

**JAVA**

TRABAJO PRÁCTICO - GIMNASIO

# **Comisión**: 3ELEC2

**Integrante**:

|  |  |
| --- | --- |
| Legajo | Apellido y Nombre |
| 46472 | Raselli Gianfranco |

**Información**

**Link repositorio del código fuente**: <https://github.com/GianfrancoRaselli/TP-GIMNASIO-JAVA>

**Link página de Inicio de la aplicación web**: <http://gimnasio-java.jelastic.saveincloud.net/TP-Gimnasio/Inicio.jsp>

**Credenciales de usuarios**:

Nivel de usuario: administrador:

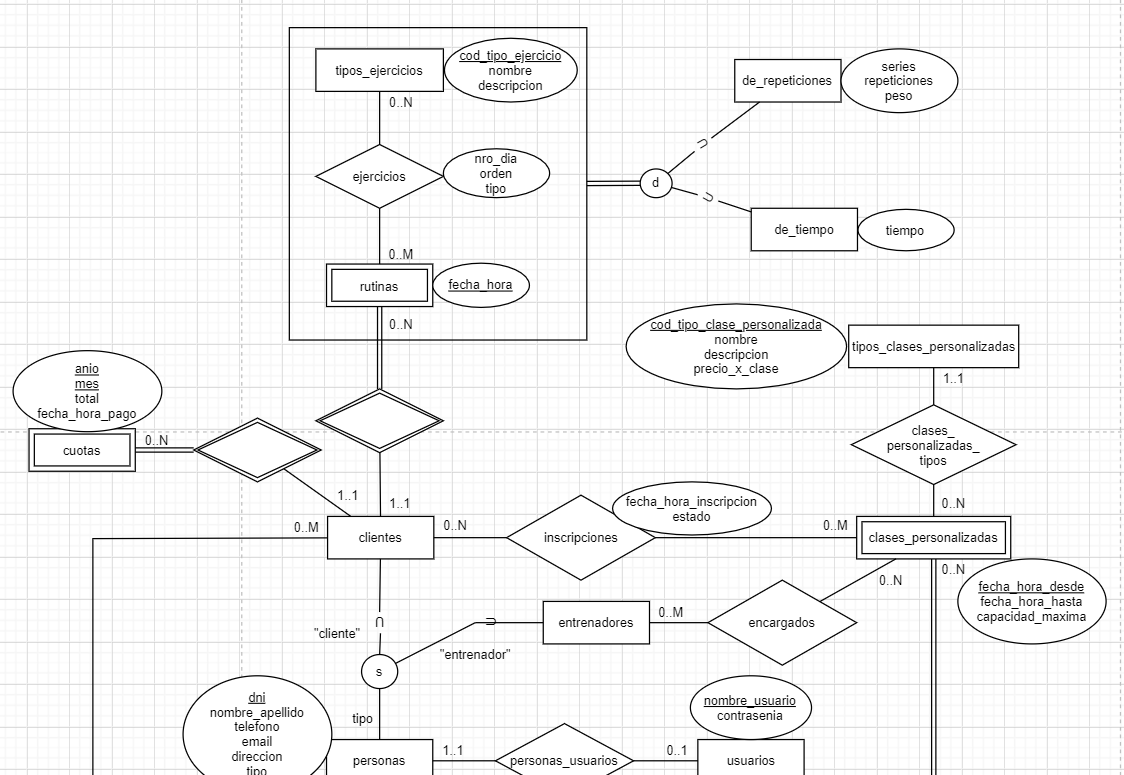
* Nombre de usuario: Administrador
* Contraseña: Administrador

Nivel de usuario: usuario:

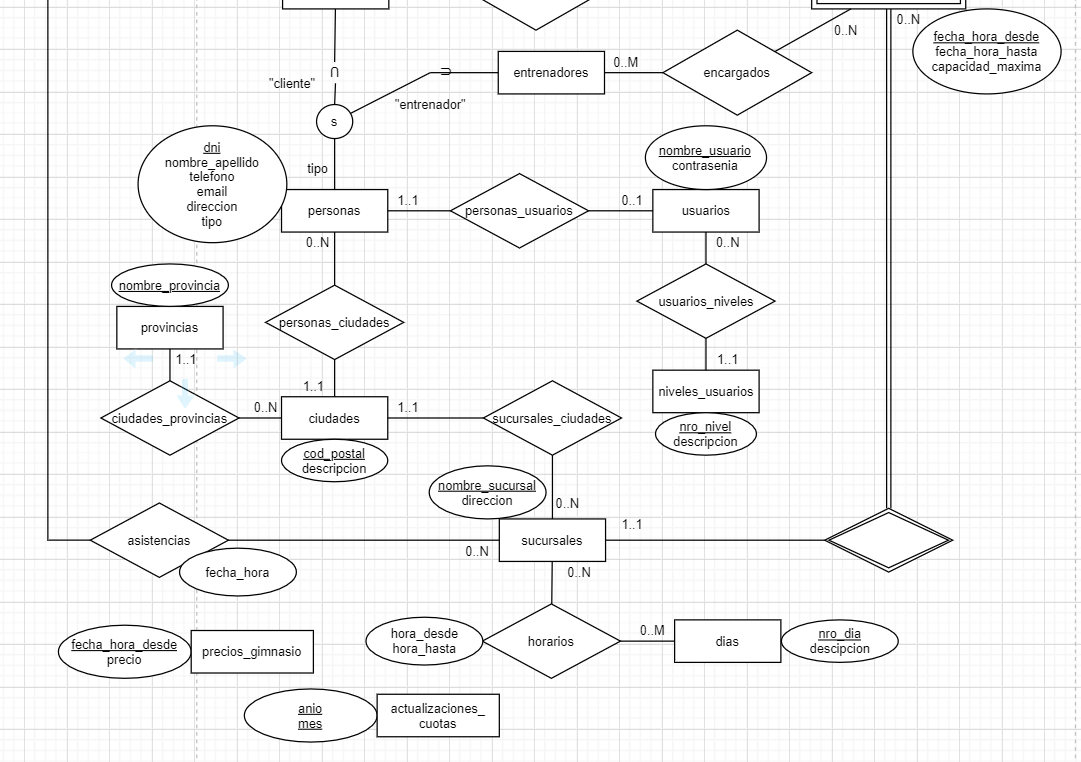
* Nombre de usuario: Usuario
* Contraseña: Usuario

**Diagrama Entidad-Relación**

**Parte A**



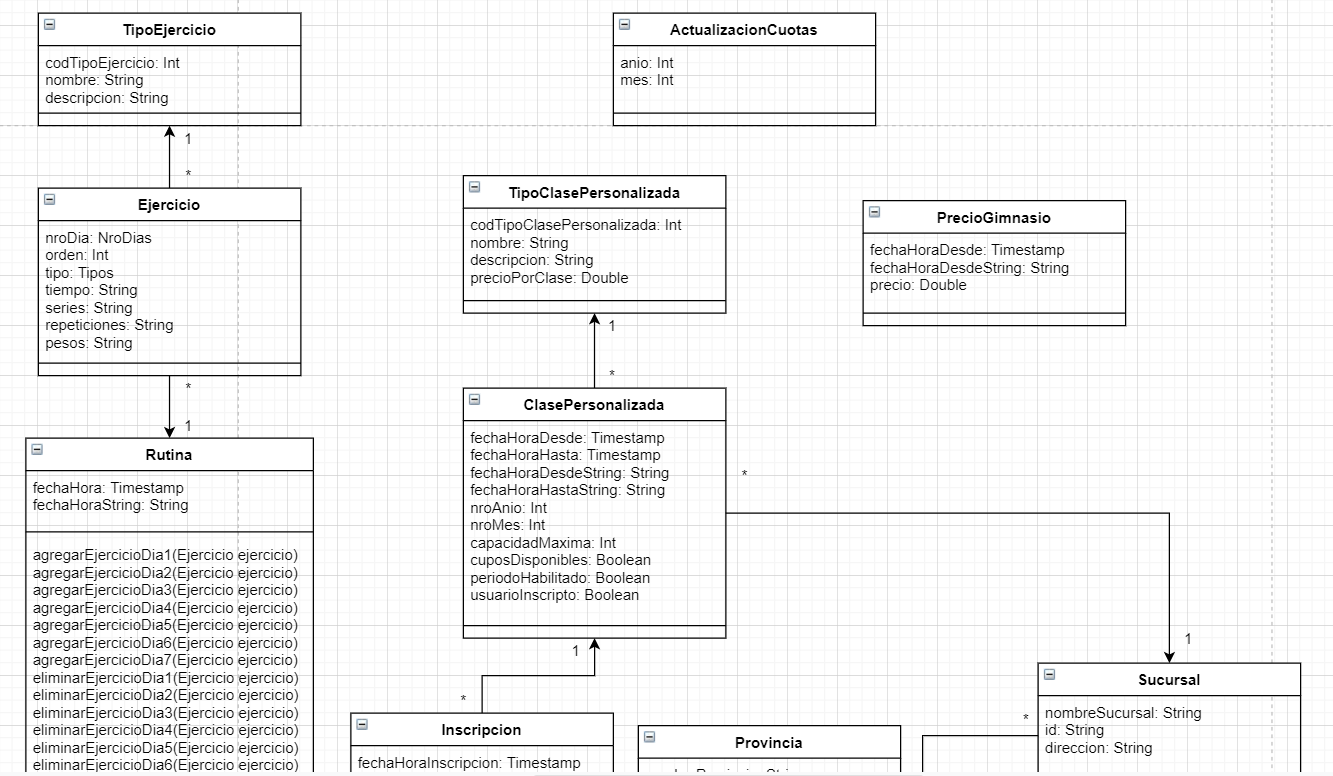
**Parte B**



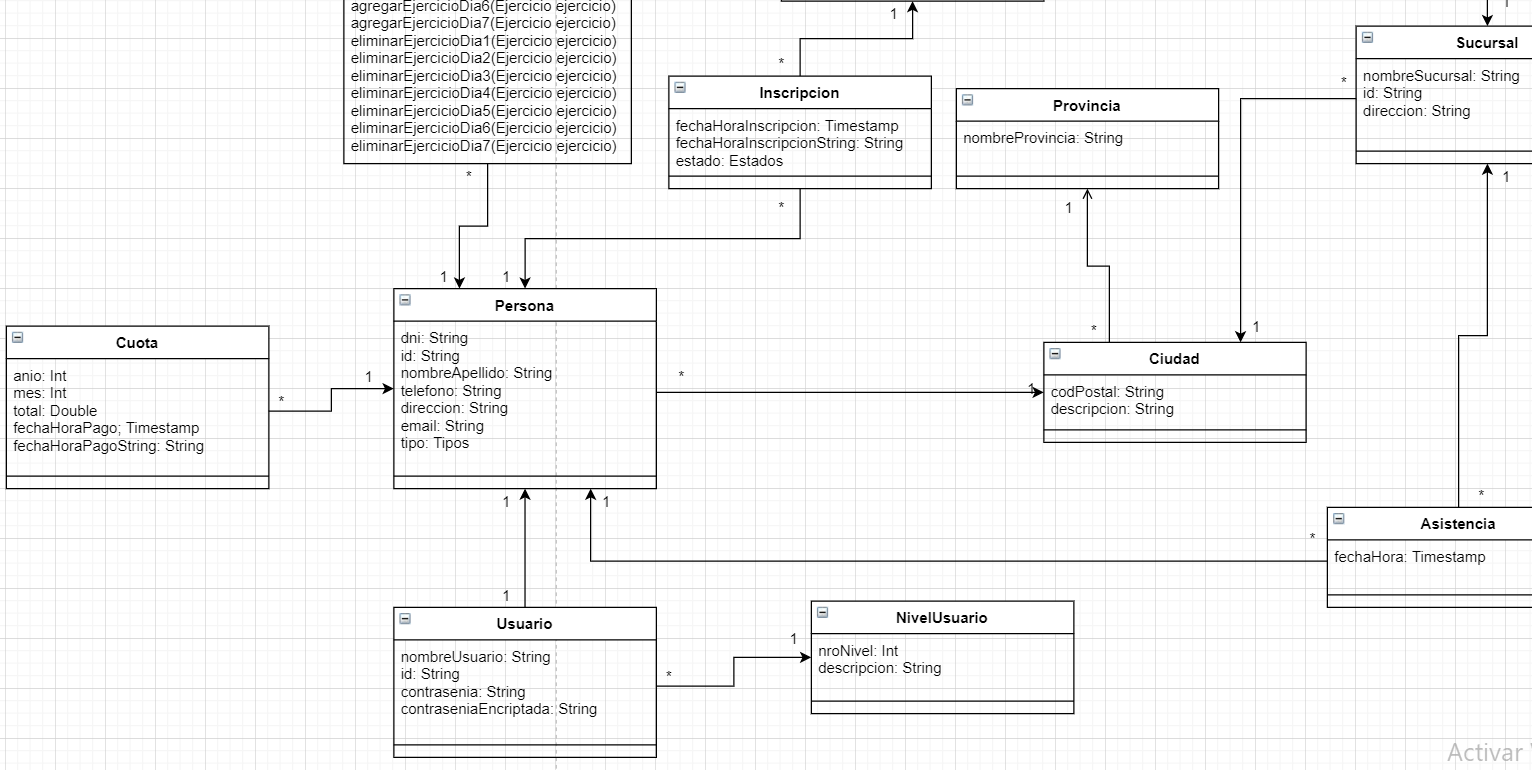
**Diagrama de Clases**

Todas las clases tienen los métodos Getters y Setters de sus atributos correspondientes. Además, poseen un constructor genérico sin parámetros; como también, los métodos toString(), hashCode() y equals() . Por lo tanto, no se incluyeron en el diagrama de clases por ser genéricos en todas ellas.

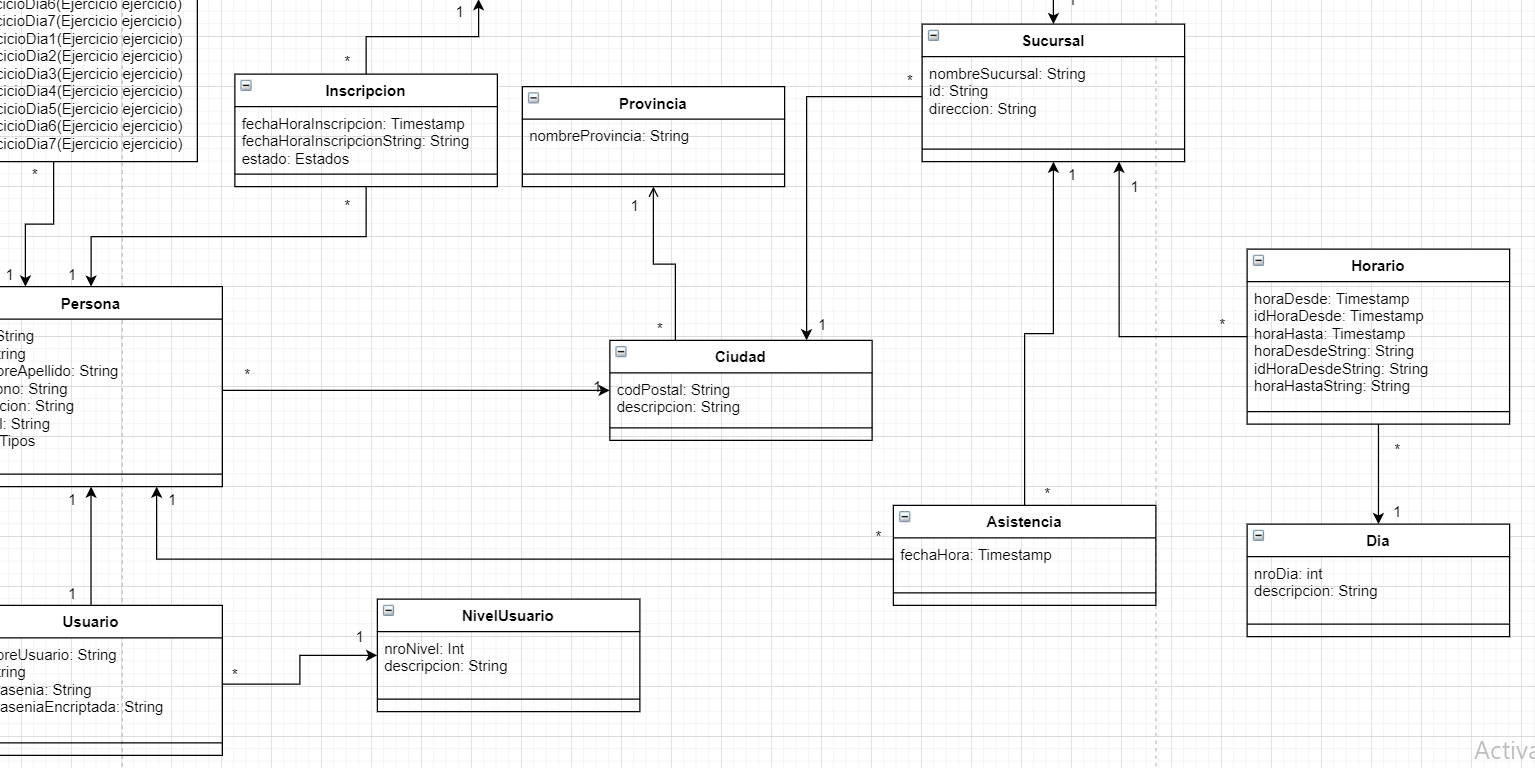
**Parte A**



**Parte B**



**Parte C**



**Script de Base de Datos**

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

-- -----------------------------------------------------

-- Schema gimnasio

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema gimnasio

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `gimnasio` DEFAULT CHARACTER SET utf8 ;

USE `gimnasio` ;

-- -----------------------------------------------------

-- Table `gimnasio`.`precios\_gimnasio`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`precios\_gimnasio` (

`fecha\_hora\_desde` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`precio` REAL NOT NULL,

PRIMARY KEY (`fecha\_hora\_desde`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`dias`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`dias` (

`nro\_dia` INT NOT NULL AUTO\_INCREMENT,

`descripcion` VARCHAR(30) NOT NULL,

PRIMARY KEY (`nro\_dia`),

UNIQUE INDEX `descripcion\_UNIQUE` (`descripcion` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`provincias`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`provincias` (

`nombre\_provincia` VARCHAR(30) NOT NULL,

PRIMARY KEY (`nombre\_provincia`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`ciudades`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`ciudades` (

`cod\_postal` VARCHAR(15) NOT NULL,

`descripcion` VARCHAR(30) NOT NULL,

`nombre\_provincia` VARCHAR(30) NOT NULL,

PRIMARY KEY (`cod\_postal`),

UNIQUE INDEX `descipcion\_UNIQUE` (`descripcion` ASC),

INDEX `fk\_ciudades\_provincias1\_idx` (`nombre\_provincia` ASC),

CONSTRAINT `fk\_ciudades\_provincias1`

FOREIGN KEY (`nombre\_provincia`)

REFERENCES `gimnasio`.`provincias` (`nombre\_provincia`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`sucursales`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`sucursales` (

`nombre\_sucursal` VARCHAR(30) NOT NULL,

`direccion` VARCHAR(30) NOT NULL,

`cod\_postal` VARCHAR(15) NOT NULL,

PRIMARY KEY (`nombre\_sucursal`),

INDEX `fk\_sucursales\_ciudades1\_idx` (`cod\_postal` ASC),

CONSTRAINT `fk\_sucursales\_ciudades1`

FOREIGN KEY (`cod\_postal`)

REFERENCES `gimnasio`.`ciudades` (`cod\_postal`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`horarios`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`horarios` (

`nombre\_sucursal` VARCHAR(30) NOT NULL,

`nro\_dia` INT NOT NULL,

`hora\_desde` TIME NOT NULL,

`hora\_hasta` TIME NOT NULL,

PRIMARY KEY (`nombre\_sucursal`, `nro\_dia`, `hora\_desde`),

INDEX `fk\_sucursales\_has\_dias\_dias1\_idx` (`nro\_dia` ASC),

INDEX `fk\_sucursales\_has\_dias\_sucursales\_idx` (`nombre\_sucursal` ASC),

CONSTRAINT `fk\_sucursales\_has\_dias\_sucursales`

FOREIGN KEY (`nombre\_sucursal`)

REFERENCES `gimnasio`.`sucursales` (`nombre\_sucursal`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `fk\_sucursales\_has\_dias\_dias1`

FOREIGN KEY (`nro\_dia`)

REFERENCES `gimnasio`.`dias` (`nro\_dia`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`niveles\_usuarios`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`niveles\_usuarios` (

`nro\_nivel` INT NOT NULL AUTO\_INCREMENT,

`descripcion` VARCHAR(15) NOT NULL,

PRIMARY KEY (`nro\_nivel`),

UNIQUE INDEX `descripcion\_UNIQUE` (`descripcion` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`personas`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`personas` (

`dni` VARCHAR(20) NOT NULL,

`nombre\_apellido` VARCHAR(45) NOT NULL,

`telefono` VARCHAR(30) NOT NULL,

`email` VARCHAR(30) NOT NULL,

`direccion` VARCHAR(30) NOT NULL,

`tipo` INT NOT NULL,

`cod\_postal` VARCHAR(15) NOT NULL,

PRIMARY KEY (`dni`),

INDEX `fk\_personas\_ciudades1\_idx` (`cod\_postal` ASC),

CONSTRAINT `fk\_personas\_ciudades1`

FOREIGN KEY (`cod\_postal`)

REFERENCES `gimnasio`.`ciudades` (`cod\_postal`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`usuarios`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`usuarios` (

`nombre\_usuario` VARCHAR(20) NOT NULL,

`contrasenia` VARCHAR(100) NOT NULL,

`nro\_nivel` INT NOT NULL,

`dni` VARCHAR(20) NOT NULL,

PRIMARY KEY (`nombre\_usuario`),

INDEX `fk\_usuarios\_niveles\_usuarios1\_idx` (`nro\_nivel` ASC),

INDEX `fk\_usuarios\_personas1\_idx` (`dni` ASC),

CONSTRAINT `fk\_usuarios\_niveles\_usuarios1`

FOREIGN KEY (`nro\_nivel`)

REFERENCES `gimnasio`.`niveles\_usuarios` (`nro\_nivel`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `fk\_usuarios\_personas1`

FOREIGN KEY (`dni`)

REFERENCES `gimnasio`.`personas` (`dni`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`asistencias`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`asistencias` (

`fecha\_hora` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`dni` VARCHAR(20) NOT NULL,

`nombre\_sucursal` VARCHAR(30) NOT NULL,

PRIMARY KEY (`fecha\_hora`, `dni`, `nombre\_sucursal`),

INDEX `fk\_asistencias\_personas1\_idx` (`dni` ASC),

INDEX `fk\_asistencias\_sucursales1\_idx` (`nombre\_sucursal` ASC),

CONSTRAINT `fk\_asistencias\_personas1`

FOREIGN KEY (`dni`)

REFERENCES `gimnasio`.`personas` (`dni`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `fk\_asistencias\_sucursales1`

FOREIGN KEY (`nombre\_sucursal`)

REFERENCES `gimnasio`.`sucursales` (`nombre\_sucursal`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`cuotas`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`cuotas` (

`anio` INT NOT NULL,

`mes` INT NOT NULL,

`total` REAL NOT NULL,

`fecha\_hora\_pago` TIMESTAMP NULL,

`dni` VARCHAR(20) NOT NULL,

PRIMARY KEY (`anio`, `mes`, `dni`),

INDEX `fk\_cuotas\_personas1\_idx` (`dni` ASC),

CONSTRAINT `fk\_cuotas\_personas1`

FOREIGN KEY (`dni`)

REFERENCES `gimnasio`.`personas` (`dni`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`tipos\_clases\_personalizadas`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`tipos\_clases\_personalizadas` (

`cod\_tipo\_clase\_personalizada` INT NOT NULL AUTO\_INCREMENT,

`nombre` VARCHAR(20) NOT NULL,

`descripcion` VARCHAR(70) NULL,

`precio\_x\_clase` REAL NULL,

PRIMARY KEY (`cod\_tipo\_clase\_personalizada`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`clases\_personalizadas`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`clases\_personalizadas` (

`nombre\_sucursal` VARCHAR(30) NOT NULL,

`fecha\_hora\_desde` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`fecha\_hora\_hasta` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`capacidad\_maxima` INT NOT NULL,

`cod\_tipo\_clase\_personalizada` INT NOT NULL,

PRIMARY KEY (`nombre\_sucursal`, `fecha\_hora\_desde`),

INDEX `fk\_clases\_personalizadas\_tipos\_clases\_personalizadas1\_idx` (`cod\_tipo\_clase\_personalizada` ASC),

CONSTRAINT `fk\_clases\_personalizadas\_sucursales1`

FOREIGN KEY (`nombre\_sucursal`)

REFERENCES `gimnasio`.`sucursales` (`nombre\_sucursal`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `fk\_clases\_personalizadas\_tipos\_clases\_personalizadas1`

FOREIGN KEY (`cod\_tipo\_clase\_personalizada`)

REFERENCES `gimnasio`.`tipos\_clases\_personalizadas` (`cod\_tipo\_clase\_personalizada`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`encargados`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`encargados` (

`nombre\_sucursal` VARCHAR(30) NOT NULL,

`fecha\_hora\_desde` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`dni` VARCHAR(20) NOT NULL,

PRIMARY KEY (`nombre\_sucursal`, `fecha\_hora\_desde`, `dni`),

INDEX `fk\_clases\_personalizadas\_has\_personas\_personas1\_idx` (`dni` ASC),

INDEX `fk\_clases\_personalizadas\_has\_personas\_clases\_personalizadas\_idx` (`nombre\_sucursal` ASC, `fecha\_hora\_desde` ASC),

CONSTRAINT `fk\_clases\_personalizadas\_has\_personas\_clases\_personalizadas1`

FOREIGN KEY (`nombre\_sucursal` , `fecha\_hora\_desde`)

REFERENCES `gimnasio`.`clases\_personalizadas` (`nombre\_sucursal` , `fecha\_hora\_desde`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_clases\_personalizadas\_has\_personas\_personas1`

FOREIGN KEY (`dni`)

REFERENCES `gimnasio`.`personas` (`dni`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`inscripciones`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`inscripciones` (

`dni` VARCHAR(20) NOT NULL,

`estado` INT NOT NULL,

`fecha\_hora\_inscripcion` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`nombre\_sucursal` VARCHAR(30) NOT NULL,

`fecha\_hora\_desde` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

PRIMARY KEY (`dni`, `fecha\_hora\_inscripcion`),

INDEX `fk\_personas\_has\_clases\_personalizadas\_personas1\_idx` (`dni` ASC),

INDEX `fk\_inscripciones\_clases\_personalizadas1\_idx` (`nombre\_sucursal` ASC, `fecha\_hora\_desde` ASC),

CONSTRAINT `fk\_personas\_has\_clases\_personalizadas\_personas1`

FOREIGN KEY (`dni`)

REFERENCES `gimnasio`.`personas` (`dni`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `fk\_inscripciones\_clases\_personalizadas1`

FOREIGN KEY (`nombre\_sucursal` , `fecha\_hora\_desde`)

REFERENCES `gimnasio`.`clases\_personalizadas` (`nombre\_sucursal` , `fecha\_hora\_desde`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`tipos\_ejercicios`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`tipos\_ejercicios` (

`cod\_tipo\_ejercicio` INT NOT NULL AUTO\_INCREMENT,

`nombre` VARCHAR(20) NOT NULL,

`descripcion` VARCHAR(70) NULL,

PRIMARY KEY (`cod\_tipo\_ejercicio`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`rutinas`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`rutinas` (

`fecha\_hora` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`dni` VARCHAR(20) NOT NULL,

PRIMARY KEY (`dni`, `fecha\_hora`),

INDEX `fk\_rutina\_personas1\_idx` (`dni` ASC),

CONSTRAINT `fk\_rutina\_personas1`

FOREIGN KEY (`dni`)

REFERENCES `gimnasio`.`personas` (`dni`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`ejercicios`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`ejercicios` (

`orden` INT NOT NULL,

`tipo` VARCHAR(20) NOT NULL,

`tiempo` VARCHAR(20) NULL,

`cod\_tipo\_ejercicio` INT NOT NULL,

`dni` VARCHAR(20) NOT NULL,

`fecha\_hora` TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`series` VARCHAR(20) NULL,

`repeticiones` VARCHAR(45) NULL,

`pesos` VARCHAR(45) NULL,

`nro\_dia` INT NOT NULL,

PRIMARY KEY (`orden`, `dni`, `fecha\_hora`, `nro\_dia`),

INDEX `fk\_ejercicios\_tipos\_ejercicios1\_idx` (`cod\_tipo\_ejercicio` ASC),

INDEX `fk\_ejercicios\_rutinas1\_idx` (`dni` ASC, `fecha\_hora` ASC),

CONSTRAINT `fk\_ejercicios\_tipos\_ejercicios1`

FOREIGN KEY (`cod\_tipo\_ejercicio`)

REFERENCES `gimnasio`.`tipos\_ejercicios` (`cod\_tipo\_ejercicio`)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT `fk\_ejercicios\_rutinas1`

FOREIGN KEY (`dni` , `fecha\_hora`)

REFERENCES `gimnasio`.`rutinas` (`dni` , `fecha\_hora`)

ON DELETE CASCADE

ON UPDATE CASCADE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `gimnasio`.`actualizaciones\_cuotas`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `gimnasio`.`actualizaciones\_cuotas` (

`anio` INT NOT NULL,

`mes` INT NOT NULL,

PRIMARY KEY (`anio`, `mes`))

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

**Casos de Uso**

**Código y Nombre del CASO DE USO: CURSR1 – Realizar Rutina**

**Dimensiones de clasificación:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Resumen | Estructurado | Sistema | Negra | Real | Semántica |

**Meta del CASO DE USO:** Realizar la rutina del día.

**Actor Primario:** Cliente. **Otros:** Administrativo.

**PRECONDICIONES (de sistema):**El cliente posee un usuario. El cliente tiene rutinas registradas.

**DISPARADOR:**  El cliente quiere asistir a la sucursal más cercana a realizar la rutina.

**CAMINO BÁSICO:**

|  |  |
| --- | --- |
| 1 | El cliente inicia sesión invocando el CUU1.1 Iniciar Sesion |
| 2 | El cliente busca la sucursal más cercana invocando el CUU1.2 Buscar Sucursal |
| 3 | El Administrativo valida si el cliente adeuda cuotas invocando el CUU1.3 Validar Entrada |
| 4 | El cliente busca su rutina actual invocando el CUU1.4 Buscar Rutina |

**CAMINOS ALTERNATIVOS:**

|  |  |  |  |
| --- | --- | --- | --- |
| 3.a | El cliente paga las cuotas que adeuda | 3.a.1 El administrativo cobra las cuotas invocando el CUU1.5 Pagar Cuotas | Vuelve al paso 4 |

**POSTCONDICIONES (de sistema):**

**Éxito:**El cliente busca su rutina.

**Fracaso:** El cliente no puede ingresar por adeudar cuotas.

**Código y Nombre del CASO DE USO: CUU1 – Iniciar Sesión**

**Dimensiones de clasificación:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Usuario | Sin estructurar | Sistema | Negra | Real | Semántica |

**Meta del CASO DE USO:** Ingresar al sistema.

**Actor Primario:** Cliente.

**PRECONDICIONES (de sistema):**El cliente posee un usuario.

**DISPARADOR:**  El cliente quiere ingresar al sistema.

**CAMINO BÁSICO:**

|  |  |
| --- | --- |
| 1 | El cliente ingresa nombre de usuario y contraseña. El sistema valida los datos contra la base de datos y le permite ingresar. |

**CAMINOS ALTERNATIVOS:**

|  |  |  |  |
| --- | --- | --- | --- |
| 1.a | Los datos ingresados son incorrectos | 1.a.1 El sistema informa el error | Vuelve al paso 1 |

**POSTCONDICIONES (de sistema):**

**Éxito:**El cliente ingresa al sistema.

**Fracaso:** El cliente no puede ingresar al sistema.

**Código y Nombre del CASO DE USO: CUU3 – Validar Entrada**

**Dimensiones de clasificación:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Usuario | Sin estructurar | Sistema | Negra | Real | Semántica |

**Meta del CASO DE USO:** Ingresar al gimnasio.

**Actor Primario:** Cliente. **Otros:** Administrativo.

**PRECONDICIONES (de sistema):**El cliente posee un usuario.

**DISPARADOR:**  El cliente quiere ingresar al gimnasio.

**CAMINO BÁSICO:**

|  |  |
| --- | --- |
| 1 | El administrativo valida si el cliente tiene las cuotas al día. El sistema informa que puede ingresar. |

**CAMINOS ALTERNATIVOS:**

|  |  |  |  |
| --- | --- | --- | --- |
| 1.a | El cliente adeuda cuotas | 1.a.1 El sistema informa la situación | Fin del caso de uso |

**POSTCONDICIONES (de sistema):**

**Éxito:**El sistema informa que el cliente puede ingresar.

**Fracaso:** El sistema informa que el cliente no puede ingresar.

**Código y Nombre del CASO DE USO: CUU5 – Pagar cuotas**

**Dimensiones de clasificación:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nivel** | **Estructura** | **Alcance** | **Caja** | **Instanciación** | **Interacción** |
| Usuario | Sin estructurar | Sistema | Negra | Real | Semántica |

**Meta del CASO DE USO:** Ingresar al gimnasio.

**Actor Primario:** Cliente. **Otros:** Administrativo.

**PRECONDICIONES (de sistema):**El cliente posee un usuario.

**DISPARADOR:**  El cliente se presenta para pagar las cuotas.

**CAMINO BÁSICO:**

|  |  |
| --- | --- |
| 1 | El administrativo busca las cuotas que el cliente adeuda. El sistema informa. |
| 2 | El cliente paga la cuota, el administrativo marca la cuota como pagada. El sistema registra el pago. |

**CAMINOS ALTERNATIVOS:**

|  |  |  |  |
| --- | --- | --- | --- |
| 1.a | El cliente no adeuda cuotas | 1.a.1 El sistema informa la situación | Fin del caso de uso |

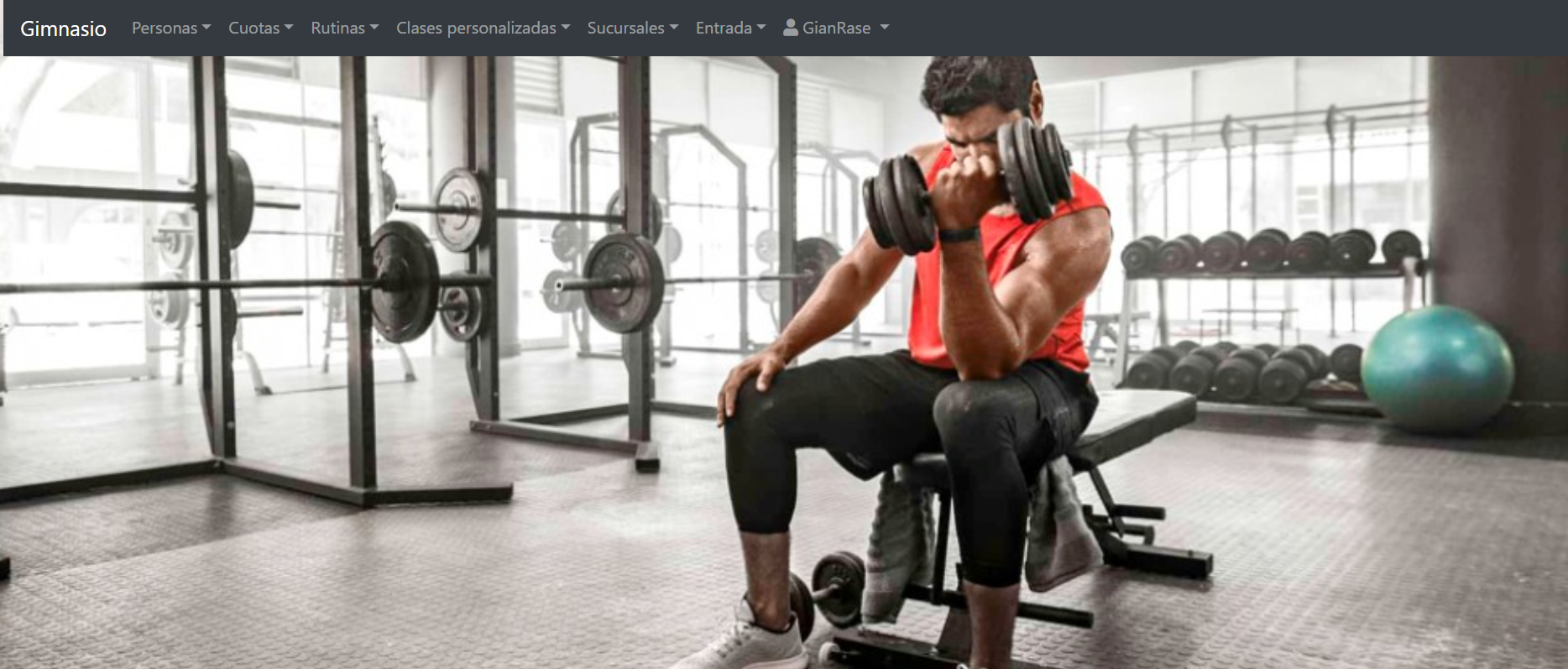
**POSTCONDICIONES (de sistema):**

**Éxito:**El sistema registra el pago de las cuotas.

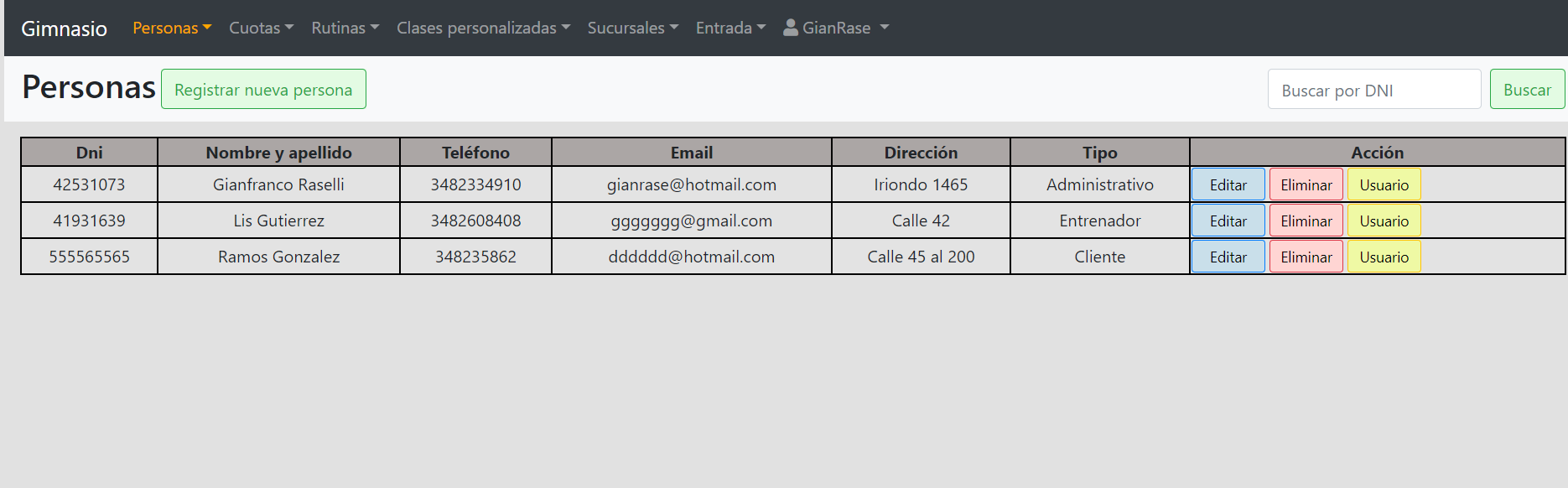
**Fracaso:** El sistema informa que el cliente no adeuda cuotas.

**Capturas de pantallas de la aplicación**

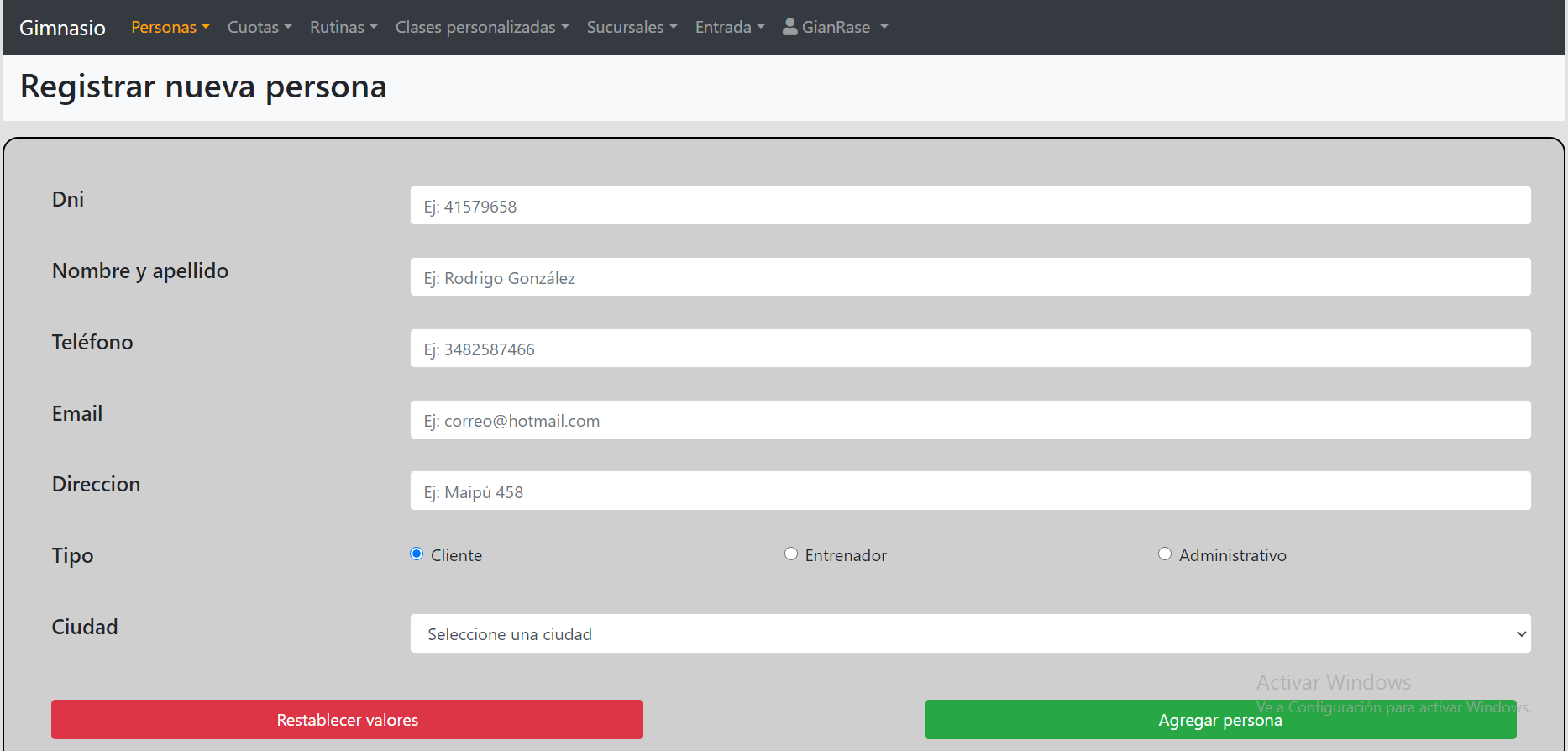
* Página de inicio



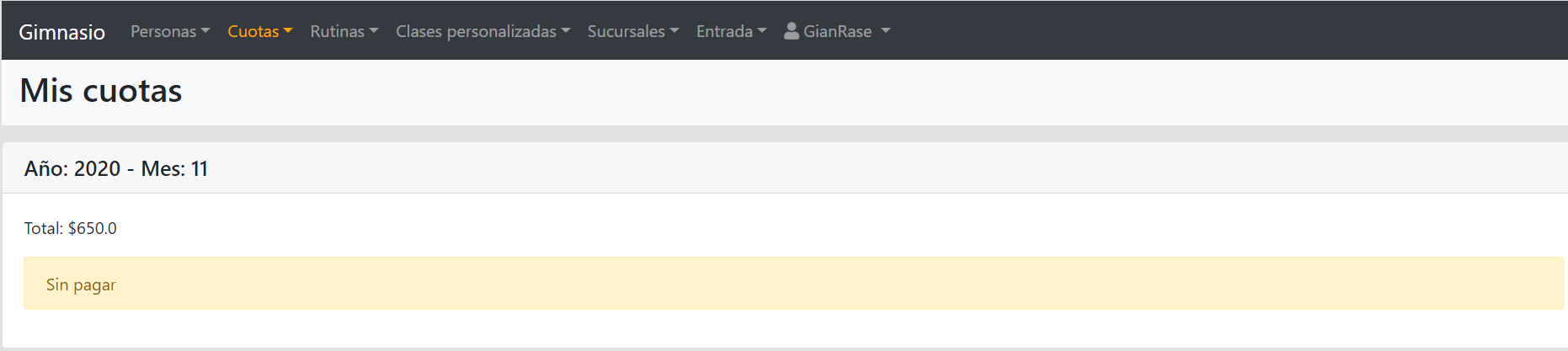
* Página de personas



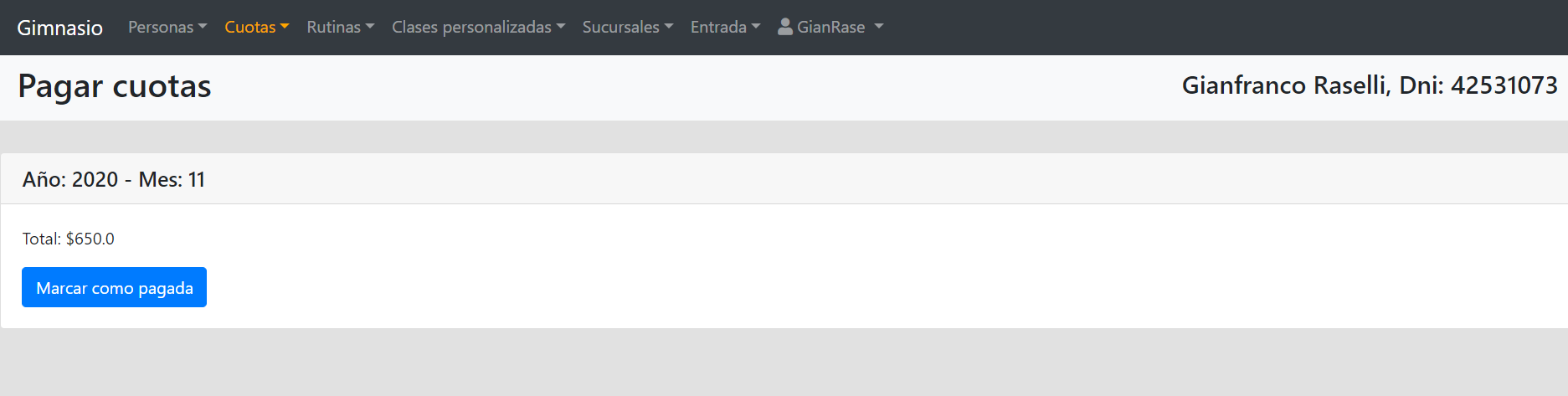
* Página para registrar una nueva persona



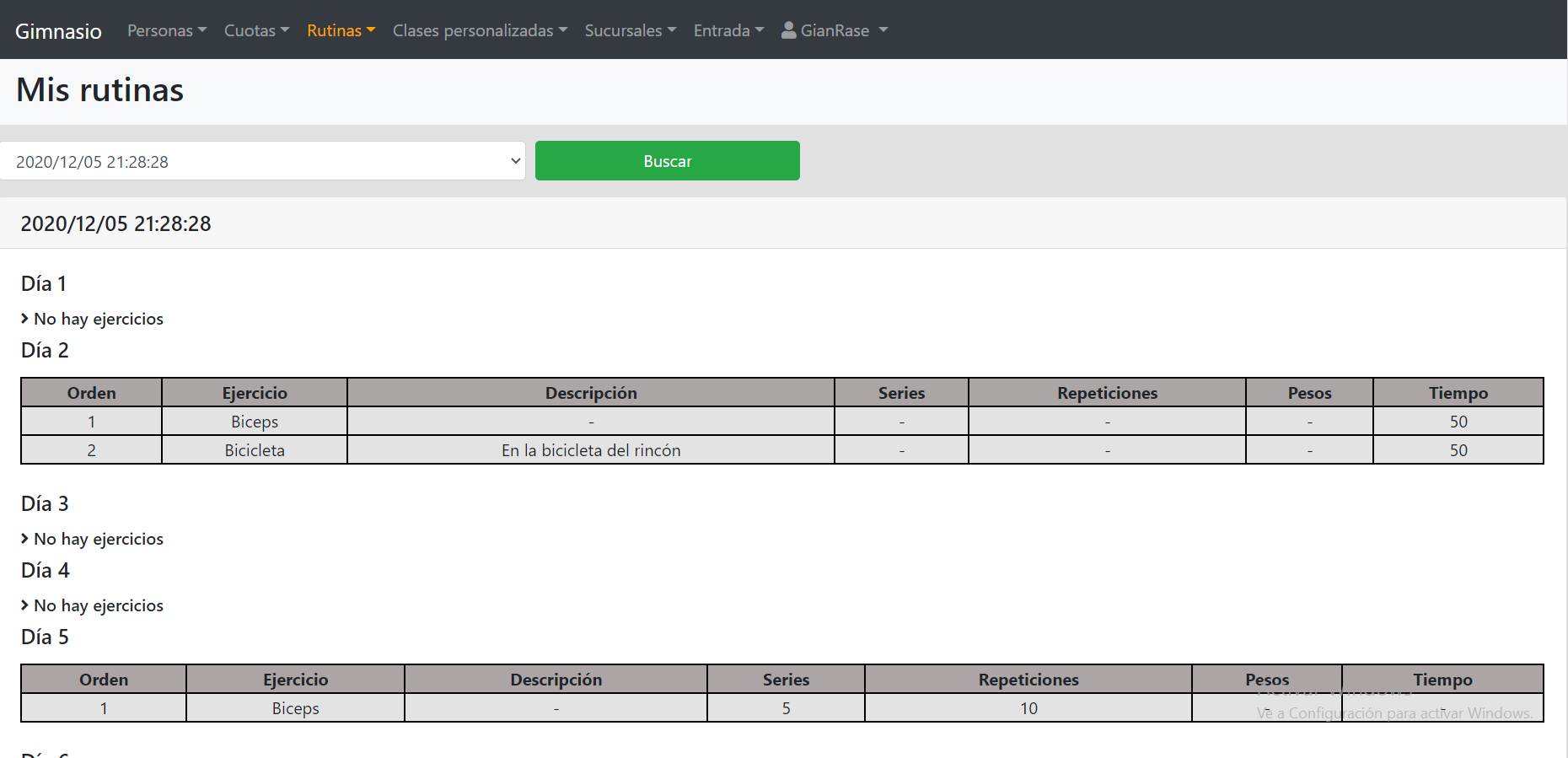
* Página de mis cuotas



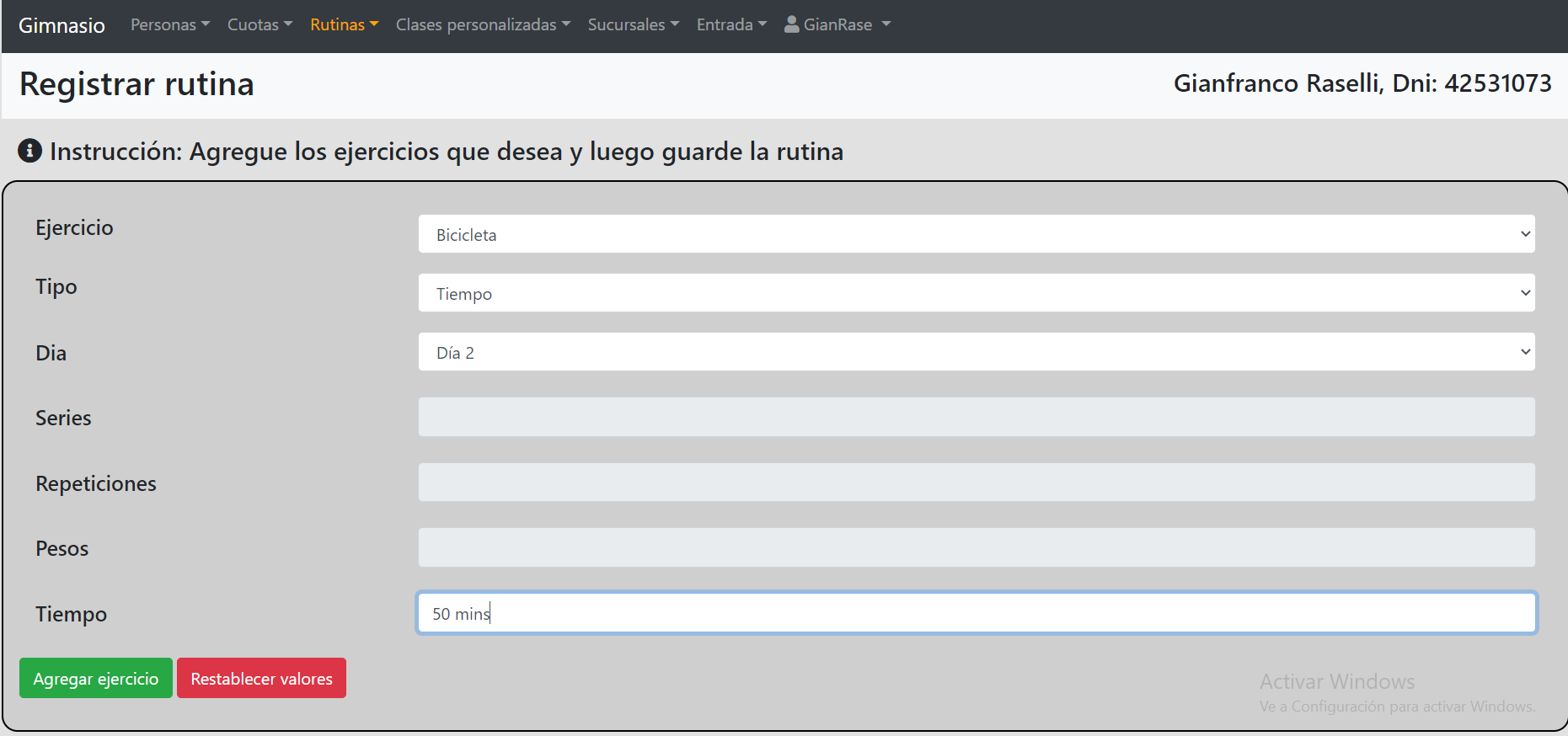
* Página para pagar cuotas



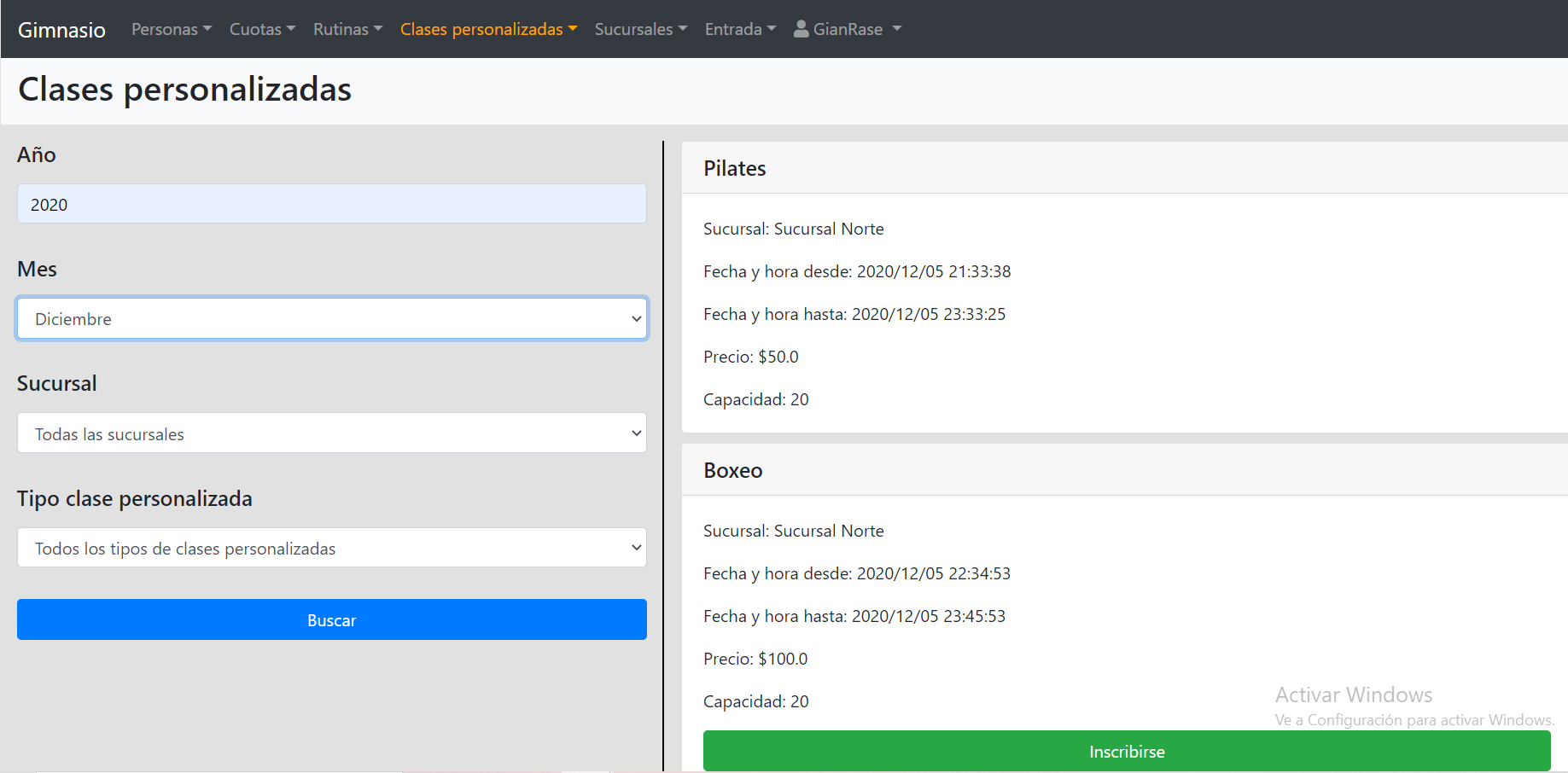
* Página de mis rutinas



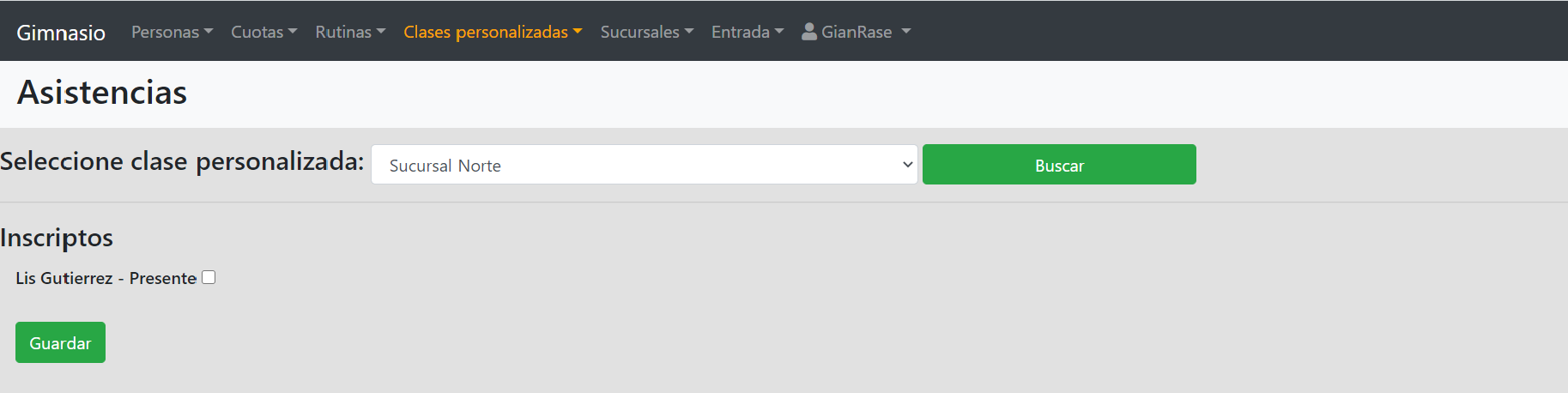
* Página para registrar una rutina



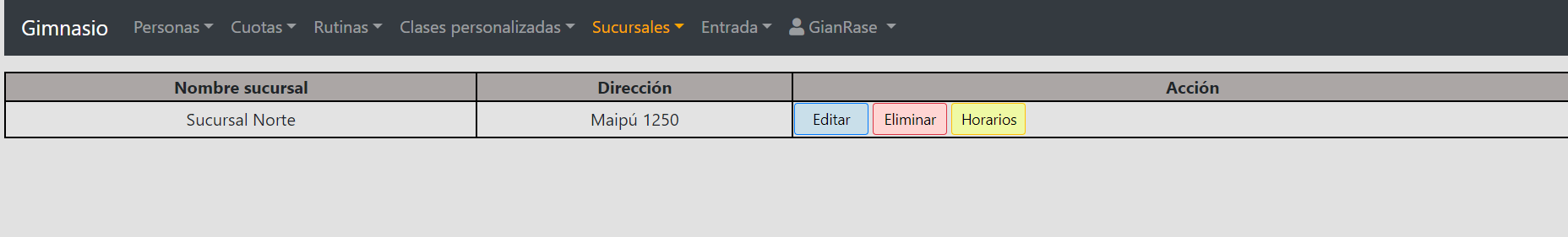
* Página para inscribirse a clases personalizadas



* Página para registrar asistencia a las clases personalizadas



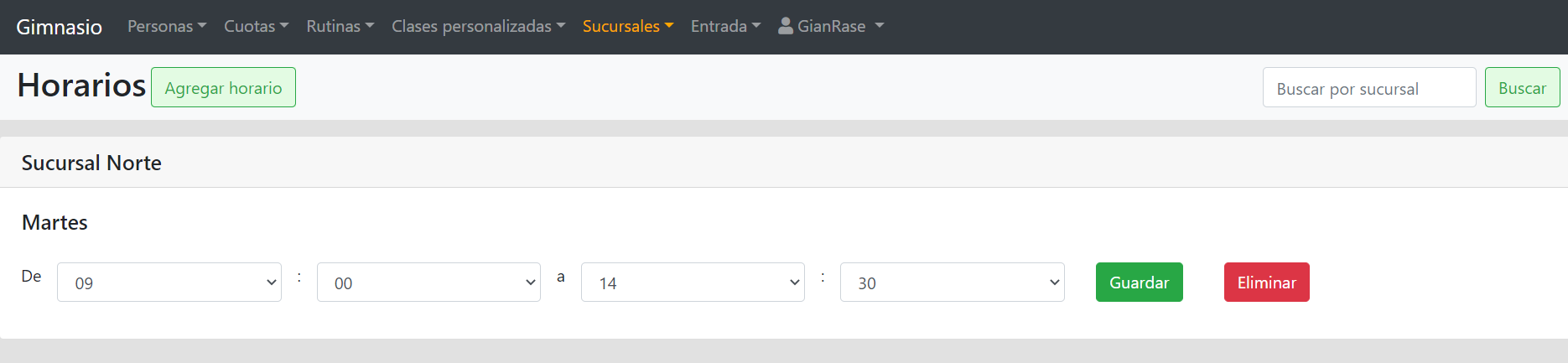
* Página de sucursales



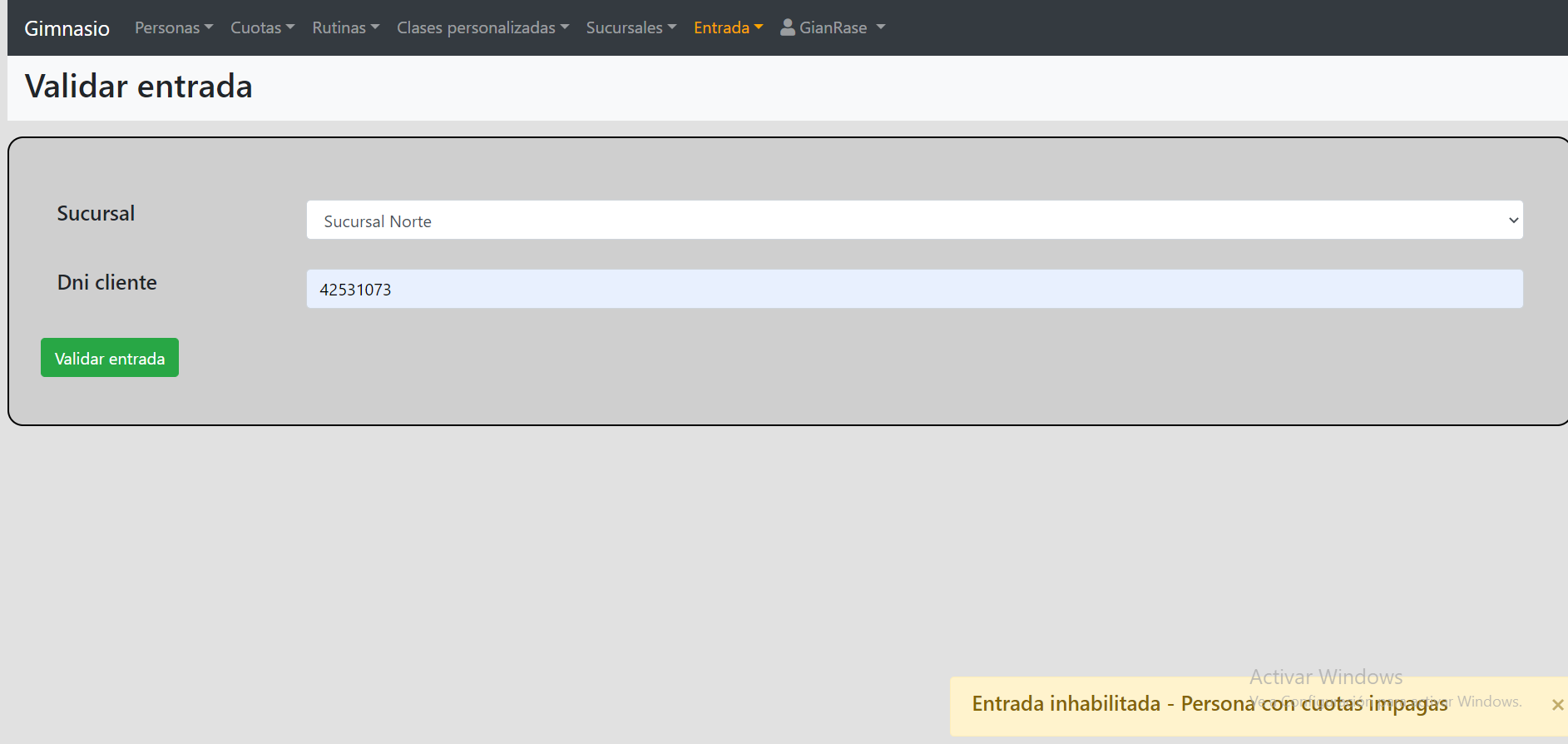
* Página para registrar una nueva sucursal



* Página para registrar horarios de sucursales



* Página para validar la entrada del cliente

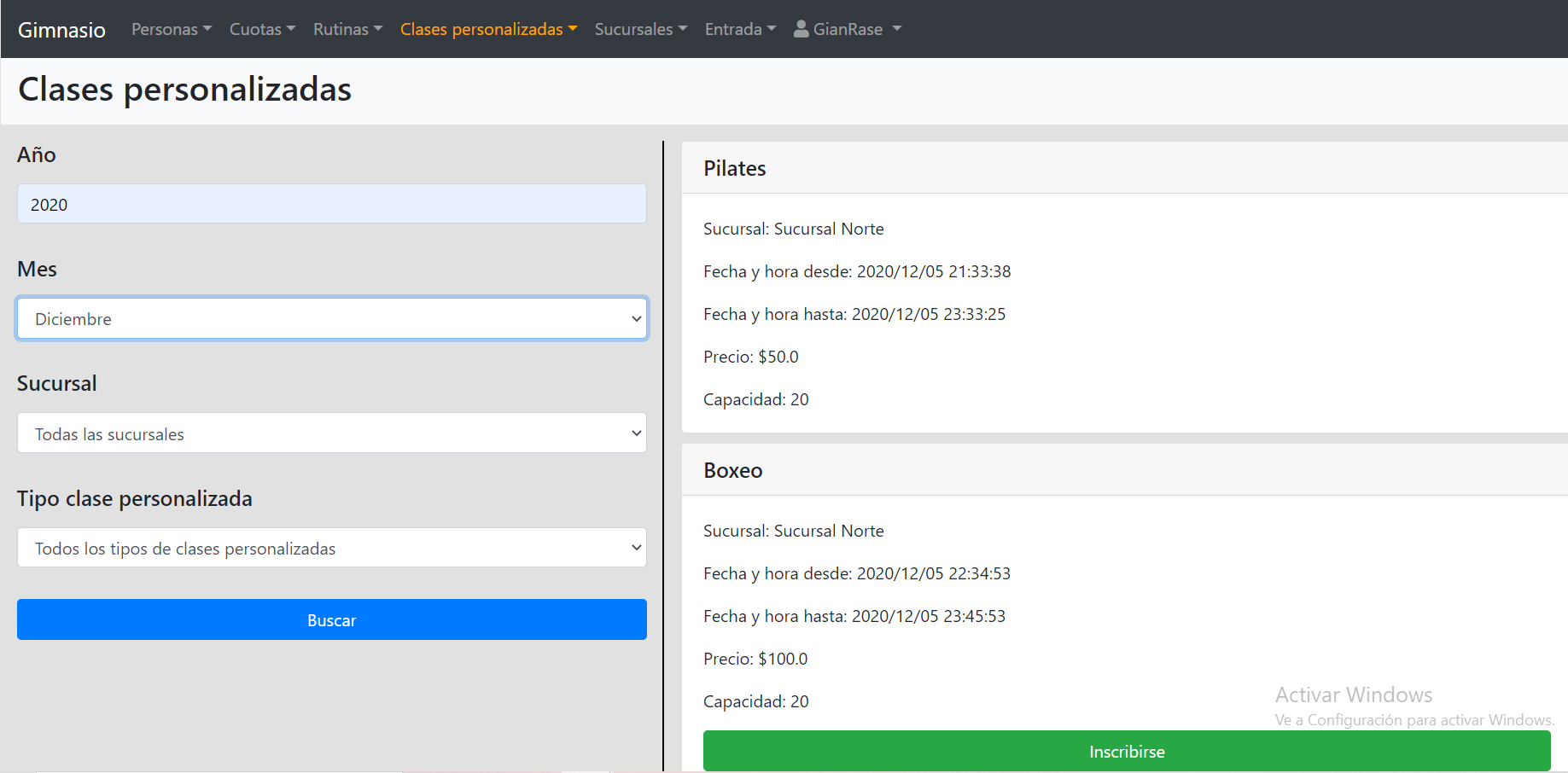


* Página de perfil

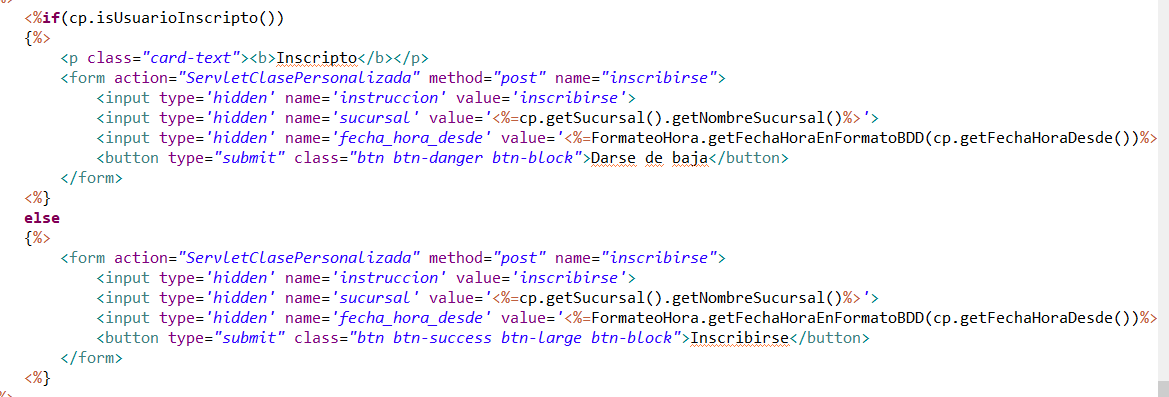


**Fragmentos de código**

**Caso de uso**: Inscribirse a una clase personalizada



El cliente puede buscar las clases personalizadas cambiando los parámetros de búsqueda, si la clase personalizada no empezó, le permitirá inscribirse, o en caso de ya estar inscripto, se podrá dar de baja.



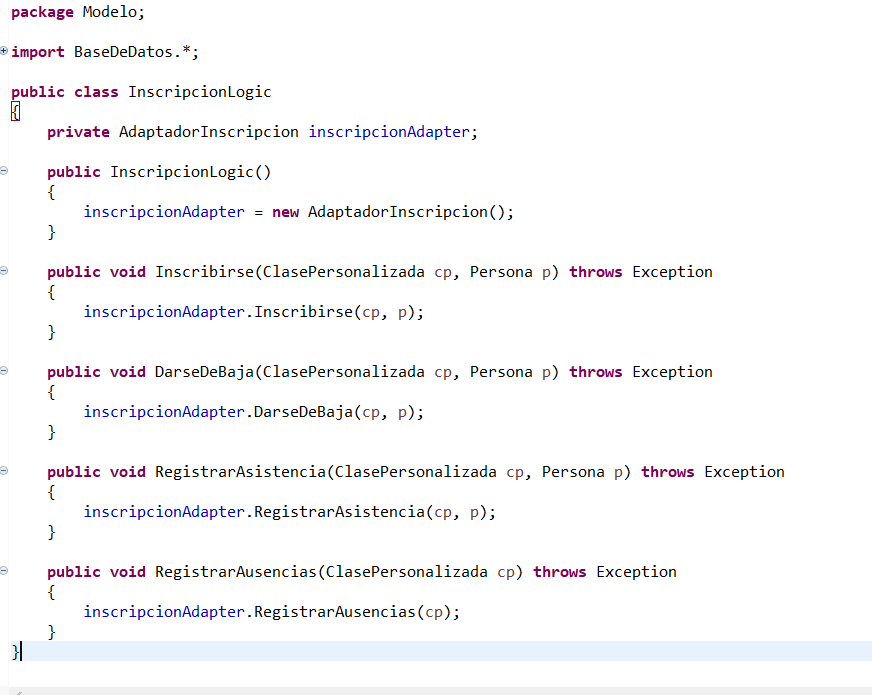
Acá vemos el código HTML en la página JSP, con los dos formularios que se envían al Servlet llamado “ServletClasePersonalizada”, dependiendo de si el cliente ya está inscripto o no manda cierta información sino manda otra.



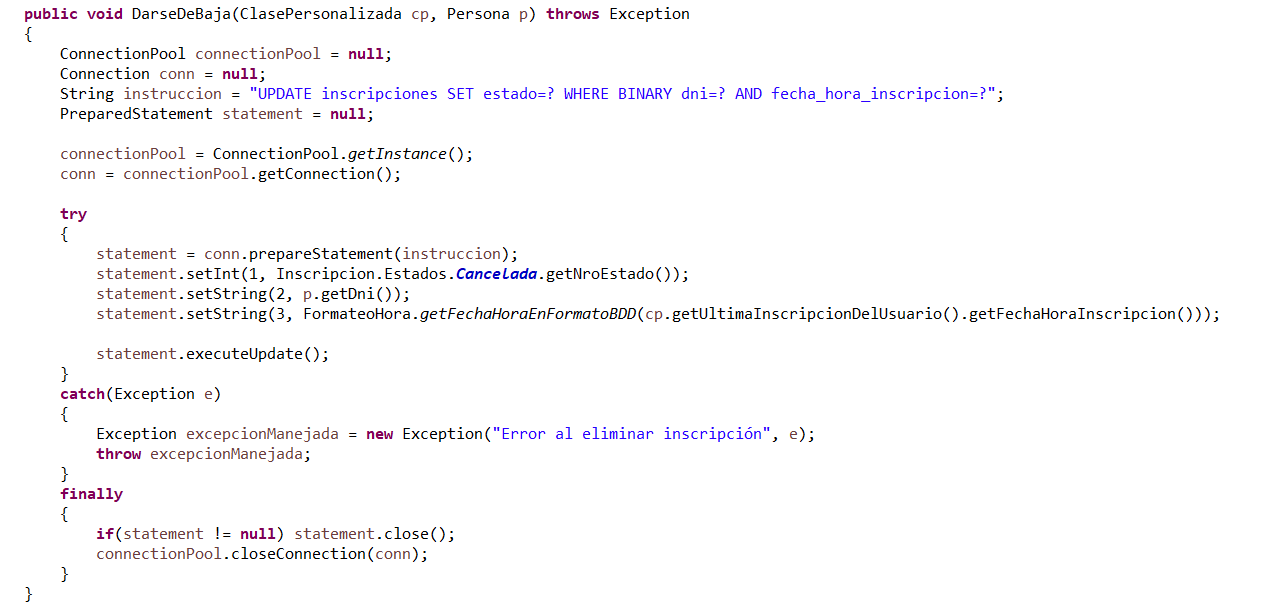
El Servlet en el método doPost recupera el parámetro “insctruccion”, que depende la información que contiene, redirige el flujo del programa a un método o a otro.



El método Inscribirse, recupera los datos de la clase personalizada, y de las inscripciones de dicho cliente a esta. Realiza algunas verificaciones y luego inscribe al cliente o lo da de baja en caso de estar habilitado, en caso contrario informa el inconveniente.

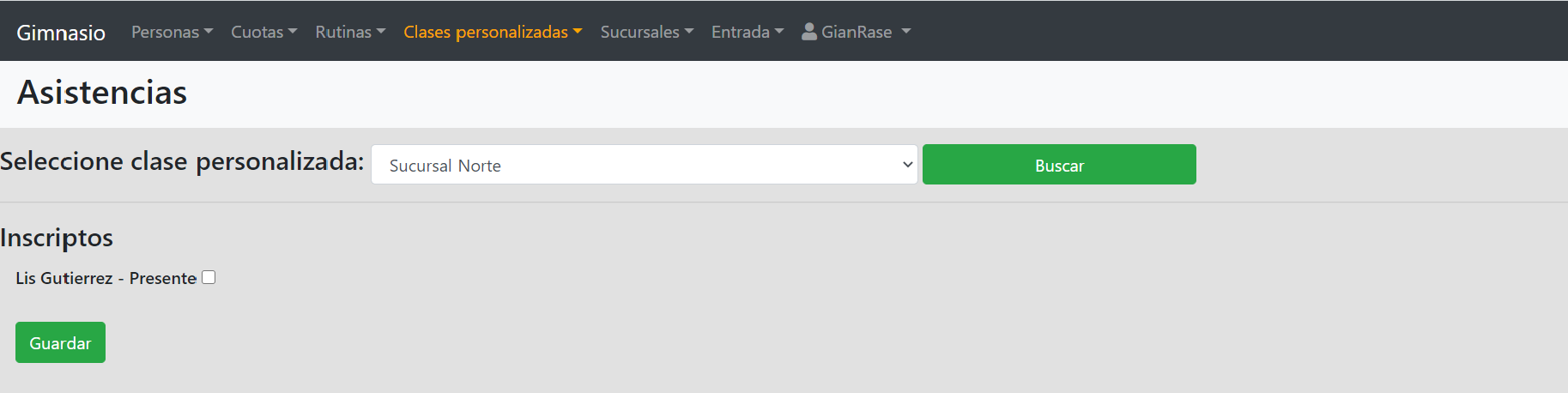


La clase InscripcionLogic se encarga de interactuar con la capa de datos.



Finalmente, en la capa de datos, se persiste la inscripción del usuario en una base de datos física.

**Caso de uso**: Registrar asistencias de una clase personalizada



Los entrenadores pueden ver las personas inscriptas a la clase personalizada que se está desarrollando en el momento (una por vez en cada sucursal) y que está a cargo, y puede marcar quienes asistieron y quienes no.



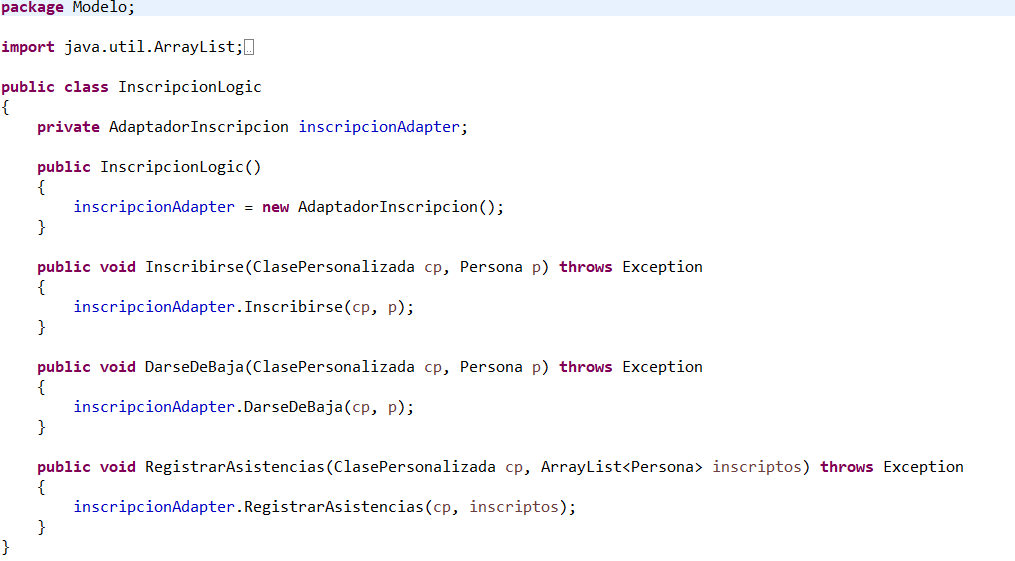
Acá se ve el formulario de la página JSP que envía los datos al servlet ServletInscripcion.



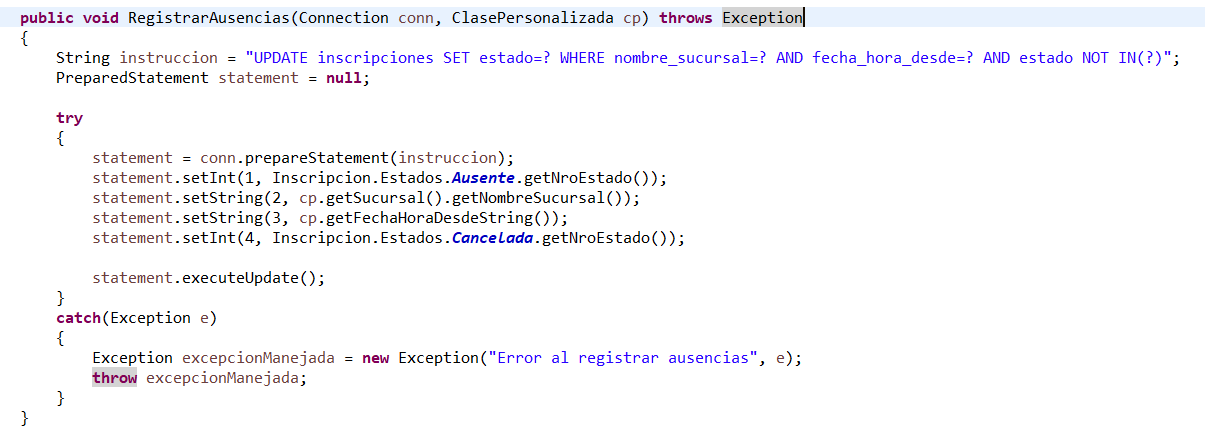
El servlet en el método doPost recupera el parámetro “instrucción” enviado por el form, y en base al valor que tiene realiza un método u otro.



En este método se recupera todos los datos enviados por el form mediante el método “request.getParameter()” y “request.getParameterValues()”. Luego llama al método RegistrarAsistencias de la clase InscripcionLogic.

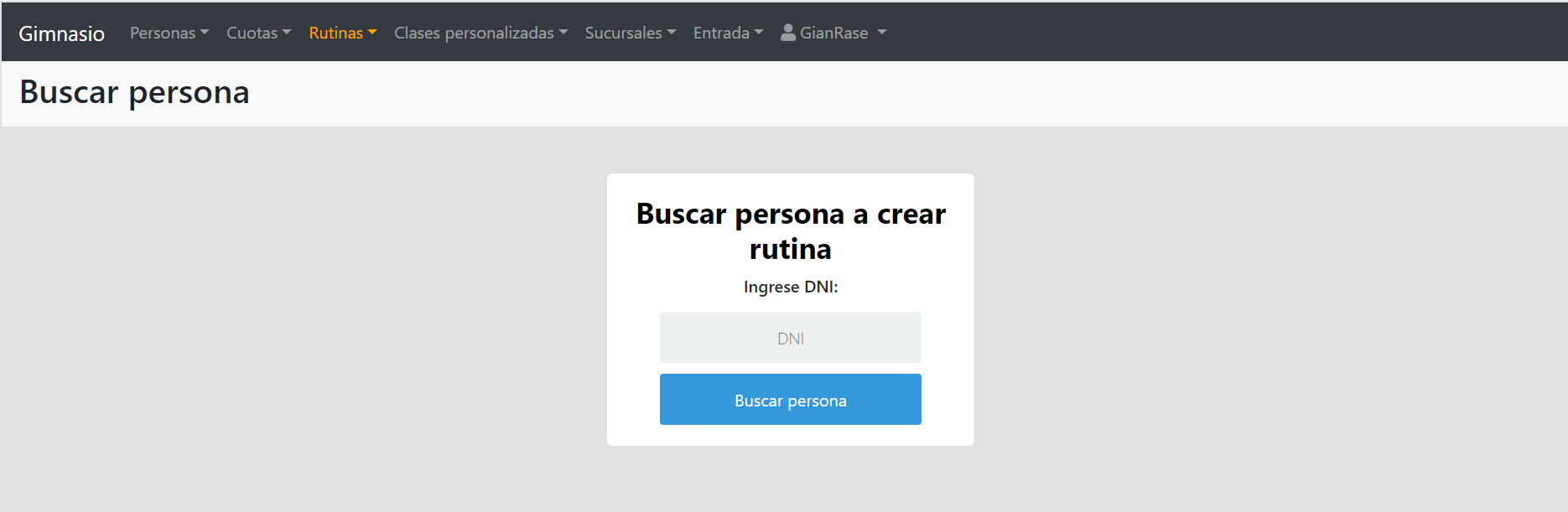


Esta clase se encarga de interactuar con la capa de datos.



Finalmente, estos dos métodos son los que se encargar de persistir los datos en la base de datos. Primero abre una conexión física, luego realiza las actualizaciones, y por ultimo cierra la conexión para no desperdiciar recursos. Todas las actualizaciones que realizan estos dos métodos se realizan dentro de una misma transacción en la base de datos.

**Caso de uso**: Registrar rutina de una persona



En “Registrar nueva rutina” ingresamos el DNI de la persona que queremos agregarle una rutina, para hacer eso debemos haber ingresado con el nivel de usuario de administrador.



El Form envía el DNI ingresado al servlet “ServletRegistrarRutina” para su validación.



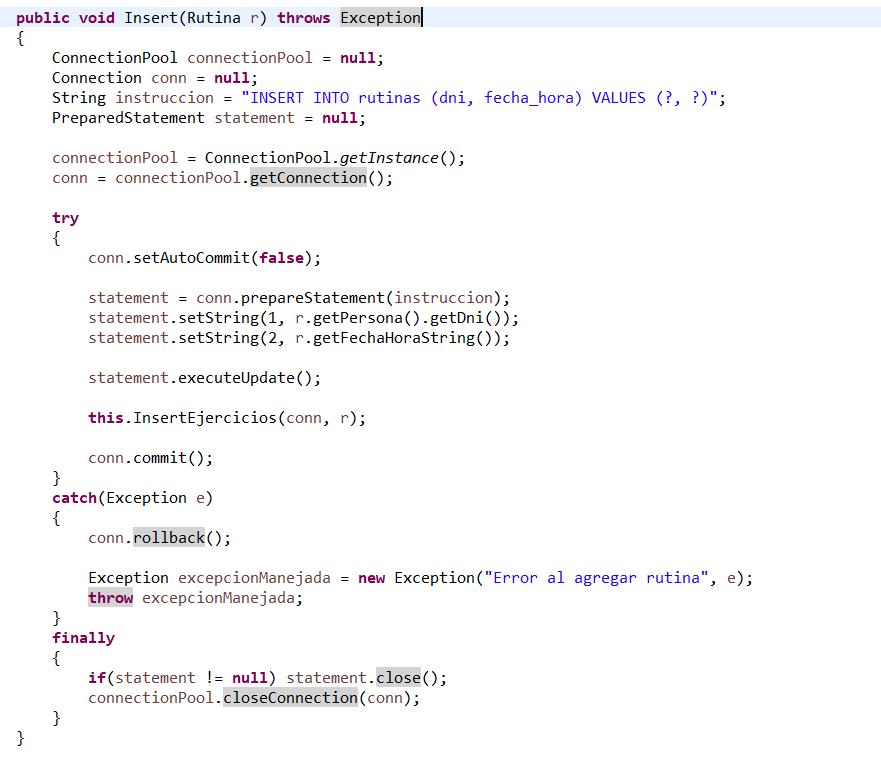
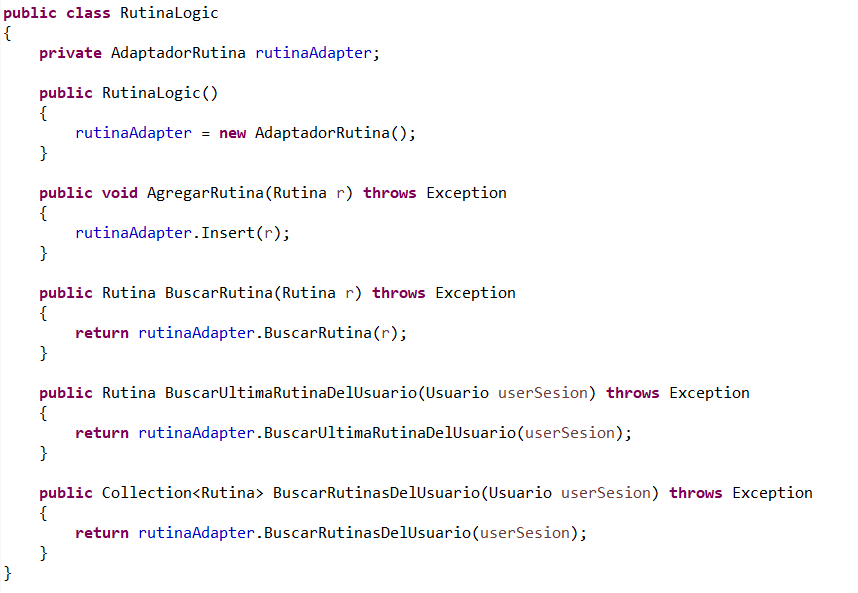
El método BuscarPersona crea una rutina de la persona en caso de encontrarla, en caso contrario informa la situación.



Una vez que la persona fue encontrada y la rutina creada, se pueden ir agregando y eliminando ejercicios. En este caso se observa el código del método AgregarEjercicio únicamente, ya que, el método EliminarEjercicio es muy similar.



El método RegistrarRutina verifica que haya al menos un ejercicio registrado en algún día de la rutina para guardarla.



Este método inicia una transacción en la base de datos para guardar los datos de la rutina y de los ejercicios como una sola operación, y en caso de que haya algún fallo no se registren los datos de ninguna de las dos clases.

