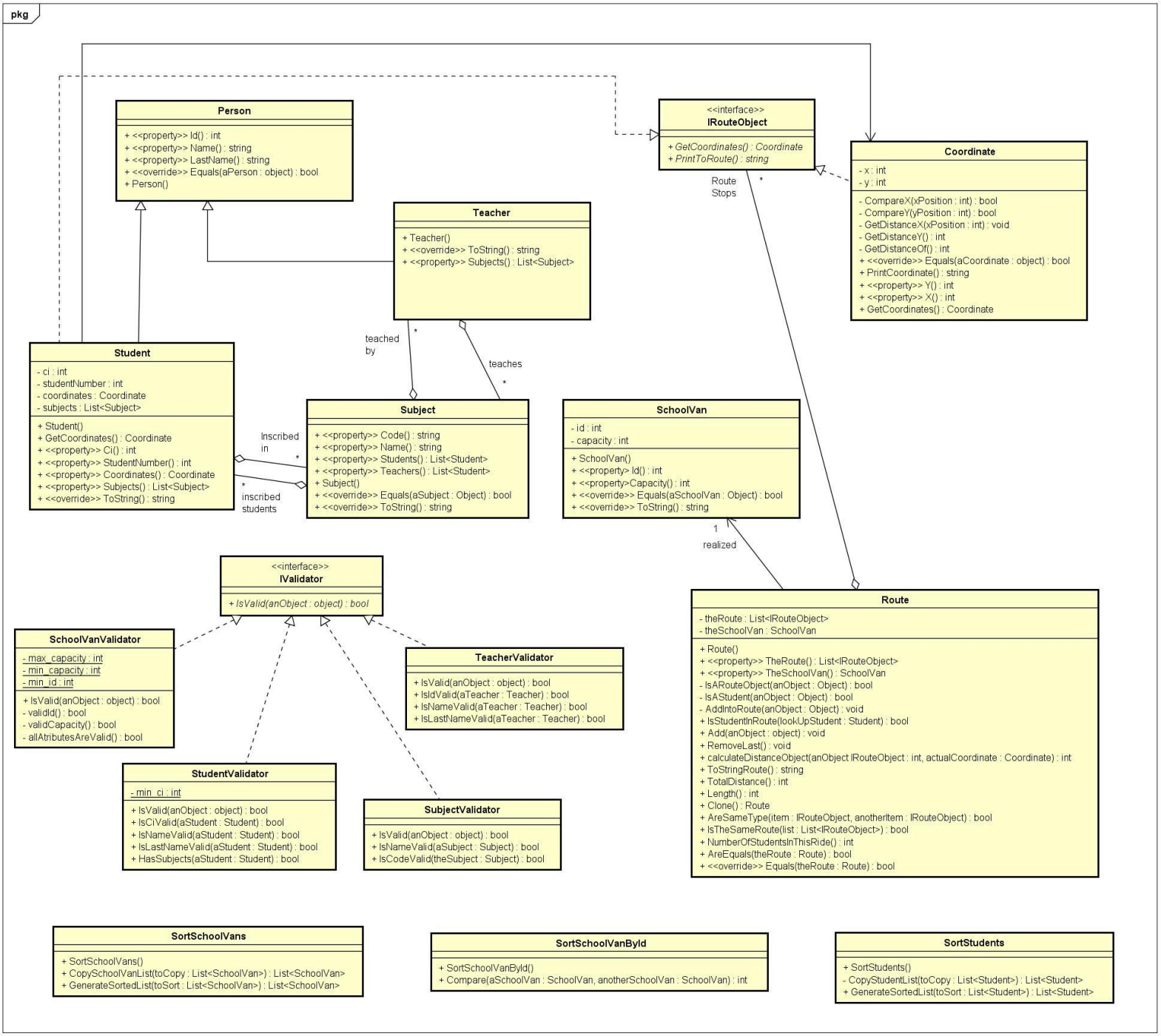


1.5.Diagrama de paquetes

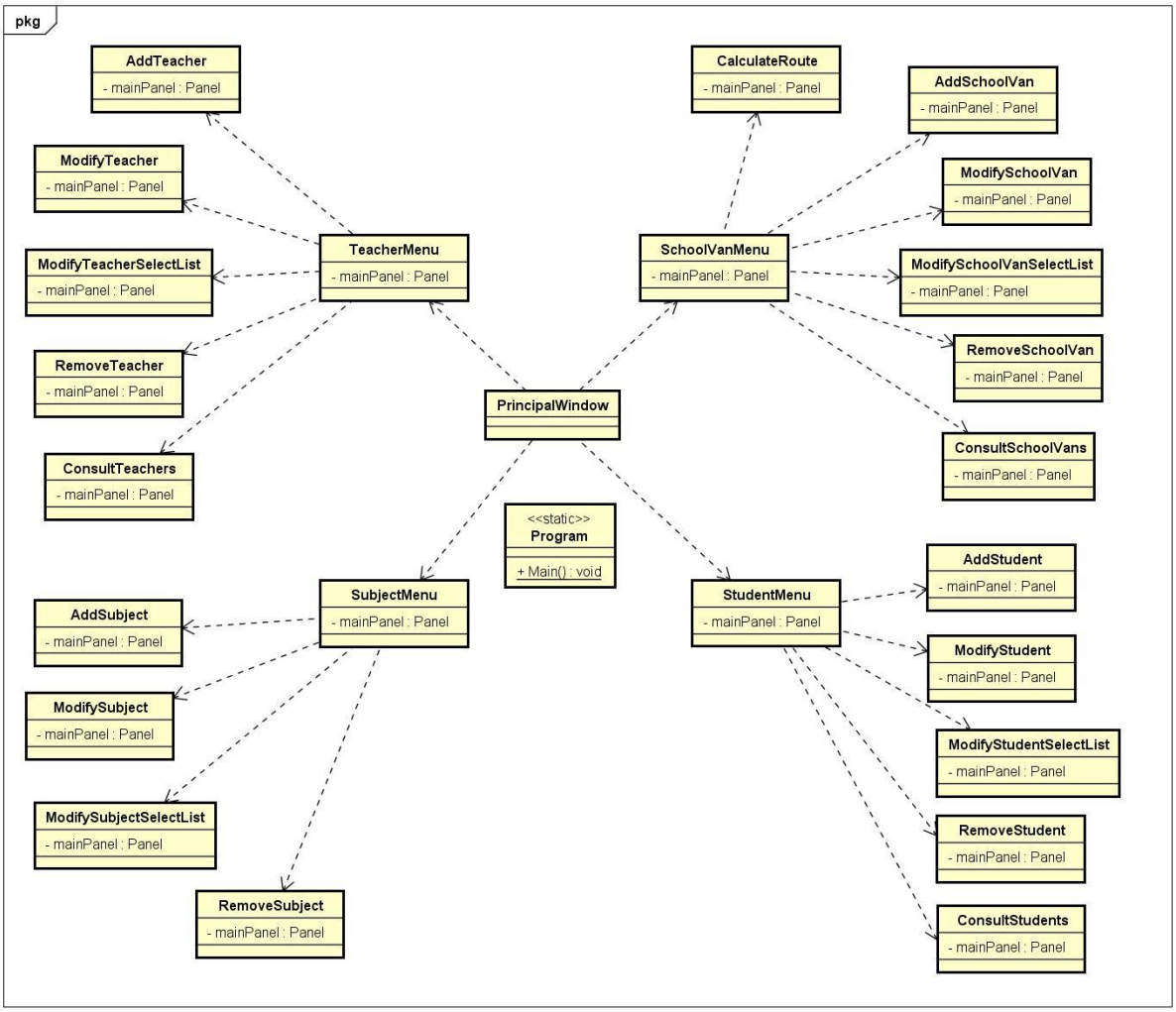


1.6.Diagramas de clases

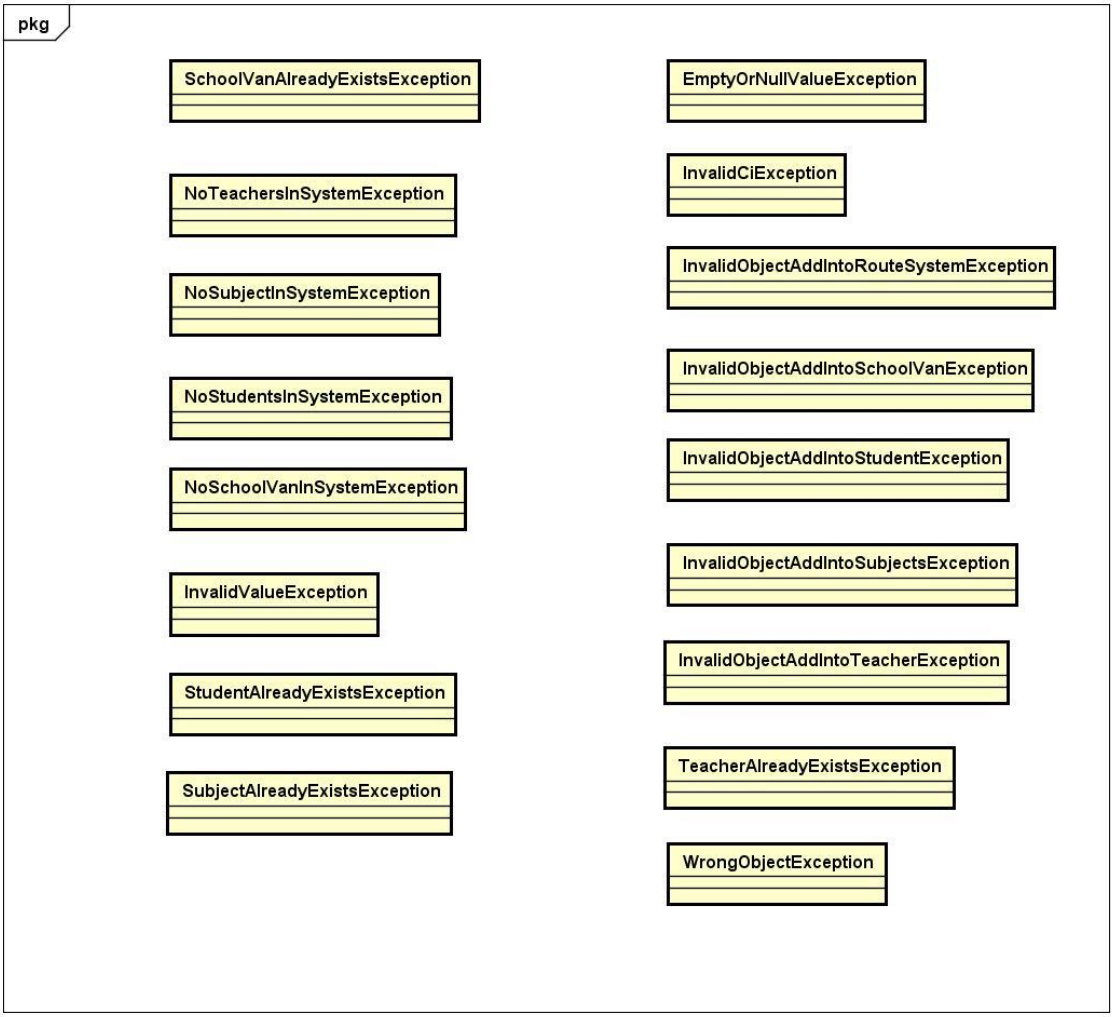
1.6.1. Dominio



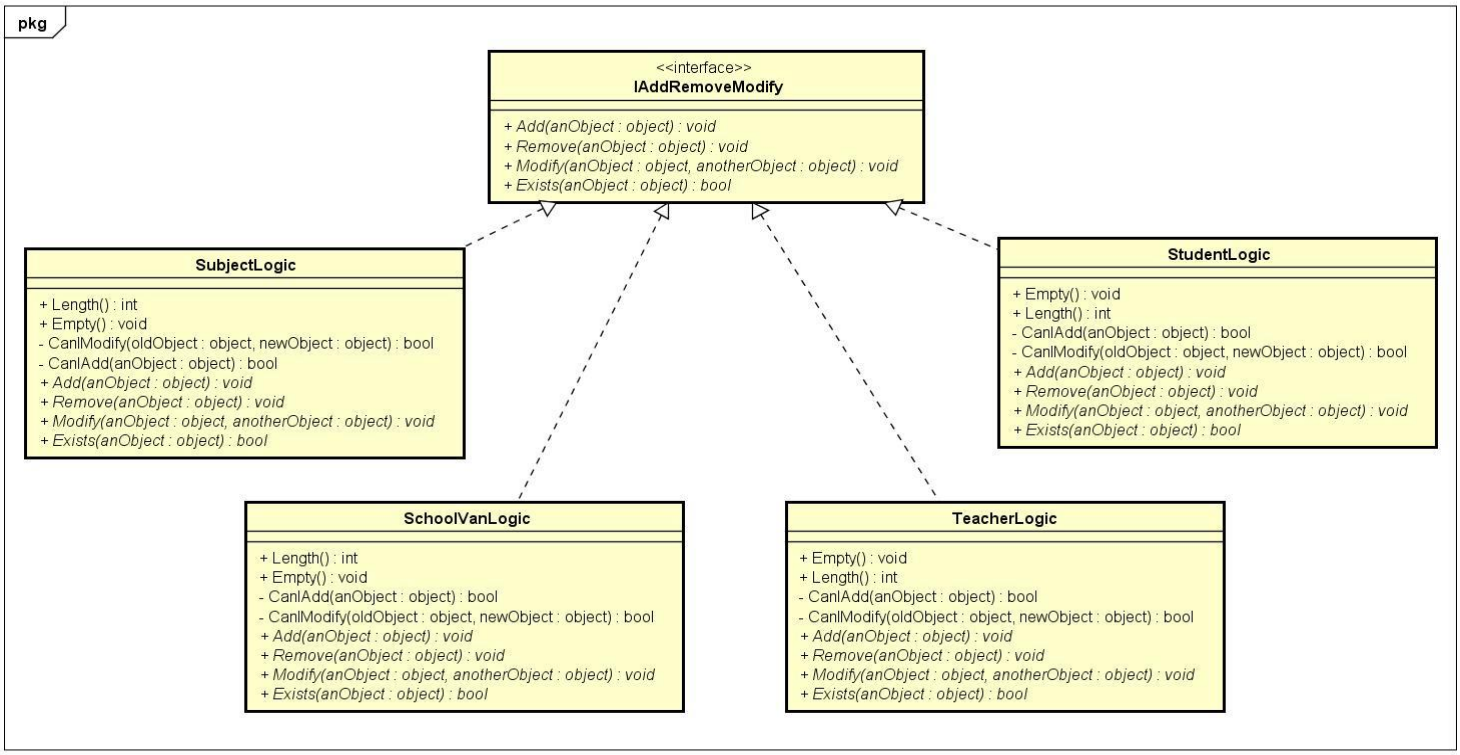
1.6.2. ERPSchoolUI



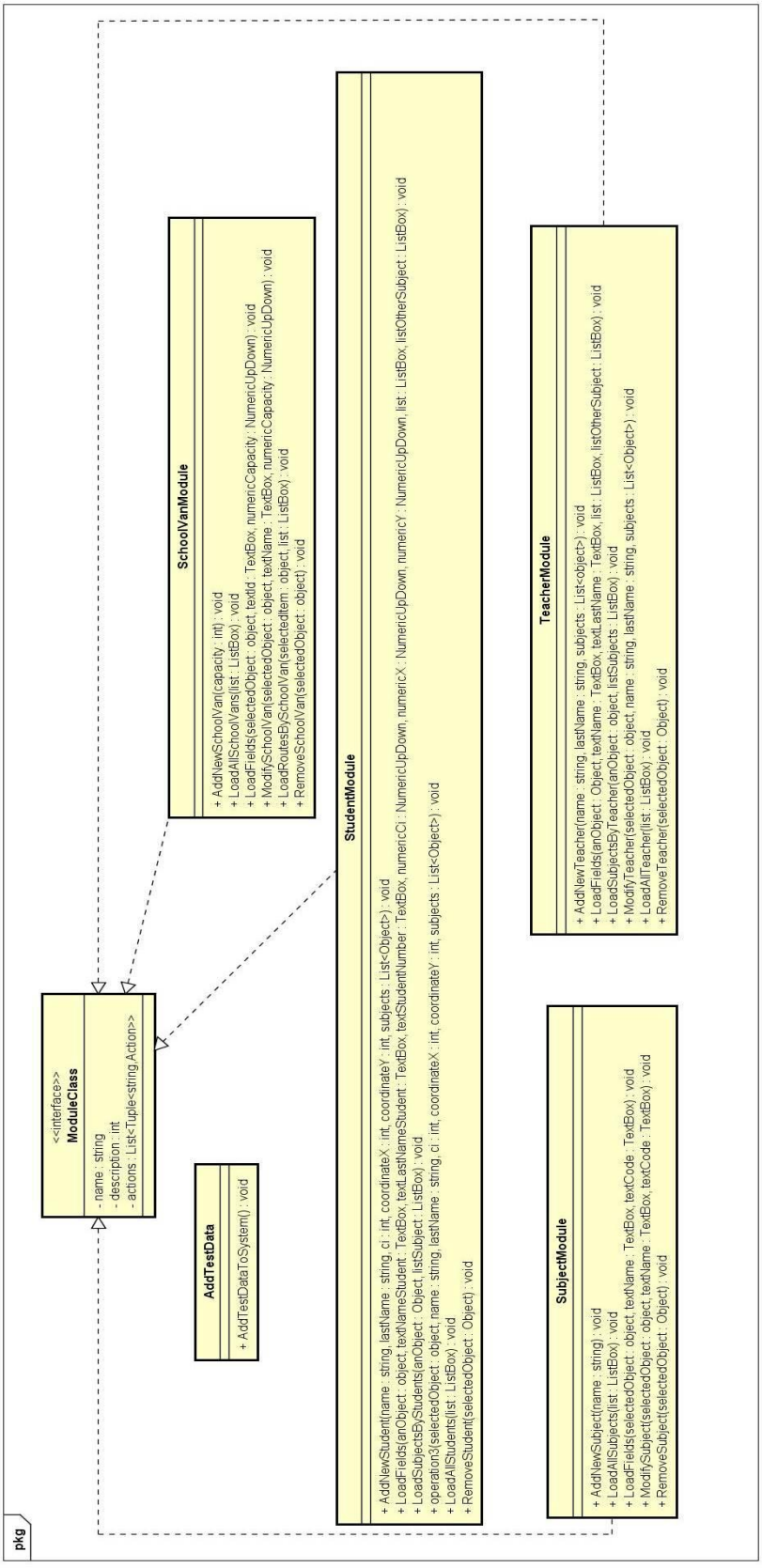
1.6.3. Exceptions



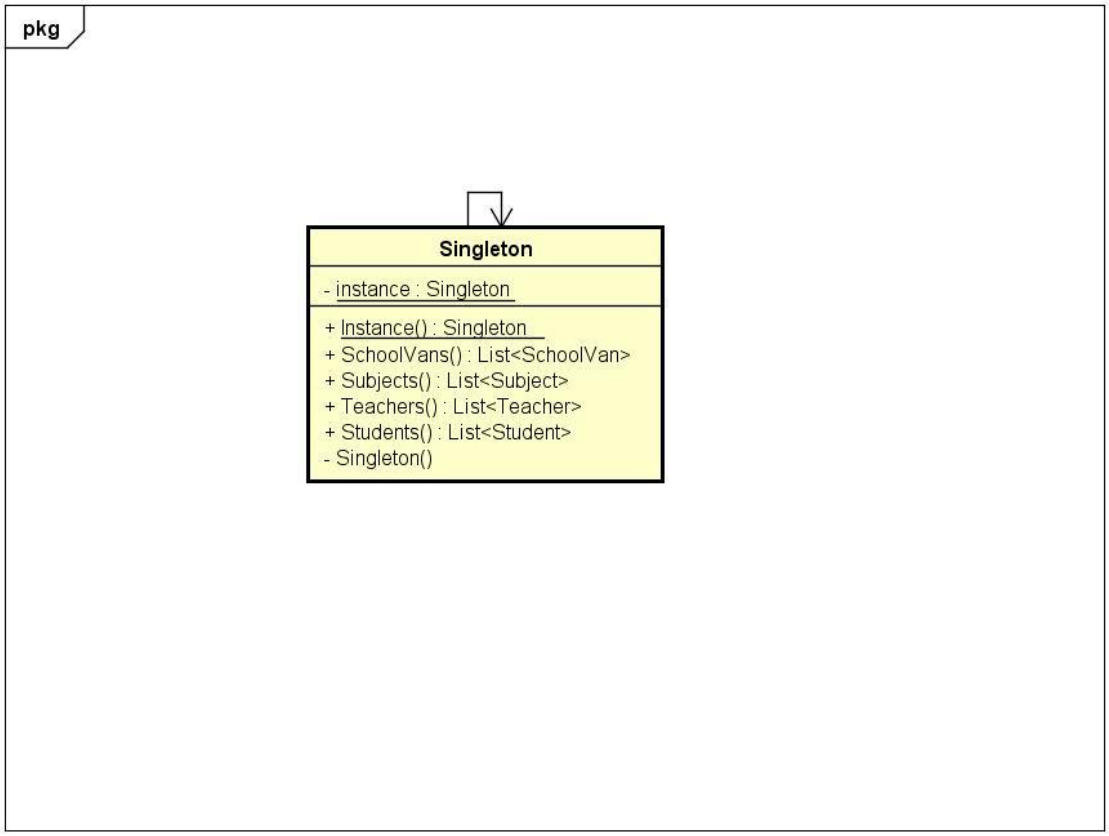
1.6.4. Logic


pkg
<<interface>>
IAddRemoveModify
+ Add(anObject : object) : void
+ Remove(anObject : object) : void
+ Modify(anObject : object, anotherObject : object) : void
+ Exists(anObject : object) : bool
SubjectLogic
+ Length() : int
+ Empty() : void
- CanIModify(oldObject : object, newObject : object) : bool
- CanIAdd(anObject : object) : bool
+ Add(anObject : object) : void
+ Remove(anObject : object) : void
+ Modify(anObject : object, anotherObject : object) : void
+ Exists(anObject : object) : bool
StudentLogic
+ Empty() : void
+ Length() : int
- CanIAdd(anObject : object) : bool
- CanIModify(oldObject : object, newObject : object) : bool
+ Add(anObject : object) : void
+ Remove(anObject : object) : void
+ Modify(anObject : object, anotherObject : object) : void
+ Exists(anObject : object) : bool
SchoolVanLogic
+ Length() : int
+ Empty() : void
- CanIAdd(anObject : object) : bool
- CanIModify(oldObject : object, newObject : object) : bool
+ Add(anObject : object) : void
+ Remove(anObject : object) : void
+ Modify(anObject : object, anotherObject : object) : void
+ Exists(anObject : object) : bool
TeacherLogic
+ Empty() : void
+ Length() : int
- CanIAdd(anObject : object) : bool
- CanIModify(oldObject : object, newObject : object) : bool
+ Add(anObject : object) : void
+ Remove(anObject : object) : void
+ Modify(anObject : object, anotherObject : object) : void
+ Exists(anObject : object) : bool

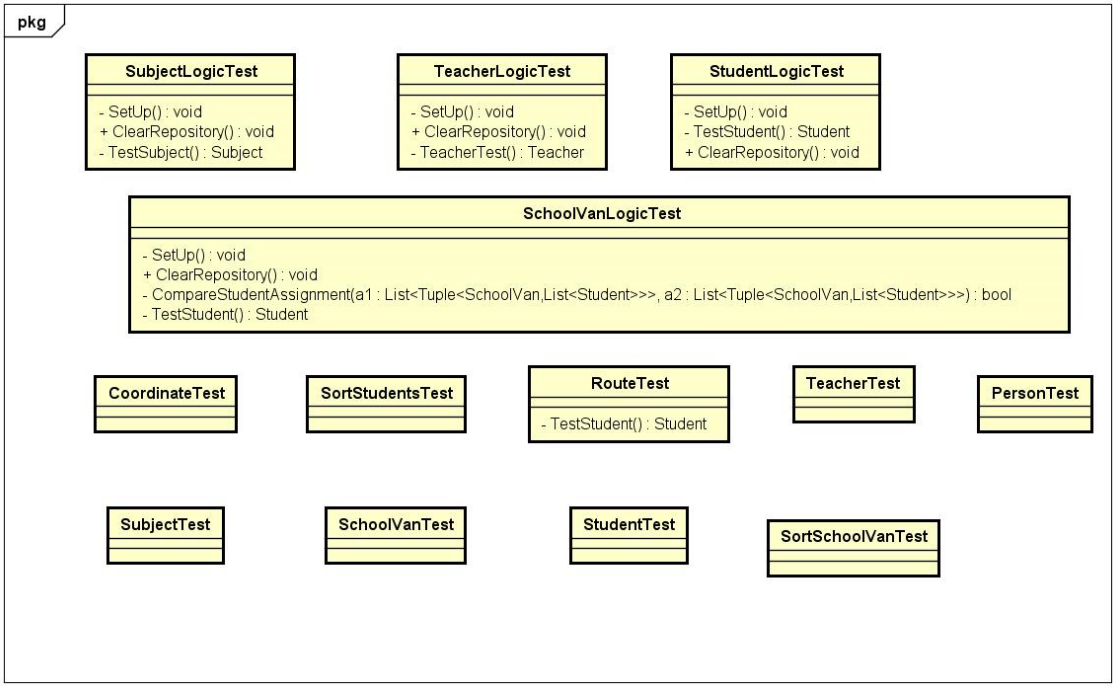
1.6.5. Module


pkg
<<interface>>
ModuleClass
- name : string
- description : int
- actions : List<Tuple<string,Action>>
AddTestData
+ AddTestDataToSystem() : void
SchoolVanModule
+ AddNewSchoolVan(capacity : int) : void
+ LoadAllSchoolVans(list : ListBox) : void
+ LoadFields(selectedObject : object, textId : TextBox, numericCapacity : NumericUpDown) : void
+ ModifySchoolVan(selectedObject : object, textName : TextBox, numericCapacity : NumericUpDown) : void
+ LoadRoutesBySchoolVan(selectedItem : object, list : ListBox) : void
+ RemoveSchoolVan(selectedObject : object) : void
StudentModule
+ AddNewStudent(name : string, lastName : string, ci : int, coordinateX : int, coordinateY : int, subjects : List<Object>) : void
+ LoadFields(anObject : object, textNameStudent : TextBox, textLastNameStudent : TextBox, textStudentNumber : TextBox, numericCi : NumericUpDown, numericX : NumericUpDown, numericY : NumericUpDown, list : ListBox, listOtherSubject : ListBox) : void
+ LoadSubjectsByStudents(anObject : Object, listSubject : ListBox) : void
+ operation3(selectedObject : object, name : string, lastName : string, ci : int, coordinateX : int, coordinateY : int, subjects : List<Object>) : void
+ LoadAllStudents(list : ListBox) : void
+ RemoveStudent(selectedObject : Object) : void
TeacherModule
+ AddNewTeacher(name : string, lastName : string, subjects : List<object>) : void
+ LoadFields(anObject : Object, textName : TextBox, textLastName : TextBox, list : ListBox, listOtherSubject : ListBox) : void
+ LoadSubjectsByTeacher(anObject : object, listSubjects : ListBox) : void
+ ModifyTeacher(selectedObject : object, name : string, lastName : string, subjects : List<Object>) : void
+ LoadAllTeacher(list : ListBox) : void
+ RemoveTeacher(selectedObject : Object) : void
SubjectModule
+ AddNewSubject(name : string) : void
+ LoadAllSubjects(list : ListBox) : void
+ LoadFields(selectedObject : object, textName : TextBox, textCode : TextBox) : void
+ ModifySubject(selectedObject : object, textName : TextBox, textCode : TextBox) : void
+ RemoveSubject(selectedObject : Object) : void

1.6.6. Repository



1.6.7. Testing



1.7.Casos de uso

1.7.1. Diagrama

1.7.2. Especificación

Caso de uso 1: Alta de Materia

<table>
<tr><td colspan="2">ID: CU1</td></tr>
<tr><td colspan="2">Nombre: Alta Materia</td></tr>
<tr><td colspan="2">Descripción: Se quiere dar de alta una nueva materia</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición:</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador ingresa el nombre de la materia.
2- El administrador presiona el botón Guardar.</td><td></td></tr>
<tr><td></td><td>3- El sistema da de alta la materia con un Código de Materia generado automáticamente y mostrado en la interfaz.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">1.1- El administrador deja vacía la casilla del nombre
El sistema muestra un mensaje de error

1.2- El administrador ingresa un nombre de materia ya existente
El sistema muestra un mensaje de error.</td></tr>
<tr><td colspan="2">Post-condición: La materia es dada de alta en el sistema.</td></tr>
</table>

Caso de uso 2: Modificar Materia

<table>
<tr><td colspan="2">ID: CU2</td></tr>
<tr><td colspan="2">Nombre: Modificar Materia</td></tr>
<tr><td colspan="2">Descripción: Se quiere modificar los datos de una materia ya ingresada en el sistema.</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición: Existe al menos una materia para modificar en el sistema.</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador selecciona una materia y presiona el botón Continuar.
2- El administrador visualiza el nombre de la materia seleccionada pudiendo modificar el mismo.
3- El administrador presiona el botón Guardar.</td><td></td></tr>
<tr><td></td><td>4- El sistema modifica la materia seleccionada existente en el sistema.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">2.1- El administrador deja vacía la casilla del nombre
El sistema muestra un mensaje de error

2.2- El administrador ingresa un nombre de materia ya existente
El sistema muestra un mensaje de error.</td></tr>
<tr><td colspan="2">Post-condición: Es modificada la materia en el sistema.</td></tr>
</table>

Caso de uso 3: Baja Materia

<table>
<tr><td colspan="2">ID: CU3</td></tr>
<tr><td colspan="2">Nombre: Baja Materia</td></tr>
<tr><td colspan="2">Descripción: Se quiere dar de baja una materia del sistema.</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición: Existe al menos una materia para dar de baja en el sistema.</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador selecciona las materias que desea dar de baja de una lista de todas las materias.
2- El administrador presiona el botón Confirmar.</td><td></td></tr>
<tr><td></td><td>3- El sistema da de baja las materias que fueron seleccionadas en el sistema.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">1.1- El usuario no selecciono ninguna materia para dar de baja del sistema
Se muestra un mensaje de error</td></tr>
<tr><td colspan="2">Post-condición: Se da de baja la materia del sistema</td></tr>
</table>

Caso de uso 4: Alta Alumno

<table>
<tr><td colspan="2">ID: CU4</td></tr>
<tr><td colspan="2">Nombre: Alta Alumno</td></tr>
<tr><td colspan="2">Descripción: Se quiere dar de alta un nuevo alumno en el sistema</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición:</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador ingresa la Cedula de Identidad, Nombre y Apellido del estudiante y selecciona de una lista de todas las materias aquellas a las que está inscripto.
2- El administrador presiona el botón Guardar.</td><td></td></tr>
<tr><td></td><td>3- El sistema da de alta al nuevo alumno en el sistema junto con un Número de estudiante generado automáticamente y mostrado en la interfaz.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">1.1- El administrador deja vacía la casilla de la Cedula de identidad
El sistema muestra un mensaje de error
1.2- El administrador ingresa una Cedula de Identidad ya existente en el sistema.
El sistema muestra un mensaje de error diciendo que no pueden haber dos cedulas repetidas.
1.3- El administrador deja vacía la casilla del Nombre y Apellido
El sistema muestra un mensaje de error</td></tr>
<tr><td colspan="2">Post-condición: Se da de alta un nuevo alumno</td></tr>
</table>

Caso de uso 5: Modificar Alumno

<table>
<tr><td colspan="2">ID: CU5</td></tr>
<tr><td colspan="2">Nombre: Modificar Alumno</td></tr>
<tr><td colspan="2">Descripción: Se quiere modificar los datos de un alumno</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición: Existe un alumno ingresado en el sistema</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador selecciona un alumno y presiona el botón Continuar.
2- El administrador visualiza la cedula de identidad, el nombre y apellido del alumno y las materias que está inscripto seleccionado pudiendo modificar el mismo.
3- El administrador presiona el botón Guardar.</td><td></td></tr>
<tr><td></td><td>4- Se modifican los datos del alumno en el sistema.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">2.1- El administrador deja vacía la casilla de la Cedula de identidad
El sistema muestra un mensaje de error
2.2- El administrador ingresa una Cedula de Identidad ya existente en el sistema.
El sistema muestra un mensaje de error diciendo que no pueden haber dos cedulas repetidas.
2.3- El administrador deja vacía la casilla del Nombre y Apellido
El sistema muestra un mensaje de error</td></tr>
</table>

Post-condición: Se modifican los datos del alumno seleccionado en el sistema

Caso de uso 6: Baja Alumno

ID: CU6	
Nombre: Baja Alumno	
Descripción: Se quiere dar de baja un alumno en el sistema	
Actores: Administrador	
Pre-condición: Existe al menos un alumno en el sistema	
Curso normal:	
Administrador	**Sistema**
1- El administrador selecciona los alumnos de una lista de todos los alumnos del sistema para darlos de baja. 2- El administrador presiona el botón Confirmar.	
	3- El sistema da de baja a los alumnos seleccionados.
Curso Alternativo:	
1.1- El usuario no selecciono ningún alumno para dar de baja del sistema Se muestra un mensaje de error	
Post-condición: Se dan de baja los alumnos seleccionados en el sistema.	

Caso de uso 7: Listado de Alumnos

<table>
<tr><td colspan="2">ID: CU7</td></tr>
<tr><td colspan="2">Nombre: Listado Alumnos</td></tr>
<tr><td colspan="2">Descripción: Se muestra un listado de todos los alumnos registrados en el sistema.</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición:</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El usuario ingresa a la opción para ver el listado de alumnos.
2- El usuario puede seleccionar un alumno del listado para ver la lista de materias que está inscripto</td><td></td></tr>
<tr><td></td><td>3- El sistema muestra la lista de alumnos de todo el sistema y al seleccionar uno de ellos despliega la lista que muestra las materias a las que está inscripto.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td colspan="2">Post-condición: Se muestra la lista de alumnos del sistema</td></tr>
</table>

Caso de uso 8: Alta Docente

<table>
<tr><td colspan="2">ID: CU8</td></tr>
<tr><td colspan="2">Nombre: Alta Docente</td></tr>
<tr><td colspan="2">Descripción: Se quiere dar de alta un docente en el sistema.</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición:</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador ingresa la, Nombre y Apellido del docente y selecciona de una lista de todas las materias aquellas las que dicho docente dicta.
2- El administrador presiona el botón Guardar.</td><td></td></tr>
<tr><td></td><td>3- El sistema da de alta al nuevo docente en el sistema junto con un Id de Docente generado automáticamente y mostrado en la interfaz.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">1.1- El administrador deja vacía la casilla del Nombre y Apellido
El sistema muestra un mensaje de error</td></tr>
<tr><td colspan="2">Post-condición: Se da de alta al docente en el sistema.</td></tr>
</table>

Caso de uso 9: Modificar Docente

<table>
<tr><td colspan="2">ID: CU9</td></tr>
<tr><td colspan="2">Nombre: Modificar Docente</td></tr>
<tr><td colspan="2">Descripción: Se quiere modificar los datos de un docente en el sistema.</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición: Existe al menos un docente registrado en el sistema.</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador selecciona un docente y presiona el botón Continuar.
2- El administrador visualiza el nombre y apellido del docente y las materias que dicta pudiendo modificar estos datos.
3- El administrador presiona el botón Guardar.</td><td></td></tr>
<tr><td></td><td>4- Se modifican los datos del docente en el sistema.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">2.1- El administrador deja vacía la casilla del Nombre y Apellido
El sistema muestra un mensaje de error</td></tr>
<tr><td colspan="2">Post-condición: Se modifica al</td></tr>
</table>

Caso de uso 10: Baja Docente

<table>
<tr><td colspan="2">ID: CU10</td></tr>
<tr><td colspan="2">Nombre: Baja Docente</td></tr>
<tr><td colspan="2">Descripción: Se desea dar de baja a un docente del sistema.</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición: Existe al menos un docente registrado en el sistema.</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador selecciona los docentes de una lista de todos los docentes del sistema para darlos de baja.
2- El administrador presiona el botón Confirmar.</td><td></td></tr>
<tr><td></td><td>3- El sistema da de baja a los docentes seleccionados.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">1.1- El usuario no selecciono ningún docente para dar de baja del sistema
Se muestra un mensaje de error</td></tr>
<tr><td colspan="2">Post-condición: Se da de baja al docente en el sistema.</td></tr>
</table>

Caso de uso 11: Listado de Docentes

<table>
<tr><td colspan="2">ID: CU11</td></tr>
<tr><td colspan="2">Nombre: Listado Docentes</td></tr>
<tr><td colspan="2">Descripción: Se muestra un listado de todos los docentes disponibles en el sistema y que materias dicta cada uno de ellos</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición:</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador ingresa a la opción para ver el listado de docentes.
2- El administrador puede seleccionar un docente del listado para ver la lista de materias que dicta</td><td></td></tr>
<tr><td></td><td>3- El sistema muestra la lista de docentes de todo el sistema y al seleccionar uno de ellos despliega la lista que muestra las materias que dicta</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td colspan="2">Post-condición: Se muestra el listado de docentes del sistema.</td></tr>
</table>

Caso de uso 12: Alta Camioneta

<table>
<tr><td colspan="2">ID: CU12</td></tr>
<tr><td colspan="2">Nombre: Alta Camioneta</td></tr>
<tr><td colspan="2">Descripción: Se quiere dar de alta una nueva camioneta en el sistema</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición:</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador ingresa la capacidad máxima de la camioneta
2- El administrador presiona el botón Guardar.</td><td></td></tr>
<tr><td></td><td>3- El sistema da de alta la camioneta con un Id de Camioneta generado automáticamente y mostrado en la interfaz.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td colspan="2">Post-condición: Se da de alta la camioneta en el sistema</td></tr>
</table>

Caso de uso 13: Modificar Camioneta

<table>
<tr><td colspan="2">ID: CU13</td></tr>
<tr><td colspan="2">Nombre: Modificar Camioneta</td></tr>
<tr><td colspan="2">Descripción: Se quiere modificar una camioneta en el sistema.</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición: Existe al menos una camioneta registrada en el sistema.</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador selecciona una camioneta y presiona el botón Continuar.
2- El administrador visualiza la capacidad de la camioneta pudiendo modificar este dato.
3- El administrador presiona el botón Guardar.</td><td></td></tr>
<tr><td></td><td>4- Se modifican los datos de la camioneta en el sistema.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2"></td></tr>
<tr><td colspan="2">Post-condición: Se modifica la camioneta en el sistema</td></tr>
</table>

Caso de uso 14: Baja Camioneta

<table>
<tr><td colspan="2">ID: CU14</td></tr>
<tr><td colspan="2">Nombre: Baja Camioneta</td></tr>
<tr><td colspan="2">Descripción:</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición: Existe al menos una camioneta registrada en el sistema</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El administrador selecciona las camionetas de una lista de todas las camionetas del sistema para darlos de baja.
2- El administrador presiona el botón Confirmar.</td><td></td></tr>
<tr><td></td><td>3- El sistema da de baja a las camionetas seleccionadas.</td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">1.1- El usuario no selecciono ninguna camioneta para dar de baja del sistema
Se muestra un mensaje de error</td></tr>
<tr><td colspan="2">Post-condición: Se da de baja la camioneta en el sistema</td></tr>
</table>

Caso de uso 15: Ver camionetas

ID: CU15	
Nombre: Ver Camionetas	
Descripción: Se muestra un listado de todas las camionetas del sistema	
Actores: Administrador	
Pre-condición:	
Curso normal:	
Administrador	**Sistema**
1- El administrador ingresa a la opción para visualizar las camionetas del sistema y se muestra un listado de todas las camionetas a disposición.	
Curso Alternativo:	
Post-condición: Se muestra el listado de camionetas del sistema	

Caso de uso 16: Calcular Rutas

<table>
<tr><td colspan="2">ID: CU16</td></tr>
<tr><td colspan="2">Nombre: Calcular Rutas</td></tr>
<tr><td colspan="2">Descripción: Se muestran las mejores rutas para cada camioneta registrada en el sistema</td></tr>
<tr><td colspan="2">Actores: Administrador</td></tr>
<tr><td colspan="2">Pre-condición: Existe al menos una camioneta registrada en el sistema</td></tr>
<tr><td colspan="2">Curso normal:</td></tr>
<tr><td>Administrador</td><td>Sistema</td></tr>
<tr><td>1- El usuario ingresa a la opción de calcular rutas y selecciona una camioneta de la lista de camionetas
2- Se muestra en la lista de rutas a realizar, las mejores rutas (que impliquen la menor distancia posible) para la camioneta seleccionada</td><td></td></tr>
<tr><td></td><td></td></tr>
<tr><td colspan="2">Curso Alternativo:</td></tr>
<tr><td colspan="2">2.1- No se seleccionó ninguna camioneta
No se muestra ninguna ruta en la lista de rutas a realizar</td></tr>
<tr><td colspan="2">Post-condición: Se muestra la lista de rutas a realizar de la camioneta seleccionada.</td></tr>
</table>

1.8.Clean Code

1.8.1. Practicas Aplicadas

Para el desarrollo de la solución, se aplicaron varias de las prácticas descritas en el libro de Clean Code que serán enumeradas a continuación:

Nombres nemotécnicos: Se buscó utilizar nombres lo más representativos posible para cada uno de los atributos, métodos, clases y paquetes.

Uso de comentarios: Se evitó totalmente el uso de comentarios, la necesidad de comentarios para aclarar algo es síntoma de que hay código mal escrito que debería ser rediseñado. Es preferible expresarse mediante el propio código.

Consistencia de nombres: Se buscó utilizar siempre la misma palabra para describir el mismo concepto a lo largo del código.

Formateo vertical: Se buscó que los métodos estén agrupados de forma lógica y que el código se pueda leer de arriba hacia abajo. Las clases se organizan primero constantes, luego variables y finalmente métodos.

SRP: Se aplicó el principio de responsabilidad única para así conseguir un mayor nivel de cohesión en las clases.

Manejo de errores: Se usaron excepciones en lugar de códigos de retorno. Las excepciones proporcionan suficiente información sobre el error y el momento en que se ha producido.

Evidencia Clean Code

Entendemos que nuestro código cumple con los estándares planteados anteriormente ya que en cada fase de “Refactor” en el proceso de TDD aplicamos dichos estándares en nuestro código sin modificar la funcionalidad del mismo.

```
public class SchoolVanValidator : IValidator
{
    private static int max_capacity = 15;
    private static int min_capacity = 1;
    private static int min_id = 0;
    private bool ValidCapacity(SchoolVan aSchoolVan)
    {
        if (!(aSchoolVan.Capacity >= min_capacity && aSchoolVan.Capacity <= max_capacity))
        {
            throw new InvalidValueException("La camioneta debe tener la capacidad entre ["+ min_capacity + "-"+ max_capacity + "]");
        }
        return true;
    }
    private bool ValidId(SchoolVan aSchoolVan)
    {
        if (aSchoolVan.Id < min_id)
        {
            throw new EmptyOrNullValueException("El id debe ser mayor a "+min_id);
        }
        return true;
    }
    private bool AllAtributesAreValid(SchoolVan theSchoolVan)
    {
        bool capacityValidation = ValidCapacity(theSchoolVan);
        bool idValidation = ValidId(theSchoolVan);
        return (capacityValidation && idValidation);
    }
```

En la imagen podemos apreciar el manejo de “Exceptions”, los nombres de métodos y de las variables revelan su intención, los métodos son cortos y realizan solo una acción.

De todas maneras, en algunos métodos no cumplimos con todos los principios de Clean Code, por ejemplo:

```
List<Student> SelectLessStudents(int schoolVansWithMoreStudents, int actualLoop, int studentsPerSchoolVanFloor)
{
    Singleton theRepository = Singleton.Instance;
    List<Student> sortedStudent = GetStudentsSortedById();
    List<Student> studentsToSchoolVan = new List<Student>();
    for (int i = actualLoop; i < actualLoop + studentsPerSchoolVanFloor; i++)
    {
        studentsToSchoolVan.Add(sortedStudent.ElementAt(i));
    }
    actualLoop = actualLoop + studentsPerSchoolVanFloor;
    return studentsToSchoolVan;
}
List<Student> SelectStudents(int schoolVansWithMoreStudents, int actualLoop, int studentsPerSchoolVanFloor)
{
    List<Student> sortedStudent = GetStudentsSortedById();
    List<Student> studentsToSchoolVan = new List<Student>();
    if (schoolVansWithMoreStudents >= actualLoop)
    {
        studentsToSchoolVan = SelectLessStudents(schoolVansWithMoreStudents, actualLoop, studentsPerSchoolVanFloor + 1);
        return studentsToSchoolVan;
    }
    studentsToSchoolVan = SelectLessStudents(schoolVansWithMoreStudents, actualLoop, studentsPerSchoolVanFloor);
    return studentsToSchoolVan;
}
```

Por la complejidad del algoritmo y el poco tiempo para resolverlo no fuimos capaces de mejorar la calidad de estos métodos siendo conscientes que pasar 3 o más objetos por parámetro podría ser un problema.

1.9.Diseño de los casos de prueba

1.10. Resultado de la ejecución de pruebas

Inicio del Sistema

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Menú de Gestión de Materias	Se presiona el botón “Gestion de Materias” del menu principal	Se muestra el panel del módulo de Gestión de Materias con los botones: “Agregar materia”,	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
		"Modificar materia" y "Eliminar Materia"	
Menú de Gestión de Alumnos	Se presiona el botón "Gestion de Alumnos" del menu principal	Se muestra el panel del módulo de Gestión de Alumnos con los botones: "Agregar alumno", "Modificar alumno" , "Eliminar alumno" y "Ver alumnos"	SI
Menú de Gestión de Docentes	Se presiona el botón "Gestion de Docentes" del menu principal	Se muestra el panel del módulo de Gestión de Docentes con los botones: "Agregar docente", "Modificar docente" , "Eliminar docente" y "Ver docentes"	SI
Menú de Gestión de Camionetas	Se presiona el botón "Gestion de Camionetas" del menu principal	Se muestra el panel del módulo de Gestión de Camionetas con los botones: "Agregar	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
		camioneta", "Modificar camioneta" , "Eliminar camioneta", "Listado camionetas" y "Calcular Rutas"	
Generación de datos de prueba	Se presiona el botón "Generar Datos de Prueba" del menu principal	Se cargan en el sistema materias, alumnos con materias asignadas, docentes con materias asignadas y camionetas.	SI

<u>Agregar Materia</u>

Situación 1:

- ***No hay ninguna materia registrada en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Alta de nueva materia satisfactoriamente	Nombre: “Diseño de Aplicaciones 1” Se presiona el botón Guardar.	Se da de alta la materia en el sistema y se muestra un mensaje: “Materia ingresada con éxito".	SI
Alta de nueva materia con la casilla del nombre vacía	Nombre: “ ” Se deja vacío el campo de Nombre y se presiona el botón “Guardar”	No se da de alta la materia en el sistema y se muestra un mensaje de error: “No se ha ingresado un nombre a la materia”	SI

Situación 2:

- ***Se generan los datos de prueba automáticos considerando que existe una materia llamada “Idioma Español”***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Alta de nueva materia con nombre repetido	Nombre: “Idioma Español”	No se da de alta la materia en el sistema y se muestra	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
	Se presiona el botón Guardar.	un mensaje de error: "Ya existe una materia con ese nombre"	

Modificar Materia

Situación 1:

- ***No hay ninguna materia registrada en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de modificar materia sin materias registradas en el sistema	Se presiona el botón "Modificar Materia"	Se muestra un mensaje informando que no hay materias en el sistema	SI

Situación 2:

- ***Se generan los datos de prueba automáticos considerando que existe una materia llamada "Idioma Español" e "Ingles".***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Modificar una materia satisfactoriamente.	Se modifica el nombre de la materia a “Lenguaje” y se presiona el botón “Guardar”.	Se modifica el nombre de la materia satisfactoriamente a “Lenguaje” y se notifica con un mensaje. El nombre “Lenguaje” ahora será visible a la hora de seleccionar una materia para modificar	SI
Modificar una materia con un nombre ya ingresado.	Se modifica la materia anterior “Lenguaje” con el nombre “Ingles” (ya ingresado en el sistema”	No se modifica la materia y se muestra mensaje de error: “Ya existe una materia con ese nombre”	SI
Modificar una materia dejando la casilla de nombre vacia	Se modifica la materia “Lenguaje” y se deja el campo vacio.	No se modifica la materia en el sistema y se muestra un mensaje de error: “No se ha ingresado un nombre a la materia”	SI

Eliminar Materia

Situación 1:

- ***No hay ninguna materia registrada en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de eliminar materia sin materias registradas en el sistema	Se presiona el botón "Eliminar Materia"	Se muestra un mensaje informando que no hay materias en el sistema	SI

Situación 2:

- ***Existe una materia en el sistema llamada "Lenguaje"***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Eliminar una materia satisfactoriamente	Se selecciona la materia "Lenguaje" de la lista de materias totales, cambiándola a la lista de materias a eliminar y se	Se da de baja la materia "Lenguaje" en el sistema y se muestra un mensaje de confirmación.	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
	presiona el botón “Confirmar”.		

<u>Agregar Alumno</u>

Situación 1:

- ***No hay ningún alumno registrado en el sistema***
- ***Existen las materias “Ingles” y “Diseño de Aplicaciones 1”***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Agregar un nuevo alumno satisfactoriamente	Se ingresan los datos del alumno: Cedula: 1234567 Nombre: “Juan” Apellido: “Perez” Coordenada X: 25 Coordenada Y: 22 Materias seleccionadas: “Ingles” Se presiona el botón “Guardar”	Se da de alta el alumno en el sistema y se muestra un mensaje: “Alumno ingresado con éxito”	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Intento agregar un nuevo alumno sin materias	Se ingresan los datos del alumno: Cedula: 7654321 Nombre: “Pedro” Apellido: “Perez” Coordenada X: 25 Coordenada Y: 22 Se presiona el botón “Guardar”	No se da de alta el alumno por no asignarle alguna materia. Se muestra un mensaje de error que informa al usuario.	SI
Agregar alumno con el campo de CI vacío	Se ingresan los datos del alumno: Cedula: <Vacío> Nombre: “Miguel” Apellido: “Perez” Coordenada X: 25 Coordenada Y: 22 Se presiona el botón “Guardar”	No se da de alta el alumno en el sistema y se muestra un mensaje de error.	SI
Agregar alumno con el campo de nombre y apellido vacío	Se ingresan los datos del alumno: Cedula: 1234567 Nombre: “ ” Apellido: “ ”	No se da de alta el alumno en el sistema y se muestra un mensaje de error	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
	Coordenada X: 25 Coordenada Y: 22 Se presiona el botón "Guardar"		

Situación 1:

- ***Se generan los datos de prueba automáticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Agregar un alumno con una CI ya existente	Se ingresan los datos del alumno: Cedula: 52848524 Nombre: "Juan" Apellido: "Perez" Coordenada X: 25 Coordenada Y: 22 Materias seleccionadas: "Ingles" Se presiona el botón "Guardar"	No se da de alta el alumno y se muestra un mensaje de error: "Ya existe un alumno con la CI ingresada"	SI

Modificar Alumno

Situación 1:

- ***No hay alumnos registrados en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de modificar alumnos sin alumnos registradas en el sistema	Se presiona el botón "Modificar Alumno"	Se muestra un mensaje informando que no hay alumnos en el sistema	SI

Situación 2:

- ***Se generan los datos de prueba automáticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Modificar alumno satisfactoriamente	Se modifican los datos del alumno con los siguientes datos: ID: 1 CI: 46702345 Nombre: Juan Apellido: Rodriguez	Se modifica el alumno correctamente a los datos ingresados. Ahora la materia solo es Geografía,	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
	Coordenada X: 24 Coordenada Y: 9 Materias: Matemática y Geografia Se modifican los datos a los siguientes: Cedula: 9999999 Nombre: “Martin” Apellido: “Perez” Coordenada X: 25 Coordenada Y: 22 Materias seleccionadas: Matematica Se presiona el botón “Guardar”	removiendo “Matemática”	
Modificar alumno con CI existente	Se selecciona los datos del alumno “Martin Perez” con CI: 99999999 Y se modifican a la CI: 52848524 (ya existente en el sistema)	No se modifica la CI y se muestra un mensaje de error notificando que ya existe esa Cedula.	SI
Modificar alumno dejando	Se modifica el alumno Martin Perez y se borra el	No se modifica el alumno y se	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
el campo de Nombre y Apellido vacío	campo del nombre y el del apellido dejándolo vacío. Se presiona el botón "Guardar"	muestra un mensaje de error notificando que hay campos vacios	

<u>Eliminar Alumno</u>

Situación 1:

- ***No hay ningún alumno registrado en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de eliminar alumno sin alumnos registrados en el sistema	Se presiona el botón "Eliminar Alumnos"	Se muestra un mensaje informando que no hay alumnos en el sistema	SI

Situación 2:

- ***Se utilizan los datos de prueba automáticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Eliminar un alumno satisfactoriamente	Se selecciona un alumno de la lista de alumnos totales, cambiándola a la lista de alumnos a eliminar y se presiona el botón "Confirmar".	Se da de baja el alumno seleccionado en el sistema y se muestra un mensaje de confirmación.	SI

<u>Ver Alumnos</u>

Situación 1:

- ***No hay alumnos registrados en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de Ver alumnos sin alumnos registradas en el sistema	Se presiona el botón "Ver Alumnos"	Se muestra un mensaje informando que no hay alumnos en el sistema	SI

Situación 2:

- ***Se generan los datos de prueba automáticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ver alumnos satisfactoriamente	Se presiona el botón Ver Alumnos visualizando el listado de alumnos del sistema.	Al presionar el botón Ver Materias sobre un alumno seleccionado se despliegan las materias de ese alumno. El listado de alumnos se visualiza correctamente	SI

<u>Agregar Docente</u>

Situación 1:

- ***No hay ningún docente registrado en el sistema***
- ***Existen las materias "Ingles" y "Diseño de Aplicaciones 1"***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Agregar un nuevo docente satisfactoriamente	Se ingresan los datos del alumno: Nombre: “Ignacio” Apellido: “Perez” Materias que dicta: “Ingles” Se presiona el botón “Guardar”	Se da de alta el docente en el sistema y se muestra un mensaje: “Docente ingresado con éxito”	SI
Intento agregar un nuevo docente sin materias	Se ingresan los datos del alumno: Nombre: “Mauricio” Apellido: “Perez” Se presiona el botón “Guardar”	No se da de alta el docente por no asignarle alguna materia. Se muestra un mensaje de error que informa al usuario.	SI
Agregar docente con el campo de nombre y apellido vacío	Se ingresan los datos del docente: Nombre: “ ”	No se da de alta el docente en el sistema y se muestra un mensaje de error	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
	Apellido: " " Se presiona el botón "Guardar"		

<u>Modificar Docente</u>

Situación 1:

- ***No hay docentes registrados en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de modificar docente sin docentes registrados en el sistema	Se presiona el botón "Modificar Docente"	Se muestra un mensaje informando que no hay docentes en el sistema	SI

Situación 2:

- ***Se generan los datos de prueba automáticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Modificar docente satisfactoriamente	Se modifican los datos del docente con los siguientes datos: Nombre:Juan Apellido:Rodriguez Materias: Matemática y Geografia Se modifican los datos a los siguientes: Nombre: "Martin" Apellido: "Perez" Materias seleccionadas: Matematica Se presiona el botón "Guardar"	Se modifica el docente correctamente a los datos ingresados. Ahora la materia que dicta solo es Geografía, removiendo "Matemática"	SI
Modificar docente a nombre y apellido vacío	Se toman los datos del docente con los siguientes datos: Nombre:Juan Apellido:Rodriguez Materias: Geografia Se modifican y se dejan de la siguiente manera:	No se modifica el docente y se muestra un mensaje de error notificando que no pueden quedar el	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
	Nombre:<vacio> Apellido:<vacio> Materias: Geografia	nombre o el apellido vacíos	

<u>Eliminar Docente</u>

Situación 1:

- ***No hay ningún docente registrado en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de eliminar docente sin materias registradas en el sistema	Se presiona el botón "Eliminar Docente"	Se muestra un mensaje informando que no hay docentes en el sistema	SI

Situación 2:

- ***Se utilizan los datos de prueba automáticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Eliminar un docente satisfactoriamente	Se selecciona un docente de la lista de docentes totales, cambiándola a la lista de docentes a eliminar y se presiona el botón "Confirmar".	Se da de baja el docente seleccionado en el sistema y se muestra un mensaje de confirmación.	SI

Ver docentes

Situación 1:

- ***No hay docentes registrados en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de Ver docentes sin docentes registradas en el sistema	Se presiona el botón "Ver Docentes"	Se muestra un mensaje informando que no hay docentes en el sistema	SI

Situación 2:

- ***Se generan los datos de prueba automáticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ver docentes satisfactoriamente	Se presiona el botón Ver Docentes visualizando el listado de docentes del sistema.	Al presionar el botón Ver docentes sobre un docente seleccionado se despliegan los docentes de ese alumno. El listado de docentes se visualiza correctamente	SI

<u>Agregar Camioneta</u>

Situación 1:

- ***No hay ninguna camioneta registrada en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Agregar una nueva camioneta satisfactoriamente	Se ingresan la capacidad de la camioneta:	Se da de alta la camioneta en el sistema y se muestra	SI

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
	Capacidad: 10 Se presiona el botón "Guardar"	un mensaje: "Camioneta ingresada con éxito"	
Agregar nueva camioneta con valor de capacidad invalido	Se intenta agregar una camioneta con valor de capacidad -10	Se corrige automáticamente a capacidad: 0	SI

<u>Modificar Camioneta</u>

Situación 1:

- ***No hay ninguna camioneta registrada en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de modificar camioneta sin camionetas registrados en el sistema	Se presiona el botón "Modificar Camioneta"	Se muestra un mensaje informando que no hay camionetas en el sistema	SI

Situación 2:

- ***Se utilizan los datos de prueba automaticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Modificar una camioneta satisfactoriamente	Se selecciona la camioneta de Id 1, y se modifica la capacidad a 5 Se presiona el botón “Guardar”	Se modifica la capacidad de la camioneta correctamente.	SI
Modificar una camioneta a un valor invalido de capacidad	Se selecciona la camioneta de capacidad 1 y se modifica a la capacidad -12 Se presiona el botón “Guardar”	No se modifica la camioneta por capacidad invalida y se notifica al usuario con un error	SI

<u>Eliminar Camioneta</u>

Situación 1:

- ***No hay ninguna camioneta registrada en el sistema***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Ingreso a opción de eliminar camioneta sin camionetas registradas en el sistema	Se presiona el botón "Eliminar Camioneta"	Se muestra un mensaje informando que no hay camionetas en el sistema	SI

Situación 2:

- ***Se generan los datos de prueba automáticos***

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Eliminar una camioneta satisfactoriamente	Se selecciona la camioneta de ID 1 de la lista de camionetas totales, cambiándola a la lista de camionetas a eliminar y se presiona el botón "Confirmar".	Se da de baja la camioneta de ID 1 en el sistema y se muestra un mensaje de confirmación.	SI

Listado Camionetas

Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Listado de camionetas	Se selecciona la opción de listar las camionetas del sistema.	Se muestran las camionetas del sistema con su respectivo ID y Capacidad	SI

Calcular Rutas (Camioneta)

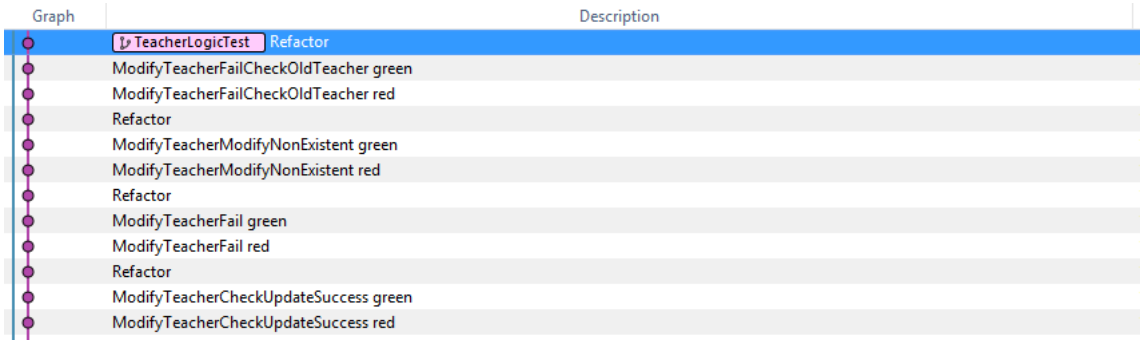
Función	Entrada de Datos	Resultado Esperado	¿Resultado Obtenido es el Esperado?
Calcular rutas	El usuario selecciona una camioneta y visualiza la ruta la lista de rutas de esa camioneta	Se muestran las rutas disponibles para esa camioneta seleccionada.	SI

1.10.1. Evidencia de Pruebas Unitarias

1.10.1.1. Evidencia de TDD

En el repositorio se puede ver la evidencia del uso de TDD. Se ven los 3 pasos que utilizamos:

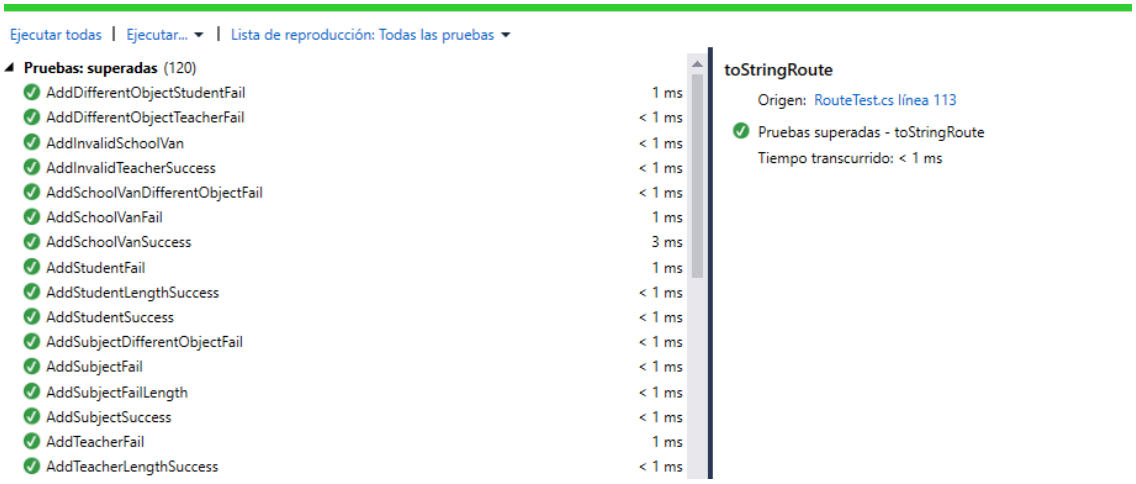
1. Paso Red: Creamos la prueba aunque no compile.
2. Paso Green: Implementamos lo mínimo necesario para que la prueba compile (en caso de que no lo haga) y luego para que pase.
3. Refactor: Modificamos el código implementado en los pasos anteriores sin modificar el comportamiento del mismo aplicando las técnicas mencionadas en el Sector de "Clean Code".



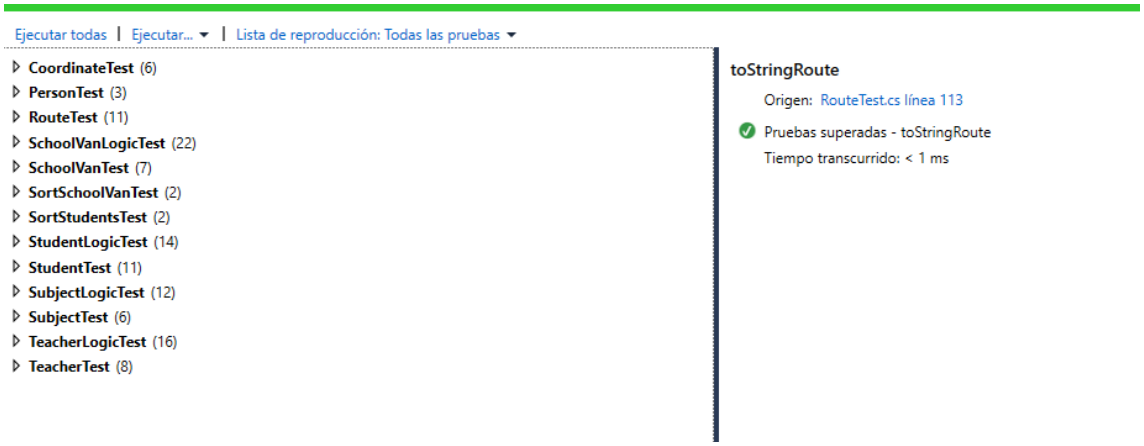
Copiamos una imagen del repositorio en la cual se puede observar claramente los pasos "Red", "Green" y "Refactor" mencionados anteriormente.

1.10.1.2. Resultado de las pruebas

Se realizaron un total de 120 pruebas unitarias independientes donde un total del 100% recibieron el resultado esperado.



Dividimos dichas pruebas en diferentes clases dependiendo de qué clase se estuviese probando.



1.10.1.3. Cobertura de las pruebas

Para verificar la cobertura de nuestras pruebas en la solución utilizamos la herramienta de cobertura de código de Visual Studio.

Jerarquía	No cubiertos (bloques)	No cubiertos (% de bloques)	Cubiertos (bloques)	Cubiertos (% de bloques)
alumnoFI_FIW10LC0718 2017-1...	24	2,15 %	1091	97,85 %
domain.dll	0	0,00 %	468	100,00 %
exceptions.dll	0	0,00 %	26	100,00 %
logic.dll	24	4,01 %	574	95,99 %
repository.dll	0	0,00 %	23	100,00 %

En el dominio logramos alcanzar el 100% de cobertura en las pruebas mientras que en la lógica cubrimos un 96% del código total. Este gran porcentaje es causado directamente por haber trabajado con la metodología de TDD.

1.11. Supuestos efectuados

- Las camionetas tienen un mínimo de capacidad que es 1 y un máximo que es 15 personas.
- El identificador de camioneta es un número que no se puede repetir dentro de las camionetas que están en el sistema. Dicho número es autogenerado.
- El identificador de docentes es un número que no se puede repetir dentro de los docentes que están en el sistema. Dicho número es autogenerado.
- El identificador de materias es un código alfanumérico que no se puede repetir dentro de las materias que están en el sistema. Dicho código es autogenerado.
- El identificador de alumnos es un número que no se puede repetir dentro de los alumnos que están en el sistema. Dicho número es autogenerado.
- No se puede repetir una CI de los alumnos que ya este ingresada en el sistema.
- El mínimo de una CI de los alumnos es 1000000.
- No se permiten nombres, apellidos vacíos en las entidades del proyecto.
- Los alumnos deben tener materias asignadas para que sea válido. De todas maneras si se borran las materias que un alumno está cursando, el alumno puede quedar sin ninguna materia anotada.
- Los docentes deben tener materias asignadas para que sea válido. De todas maneras si se borran las materias que un docente está dictando, el docente puede quedar sin ninguna materia anotada
- En el cálculo de rutas se asignan los alumnos por orden de ID, aumentando la probabilidad de que ID's consecutivos estén en la misma camioneta. Se eligió esta asignación para generar un círculo cercano entre las personas que se inscribieron en fechas cercanas.

3. Conclusiones

Como conclusión, entendimos la importancia de las prácticas de Clean Code para poder desarrollar un código de calidad y entendimos el funcionamiento de TDD, logrando adaptarnos a esta metodología de desarrollo guiada por las pruebas.

Comprendimos la importancia de lograr un buen porcentaje de cobertura para minimizar los errores, apoyado de la documentación con los respectivos casos de prueba, diagramas de caso de uso y requerimientos y diagramas de clases.

4. Referencias bibliográficas