

**Alumnos:**

José Alonso Lora Gonzáles.

Raymundo de Jesús Pérez Castellanos

Irvin Dereb Vera López.

**Experiencia Educativa:** Principios de construcción de software.

**Académico:** Juan Carlos Pérez Arriaga.

**Programa Educativo:** Ingeniería de Software.

Contenido

[Revisión de código apegada al estándar. 6](#_Toc484603901)

[Modelos de diseño. 7](#_Toc484603902)

[Modelo de Dominio. 7](#_Toc484603903)

[Modelo de Casos de Uso. 9](#_Toc484603904)

[Modelo de Entidad-Relación. 13](#_Toc484603905)

[Capa de persistencia de datos. 13](#_Toc484603906)

[Caso de Uso Administrar Reservaciones. 15](#_Toc484603907)

[Caso de Uso Administrar Inscripción. 21](#_Toc484603908)

[Caso de Uso Administrar actividad en calendario. 26](#_Toc484603909)

[31](#_Toc484603910)

[Reporte de pruebas unitarias. 32](#_Toc484603911)

[Conclusiones. 40](#_Toc484603912)

[Referencias. 40](#_Toc484603913)

**Definición del estándar de codificación.**

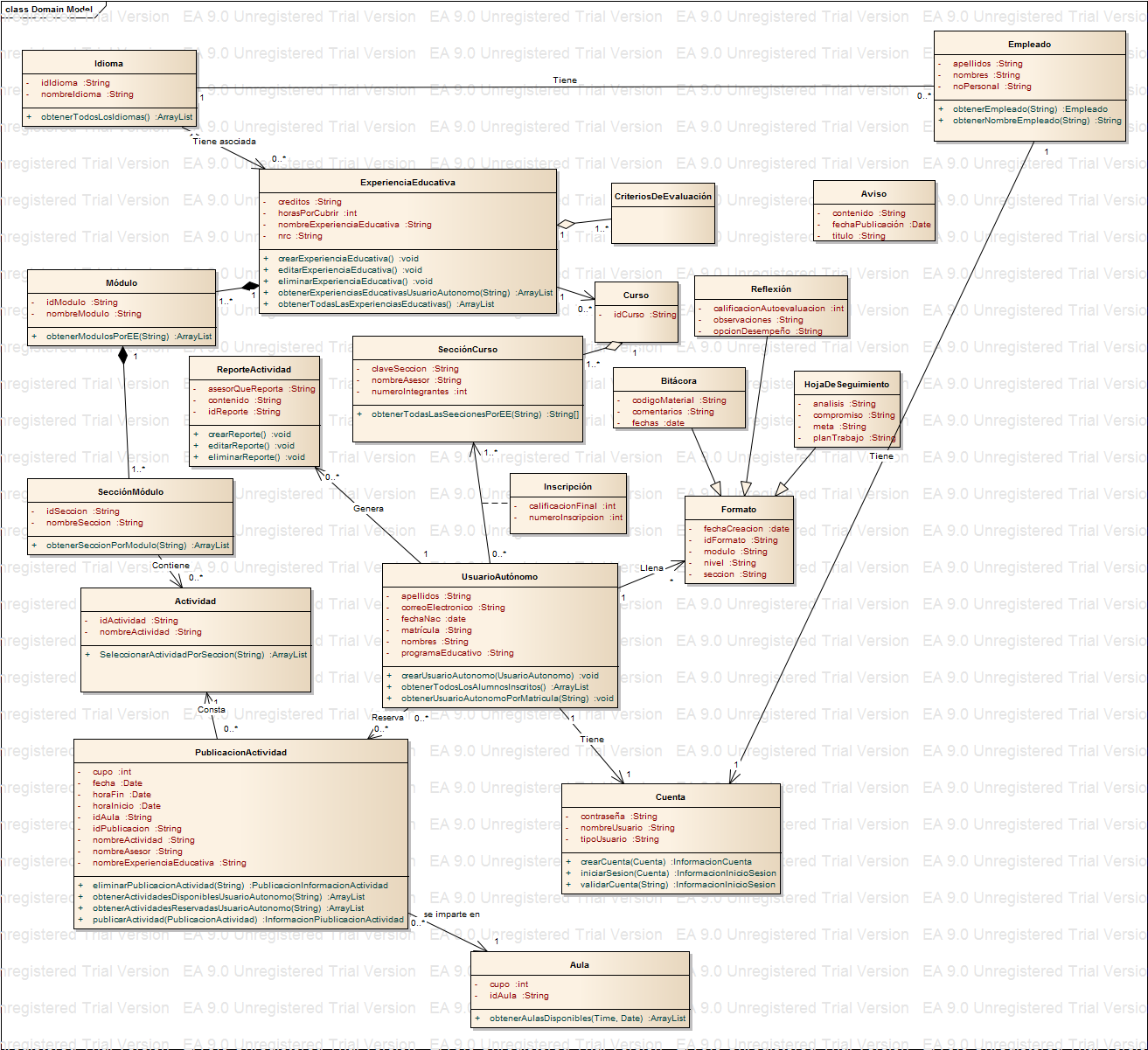
|  |  |
| --- | --- |
| Propósito | Guiar la implementación en los programas de Java |
| Cabeceras de programa | Comenzar todos los programas con una cabecera descriptiva. |
| Formato de la cabecera | /\*Identificador del programa:  Clave única del programa:    Nombre(s):  Nombres de los desarrolladores: Irvin Dereb Vera López.  Jose Alonso Lora González.  Raymundo de Jesús Perez Caastellanos.    Fecha en la que se inicio el programa :    Descripción:  Una breve descripción del programa y que es lo que hace:  \*/ |
| Listado de contenido | Proporciona un resumen del contenido listado |
| Ejemplo de contenidos | Listado de contenido:  Instrucciones de reutilización  Instrucciones de modificación  Instrucciones de compilación  Includes  Declaración de clases  SumarNumeros    Código fuente en c:/clases/SumarNumeros.java  SumarNumeros  SumarNumeros()  Empty() |
| Instrucciones de reutilización | * Describe cómo va a usarse el programa. Proporciona el formato de declaración, valores tipos y límites de los parámetros. * Proporciona advertencias de valores ilegales, condiciones sobrecargadas, u otras condiciones que potencialmente podrían resultar en una operación inapropiada |
| Ejemplo de reutilización | /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*/  /\* Instrucciones de reutilización  /\* int imprimeCadena(String Cadena) \*/  /\* Purpose: Imprimir una cadena en pantalla \*/  /\* Limitations: El tamaño de la cadena no debe sobrepasar el LINE\_LENGTH \*/  /\* Return 0 si no puede imprimir, sino 1 \*/  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*/ |
| Identificadores | Usa nombres descriptivos para todas las variables, nombres de funciones, constantes y otros identificadores. Evita abreviaciones o nombres de variables con una sola letra. |
| Ejemplo de identificadores | int numeroEstudiantes; /\* Bien \*/ Float: x4, j, ftave; /\* Mal\*/ |
| Comentarios | * Documenta el código para que el elector pueda entender la operación * Los comentarios deberían explicar el propósito y el comportamiento del código. |
| Buen comentario | /\*\*  \* Descripción  \* @author  \* @param  \* @return  \*@exception  \*/ |  |
| Mal comentario | If(record\_count > limit) /\* Determina si el contador de grabaciones excedió el límite\*/  If (record\_count > limit) /\* ¿Han sido procesadas todas las grabaciones ? \*/ |  |
| Secciones principales | Precede las secciones principales del programa mediante un comentario en bloque que describa el procesamiento que se realiza en la siguiente sección |  |
| Ejemplo | /\*\*  \* Valida los datos ingresados por el usuario para iniciar sesión.  \* @author Irvin Vera  \* @author Alonso Lora  \* @author Raymundo Pérez  \* @param cuenta Contiene los datos del usuario.  \*@exception SQLException La excepción se puede mandar cuando no se obtiene los  \* datos esperados de la consulta por alguna razón  \*@return Un objeto de la clase <MensajesParaUsuario>  \*el cual contiene información sobre el estado de inicio de sesión.  \*/  public MensajesParaUsuario iniciarSesion(Cuenta cuenta){  } |  |
| Espacios en blanco | * Escribe programas con suficiente espacio, de esta manera no estarán amontonados. * Separa cada construcción del programa con al menos un espacio |  |
| Identación | * Cada vez que se abre un nuevo bloque de sangría aumenta por dos espacios. Cuando el bloque termina, la sangría vuelve al nivel de sangría anterior.      * Las llaves abiertas deben estar en renglón de la función, ciclo o condición a la que correspondan y las cerradas deben estar en líneas por sí mismas y alineadas con el elemento que cierra |  |
|  |  |  |
| Ejemplo de identación | while (missDistance > threshold) {  successCode = moveRobot (target Location);  if (successCode == MOVE\_FAILED) {  System.Out.PrintLn(“The robot move has failed.\n”);  }  } |  |
| Capitalización | Lower Camel Case para todos los identificadores y letras minúsculas para las palabras reservadas  Los mensajes para el usuario pueden usar mayúsculas y minúsculas |  |
| Ejemplo de capitalización | System.Out.PrintLn(“ADVERTENCIA: Error de conexión con la base de datos”); |  |
|  |  |  |

# **Revisión de código apegada al estándar.**

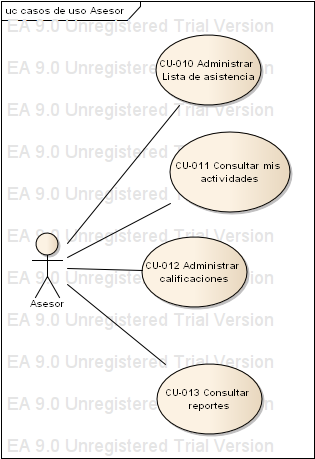
|  |  |  |
| --- | --- | --- |
|  | Revisión del código apegada al estándar |  |
| Preguntas | ¿Cumple? | Observaciones |
| ¿Existen razones suficientes para crear todos los métodos? | Si |  |
| ¿El nombre de los métodos es claro, así como su función? | Sí |  |
| ¿Cuáles son las convenciones de código que se siguen en la escritura de los métodos? | Si |  |
| ¿Las listas de parámetros es consistente? | Si |  |
| ¿Los parámetros siguen un orden específico? | Si |  |
| ¿Todos los parámetros son usados? | Si |  |
| ¿Los métodos que regresan valores, se aseguran que sean valores válidos? | Si |  |
| ¿Existe una documentación sobre los métodos y, si lanza excepciones, se especifica en qué casos ocurre? | SI |  |
| ¿El nombre de las variables es descriptivo? | Si |  |
| ¿Los métodos tiene una sola responsabilidad? | Si |  |

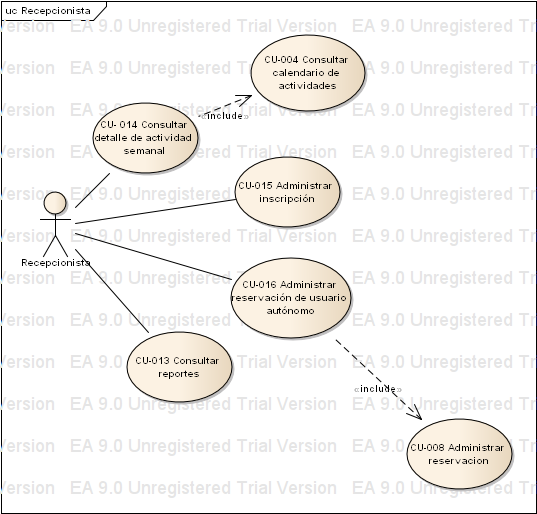
# **Modelos de diseño.**

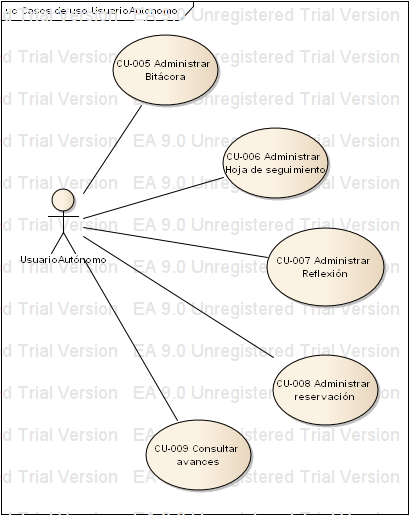
## **Modelo de Dominio.**

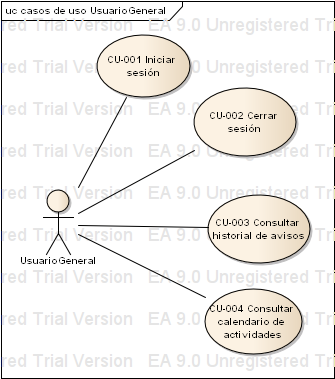


## C:\Users\irdev\AppData\Local\Microsoft\Windows\INetCache\Content.Word\caso de uso Coordinador.png**Modelo de Casos de Uso.**



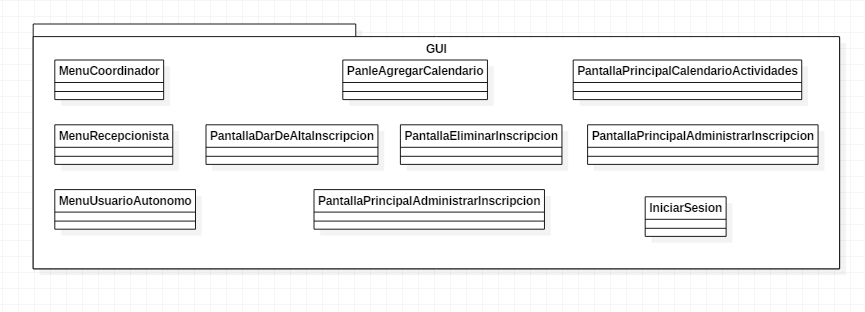


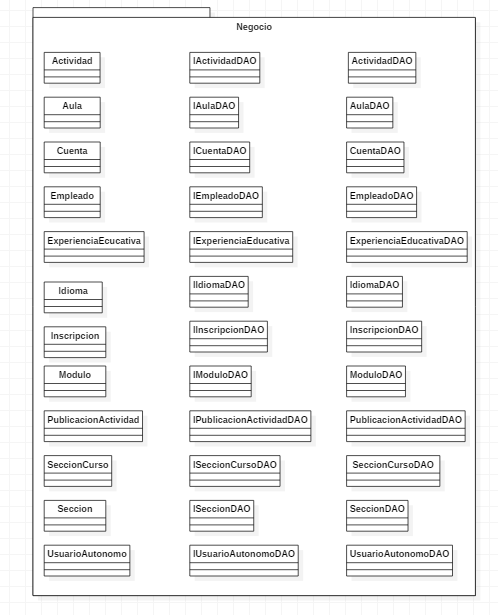


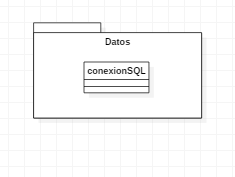


## C:\Users\irdev\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Diagrama entidad relacion.png**Modelo de Entidad-Relación.**

# **Capa de persistencia de datos.**

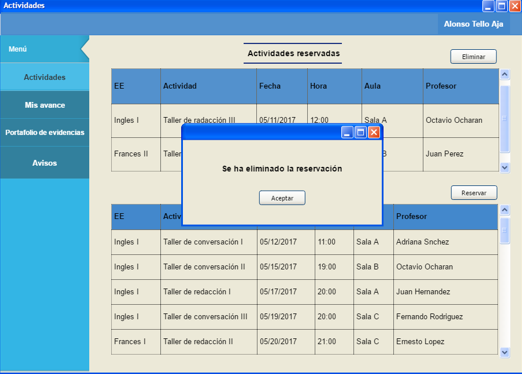


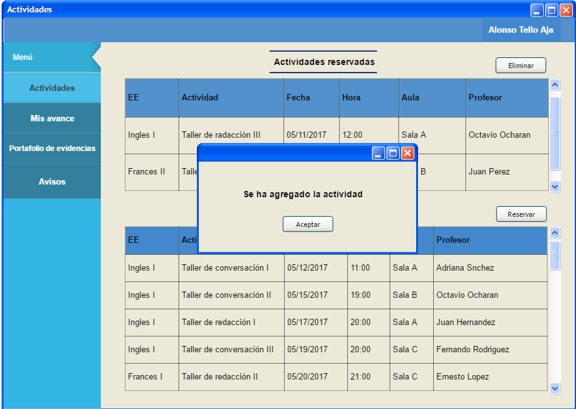




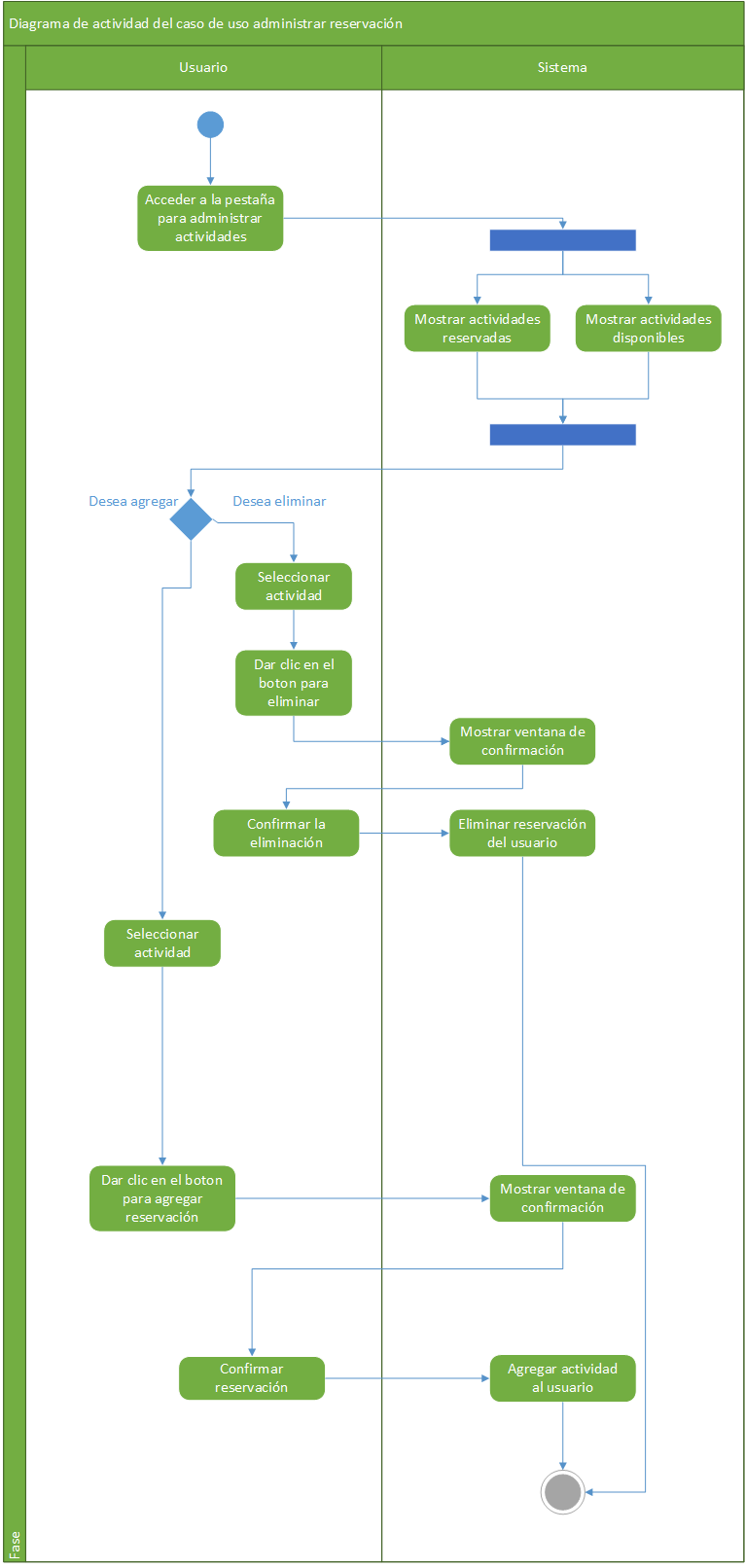
## **Caso de Uso Administrar Reservaciones.**

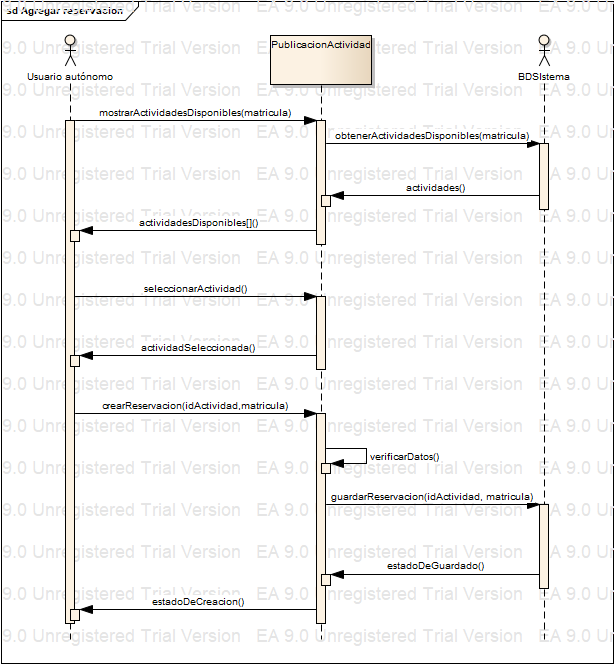


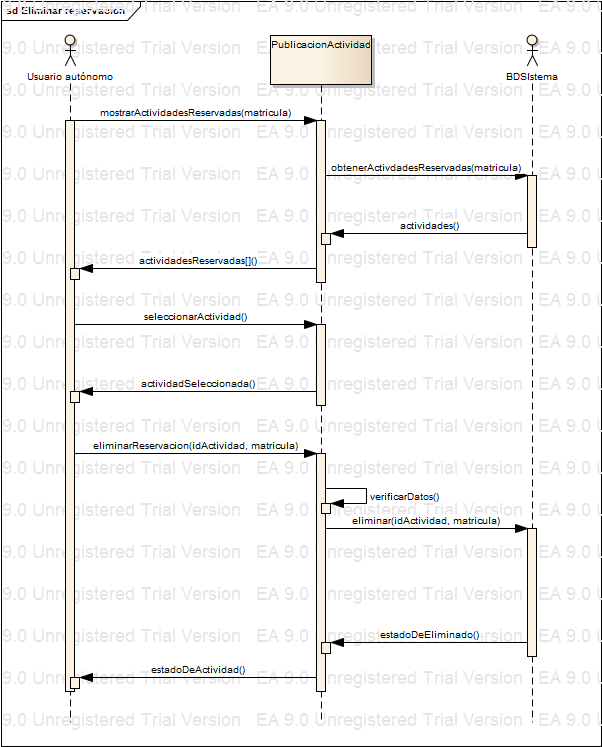




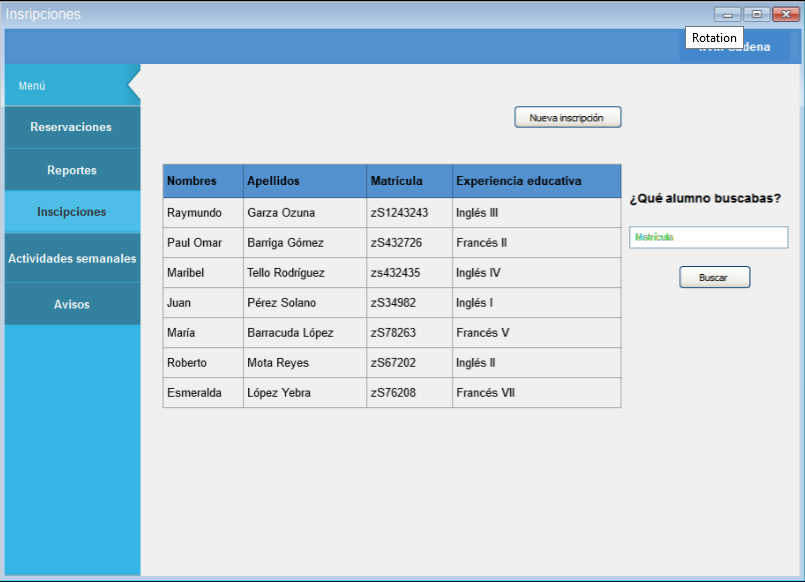
|  |  |
| --- | --- |
| **­­ID:** | CU-008 |
| **Nombre:** | Administrar reservación |
| **Autor(es):** | Raymundo Pérez, Alonso Lora, Irvin Vera |
| **Fecha de creación:** | 6/04/2017 |
| **Fecha de actualización:** | 2/05/2017 |
| **Actor(es):** | Usuario Autónomo. |
| **Descripción:** | El usuario autónomo podrá asegurar un lugar en una actividad, eliminar una reservación y consultar las actividades disponibles. |
| **Precondiciones:** | 1. El usuario autónomo debe estar autenticado. 2. Deben existir actividades disponibles 3. El usuario autónomo debe estar inscrito a un curso. |
| **Flujo Normal:** | Reservar actividad   1. El alumno accede a la pestaña para administrar sus actividades. 2. El sistema muestra una tabla con calendario de las actividades actualmente disponibles, así como otra tabla con las actividades ya reservadas por el alumno, indicando la experiencia educativa, la fecha, la hora, el aula y el profesor que la imparte. 3. El alumno selecciona una actividad disponible de la tabla y da clic en el botón para agregarla. 4. El sistema muestra una ventana de confirmación. 5. El alumno da clic en el botón aceptar. 6. El sistema regresa a la pantalla para reservar actividad. |
| **Flujos Alternos:** | (En paso 3): Eliminar actividad   1. El alumno selecciona una actividad de su tabla de actividades reservadas y da clic en el botón para eliminarla. 2. El sistema muestra una ventana de confirmación. 3. El alumno da clic en el botón aceptar. 4. El sistema regresa a la pantalla para reservar actividad. |
| **Excepciones:** | No hay conexión a la base de datos.   1. El sistema muestra un mensaje indicando que no fue posible conectarse a la base de datos. |
| **Postcondiciones:** | 1. Se debe registrar la reservación. 2. Se agregará el usuario autónomo a la lista del asesor. 3. Se debe reducir el cupo de la actividad. 4. Se debe mostrar la actividad en la tabla de actividades. reservadas del usuario autónomo. |

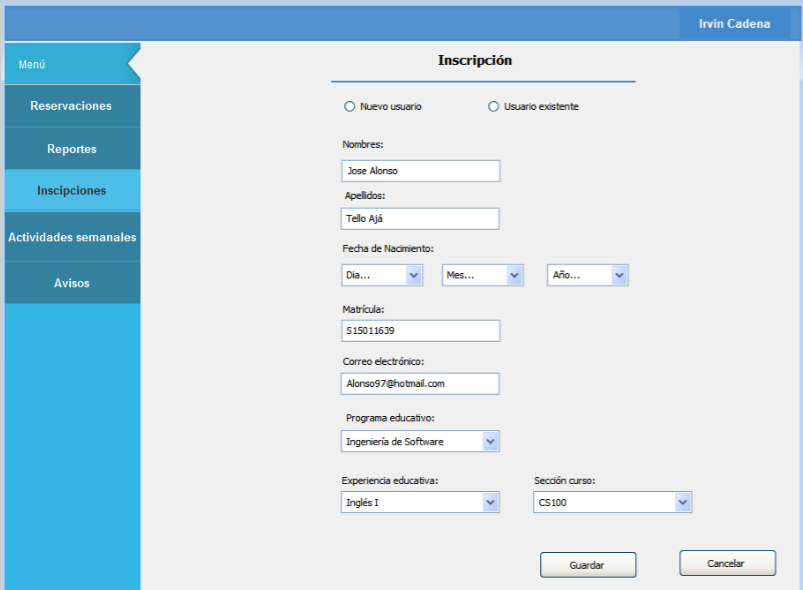


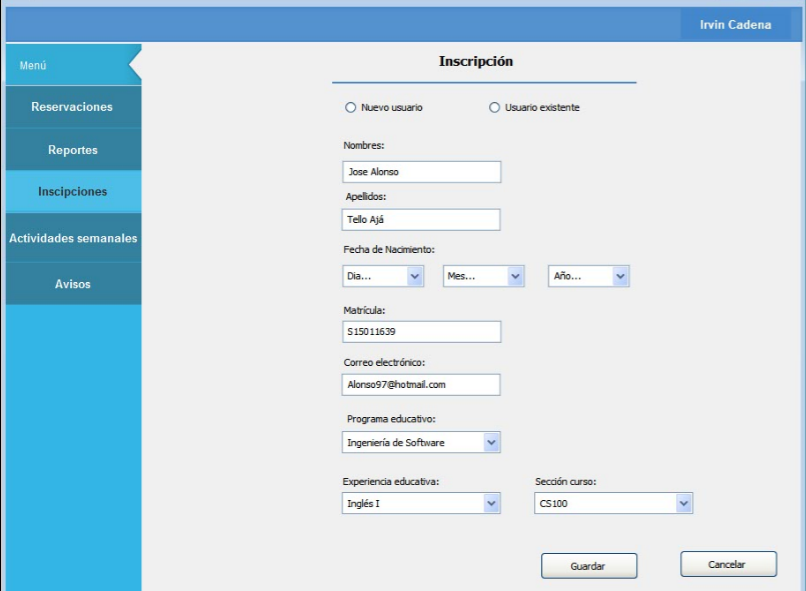


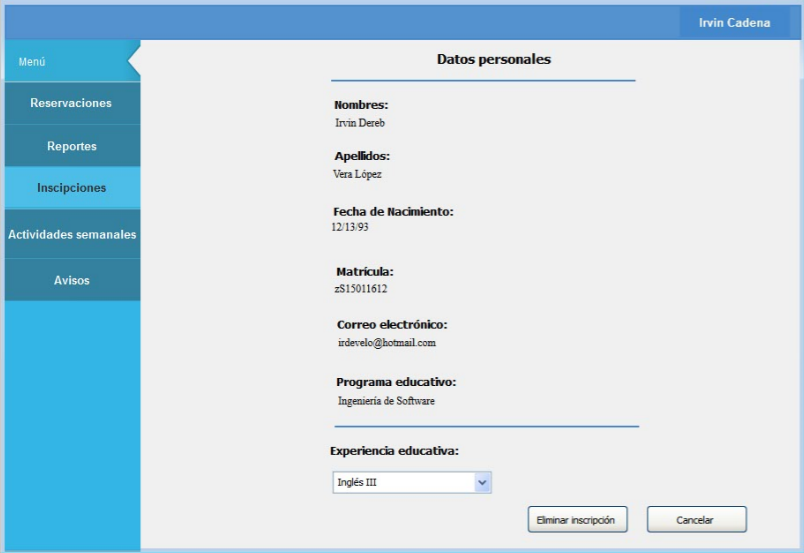


## **Caso de Uso Administrar Inscripción.**

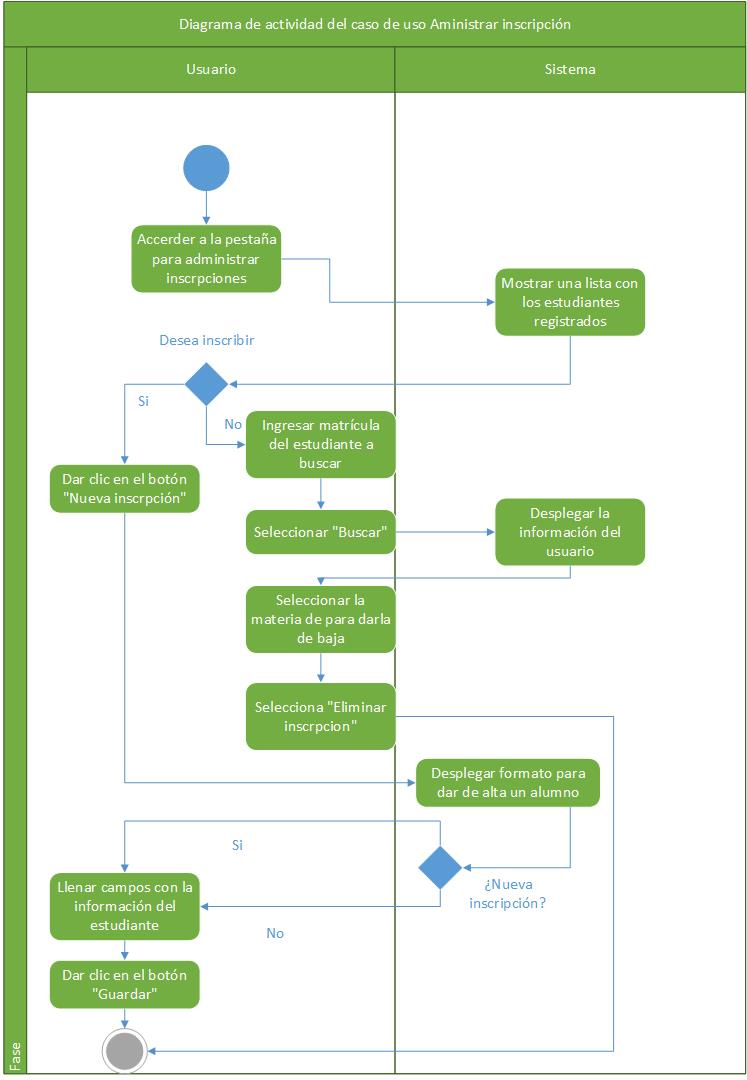


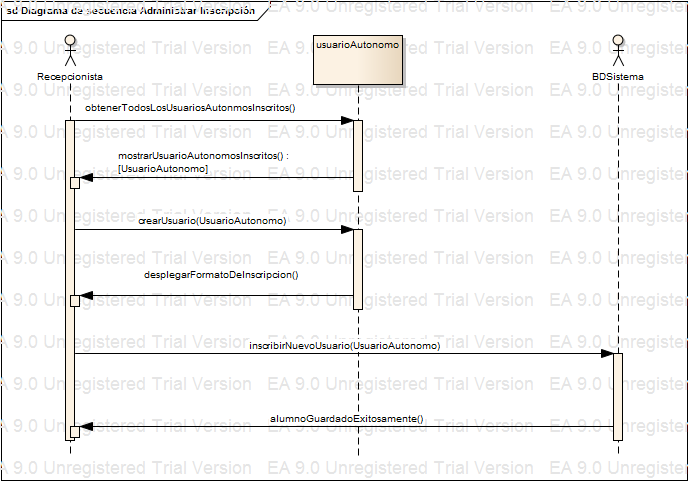


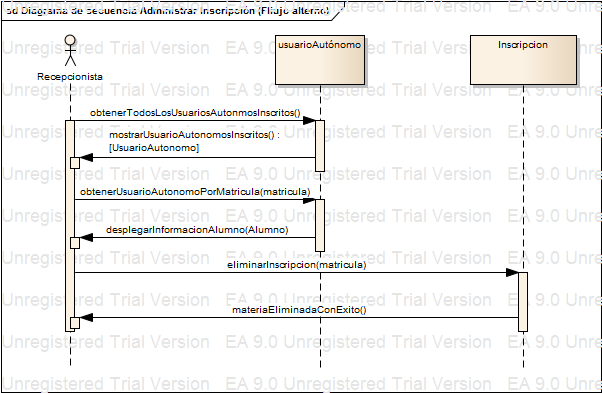




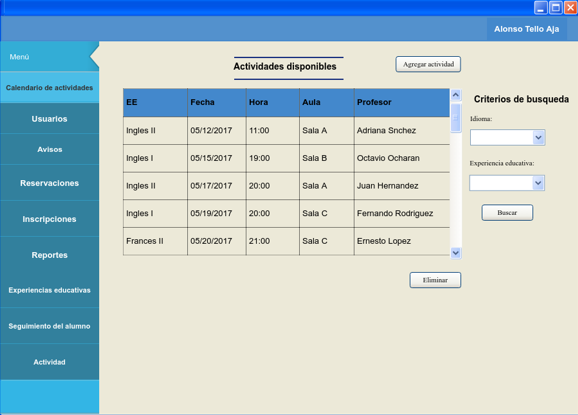
|  |  |
| --- | --- |
| **­­ID:** |  |
| **Nombre:** | Administrar inscripción |
| **Autor(es):** | Raymundo Pérez, Alonso Lora, Irvin Vera |
| **Fecha de creación:** | 03/05/2017 |
| **Actor(es):** | Recepcionista, coordinador. |
| **Descripción:** | El sistema permitirá al recepcionista agregar, buscar y eliminar una inscripción. |
| **Precondiciones:** | Debe estar autenticado el recepcionista.  Debe haber alumnos inscritos. |
| **Flujo Normal:** | 1. El usuario accede a la pestaña inscripciones. 2. El sistema muestra una lista con los estudiantes inscritos en el sistema. 3. El usuario selecciona “Nueva inscripción”. 4. El sistema despliega un formato para dar de alta un alumno. 5. El usuario selecciona el radioButton “Nuevo usuario” y llena todos los campos. 6. El usuario selecciona “Guardar”. |
| **Flujos Alternos:** | (En paso 5): Nueva inscripción (Usuario existente)  El usuario selecciona el radioButton “Usuario existente” y llena los campos.  El usuario selecciona “Guardar”  (En paso 3): Eliminar inscripción  1. El usuario ingresa la matrícula del estudiante que desea buscar y selecciona buscar.  3. El sistema despliega la información del usuario.  4. El usuario selecciona de un ComboBox la materia de la que desea dar de baja al alumno u selecciona “Eliminar inscripción”.  6. El sistema manda un mensaje de confirmación. |
| **Excepciones:** | Error de conexión con la base de datos.  No hay conexión a internet. |
| **Postcondiciones:** | En el caso de agregar una inscripción el alumno debe quedar dado de alta en la experiencia educativa.  En el caso de eliminar una inscripción el usuario debe de quedar dado de baja de dicha experiencia educativa.  En el caso de editar, la información del usuario debe aparecer actualizada. |

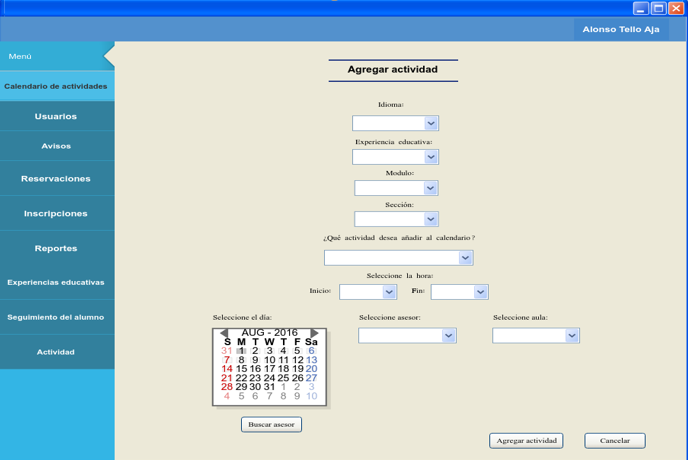


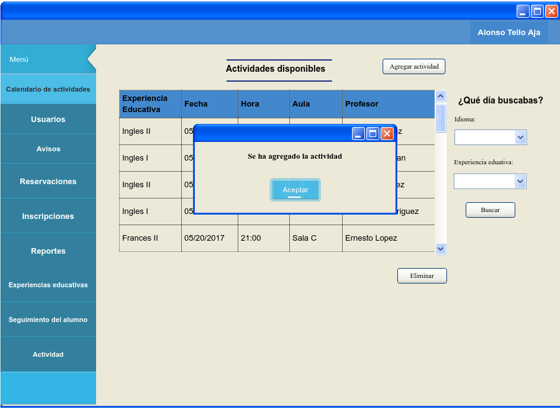


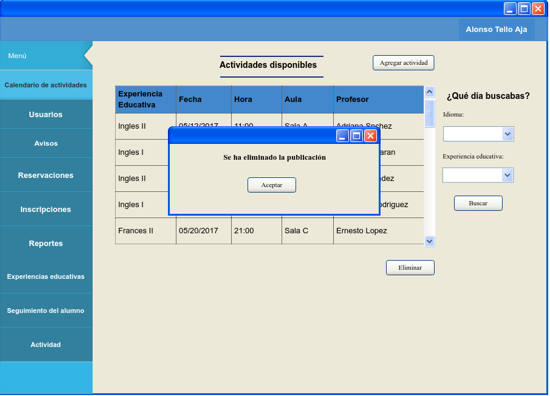


## **Caso de Uso Administrar actividad en calendario.**

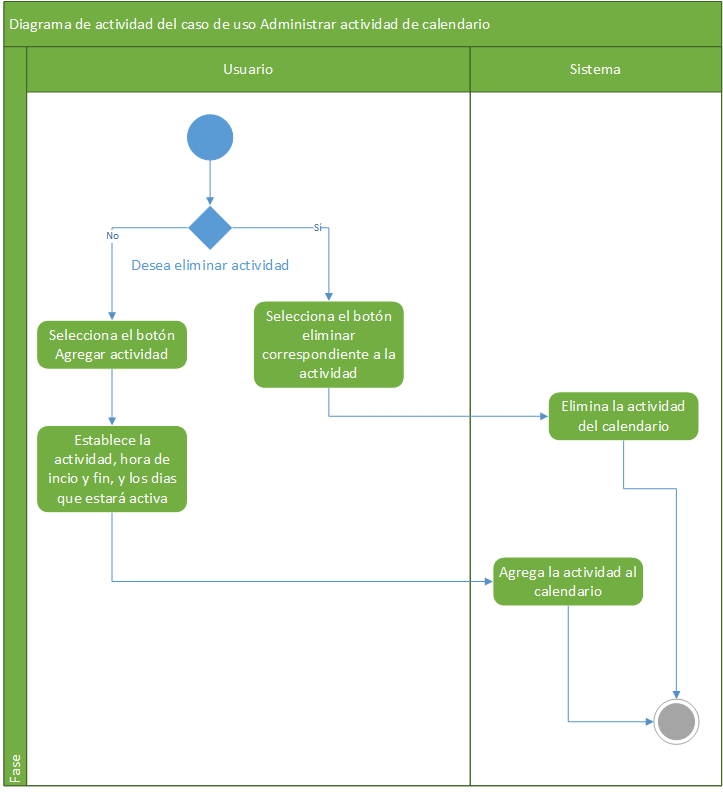


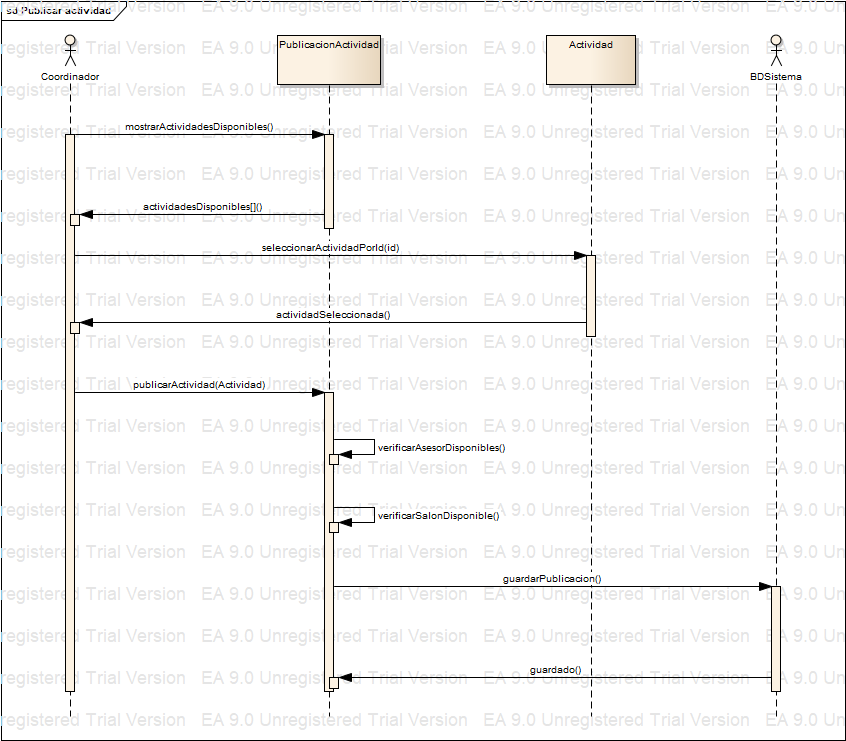






|  |  |
| --- | --- |
| **ID:** | CU-017 |
| **Nombre:** | Administrar actividad de calendario |
| **Autor(es):** | Alonso Lora, Raymundo Pérez, Irvin Vera |
| **Fecha de creación:** | 04/05/2017 |
| **Fecha de actualización:** | 06/06/2017 |
| **Actor(es):** | Coordinador |
| **Descripción:** | En este CU el coordinador publica actividades en el calendario, para que el usuario autónomo pueda reservalas, así como quitar la actividad del calendario. |
| **Precondiciones:** | 1. Deben existir actividades en el sistema. 2. Debe haber asesores disponibles. 3. Debe haber aulas disponibles. |
| **Flujo Normal:** | Agregar actividad   1. El coordinador accede a la pestaña para administrar la actividad. 2. El sistema muestra las actividades disponibles. 3. El coordinador selecciona Agregar actividad. 4. El sistema muestra una ventana para llenar la información de la actividad. 5. El coordinador selecciona la actividad, la hora de inicio y fin, así como la fecha que estará activa y selecciona buscar asesor. 6. El sistema verifica los asesores y aulas disponibles a la hora y fecha establecida. 7. El coordinador selecciona agregar actividad. 8. El sistema muestra una ventana de confirmación. |
| **Flujos Alternos:** | Eliminar actividad   1. El coordinador accede a la pestaña para administrar la actividad. 2. El sistema muestra las actividades disponibles. 3. El coordinador selecciona la actividad de la tabla y selecciona eliminar. 4. El sistema elimina todas las reservaciones hechas por los usuarios autónomos.   Eliminar actividad (2)   1. El coordinador busca por idioma y por la experiencia educativa. 2. El sistema muestra los resultados de la búsqueda. 3. El coordinador selecciona la actividad de la tabla y selecciona eliminar. 4. El sistema elimina todas las reservaciones hechas por los usuarios autónomos. |
| **Excepciones:** | No hay conexión a la base de datos.   1. El sistema informa al usuario que hubo un error al conectarse a la base datos. |
| **Postcondiciones:** | 1. La actividad debe aparecer en el calendario. |





# C:\Users\irdev\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Cancelar actividad .png

# **Reporte de pruebas unitarias.**

En esta sección se muestra el reporte acerca de las pruebas unitarias elaboradas a las funcionalidades del sistema. Las pruebas se aplicaron a los métodos que se consideraron más importantes dentro del proyecto.

|  |  |
| --- | --- |
| Método | testSeleccionarActividadPorSeccion |
| Propósito | Obtener actividades existentes dado el Id de una sección |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase Actividad |

|  |  |
| --- | --- |
| Método | testObtenerAulasDisponibles |
| Propósito | Obtener las aulas disponibles dado una hora y fecha |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase Aula |

|  |  |
| --- | --- |
| Método | testCrearCuenta |
| Propósito | Poder crear una cuenta para un Usuario Autonomo. |
| Criterio de aceptación | La prueba debe regresar un objeto de tipo InformacionCuenta así como registrar los parámetros en la base de datos |

|  |  |
| --- | --- |
| Método | testIniciarSesion |
| Propósito | Que los usuarios puedan iniciar sesión el en sistema |
| Criterio de aceptación | La prueba debe primero autenticar que el usuario esté registrado, así como el rol que cumple en rol que tiene dentro del sistema y después lo valida para iniciar sesión |

|  |  |
| --- | --- |
| Método | testValidarCuenta |
| Propósito | Valida que tipo de usuario es el que iniciará sesión |
| Criterio de aceptación | La prueba debe determinar que tipo de usuario es el que iniciará sesión en el sistema |

|  |  |
| --- | --- |
| Método | testObtenerEmpleado |
| Propósito | Obtener la información de un empleado33 dado su nombre de usuario |
| Criterio de aceptación | La prueba debe regresar un objeto de la clase Empleado |

|  |  |
| --- | --- |
| Método | testObtenerNombreEmpleado |
| Propósito | Obtener el nombre de un usuario dado su número de personal |
| Criterio de aceptación | La prueba debe regresar un el nombre de un empleado |

|  |  |
| --- | --- |
| Método | testObtenerEmpleadosDisponibles |
| Propósito | Obtener una lista de empleados disponibles dado una hora y una fecha |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase Empleado. |

|  |  |
| --- | --- |
| Método | testObtenerTodasLasExperienciasEducativas |
| Propósito | Obtener los nombres de todas las experiencias educativas. |
| Criterio de aceptación | La prueba debe regresar un arreglo con los nombres de las experiencias educativas |

|  |  |
| --- | --- |
| Método | testobtenerExperienciasEducativasUsuarioAutonomo |
| Propósito | Obtener las experiencias educativas a las que está inscrito un usuario autónomo dado su matrícula |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de tipo ExperienciaEducativa que son a las que está inscrito es estudiante |

|  |  |
| --- | --- |
| Método | testobtenerExperienciaEducativaPorIdioma |
| Propósito | Obtener las experiencias educativas que tenga un idioma dado el ID del idioma |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase ExperienciaEducativa que son las existentes dado un idioma |

|  |  |
| --- | --- |
| Método | testobtenerTodosLosIdiomas |
| Propósito | Obtener una lista de todos los idiomas existentes |
| Criterio de aceptación | La prueba debe regresar una lista de objetos de la clase Idioma. |

|  |  |
| --- | --- |
| Método | testCrearInscripcion |
| Propósito | Permite inscribir a un UsuarioAutonomo a una ExperienciaEducativa |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionInscripcion según sea el caso.  -inscripcionCreadaCorrectamente  -inscripcionNoCreada |

|  |  |
| --- | --- |
| Método | testValidarExistenciaInscripcion |
| Propósito | Validar si un UsuarioAutonomo está inscrito en alguna ExperienciaEducativa |
| Criterio de aceptación | La prueba debe regresar un valor si el UsuarioAutonomo está inscrito en la ExperienciaEducativa o regresar otro valor de ser lo contrario |

|  |  |
| --- | --- |
| Método | testBuscarInscripcion |
| Propósito | Obtener la clave sección de una inscripción dado el nrc de una ExperienciaEducativa |
| Criterio de aceptación | La prueba debe regresar la claveSeccion de la SeccionCurso |

|  |  |
| --- | --- |
| Método | testEliminarInscripcion |
| Propósito | Obtener la inscripción de un UsuarioAutonomo dado su matrícula y la claveSeccion de la ExperienciaEducativa de la cual se quiere dar de baja. |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionInscripcion según sea el caso.  -inscripcionEliminadaCorrectamente  -inscripcionNoEliminada |

|  |  |
| --- | --- |
| Método | testObtenerModulosPorEE |
| Propósito | Obtener los módulos que tiene un ExperienciaEducativa dado su nrc |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase Modulo |

|  |  |
| --- | --- |
| Método | testObtenerActividadesDisponiblesUsuarioAutonomo |
| Propósito | Obtener las actividades disponibles para un UsuarioAutonomo dada su matrícula |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase ExperienciaEducativa |

|  |  |
| --- | --- |
| Método | testAgregarReservacionPublicacionActividad |
| Propósito | Agregar una reservación |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionPublicacionActividad según sea el caso.  -reservacionNoGuardada  -reservacionGuardada |

|  |  |
| --- | --- |
| Método | testEliminarReservacionPublicacionActividad |
| Propósito | Eliminar la reservación de la actividad de un UsuarioAutonomo |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionPublicacionActividad según sea el caso.  -reservacionNoEliminada  -reservacionEliminada |

|  |  |
| --- | --- |
| Método | testPublicarActividad |
| Propósito | Publicar una actividad |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionPublicacionActividad según sea el caso.  -actividadPublicada  -reservacionNoPublicada |

|  |  |
| --- | --- |
| Método | testObtenerActividadesDisponibles |
| Propósito | Obtener todas las actividades publicadas disponibles. |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase PublicacionActividad |

|  |  |
| --- | --- |
| Método | testEliminarPublicacionActividad |
| Propósito | Eliminar la publicación de alguna Actividad dado su ID |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionPublicacionActividad según sea el caso.  -publicacionEliminada  -publicacionNoEliminada |

|  |  |
| --- | --- |
| Método | testObtenerActividadesDisponiblesPorEE |
| Propósito | Obtener las actividades de una ExperienciaEducativa dado su nrc |
| Criterio de aceptación | La prueba debe regresar una lista de objetos de la clase PublicacionActividad |

|  |  |
| --- | --- |
| Método | testReducirCupoPublicacionActividad |
| Propósito | Reducir el cupo de una PublicacionActividad dado su ID cuando esta es reservada |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionPublicacionActividad según sea el caso.  -cupoNoRestado  -cupoRestado |

|  |  |
| --- | --- |
| Método | testAumentarCupoPublicacionActividad |
| Propósito | Aumentar el cupo de una PublicacionActividad dado su ID cuando una reservación sobre dicha actividad es eliminada. |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionPublicacionActividad según sea el caso.  -cupoNoRestado  -cupoRestado |

|  |  |
| --- | --- |
| Método | testEliminarTodasLasReservacionesDeUnaPublicacion |
| Propósito | Eliminar todas las reservaciones que tenga una PublicacionActividad |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionPublicacionActividad según sea el caso.  -reservacionEliminada  -reservacionNoEliminada |

|  |  |
| --- | --- |
| Método | testObtenerTodasLasSeecionesPorEE |
| Propósito | Obtener todas las secciones que tenga un ExperienciaEducativa dada si nrc |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase Seccion |

|  |  |
| --- | --- |
| Método | testObtenerSeccionPorModulo |
| Propósito | Obtener las secciones que tenga un módulo |
| Criterio de aceptación | La prueba debe regresar una lista con objetos de la clase Seccion |

|  |  |
| --- | --- |
| Método | testCrearUsuarioAutonomo |
| Propósito | Crear un nuevo UsuarioAutonomo |
| Criterio de aceptación | La prueba debe regresar un mensaje de la clase InformacionUsuarioAutonomo según sea el caso.  -usuarioAutonomoNoCreado  -usuarioAutonomoCreadoCorrectamente |

|  |  |
| --- | --- |
| Método | testObtenerUsuarioAutonomo |
| Propósito | Obtener la información de un UsuarioAutonomo |
| Criterio de aceptación | La prueba debe regresar un objeto de la clase Empleado |

|  |  |
| --- | --- |
| Método | testValidarExistenciaUsuarioAutonomo |
| Propósito | Validar que un UsaurioAutonomo exista |
| Criterio de aceptación | La prueba debe regresar verdadero si el UsuarioAutonomo está registrado en el sistema o un valor falso si es lo contrario |

|  |  |
| --- | --- |
| Método | testObtenerTodosLosAlumnosInscritos |
| Propósito | Obtener todos los usuarios autónomos que estén inscritos a alguna ExperienciaEducativa |
| Criterio de aceptación | La prueba debe regresar una lista con objetos del tipo UsuarioAutonomo |

|  |  |
| --- | --- |
| Método | testValidarExistenciaUsuarioAutonomoConInscripcion |
| Propósito | Validar que un UsuarioAutonomo dado su matrícula esté inscrito en una ExperienciaEducativa |
| Criterio de aceptación | La prueba debe regresar verdadero si el UsuarioAutonomo tiene alguna inscrpción o un valor falso si es lo contrario |

|  |  |
| --- | --- |
| Método | testObtenerUsuarioAutonomoPorMatricula |
| Propósito | Obtener la información de un UsuarioAutonomo dado su matrícula |
| Criterio de aceptación | La prueba debe regresar un objeto de la clase UsuarioAutonomo |

# **Conclusiones.**

En el proceso de este documento se mostraron los distintos artefactos que ayudaron a lo largo de todo el desarrollo de las funcionalidades programadas a obtener un mejor entendimiento de la problemática desea sistema que se desea implementar.

Podemos concluir que a pesar de tener un diseño acerca del sistema durante la implementación van surgiendo nuevos cambios que se van necesitando y así el sistema va encontrando mejoras notables a partir de lo que se tenía pensado.

# **Referencias.**

(1993-2016). Java™ Platform, Standard Edition 7 API Specification. Parkway, Redwood City, U.S.A. <https://docs.oracle.com/javase/7/docs/api>.

2017. Mocrosoft.U.S.A.https://technet.microsoft.com/es-es/cc512759.aspx

2017. Oracle.Using AUTO\_INCREMENT.U.S.A. <https://www.mysql.com/about/contact/>