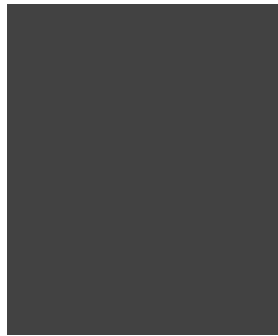


Universidad ORT Uruguay
Facultad de Ingeniería
Escuela de Tecnología

OBLIGATORIO PROGRAMACIÓN 1



L



M



[Grupo: N1]

Docente: G



Analista en Tecnologías de la Información

Fecha de entrega del documento (16-11-2023)

Índice

1. Documento de análisis.....	5
1.1. Descripción general del problema a resolver.....	5
1.1.1. Tipos de Usuarios del Sistema.....	5
1.1.2. Listado de funcionalidades.....	5
1.2. Detalle de funcionalidades.....	6
1.2.1. F01 - Registro de usuarios.....	6
1.2.1.1. Acceso.....	6
1.2.1.2. Descripción.....	6
1.2.1.3. Interfaz de Usuario.....	6
1.2.1.4. Validaciones.....	7
1.2.2. F02 - Ingreso de usuarios.....	9
1.2.2.1. Acceso.....	9
1.2.2.2. Descripción.....	9
1.2.2.3. Interfaz de Usuario.....	9
1.2.2.4. Validaciones.....	9
1.2.3. F03 - Alquiler de Máquinas Virtuales.....	10
1.2.3.1. Acceso.....	10
1.2.3.2. Descripción.....	10
1.2.3.3. Interfaz de Usuario.....	10
1.2.3.4. Validaciones.....	11
1.2.4. F04 - Visualización de máquinas alquiladas.....	11
1.2.4.1. Acceso.....	11
1.2.4.2. Descripción.....	11
1.2.4.3. Interfaz de Usuario.....	11
1.2.4.4. Validaciones.....	12
1.2.5. F05 - Encender / Apagar Máquinas Alquiladas.....	13
1.2.5.1. Acceso.....	13
1.2.5.2. Descripción.....	13
1.2.5.3. Interfaz de Usuario.....	13
1.2.5.4. Validaciones.....	13
1.2.6. F06 - Visualización del Costo Total de Alquiler.....	14
1.2.6.1. Acceso.....	14
1.2.6.2. Descripción.....	14
1.2.6.3. Interfaz de Usuario.....	14
1.2.6.4. Validaciones.....	14
1.2.7. F07 - Aprobación de Registros de Usuarios.....	15
1.2.7.1. Acceso.....	15

1.2.7.2. Descripción.....	15
1.2.7.3. Interfaz de Usuario.....	15
1.2.7.4. Validaciones.....	16
1.2.8. F08 - Bloqueo de Usuarios.....	16
1.2.8.1. Acceso.....	16
1.2.8.2. Descripción.....	16
1.2.8.3. Interfaz de Usuario.....	16
1.2.8.4. Validaciones.....	17
1.2.9. F09 - Modificación del Stock de Máquinas Virtuales.....	18
1.2.9.1. Acceso.....	18
1.2.9.2. Descripción.....	18
1.2.9.3. Interfaz de Usuario.....	18
1.2.9.4. Validaciones.....	19
1.2.10. F010 - Visualización de Informe de Máquinas Virtuales.....	20
1.2.10.1. Acceso.....	20
1.2.10.2. Descripción.....	20
1.2.10.3. Interfaz de Usuario.....	20
1.2.10.4. Validaciones.....	21
1.3. Casos de Prueba.....	22
1.3.1. F01 - Registro de usuarios.....	22
1.3.2. F02 - Ingreso de usuarios.....	26
1.3.3. F03 - Alquiler de Máquinas Virtuales.....	28
1.3.4. F04 - Visualización de máquinas alquiladas.....	33
1.3.5. F05 - Encender / Apagar Máquinas Alquiladas.....	33
1.3.6. F06 - Visualización del Costo Total de Alquiler.....	35
1.3.7. F07 - Aprobación de Registros de Usuarios.....	36
1.3.8. F08 - Bloqueo de Usuarios.....	37
1.3.9. F09 - Modificación del Stock de Máquinas Virtuales.....	38
1.3.10. F010 - Visualización de Informe de Máquinas Virtuales.....	40
2. Precarga de Datos.....	41
2.1. Usuarios Comunes:.....	41
2.1.1. Usuario 1.....	41
2.1.2. Usuario 2.....	41
2.1.3. Usuario 3.....	41
2.1.4. Usuario 4.....	42
2.1.5. Usuario 5.....	42
2.2. Usuarios Administradores.....	43
2.2.1. Administrador 1:.....	43
2.2.2. Administrador 2:.....	43
2.2.3. Administrador 3:.....	43

2.2.4. Administrador 4:.....	43
2.2.5. Administrador 5:.....	43
2.3. Tipos de Instancia.....	44
2.3.1. Optimizadas para Computo.....	44
2.3.2. Optimizadas para Memoria.....	44
2.3.3. Optimizadas para Almacenamiento.....	44
2.4. Stock de Máquinas virtuales.....	45
2.5. Alquileres.....	45
3. Código.....	46
3.1. index.html.....	46
3.2. clases.js.....	52
3.3. sistema.js.....	61
3.4. ui.js.....	102
3.5. Libreria.js.....	143
Anexo 1 Documento de análisis Original.....	149
1. Descripción general del problema a resolver.....	149
1.1. Tipos de Usuarios del Sistema.....	149
1.2. Listado de funcionalidades.....	149
2. Detalle de funcionalidades.....	150
2.1. F01 - Registro de usuarios.....	150
2.1.1. Acceso.....	150
2.1.2. Descripción.....	150
2.1.3. Interfaz de Usuario.....	150
2.1.4. Validaciones.....	151
2.2. F02 - Ingreso de usuarios.....	152
2.2.1. Acceso.....	152
2.2.2. Descripción.....	152
2.2.3. Interfaz de Usuario.....	152
2.2.4. Validaciones.....	152
2.3. F03 - Alquiler de Máquinas Virtuales.....	153
2.3.1. Acceso.....	153
2.3.2. Descripción.....	153
2.3.3. Interfaz de Usuario.....	153
2.3.4. Validaciones.....	153
2.4. F04 - Visualización de máquinas alquiladas.....	154
2.4.1. Acceso.....	154
2.4.2. Descripción.....	154
2.4.3. Interfaz de Usuario.....	154
2.4.4. Validaciones.....	155
2.5. F05 - Encender / Apagar Máquinas Alquiladas.....	156
2.5.1. Acceso.....	156
2.5.2. Descripción.....	156

2.5.3. Interfaz de Usuario.....	156
2.5.4. Validaciones.....	156
2.6. F06 - Visualización del Costo Total de Alquiler.....	157
2.6.1. Acceso.....	157
2.6.2. Descripción.....	157
2.6.3. Interfaz de Usuario.....	157
2.6.4. Validaciones.....	157
2.7. F07 - Aprobación de Registros de Usuarios.....	158
2.7.1. Acceso.....	158
2.7.2. Descripción.....	158
2.7.3. Interfaz de Usuario.....	158
2.7.4. Validaciones.....	158
2.8. F08 - Bloqueo de Usuarios.....	159
2.8.1. Acceso.....	159
2.8.2. Descripción.....	159
2.8.3. Interfaz de Usuario.....	159
2.8.4. Validaciones.....	159
2.9. F09 - Modificación del Stock de Máquinas Virtuales.....	160
2.9.1. Acceso.....	160
2.9.2. Descripción.....	160
2.9.3. Interfaz de Usuario.....	160
2.9.4. Validaciones.....	161
2.10. F010 - Visualización de Informe de Máquinas Virtuales.....	162
2.10.1. Acceso.....	162
2.10.2. Descripción.....	162
2.10.3. Interfaz de Usuario.....	162
2.10.4. Validaciones.....	163
3. Casos de prueba.....	164
3.1. F01 - Registro de usuarios.....	164
3.2. F02 - Ingreso de usuarios.....	166
3.3. F03 - Alquiler de Máquinas Virtuales.....	167
3.4. F04 - Visualización de máquinas alquiladas.....	168
3.5. F05 - Encender / Apagar Máquinas Alquiladas.....	169
3.6. F06 - Visualización del Costo Total de Alquiler.....	170
3.7. F07 - Aprobación de Registros de Usuarios.....	171
3.8. F08 - Bloqueo de Usuarios.....	172
3.9. F09 - Modificación del Stock de Máquinas Virtuales.....	173
3.10. F010 - Visualización de Informe de Máquinas Virtuales.....	175

1. Documento de análisis

1.1. Descripción general del problema a resolver

El objetivo es desarrollar una aplicación web que ofrezca el alquiler de máquinas virtuales a clientes, y la gestión de usuarios y máquinas virtuales a los administradores de la misma.

1.1.1. Tipos de Usuarios del Sistema

- Usuarios
- Administradores

1.1.2. Listado de funcionalidades

- F01 - Registro de Usuarios (Usuario)
- F02 - Ingreso de Usuarios (Usuario,Administrador)
- F03 - Alquiler de Máquinas Virtuales (Usuario)
- F04 - Visualización de Máquinas Alquiladas (Usuario)
- F05 - Encender / Apagar Máquinas Alquiladas (Usuario)
- F06 - Visualización del Costo Total de Alquiler (Usuario)
- F07 - Aprobación de Registros de Usuarios (Administrador)
- F08 - Bloqueo de Usuarios (Administrador)
- F09 - Modificación del Stock de Máquinas Virtuales (Administrador)
- F10 - Visualización de Informe de Máquinas Virtuales (Administrador)

1.2. Detalle de funcionalidades

A continuación, se presenta el detalle de cada una de las funcionalidades a resolver en el obligatorio.

1.2.1. F01 - Registro de usuarios

1.2.1.1. Acceso

Tipo de usuario que puede acceder: Usuario

1.2.1.2. Descripción

Permitir a los usuarios registrarse en la aplicación proporcionando información personal.

1.2.1.3. Interfaz de Usuario

La interfaz de usuario para el registro debe incluir un formulario que solicita los siguientes datos:

- Nombre
- Apellido
- Nombre de usuario (formato alfanumérico, case insensitive)
- Contraseña (debe cumplir con ciertos requisitos)
- Número de tarjeta de crédito (formato xxxx-xxxx-xxxx-xxxx)
- CVC (código de verificación, formato numérico de 3 dígitos)

Contiene al final 2 botones con el título:

Crear Usuario: Registra al usuario para ser aprobado por un administrador

Volver al Login: Permite volver al login principal

Registrarse

NOMBRE

APELLIDO

NOMBRE USUARIO

CONTRASEÑA

TARJETA DE CREDITO

CODIGO CVC TARJETA

[CREAR USUARIO](#) [VOLVER AL LOGIN](#)

1.2.1.4. Validaciones

Todos los campos del formulario deben ser completados.

El nombre de usuario debe cumplir con un formato alfanumérico y debe ser case insensitive.

- La contraseña debe cumplir con los siguientes requisitos:
 - Al menos 5 caracteres
 - Al menos una mayúscula
 - Al menos una minúscula
 - Al menos un número

La tarjeta de crédito debe cumplir con los siguientes requisito:

- Cumplir el formato xxxx-xxxx-xxxx-xxxx
- Tener 16 caracteres sin contar los guiones
- Debe cumplir con el algoritmo de verificación de tarjetas

El CVC debe ser un número de 3 dígitos.

No pueden existir dos usuarios con el mismo nombre de usuario

1.2.2. F02 - Ingreso de usuarios

1.2.2.1. Acceso

Tipo de usuario que puede acceder: Usuario, Administrador

1.2.2.2. Descripción

Permitir a los usuarios registrados iniciar sesión.

1.2.2.3. Interfaz de Usuario

La interfaz de usuario para el ingreso contará con:

- Nombre de Usuario: Campo de entrada de texto
- Contraseña: Campo de entrada de texto
- Botón de Inicio de Sesión
- Botón de Registrarse



The image shows a login interface within a light gray rectangular frame. At the top, the title "Inicio de sesion" is displayed in a large, bold, dark blue font. Below the title, there are two input fields. The first field is preceded by the label "INGRESE NOMBRE DE USUARIO" in a smaller, dark blue font. The second field is preceded by the label "INGRESE CONTRASENA" in the same font. At the bottom of the form, there are two dark blue buttons with white text. The left button is labeled "INICIAR SESION" and the right button is labeled "REGISTRARSE".

1.2.2.4. Validaciones

- Las credenciales ingresadas deben corresponder con las de un usuario ya registrado previamente.

- En caso de que las credenciales proporcionadas sean incorrectas, el sistema mostrará una notificación de error para informar al usuario sobre la falla en el inicio de sesión.
- El usuario debe estar habilitado para usar el sistema (no bloqueado)

1.2.3. F03 - Alquiler de Máquinas Virtuales

1.2.3.1. Acceso

Tipo de usuario que puede acceder: Usuario

1.2.3.2. Descripción

Permitir a los usuarios alquilar máquinas virtuales temporales para su uso. Al alquilar una máquina virtual, esta se pone en estado "encendido" automáticamente y, en esa primera ocasión, no se cobra el encendido, solo el alquiler.

1.2.3.3. Interfaz de Usuario

La interfaz de usuario para el alquiler contará con:

- ❖ Tipo de Instancia: Combo desplegable con las opciones :
 - Optimizadas para Computo
 - Optimizadas para Memoria
 - Optimizadas para Almacenamiento
- ❖ Disponibilidad: Área de texto o etiqueta que muestra si la instancia seleccionada está disponible o no.
- ❖ Costo de Alquiler: Área de texto o etiqueta que muestra el costo del alquiler.
- ❖ Costo por Encendido: Área de texto o etiqueta que muestra el costo del encendido.
- ❖ Botón de Alquiler: Botón con el texto "Alquilar"

ALQUILAR INSTANCIAS

CATEGORÍA

Optimizadas para Cómputo

Instancia	Costo Alquiler	Costo por Encendido	Disponible	Operar
c7.small	US\$ 20	US\$ 2.5	Si	ALQUILAR
c7.medium	US\$ 30	US\$ 3.5	Si	ALQUILAR
c7.large	US\$ 50	US\$ 6	Si	ALQUILAR

1.2.3.4. Validaciones

- ❖ El usuario debe estar autorizado a usar el sistema (no bloqueado)
- ❖ El tipo de instancia seleccionada debe ser válida (corresponde a las opciones del combobox)
- ❖ El tipo de instancia seleccionada debe tener stock disponible
- ❖ Solo se puede alquilar una instancia a la vez, por lo que si se desean alquilar múltiples instancias, se debe reiniciar el flujo de esta funcionalidad.

1.2.4. F04 - Visualización de máquinas alquiladas

1.2.4.1. Acceso

Tipo de usuario que puede acceder: Usuario

1.2.4.2. Descripción

Permitir a los usuarios ver una lista de las máquinas virtuales que han alquilado, así como información sobre el estado de cada instancia.

1.2.4.3. Interfaz de Usuario

La interfaz de usuario para el ingreso contará con:

- ❖ Tipo de Instancia: Combo desplegable con opciones cargadas dinámicamente, deben corresponderse con los tipos de instancia ya definidas, incluyendo “Todas las instancias”
- ❖ Tabla de instancias: Tabla que mostrará las instancias alquiladas por el usuario, contiene los siguientes datos:
 - ID - Id de la instancia
 - Tipo- Tipo de instancia (ejemplo C7.small)
 - Estado- Muestra el estado actual de la instancia (Encendido o Apagado)
 - Veces Iniciada - cantidad de veces que se prendió la instancia
 - Operar- contiene un botón que puede tener el título “Apagar” o “Encender” acorde al estado actual de la instancia.

MIS INSTANCIAS				
<div>Todas las instancias</div>				
ID	Tipo	Estado	Veces iniciada	Operar
INSTANCE_ID_2	c7.small	APAGADA	1 vez	ENCENDER
INSTANCE_ID_4	c7.medium	ENCENDIDA	1 vez	APAGAR

1.2.4.4. Validaciones

- El usuario debe estar autorizado a usar el sistema (no bloqueado)
- El usuario debe estar logueado

1.2.5. F05 - Encender / Apagar Máquinas Alquiladas

1.2.5.1. Acceso

Tipo de usuario que puede acceder: Usuario

1.2.5.2. Descripción

Permitir a los usuarios encender o apagar las Instancias.

1.2.5.3. Interfaz de Usuario

Misma Interfaz que en el punto 2.4 (F04).

MIS INSTANCIAS				
Todas las instancias				
ID	Tipo	Estado	Veces iniciada	Operar
INSTANCE_ID_2	c7.small	APAGADA	1 vez	ENCENDER
INSTANCE_ID_4	c7.medium	ENCENDIDA	1 vez	APAGAR

1.2.5.4. Validaciones

- El usuario debe estar logueado
- El usuario debe estar habilitado para operar (no bloqueado)

1.2.6. F06 - Visualización del Costo Total de Alquiler

1.2.6.1. Acceso

Tipo de usuario que puede acceder: Usuario

1.2.6.2. Descripción

Permitir a los usuarios visualizar el costo total de los alquileres de máquinas virtuales que han realizado. Esto incluye la agrupación de gastos por tipo de instancia.

1.2.6.3. Interfaz de Usuario

Tabla con los siguientes datos:

- Tipo de Instancia
- Costo por encendido
- Total de veces iniciada
- Costo total

MIS COSTOS			
Tipo de Instancia	Costo por Encendido	Total de veces encendida	Costo total
c7.small	US\$ 2.5	7	US\$ 35
c7.medium	US\$ 3.5	1	US\$ 30

1.2.6.4. Validaciones

- El usuario debe estar habilitado a operar el sistema (no bloqueado)
- El usuario debe estar logueado

1.2.7. F07 - Aprobación de Registros de Usuarios

1.2.7.1. Acceso

Tipo de usuario que puede acceder: Administrador

1.2.7.2. Descripción

Permitir a los administradores ver la lista de usuarios registrados que están en estado "pendiente" y aprobarlos para que puedan iniciar sesión en la aplicación.

1.2.7.3. Interfaz de Usuario

Lista de Usuarios: Una tabla que muestra la lista de todos los usuarios.

- Nombre
- Apellido
- Nombre de Usuario
- Estado (Activo, Pendiente o Bloqueado)
- Acción - Botón con el texto "Aprobar"

Notificación de Acción Exitosa: Área de texto o etiqueta que muestra un mensaje de confirmación cuando se realiza una acción de aprobación o bloqueo.

GESTIÓN DE USUARIOS DEL SISTEMA

Nombre	Apellido	Nombre usuario	Estado	Operar
mateo	carriqui	matteo	activo	BLOQUEAR
leandro	guzman	lean.g	activo	BLOQUEAR
Gaston	Sanguinetti	g.sangui	activo	BLOQUEAR
Mateo	Sanguinetti	m.sangui	activo	BLOQUEAR
Soy	Yo	soy.yo	activo	BLOQUEAR
Bruno	Perez	pru.per1	pendiente	ACTIVAR
Gaston	Sanguinetti	gas.ton1	pendiente	ACTIVAR

1.2.7.4. Validaciones

- Estar logueado como administrador

1.2.8. F08 - Bloqueo de Usuarios

1.2.8.1. Acceso

Tipo de usuario que puede acceder: Administrador

1.2.8.2. Descripción

Permitir a los administradores bloquear usuarios activos en la aplicación. El bloqueo de usuarios implica que el usuario no podrá iniciar sesión, y todas las máquinas virtuales alquiladas por el usuario volverán al stock, eliminando sus alquileres

1.2.8.3. Interfaz de Usuario

Lista de Usuarios: Una tabla que muestra la lista de todos los usuarios.

- Nombre
- Apellido
- Nombre de Usuario
- Estado (Activo, Pendiente o Bloqueado)
- Acción - Botón con el texto "Aprobar"

Notificación de Acción Exitosa: Área de texto o etiqueta que muestra un mensaje de confirmación cuando se realiza una acción de aprobación o bloqueo.

GESTIÓN DE USUARIOS DEL SISTEMA

Nombre	Apellido	Nombre usuario	Estado	Operar
mateo	carriqui	matteo	activo	BLOQUEAR
leandro	guzman	lean.g	activo	BLOQUEAR
Gaston	Sanguinetti	g.sangui	activo	BLOQUEAR
Mateo	Sanguinetti	m.sangui	activo	BLOQUEAR
Soy	Yo	soy.yo	activo	BLOQUEAR
Bruno	Perez	pru.per1	pendiente	ACTIVAR
Gaston	Sanguinetti	gas.ton1	pendiente	ACTIVAR

1.2.8.4. Validaciones

Estar logueado como administrador

1.2.9. F09 - Modificación del Stock de Máquinas Virtuales

1.2.9.1. Acceso

Tipo de usuario que puede acceder: Administrador

1.2.9.2. Descripción

Permitir a los administradores modificar el stock de máquinas virtuales disponibles por tipo de instancia.

1.2.9.3. Interfaz de Usuario

Lista de Tipos de Instancia: Una tabla dividida en 3 categorías que muestra la lista de tipos de instancia con las siguientes columnas:

- Tipo
- Unidades Totales

- Unidades Alquiladas
- Unidades Disponibles
- Modificar Disponibles(campo de entrada numérico)
- Botón "Guardar" (para cada tipo de instancia)

Notificación de Modificación Exitosa: Área de texto o etiqueta que muestra un mensaje de confirmación cuando se realiza una modificación exitosa

GESTIÓN DE INSTANCIAS

Optimizadas para computo

Tipo	Unidades Totales	Unidades Alquiladas	Unidades Disponibles	Modificar Disponibles	Confirmar
c7.small	3	2	1	<input type="text" value="1"/>	GUARDAR
c7.medium	5	1	4	<input type="text" value="4"/>	GUARDAR
c7.large	3	1	2	<input type="text" value="2"/>	GUARDAR

Optimizadas para memoria

Tipo	Unidades Totales	Unidades Alquiladas	Unidades Disponibles	Modificar Disponibles	Confirmar
r7.small	7	2	5	<input type="text" value="5"/>	GUARDAR
r7.medium	5	2	3	<input type="text" value="3"/>	GUARDAR
r7.large	2	0	2	<input type="text" value="2"/>	GUARDAR

Optimizadas para almacenamiento

Tipo	Unidades Totales	Unidades Alquiladas	Unidades Disponibles	Modificar Disponibles	Confirmar
i7.medium	3	2	1	<input type="text" value="1"/>	GUARDAR
i7.large	0	0	0	<input type="text" value="0"/>	GUARDAR

1.2.9.4. Validaciones

- Estar logueado como administrador
- Solo los administradores deben tener acceso para modificar el stock de máquinas virtuales.
- La cantidad a modificar debe ser numérica.
- La cantidad a modificar no debe ser negativa.
- Solo se modifican las Unidades disponibles

1.2.10. F010 - Visualización de Informe de Máquinas Virtuales

1.2.10.1. Acceso

Tipo de usuario que puede acceder: Administrador

1.2.10.2. Descripción

Permitir a los administradores acceder a un informe que muestre información sobre los tipos de instancia en alquiler, la cantidad de cada tipo de instancia en alquiler y su ingreso total actual por cada tipo de instancia. Además, se debe mostrar el ingreso total global de la empresa.

1.2.10.3. Interfaz de Usuario

Reportes: Una tabla que muestra el informe con las siguientes columnas:

- Tipo de Instancia
- Alquileres Efectuados
- Ingresos

REPORTES		
Tipo de Instancia	Alquileres efectuados	Ingresos
c7.small	2	US\$ 40
c7.medium	1	US\$ 30
c7.large	1	US\$ 50
r7.small	2	US\$ 70
r7.medium	2	US\$ 100
r7.large	0	US\$ 0
i7.medium	2	US\$ 60
i7.large	0	US\$ 0
TOTAL	10	US\$ 350

1.2.10.4. Validaciones

- Solo los administradores deben tener acceso para ver el informe de máquinas virtuales.
- El administrador debe estar logueado

1.3.

1.4. Casos de Prueba

1.3.1. F01 - Registro de usuarios

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F01-T01	Registro exitoso (Administrador aprueba)	<p>1.El usuario ingresa un nombre, apellido, nombre de usuario, contraseña, número de tarjeta de crédito, CVC. El usuario solicita el registro.</p> <p>2.El administrador desde su cuenta aprueba al usuario nuevo</p>	<p><u>Nombre:</u> Leandro</p> <p><u>Apellido:</u> Guzmán</p> <p><u>Nombre De Usuario:</u> Prog1</p> <p><u>Contraseña:</u> Tarea1</p> <p><u>Número de Tarjeta De Crédito:</u>4970-1000-0000-0014</p> <p><u>CVC:</u> 123</p>	<p>1.El sistema registra al usuario como "pendiente" y muestra un mensaje indicando que su registro será aprobado por un administrador.</p> <p>2. El sistema muestra al administrador la solicitud de pendiente del usuario nuevo en la ventana "Gestión de Usuarios del Sistema" y con el botón "Activar" lo aprueba.</p>	<p>1-Para el usuario se muestra el mensaje: "Usuario Creado exitosamente. Pendiente a aprobación"</p> <p>2-Para el administrador el estado del usuario pasa de "Pendiente" a "Activo"</p>	P
F01-T02	Intento de registro con una contraseña débil.	<p>1.El usuario ingresa una contraseña que no cumple con los requisitos (por ejemplo, una contraseña de 3 caracteres sin mayúsculas ni números).</p>	<p><u>Nombre:</u> Leandro</p> <p><u>Apellido:</u> Guzmán</p> <p><u>Nombre De Usuario:</u> Prog1</p>	<p>El sistema muestra un mensaje de error indicando que la contraseña no cumple con los requisitos.</p>	<p>Se muestra el mensaje :</p> <p>"Debe ingresar una contraseña válida. Mínimo de 5 caracteres,</p>	P

		2.El usuario solicita el registro	<u>Contraseña:</u> Tar <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> 123		contando con al menos una mayúscula, una minúscula y un número."	
F01-T03	Intento de registro con una tarjeta de crédito inválida.	1.El usuario ingresa un número de tarjeta de crédito que no tiene 16 dígitos o no pasa la validación del algoritmo de tarjetas. 2.El usuario solicita el registro.	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> Prog1 <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 1234-56 <u>CVC:</u> 123	El sistema muestra un mensaje de error indicando que el número de tarjeta de crédito es inválido.	Se muestra el mensaje: "Tarjeta invalida."	P
F01-T04	Intento de registro con un nombre de usuario ya en uso.	1.El usuario ingresa un nombre de usuario que ya está en uso por otro usuario. 2.El usuario solicita el registro.	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> Prog1 (ya está registrado) <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> 123	El sistema muestra un mensaje de error indicando que el nombre de usuario ya está en uso.	Se muestra el mensaje: "El nombre de usuario ya existe en el sistema"	P
F01-T05	Intento de registro con un nombre de usuario invalido.	1.El usuario ingresa un nombre de usuario que tenga espacios o símbolos (Ejemplo: - , + " \$). 2.El usuario solicita el registro.	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> Prog1+	El sistema muestra un mensaje de error indicando que el nombre de usuario es invalido..	Se muestra el mensaje: "Nombre de usuario inválido. Solo	P

			<u>Prog1-</u> <u>Prog1”</u> <u>Prog 1</u> <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> 123		se admiten números, letras, puntos y guiones bajos”	
F01-T06	Intento de registro con CVC invalido	<p>El usuario ingresa datos válidos excepto por el CVC.</p> <p>El usuario solicita el registro.</p>	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> Prog1 <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> 12	El sistema muestra mensaje de error	Se muestra el mensaje “CVC invalido.”	P
F01-T07	Intento de registro con campos vacíos(Nombre)	El usuario carga los datos requeridos para el registro omitiendo el campo “Nombre” que quedará vacío.	<u>Nombre:</u> (vacío) <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> Prog1 <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> 123	El sistema informa al usuario que ingrese un nombre.	Se muestra el mensaje : “Debe Ingresar Nombre”	P

F01-T08	Intento de registro con campos vacíos(Apellido)	El usuario carga los datos requeridos para el registro omitiendo el campo "Apellido" que quedará vacío.	<u>Nombre:</u> Leandro <u>Apellido:</u> (Vacío) <u>Nombre De Usuario:</u> Prog1 <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> 123	El sistema informa al usuario que ingrese un Apellido.	Se muestra el mensaje : "Debe Ingresar un Apellido"	P
F01-T09	Intento de registro con campos vacíos(Nombre Usuario)	El usuario carga los datos requeridos para el registro omitiendo el campo "Nombre Usuario" que quedará vacío.	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> (Vacío) <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> 123	El sistema informa al usuario que ingrese un Nombre de Usuario.	Se muestra el mensaje : "Debe Ingresar un Nombre de Usuario"	P
F01-T10	Intento de registro con campos vacíos(Contraseña)	El usuario carga los datos requeridos para el registro omitiendo el campo "Contraseña" que quedará vacío	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> Prog1 <u>Contraseña:</u> (Vacío) <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> 123	El sistema informa al usuario que ingrese una contraseña válida	Se muestra el mensaje : "Debe ingresar una contraseña válida. Mínimo de 5 caracteres, contando con al menos una mayúscula, una minúscula y un número."	P

F01-T11	Intento de registro con campos vacíos(Tarjeta de crédito)	El usuario carga los datos requeridos para el registro omitiendo el campo "Tarjeta de crédito" que quedará vacío	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> Prog1 <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> (Vacío) <u>CVC:</u> 123	El sistema informa al usuario que ingrese una tarjeta de crédito válida.	Se muestra el mensaje : "Tarjeta invalida."	P
F01-T12	Intento de registro con campos vacíos(CVC)	El usuario carga los datos requeridos para el registro omitiendo el campo "CVC" que quedará vacío	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> Prog1 <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 4970-1000-0000-0014 <u>CVC:</u> (Vacío)	El sistema informa al usuario que ingrese el CVC	Se muestra el mensaje: "CVC invalido."	P

1.3.2. F02 - Ingreso de usuarios

Precondiciones :El usuario cuenta con un usuario creado.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F02-T01	Ingreso exitoso con credenciales válidas.	El usuario ingresa su nombre de usuario y contraseña correctos. El usuario solicita el ingreso.	<u>Nombre De Usuario:</u> matteo <u>Contraseña:</u> 1234La	El sistema permite el ingreso	El usuario ingresa exitosamente quedando por defecto en la ventana “Mis Instancias”	P
F02-T02	Intento de ingreso con un nombre de usuario incorrecto.	El usuario ingresa un nombre de usuario incorrecto. El usuario ingresa la contraseña correcta. El usuario solicita el ingreso.	<u>Nombre De Usuario:</u> matte0 <u>Contraseña:</u> 1234La	El sistema muestra un mensaje de error sin especificar si existe el usuario	El sistema muestra el mensaje : “Usuario y/o contraseña incorrecto”	P
F02-T03	Intento de ingreso con una contraseña incorrecta.	El usuario ingresa su nombre de usuario correcto. El usuario ingresa una contraseña incorrecta. El usuario solicita el ingreso.	<u>Nombre De Usuario:</u> matteo <u>Contraseña:</u> 123	El sistema muestra un mensaje de error	El sistema muestra el mensaje : “Usuario y/o contraseña incorrecto”	

F02-T04	Intento de ingreso sin ingresar una contraseña	El usuario ingresa su nombre de usuario correcto. El usuario no ingresa contraseña	<u>Nombre De Usuario:</u> matteo <u>Contraseña:</u> (Vacio)	El sistema muestra un mensaje de error	El sistema muestra el mensaje : "Usuario y/o contraseña incorrecto"	
F02-T05	Intento de ingreso sin ingresar una su nombre de usuario	El usuario omite su nombre de usuario.. El usuario ingresa contraseña	<u>Nombre De Usuario:</u> (Vacio) <u>Contraseña:</u> 1234La	El sistema muestra un mensaje de error	El sistema muestra el mensaje : "Usuario y/o contraseña incorrecto"	

1.3.3. F03 - Alquiler de Máquinas Virtuales

Precondiciones :El usuario ingreso al sistema

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falló, P=Pasó)
F03-T01	Alquiler exitoso de una máquina virtual. (Optimizada para Computo-c7.Small)	El usuario elige el tipo de instancia a alquilar. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para computo: c7.small	El sistema muestra el mensaje “La máquina c7.small fue alquilada correctamente.” Una nueva máquina c7.small encendida está disponible en la ventana “Gestionar Instancias”	El sistema muestra el mensaje “La máquina c7.small fue alquilada correctamente.” Una nueva máquina c7.small encendida está disponible en la ventana “Gestionar Instancias”	P
F03-T02	Intento de alquiler de una máquina virtual no disponible. (Optimizada para Computo-c7.Small)	El usuario elige un tipo de instancia que no está disponible en el stock. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para computo: c7.small(en este caso no disponible)	El sistema muestra un mensaje de error indicando que el tipo de instancia no está disponible.	El sistema muestra el mensaje La máquina c7.small no tiene stock disponible por el momento. Intente de nuevo más tarde.	P
F03-T03	Alquiler exitoso de una máquina virtual. (Optimizada para Computo-c7.medium)	El usuario elige el tipo de instancia a alquilar. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para computo: c7.medium	El sistema muestra el mensaje “La máquina c7.medium fue alquilada correctamente.”	El sistema muestra el mensaje “La máquina c7.small fue alquilada correctamente.” Una nueva máquina c7.small encendida está	P

				Una nueva máquina c7.medium encendida está disponible en la ventana "Gestionar Instancias"	disponible en la ventana "Gestionar Instancias"	
F03-T04	Intento de alquilar de una máquina virtual no disponible. (Optimizada para Computo-c7.medium)	El usuario elige un tipo de instancia que no está disponible en el stock. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para computo: c7.medium(en este caso no disponible)	El sistema muestra un mensaje de error indicando que el tipo de instancia no está disponible.	El sistema muestra el mensaje La máquina c7.medium no tiene stock disponible por el momento. Intente de nuevo más tarde.	P
F03-T05	Alquiler exitoso de una máquina virtual. (Optimizada para Computo-c7.large)	El usuario elige el tipo de instancia a alquilar. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para computo: c7.large	El sistema muestra el mensaje "La máquina c7.large fue alquilada correctamente." Una nueva máquina c7.large encendida está disponible en la ventana "Gestionar Instancias"	El sistema muestra el mensaje "La máquina c7.large fue alquilada correctamente." Una nueva máquina c7.large encendida está disponible en la ventana "Gestionar Instancias"	P
F03-T06	Intento de alquilar de una máquina virtual no disponible. (Optimizada para Computo-c7.large)	El usuario elige un tipo de instancia que no está disponible en el stock. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para computo: c7.large(en este caso no disponible)	El sistema muestra un mensaje de error indicando que el tipo de instancia no está disponible.	El sistema muestra el mensaje La máquina c7.large no tiene stock disponible por el momento. Intente de nuevo más tarde.	P
F03-T07	Alquiler exitoso de una máquina virtual.	El usuario elige el tipo de instancia a alquilar.	Tipo de instancia:	El sistema muestra el mensaje "La	El sistema muestra el mensaje "La máquina	P

	(Optimizada para Memoria-r7.Small)	El usuario solicita el alquiler.	Optimizadas para memoria: r7.small	máquina r7.small fue alquilada correctamente.” Una nueva máquina r7.small encendida está disponible en la ventana “Gestionar Instancias”	r7.small fue alquilada correctamente.” Una nueva máquina r7.small encendida está disponible en la ventana “Gestionar Instancias”	
F03-T08	Intento de alquiler de una máquina virtual no disponible. (Optimizada para Memoria-r7.Small)	El usuario elige un tipo de instancia que no está disponible en el stock. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para memoria: r7.small(en este caso no disponible)	El sistema muestra un mensaje de error indicando que el tipo de instancia no está disponible.	El sistema muestra el mensaje La máquina r7.small no tiene stock disponible por el momento. Intente de nuevo más tarde.	P
F03-T09	Alquiler exitoso de una máquina virtual. (Optimizada para Memoria-r7.Medium)	El usuario elige el tipo de instancia a alquilar. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para memoria: r7.medium	El sistema muestra el mensaje “La máquina r7.medium fue alquilada correctamente.” Una nueva máquina r7.medium encendida está disponible en la ventana “Gestionar Instancias”	El sistema muestra el mensaje “La máquina r7.medium fue alquilada correctamente.” Una nueva máquina r7.medium encendida está disponible en la ventana “Gestionar Instancias”	P
F03-T10	Intento de alquiler de una máquina virtual no disponible. (Optimizada para Memoria-r7.Medium)	El usuario elige un tipo de instancia que no está disponible en el stock. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para memoria: r7.medium(en este caso no disponible)	El sistema muestra un mensaje de error indicando que el tipo de instancia no está disponible.	El sistema muestra el mensaje La máquina r7.medium no tiene stock disponible por el momento. Intente de nuevo más tarde.	P

F03-T11	Alquiler exitoso de una máquina virtual. (Optimizada para Memoria-r7.large)	El usuario elige el tipo de instancia a alquilar. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para memoria: r7.large	El sistema muestra el mensaje “La máquina r7.large fue alquilada correctamente.” Una nueva máquina r7.large encendida está disponible en la ventana “Gestionar Instancias”	El sistema muestra el mensaje “La máquina r7.large fue alquilada correctamente.” Una nueva máquina r7.large encendida está disponible en la ventana “Gestionar Instancias”	P
F03-T12	Intento de alquiler de una máquina virtual no disponible. (Optimizada para Memoria-r7.large)	El usuario elige un tipo de instancia que no está disponible en el stock. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para memoria: r7.large(en este caso no disponible)	El sistema muestra un mensaje de error indicando que el tipo de instancia no está disponible.	El sistema muestra el mensaje La máquina r7.large no tiene stock disponible por el momento. Intente de nuevo más tarde.	P
F03-T13	Alquiler exitoso de una máquina virtual. (Optimizada para Almacenamiento-i7.medium)	El usuario elige el tipo de instancia a alquilar. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para Almacenamiento: i7.medium	El sistema muestra el mensaje “La máquina i7.medium fue alquilada correctamente.” Una nueva máquina i7.medium encendida está disponible en la ventana “Gestionar Instancias”	El sistema muestra el mensaje “La máquina i7.medium fue alquilada correctamente.” Una nueva máquina i7.medium encendida está disponible en la ventana “Gestionar Instancias”	P
F03-T14	Intento de alquiler de una máquina virtual no disponible.	El usuario elige un tipo de instancia que no está disponible en el stock.	Tipo de instancia: Optimizadas para Almacenamiento:	El sistema muestra un mensaje de error indicando que el tipo de instancia	El sistema muestra el mensaje La máquina i7.medium no tiene stock	P

	(Optimizada para Almacenamiento-i7.medium)	El usuario solicita el alquiler.	i7.medium(en este caso no disponible)	no está disponible.	disponible por el momento. Intente de nuevo más tarde.	
F03-T15	Intento de alquiler de una máquina virtual no disponible. (Optimizada para Almacenamiento-i7.large)	El usuario elige el tipo de instancia a alquilar. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para Almacenamiento: i7.large	El sistema muestra el mensaje “La máquina i7.large fue alquilada correctamente.” Una nueva máquina i7.large encendida está disponible en la ventana “Gestionar Instancias”	El sistema muestra el mensaje “La máquina i7.large fue alquilada correctamente.” Una nueva máquina i7.large encendida está disponible en la ventana “Gestionar Instancias”	P
F03-T16	Intento de alquiler de una máquina virtual no disponible. (Optimizada para Almacenamiento-i7.large)	El usuario elige un tipo de instancia que no está disponible en el stock. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para Almacenamiento: i7.large(en este caso no disponible)	El sistema muestra un mensaje de error indicando que el tipo de instancia no está disponible.	El sistema muestra el mensaje La máquina i7.large no tiene stock disponible por el momento. Intente de nuevo más tarde.	P

1.3.4. F04 - Visualización de máquinas alquiladas

Precondiciones :El usuario ingreso al sistema

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla , P=pasa)

F04-T01	Visualización exitosa de las máquinas alquiladas.	El usuario ingresa a la pantalla de visualización de máquinas alquiladas.		El sistema muestra la lista de máquinas alquiladas por el usuario, incluyendo el tipo de instancia, estado, y el número de veces que se ha iniciado.	El sistema muestra la lista de máquinas alquiladas por el usuario, incluyendo el tipo de instancia, estado, y el número de veces que se ha iniciado.	P
F04-T02	Usuario sin máquinas alquiladas.	El usuario ingresa a la pantalla de visualización de máquinas alquiladas.		El sistema muestra la lista de máquinas alquiladas vacía	El sistema muestra la lista de máquinas alquiladas vacía	P

1.3.5. F05 - Encender / Apagar Máquinas Alquiladas

Precondiciones :El usuario ingresó al sistema y cuenta con máquinas alquiladas.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falló, P=Pasó)
F05-T01	Encendido exitoso de una máquina alquilada	El usuario ingresa a la pantalla de visualización de máquinas alquiladas. El usuario selecciona una máquina alquilada que está apagada y da		El sistema enciende la máquina alquilada, cobra el costo correspondiente, cambia el estado de apagada a encendida y suma un nuevo encendido en la	El sistema enciende la máquina alquilada, cobra el costo correspondiente, cambia el estado de apagada a encendida y suma un nuevo encendido en la columna "Veces Iniciada".	P

		click al botón “Encender”.		columna “Veces Iniciada” . El botón de “Encender” cambia por uno de “Apagar”	El botón de “Encender” cambia por uno de “Apagar”	
F05-T02	Apagado exitoso de una máquina alquilada.	El usuario ingresa a la pantalla de visualización de máquinas alquiladas. El usuario selecciona una máquina alquilada que está encendida y da click al botón “Apagar”.		El sistema apaga la máquina alquilada con éxito y cambia el estado de Encendida a Apagada. El botón de “Apagar” cambia por uno de “Encender”	El sistema apaga la máquina alquilada con éxito y cambia el estado de Encendida a Apagada. El botón de “Apagar” cambia por uno de “Encender”	P

1.3.6. F06 - Visualización del Costo Total de Alquiler

Precondiciones :El usuario ingreso al sistema.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)

F06-T01	Visualización exitosa del costo total de alquiler por usuario.	El usuario ingresa a la pantalla “Ver Costo”.		El sistema muestra una tabla con los tipos de instancia, el costo por encendido, el total de veces encendida y el costo total.	El sistema muestra una tabla con los tipos de instancia, el costo por encendido, el total de veces encendida y el costo total.	P
F06-T02	Usuario no tiene máquinas	El usuario ingresa a la pantalla de visualización del costo total de alquiler.		La tabla se muestra vacía	La tabla se muestra vacía	P

1.3.7. F07 - Aprobación de Registros de Usuarios

Precondiciones :El usuario ingresa al sistema como administrador.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F07-T01	Aprobación exitosa de un registro de usuario pendiente.	El administrador ingresa a la pantalla "Gestion de usuarios". El administrador selecciona un registro de usuario pendiente. El administrador aprueba el registro.		El sistema aprueba el registro de usuario pendiente y cambia su estado a "activo". El usuario puede iniciar sesión en la aplicación.	El sistema aprueba el registro de usuario pendiente y cambia su estado a "activo". El usuario puede iniciar sesión en la aplicación.	P
F07-T02	Sin Usuarios Pendientes	El administrador ingresa a la pantalla "Gestion de usuarios".		Lista de usuarios por aprobar vacía.	Lista de usuarios por aprobar vacía.	P

1.3.8. F08 - Bloqueo de Usuarios

Precondiciones :El usuario ingresa al sistema como administrador.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F08-T01	Bloqueo exitoso de un usuario activo por parte de un administrador.	El administrador ingresa a la pantalla de "Gestion de usuarios". El administrador selecciona un usuario activo. El administrador bloquea al usuario.		El sistema bloquea al usuario, cambia su estado a "bloqueado" y todas las máquinas virtuales alquiladas por el usuario vuelven al stock, eliminando sus alquileres.	El sistema bloquea al usuario, cambia su estado a "bloqueado" y todas las máquinas virtuales alquiladas por el usuario vuelven al stock, eliminando sus alquileres.	P
F08-T02	Lista de Usuarios para bloquear vacía	El administrador ingresa a la pantalla de bloqueo de usuarios.		Lista de usuarios por bloquear vacía.	Lista de usuarios por bloquear vacía.	P

1.3.9. F09 - Modificación del Stock de Máquinas Virtuales

Precondiciones :El usuario ingresa al sistema como administrador.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F09-T01	Modificación exitosa del stock de máquinas virtuales disponibles por tipo de instancia.	<p>El administrador ingresa a la pantalla de modificación del stock de “Gestionar Instancia”</p> <p>El administrador selecciona un tipo de instancia.</p> <p>El administrador modifica la cantidad de máquinas virtuales disponibles.</p>	Cantidad a modificar: 5	El sistema permite al administrador modificar la cantidad de máquinas virtuales disponibles y actualiza el stock con el nuevo valor.	El stock disponible se actualiza al número cargado por el administrador.	P
F09-T02	Intento de modificar el stock a una cantidad negativa.	<p>El administrador ingresa a la pantalla de modificación del stock de “Gestionar Instancia”</p> <p>El administrador selecciona un tipo de instancia.</p> <p>El administrador intenta modificar la cantidad de máquinas virtuales disponibles a un valor negativo.</p>	Cantidad a modificar: -5	El sistema muestra un mensaje de error indicando que no se puede establecer un stock negativo para las máquinas virtuales.	Se muestra el mensaje : Valor inválido.	P

F09-T03	Intento de modificar el stock por un simbolo (Ejemplo: . , + -)	<p>El administrador ingresa a la pantalla de modificación del stock de “Gestionar Instancia”</p> <p>El administrador selecciona un tipo de instancia.</p> <p>El administrador intenta modificar la cantidad de máquinas virtuales disponibles por un simbolo..</p>	Cantidad a modificar: + - , .	El sistema muestra un mensaje de error indicando que se ingrese un valor valido.	Se muestra el mensaje : Valor inválido.	P
---------	---	--	-------------------------------	--	---	---

1.3.10. F010 - Visualización de Informe de Máquinas Virtuales

Precondiciones :El usuario ingresa al sistema como administrador.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F010-T01	Visualización exitosa del informe de máquinas virtuales.	El administrador ingresa a la pantalla Reportes. El administrador revisa el informe con la información de tipos de instancia, cantidad y su ingreso total actual por cada tipo de instancia.		El sistema muestra el informe de máquinas virtuales con la información: <ul style="list-style-type: none"> • Tipo de Instancia. • Alquileres Efectuados. • Ingresos. 	El sistema muestra el informe de máquinas virtuales con la información: <ul style="list-style-type: none"> • Tipo de Instancia. • Alquileres Efectuados. • Ingresos. 	P
F010-T02	Visualización del informe cuando no hay máquinas virtuales en el sistema.	El administrador ingresa a la pantalla Reportes.		El sistema muestra el informe vacío, indicando que no hay máquinas virtuales alquiladas en el sistema.	El sistema muestra el informe vacío, indicando que no hay máquinas virtuales alquiladas en el sistema.	

2. Precarga de Datos

2.1. Usuarios Comunes:

2.1.1. Usuario 1

- Nombre: [REDACTED]
- Apellido: [REDACTED]
- Nombre Usuario: matteo
- Contraseña: 1234La
- Numero de Tarjeta de crédito: 4539-6253-0847-8250
- Codigo de Verificacion: 323

2.1.2. Usuario 2

- Nombre: [REDACTED]
- Apellido: [REDACTED]
- Nombre Usuario: lean.g
- Contraseña: 1234La
- Numero de Tarjeta de crédito: 4539-6253-0847-8250
- Codigo de Verificacion: 323

2.1.3. Usuario 3

- Nombre: [REDACTED]
- Apellido: [REDACTED]
- Nombre Usuario: [REDACTED]
- Contraseña: 1234La
- Numero de Tarjeta de crédito: 4539-6253-0847-8250

- Código de Verificación: 323

2.1.4. Usuario 4

- Nombre: [REDACTED]
- Apellido: [REDACTED]
- Nombre Usuario: [REDACTED]
- Contraseña: 1234La
- Número de Tarjeta de crédito: 4539-6253-0847-8250
- Código de Verificación: 323

2.1.5. Usuario 5

- Nombre: Soy
- Apellido: Yo
- Nombre Usuario: soy.yo
- Contraseña: 1234La
- Número de Tarjeta de crédito: 4539-6253-0847-8250
- Código de Verificación: 323

2.2. Usuarios Administradores

2.2.1. Administrador 1:

- **Nombre Usuario:** [REDACTED]
- **Contraseña:** 1234La

2.2.2. Administrador 2:

- **Nombre Usuario:** mateo
- **Contraseña:** 1234La

2.2.3. Administrador 3:

- **Nombre Usuario:** matt1
- **Contraseña:** 1234La

2.2.4. Administrador 4:

- **Nombre Usuario:** lean1
- **Contraseña:** 1234La

2.2.5. Administrador 5:

- **Nombre Usuario:** admin
- **Contraseña:** 1234La

2.3. Tipos de Instancia

2.3.1. Optimizadas para Computo

- c7.small:
 - Costo alquiler :US\$ 20
 - Costo Encendido: US\$ 2.5
- c7.medium:
 - Costo alquiler :US\$ 30
 - Costo Encendido:US\$ 3.5
- c7.large:
 - Costo alquiler:US\$ 50
 - Costo Encendido:US\$ 6

2.3.2. Optimizadas para Memoria

- r7.small:
 - Costo alquiler :US\$ 35
 - Costo Encendido: US\$ 4
- r7.medium:
 - Costo alquiler :US\$ 50
 - Costo Encendido:US\$ 6.5
- r7.large:
 - Costo alquiler:US\$ 60
 - Costo Encendido:US\$ 7

2.3.3. Optimizadas para Almacenamiento

- i7.medium:
 - Costo alquiler :US\$ 30
 - Costo Encendido: US\$ 3.5
- i7.large:
 - Costo alquiler :US\$ 50
 - Costo Encendido:US\$ 6.5

2.4. Stock de Máquinas virtuales

- c7.small: 3
- c7.medium: 5
- c7.large: 3
- r7.small: 7
- r7.medium: 5
- r7.large: 2
- i7.medium: 3
- i7.large: 3

2.5. Alquileres

- matteo:
 - c7.small :1
 - r7.small :1
- lean.g:
 - c7.small: 1
 - c7.medium: 1
- g.sangui:
 - c7.large: 1
 - r7.small: 1
- m.sangui
 - r7.medium: 2
- soy.yo
 - i7.medium: 2

3. Código

3.1. index.html

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="stylesheet" href="estilos/estilos.css" />

    <title>Obligatorio P1 2023 - Mateo Carriquí, Leandro Guzmán</title>

  </head>

  <body>

    <div id="contenedor">

      <div id="cabezal" class="clearfix" style="display: none">

        <ul id="navPrincipal">

          <!-- ADMIN NAV -->
```



```

<li>

    <a id="aGestionarInstancias" class="nav-item-admin"

        >Gestionar Instancias</a

    >

</li>

<li>

    <a id="aListadoUsuarios" class="nav-item-admin"

        >Gestionar usuarios</a

    >

</li>

<li>

    <a id="aReportes" class="nav-item-admin">Reportes</a>

</li>

<!-- FIN ADMIN NAV -->


<!-- USER NAV -->

<li>

    <a id="aMisInstancias" class="nav-item-usuario"

        >Gestionar Instancias</a

    >

</li>

<li>

    <a id="aMisCostos" class="nav-item-usuario">Ver Costos</a>

</li>

```

```

<li>

    <a id="aAlquilar" class="nav-item-usuario">Alquilar Instancias</a>

</li>

<!-- FIN USER NAV -->


<!-- COMPARTIDO -->

<li>

    <a id="aCerrarSesion">Cerrar Sesión</a>

</li>

</ul>

</div>

<div id="contenido">

    <div id="divLogin">

        <h1>Inicio de sesion</h1>

        <label for="txtNomUsu">Ingrese nombre de usuario</label>

        <input type="text" id="txtNomUsu" />

        <label for="txtPassword">Ingrese contrasena</label>

        <input type="password" id="txtPassword" />

        <input type="button" id="btnIniciarSesion" value="Iniciar Sesion" />

        <input type="button" id="btnRegistrarse" value="Registrarse" />

        <p id="pMensajesLogin"></p>

    </div>

```

```

<div id="divRegistroUsuario" style="display: none">

    <h1>Registrarse</h1>

    <label for="txtNombreUsuarioRegistro">Nombre</label>

    <input type="text" id="txtNombreUsuarioRegistro" />

    <label for="txtApellidoRegistro">Apellido</label>

    <input type="text" id="txtApellidoRegistro" />

    <label for="txtNomUsuRegistro">Nombre usuario</label>

    <input type="text" id="txtNomUsuRegistro" />

    <label for="txtContrasenaRegistro">Contraseña</label>

    <input type="text" id="txtContrasenaRegistro" />

    <label for="txtTarjeta">Tarjeta de credito</label>

    <input type="text" id="txtTarjeta" />

    <label for="txtCvcTarjeta">Codigo CVC Tarjeta</label>

    <input type="text" id="txtCvcTarjeta" />

    <input type="button" id="btnCrearUsuario" value="Crear usuario" />

    <input type="button" id="btnVolverLogin" value="Volver al login" />

    <p id="pInfoRegistroUsuario"></p>

</div>


<div id="divContenidoAdministrador" style="display: none">

    <div id="divAdminStockInstancias"></div>

    <div id="divAdminAdministrarUsuario"></div>

    <div id="divAdminReportes"></div>

</div>

```

```

<div id="divContenidoUsuario" style="display: none">

    <div id="divUsuarioMisInstancias"></div>

    <div id="divUsuarioMisCostos"></div>

    <div id="divUsuarioAlquilar">

        <h2>Alquilar Instancias</h2>

        <label for="slcOpcionesAlquiler" style="margin-left: 2px"

            >Categoría</label

        >

        <select

            id="slcOpcionesAlquiler"

            style="width: 100%; margin-top: 4px"

        >

            <option value="Optimizadas para computo">

                Optimizadas para Cómputo

            </option>

            <option value="Optimizadas para memoria">

                Optimizadas para Memoria

            </option>

            <option value="Optimizadas para almacenamiento">

                Optimizadas para Almacenamiento

            </option>

        </select>

        <div id="divUsuarioAlquilarContenedorTabla"></div>

```

```
        </div>

    </div>

</div>

<div id="footer">

    <a href="https://www.youtube.com/watch?v=-HB5XU18IsU">Footer</a>

</div>

</div>

<script src="codigo/libreria.js"></script>

<script src="codigo/clases.js"></script>

<script src="codigo/sistema.js"></script>

<script src="codigo/ui.js"></script>

</body>

</html>
```

3.2. clases.js

```
class UsuarioAdministrador {  
  
    /**  
     * @param {string} pNombreUsuario  
     * @param {string} pContrasena  
     */  
  
    constructor(pNombreUsuario, pContrasena) {  
  
        /**  
         * @type {string}  
         */  
  
        this.nombreUsuario = pNombreUsuario;  
  
        /**  
         * @type {string}  
         */  
  
        this.contrasena = pContrasena;  
    }  
}  
  
class UsuarioComun {  
  
    static contadorID = 1;  
  
    /**  
     * @param {string} pNombre  
     * @param {string} pApellido
```

```

* @param {string} pNomUsu

* @param {string} pContrasena

* @param {string} pNroTjt

* @param {number} pCvcTjt

*/

constructor(pNombre, pApellido, pNomUsu, pContrasena, pNroTjt, pCvcTjt) {

    /**

    * @type {number}

    */

    this.ID = UsuarioComun.contadorID;

    /**

    * @type {string}

    */

    this.nombre = pNombre;

    /**

    * @type {string}

    */

    this.apellido = pApellido;

    /**

    * @type {string}

    */

    this.nombreUsuario = pNomUsu;

    /**

    * @type {string}

```

```

    */

    this.contrasena = pContrasena;

    /**
     * @type {string}
     */

    this.nroTarjeta = pNroTjt;

    /**
     * @type {string}
     */

    this.cvcTarjeta = pCvcTjt;

    /**
     * @type {"pendiente" | "activo" | "bloqueado"}
     */

    this.estado = "pendiente";

    UsuarioComun.contadorID++;

}

}

class TipoInstancia {

    /**
     * @param {string} pTipo nuevo tipo a crear @example "c7small"

```



```

    * @param {number} pCostoAlquiler costo que se cobra cada vez que un usuario
presiona `Alquilar`

    * @param {number} pCostoEncendido costo que se cobra cada vez que un usuario
presiona `Encender`, no se cobra la primera

    * @param {string} pCategoria una categoría previamente cargada a
`sistema.categorias` @example "Optimizadas para computo"

*/

constructor(pTipo, pCostoAlquiler, pCostoEncendido, pCategoria) {

    /**

    * el nombre del tipo @example c7small

    * @type {string}

    */

    this.tipo = pTipo;

    /**

    * @type {number}

    */

    this.costoAlquiler = pCostoAlquiler;

    /**

    * @type {number}

    */

    this.costoEncendido = pCostoEncendido;

    /**

    * @type {string}

    */

```

```

    this.categoria = pCategoria;

}

/**
 * costoAlquiler + (cantidadEncendidos -1) * costoEncendido
 *
 * @param {number} cantidadEncendidos
 * @returns {number}
 */
calcularCostos(cantidadEncendidos) {
    return this.costoAlquiler + (cantidadEncendidos - 1) * this.costoEncendido;
}
}

class MaquinaVirtual {

    static contadorID = 1;

    /**
     * @param {TipoInstancia} pTipo
     */
    constructor(pTipo) {

        /**
         * objeto TipoInstancia - Preset de Informacion sobre un tipo X
         *
         * @type {TipoInstancia}
         */
    }
}

```

```

this.tipo = pTipo;

/**
 * @type {number}
 */

this.ID = MaquinaVirtual.contadorID;

/**
 * @type {"APAGADA" | "ENCENDIDA"}
 */

this.estado = "APAGADA";


/**
 * Para efectuar soft deletes en lugar de borrados permanentes
 * @type {boolean}
 */

this.habilitada = true;

MaquinaVirtual.contadorID++;

this.tipo.categoria

}

/**

```

```

    * @returns {boolean} `false` si la maquina ya estaba prendida, `true` si estaba
    APAGADA.

    */

    encender() {

        if (this.estado !== "APAGADA") {

            return false;

        }

        this.estado = "ENCENDIDA";

        return true;

    }

    /**

    * Apaga la maquina virtual.

    */

    apagar() {

        this.estado = "APAGADA";

    }

    /**

    * baja lógica para evitar borrados permanentes.

    */

    deshabilitar() {

        this.habilitada = false;

```

```

    }

}

/**
 * Un `Alquiler` se da entre un `UsuarioComun` y una `MaquinaVirtual`.
 *
 * @see {@link UsuarioComun}
 * @see {@link MaquinaVirtual}
 */
class Alquiler {

    /**
     * @param {number} instancia
     * @param {string} nomUsuario
     */
    constructor(idInstancia, nomUsuario) {

        /**
         * @type {number}
         */
        this.idInstancia = idInstancia;

        /**
         * @type {string}
         */
        this.nomUsuario = nomUsuario;
    }
}

```

```

/**
 * Cuantas veces el usuario encendió esta instancia durante su alquiler.
 *
 * @type {number}
 */

    this.contadorEncendido = 0; // al alquilar una instancia esta se enciende
    automáticamente, no se el cobra primer encendido.

/**
 * Determina si este alquiler está activo o no.
 *
 * Cuando se bloquea a un usuario, sus instancias alquiladas vuelven a estar
    disponibles para alquilar por otros usuarios. Pero no queremos borrar el alquiler
    del arreglo de alquileres, porque queremos mantener un registro de todos los
    alquileres que se hicieron. Y así poder determinar apropiadamente los ingresos
    finales del sistema. Usamos una baja lógica para lograr el efecto deseado.
 *
 * @type {boolean}
 */

    this.activo = true;
}

desactivar() {
    this.activo = false;
}
}

```

3.3. sistema.js

```
class Sistema {

  constructor() {

    /**

     * @type {UsuarioAdministrador[]}

     */

    this.arrayUsuariosAdmin = [];

    /**

     * @type {UsuarioComun[]}

     */

    this.arrayUsuariosComunes = [];

    /**

     * @type {TipoInstancia[]}

     */

    this.arrayTiposInstancia = [];

    /**

     * @type {MaquinaVirtual[]}

     */

    this.arrayInstancias = [];

    /**

     * @type {Alquiler[]}

     */

    this.arrayAlquileres = [];

  }

}
```

```

/**
 * @type {string[]} Categorías de instancias.
 */

this.categorias = [

    "Optimizadas para computo",

    "Optimizadas para memoria",

    "Optimizadas para almacenamiento",

];

this.precargaDeDatos();

}

/**
 * `true` si las credenciales son válidas, `false` si no.
 * @param {string} pNomUsuario
 * @param {string} pContrasena
 * @returns {boolean}
 */

validarLogin(pNomUsuario, pContrasena) {

    let i = 0;

    while (i < this.arrayUsuariosAdmin.length) {

        let unUsuario = this.arrayUsuariosAdmin[i];

        if (unUsuario.nombreUsuario === pNomUsuario) {

```



```

        if (unUsuario.contrasena === pContrasena) {

            return true;

        } else {

            return false;

        }

    }

    i++;

}

let j = 0;

while (j < this.arrayUsuariosComunes.length) {

    let unUsuario = this.arrayUsuariosComunes[j];

    if (unUsuario.nombreUsuario === pNomUsuario) {

        if (unUsuario.contrasena === pContrasena) {

            return true;

        } else {

            return false;

        }

    }

    j++;

}

return false;

```

```

}

/**
 * @param {string} pNomUsuario
 * @returns {boolean}
 */
usuarioEstaActivo(pNomUsuario) {
    let usuario = this.buscarUsuarioObjeto(pNomUsuario);

    if (usuario !== null) {
        if (usuario.estado === "activo") {
            return true;
        }
    }

    return false;
}

/**
 * `true` si el usuario es administrador, `false` si no.
 * @param {string} pNomUsuario
 * @returns {boolean}
 */
esAdmin(pNomUsuario) {
    let esAdmin = false;

```

```

for (let i = 0; i < this.arrayUsuariosAdmin.length; i++) {

    if (this.arrayUsuariosAdmin[i].nombreUsuario === pNomUsuario) {

        return true;

    }

}

return esAdmin;

}

/**
 * Crea un usuario común con los datos provistos. Pendiente de aprobación.
 *
 * En caso de que ya exista un usuario o admin con el mismo nombre de usuario,
false.
 *
 * En caso de que no validen los datos proporcionados, false
 *
 * En caso de que esté todo bien, true
 *
 * @param {string} pNombre
 * @param {string} pApellido
 * @param {string} pNomUsu
 * @param {string} pContrasena
 * @param {number} pNroTjt

```

```

* @param {number} pCvcTjt

* @returns {boolean}

*/

registrarUsuario(pNombre, pApellido, pNomUsu, pContrasena, pNroTjt, pCvcTjt) {

    if (

        !this.validarDatosUsuario(

            pNombre,

            pApellido,

            pNomUsu,

            pContrasena,

            pNroTjt,

            pCvcTjt

        )

    ) {

        return false;

    }

    pNomUsu = pNomUsu.toLowerCase(); // para que sea case insensitive

    let usuario = this.buscarUsuarioObjeto(pNomUsu);

    let admin = this.buscarAdminObjeto(pNomUsu);

    if (usuario === null && admin === null) {

        usuario = new UsuarioComun(

```

```

        pNombre,

        pApellido,

        pNomUsu,

        pContrasena,

        pNroTjt,

        pCvcTjt

    );

    this.arrayUsuariosComunes.push(usuario);

    return true;

}

return false;

}

```

```

validarDatosUsuario(

    pNombre,

    pApellido,

    pNomUsu,

    pContrasena,

    pNroTjt,

    pCvcTjt

) {

    if (

        !hayCaracteres(pNombre) ||

        !hayCaracteres(pApellido) ||
    
```

```

    !hayCaracteres(pNomUsu) ||

    !hayCaracteres(pContrasena) ||

    !hayCaracteres(pNroTjt) ||

    !hayCaracteres(pCvcTjt) ||

    pCvcTjt.length != 3 ||

    isNaN(pCvcTjt) ||

    !this.validarContrasena(pContrasena) ||

    !this.validarTarjeta(pNroTjt) ||

    !this.validarNombreUsuario(pNomUsu)
) {

    return false;

}

return true;

}

/**
 * Formato string alfanumérico, case insensitive.
 *
 * Acepta "." y "_" como caracteres especiales.
 *
 * A modo de ejemplo: martin.luz01.
 *
 * @param {string} pNomUsu
 * @returns {boolean}

```

```

*/

validarNombreUsuario(pNomUsu) {

    pNomUsu = pNomUsu.toLowerCase();

    for (let i = 0; i < pNomUsu.length; i++) {

        let char = pNomUsu.charAt(i);

        let esCharEspecial = char === "." || char === "_";

        let esNumerico = !isNaN(char) && char !== " "; // Number(" ") retorna 0.

        let esLetra = char >= "a" && char <= "z";

        let esValido = esCharEspecial || esNumerico || esLetra;

        if (!esValido) {

            return false;

        }

    }

    return true;

}

/**

    * Debe tener al menos 5 caracteres, al menos una mayúscula, al menos una
    minúscula y al menos un número.

```

```

* @param {string} pContrasena

* @returns {boolean}

*/

validarContrasena(pContrasena) {

    let cumpleLargo = pContrasena.length >= 5;

    let tieneMayus = false;

    let tieneMinus = false;

    let tieneNumero = false;

    for (let i = 0; i < pContrasena.length; i++) {

        let caracterActual = pContrasena.charAt(i);

        if (!isNaN(caracterActual)) {

            tieneNumero = true;

        } else if (caracterActual === caracterActual.toUpperCase()) {

            tieneMayus = true;

        } else if (caracterActual === caracterActual.toLowerCase()) {

            tieneMinus = true;

        }

    }

    return cumpleLargo && tieneMayus && tieneMinus && tieneNumero;

}

/**

```



```

* Dado un nombre de usuario buscará al usuario entre los usuarios comunes.

* @param {string} pNombreUsuario

* @returns {?UsuarioComun}

*/

buscarUsuarioObjeto(pNombreUsuario) {

    let i = 0;

    while (i < this.arrayUsuariosComunes.length) {

        let unUsuario = this.arrayUsuariosComunes[i];

        if (unUsuario.nombreUsuario === pNombreUsuario) {

            return unUsuario;

        }

        i++;

    }

    return null;

}

/**

* Dado un nombre de usuario buscará al usuario entre los administradores.

* @param {string} pNombreUsuario

* @returns {?UsuarioAdministrador}

*/

buscarAdminObjeto(pNombreUsuario) {

```

```

let i = 0;

while (i < this.arrayUsuariosAdmin.length) {

    let unAdmin = this.arrayUsuariosAdmin[i];

    if (unAdmin.nombreUsuario === pNombreUsuario) {

        return unAdmin;

    }

    i++;

}

return null;

}

/**
 * Valida una tarjeta de crédito usando el algoritmo de luhn.
 *
 * La tarjeta debe cumplir el formato xxxx-xxxx-xxxx-xxxx.
 *
 * De forma tal que cada x es un número, y estos sean 16 en total.
 *
 * Esta función prepara el string a usar en la función `validarLuhn` removiendo
los guiones.
 *
 * @param {string} pNumeroTarjeta

```

```

* @returns {boolean}

*/

validarTarjeta(pNumeroTarjeta) {

    if (pNumeroTarjeta.length !== 19) {

        return false;

    }

    if (

        pNumeroTarjeta.charAt(4) !== "-" ||

        pNumeroTarjeta.charAt(9) !== "-" ||

        pNumeroTarjeta.charAt(14) !== "-"

    ) {

        return false;

    }

    //aca vamos a tener eventualmente solo los numeros de la tarjeta sin los guiones

    let tarjetaFormateada = this.sacarGuiones(pNumeroTarjeta);

    if (tarjetaFormateada.length !== 16) {

        return false;

    }

    if (isNaN(tarjetaFormateada)) {

        return false;

```

```

}

if (!this.validarLuhn(tarjetaFormateada)) {

    return false;

}

return true;

}

/**
 * Recibe una tarjeta ya formateada y validada para comprobar si es válida usando
el algoritmo de luhn.
 * @param {string} pNumeroTjtFormateado
 * @returns {boolean}
 */
validarLuhn(pNumeroTjtFormateado) {

    let suma = 0;

    let duplicar = true;

    for (let i = pNumeroTjtFormateado.length - 2; i >= 0; i--) {

        let numeroActual = Number(pNumeroTjtFormateado.charAt(i));

        if (duplicar) {

            let duplicado = numeroActual * 2;

            if (duplicado > 9) {

                duplicado = duplicado - 9;

```

```

    }

    suma += duplicado;

} else {

    suma += numeroActual;

}

duplicar = !duplicar;

}

let por9 = suma * 9;

let mod10Final = por9 % 10;

let ultimoDigitoDeTarjeta = Number(

    pNumeroTjtFormateado.charAt(pNumeroTjtFormateado.length - 1)

);

let luhnValido = false;

if (ultimoDigitoDeTarjeta === mod10Final) {

    luhnValido = true;

}

return luhnValido;

}

sacarGuiones(pNumeroTarjeta) {

```

```

let tarjetaSinGuiones = ""; //4563685

for (let i = 0; i < pNumeroTarjeta.length; i++) {

    if (pNumeroTarjeta.charAt(i) !== "-") {

        tarjetaSinGuiones += pNumeroTarjeta.charAt(i);

    }

}

return tarjetaSinGuiones;

}

/**
 * Marca como "activo" al usuario dado.
 * @param {string} pNombreUsuario
 * @returns {boolean}
 */
activarUsuario(pNombreUsuario) {

    let usuario = this.buscarUsuarioObjeto(pNombreUsuario);

    if (usuario !== null) {

        usuario.estado = "activo";

        return true;

    }

    return false;
}

```

```

}

/**
 * Marca como "bloqueado" al usuario dado.
 *
 * @param {string} pNombreUsuario
 */
bloquearUsuario(pNombreUsuario) {
    let usuario = this.buscarUsuarioObjeto(pNombreUsuario);

    if (usuario !== null) {
        usuario.estado = "bloqueado";

        this.liberarInstanciasAlquiladas(pNombreUsuario);

        return true;
    }

    return false;
}

/**
 * Remueve los alquileres efectuados del usuario dado.
 *
 * Esta función es llamada al bloquear un usuario. NO debe usarse en otro
contexto.
 *
 * @param {string} pNombreUsuario
 *
 * @private

```

```

*/

liberarInstanciasAlquiladas(pNombreUsuario) {

    for (let i = 0; i < this.arrayAlquileres.length; i++) {

        let alquiler = this.arrayAlquileres[i];

        if (alquiler.nomUsuario === pNombreUsuario) {

            alquiler.desactivar();

            let instancia = this.buscarInstanciaPorID(alquiler.idInstancia);

            instancia.apagar();

        }

    }

}

/**
 * Agrega una instancia del tipo dado.
 * @param {TipoInstancia} pTipo
 */

agregarInstancia(pTipo) {

    let instancia = new MaquinaVirtual(pTipo);

    this.arrayInstancias.push(instancia);

}

/**

```



```

* Intenta agregar la cantidad de instancias indicadas para el tipo dado.

* En caso de que la cantidad no sea un número entero válido, no se agregará
ninguna instancia.

*

* Ejemplo: `agregarInstancias("c7small", 5)` agregará 5 instancias del tipo
"c7small".

*

* Ejemplo: `agregarInstancias("c7small", 5.5)` NO agregará ninguna instancia.

*

* @param {string} pTipo

* @param {number} cantidadAgregar

* @returns {boolean}

*/

agregarInstancias(pTipo, cantidadAgregar) {

    if (

        !esNumeroEnteroValido(cantidadAgregar) &&

        !esUnNumeroRazonable(cantidadAgregar)

    ) {

        return false;

    }

    let tipoI = this.buscarTipo(pTipo);

    if (!tipoI) {

        return false;

    }

```

```

}

for (let i = 0; i < cantidadAgregar; i++) {

    this.agregarInstancia(tipoI);

}

return true;

}

/**
 * Dada una categoría, devuelve todas las instancias que pertenecen a dicha
categoría.
 * @param {string} categoria
 * @returns {MaquinaVirtual[]}
 */
buscarInstanciasPorCategoria(categoria) {

    let instancias = [];

    for (let i = 0; i < this.arrayInstancias.length; i++) {

        let instancia = this.arrayInstancias[i];

        if (!instancia.habilitada) continue;

        if (instancia.tipo.categoria === categoria) {

            instancias.push(instancia);

```

```

    }

}

return instancias;

}

/**
 * Consigue las máquinas disponibles en el sistema dado un tipo.
 *
 * @param {string} tipo
 * @returns {MaquinaVirtual[]}
 */
buscarInstanciasLibresPorTipo(tipo) {

    let libres = [];

    for (let i = 0; i < this.arrayInstancias.length; i++) {

        let instancia = this.arrayInstancias[i];

        if (

            instancia.tipo.tipo === tipo &&

            instancia.habilitada &&

            this.maquinaEstaLibre(instancia.ID)

        ) {

            libres.push(instancia);

        }

    }

}

```

```

    }

    return libres;
}

/**
 * Buscará por el criterio proporcionado, en `this.arrayInstancias`, o en
`arrInstancias` de ser provisto.
 *
 * @param {string} tipo - nombre del tipo @example "c7small"
 *
 * @param {?MaquinaVirtual[]} arrInstancias Si no es provisto, usará
this.arrayInstancias.
 *
 * @returns {MaquinaVirtual[]}
 */
buscarInstanciasPorTipo(tipo, arrInstancias = null) {
    arrInstancias = arrInstancias || this.arrayInstancias;

    let instancias = [];

    for (let i = 0; i < arrInstancias.length; i++) {
        let instancia = arrInstancias[i];

        if (instancia.tipo.tipo === tipo && instancia.habilitada) {
            instancias.push(instancia);
        }
    }
}

```

```

    }

    return instancias;
}

/**
 * Intenta reducir el stock de instancias disponibles con una baja lógica ( @see
{MaquinaVirtual.deshabilitar()} ).
 *
 * En caso de que la cantidad no sea un número entero válido, no se reducirá el
stock.
 *
 * Ejemplo: `reducirStockDisponible("c7small", 5)` removerá 5 instancias del tipo
"c7small".
 *
 * Ejemplo: `reducirStockDisponible("c7small", 5.5)` NO removerá ninguna
instancia.
 * @param {string} tipo
 * @param {number} cantidadAReducir Cantidad de instancias a sacar de las
`libres`.
 * @returns {boolean} `false` si no se pudo, `true` si se pudo.
 */
reducirStockDisponible(tipo, cantidadAReducir) {
    if (!esNumeroEnteroValido(cantidadAReducir)) {
        return false;
    }
}

```

```

let libres = this.buscarInstanciasLibresPorTipo(tipo);

if (libres.length < cantidadAReducir) {

    return false;

}

const instanciasAEliminar = libres.splice(0, cantidadAReducir);

// buscar el indice de esta instancia en el arreglo original

for (let i = 0; i < instanciasAEliminar.length; i++) {

    const instanciaLibre = instanciasAEliminar[i];

    for (let j = 0; j < this.arrayInstancias.length; j++) {

        const instancia = this.arrayInstancias[j];

        if (instanciaLibre.ID === instancia.ID) {

            instancia.apagar(); // asegurarnos de que quede apagada.

            // this.arrayInstancias.splice(j, 1); // borrado permanente

            instancia.deshabilitar(); // soft delete

        }

    }

}

return true;

```

```

}

/**
 * Recorre this.arrayAlquileres, si encuentra una maquina con la ID proporcionada,
significará que está alquilada actualmente.
 * @param {number} id
 * @returns {boolean}
 */
maquinaEstaLibre(id) {
  for (let i = 0; i < this.arrayAlquileres.length; i++) {
    let alquiler = this.arrayAlquileres[i];

    if (alquiler.idInstancia === id && alquiler.activo) {
      return false;
    }
  }

  return true;
}

/**
 * Le asigna una instancia del tipo provisto al usuario dado en alquiler, si este
está habilitado Y si hay instancias disponibles.
 * @param {string} tipoInstancia
 * @param {string} nomUsuario

```

```

* @returns {boolean}

*/

alquilarInstancia(tipoInstancia, nomUsuario) {

    if (!this.usuarioEstaActivo(nomUsuario)) {

        return false;

    }

    let instancias = this.buscarInstanciasLibresPorTipo(tipoInstancia);

    if (!arrayTieneElementos(instancias)) {

        return false;

    }

    let alquiler = new Alquiler(instancias[0].ID, nomUsuario);

    this.arrayAlquileres.push(alquiler);

    this.encenderInstancia(instancias[0].ID); // al alquilar una instancia esta se
enciende automaticamente, no se cobra primer encendido.

    return true;

}

/**

* Busca un alquiler por estado, id de instancia y nombre de usuario.

*

```


** Ejemplo: sistema.buscarAlquiler(true, 1, "mateo") buscará un alquiler activo de la instancia 1, alquilada por el usuario "mateo".*

** @param {boolean} activo*

** @param {number} idInstancia*

** @param {string} nomUsuario*

** @returns*

**/*

```
buscarAlquiler(activo, idInstancia, nomUsuario) {
```

```
  for (let i = 0; i < this.arrayAlquileres.length; i++) {
```

```
    let alquiler = this.arrayAlquileres[i];
```

```
    if (alquiler.idInstancia === idInstancia && alquiler.activo === activo) {
```

```
      if (!nomUsuario || alquiler.nomUsuario === nomUsuario) {
```

```
        return alquiler;
```

```
      }
```

```
    }
```

```
  }
```

```
  return null;
```

```
}
```

*/***

** Consigue todas las instancias alquiladas por el usuario dado.*

** @param {string} nomUsuario*

```

* @returns {MaquinaVirtual[]}

*/

buscarInstanciasDeUsuario(nomUsuario) {

  /**

  * @type {MaquinaVirtual[]}

  */

  let arr = [];

  for (let i = 0; i < this.arrayAlquileres.length; i++) {

    let alquiler = this.arrayAlquileres[i];

    if (alquiler.nomUsuario === nomUsuario) {

      let instancia = this.buscarInstanciaPorID(alquiler.idInstancia);

      if (instancia !== null) {

        arr.push(instancia);

      }

    }

  }

  return arr;

}

/**

```

```

* Busca una maquina virtual por su ID.

* @param {number} id

* @returns {?MaquinaVirtual}

*/

buscarInstanciaPorID(id) {

    for (let i = 0; i < this.arrayInstancias.length; i++) {

        let instancia = this.arrayInstancias[i];

        if (instancia.ID === id) {

            return instancia;

        }

    }

    return null;

}

/**

* Intenta prender una maquina virtual por su ID.

* @param {number} id

* @returns {boolean}

*/

encenderInstancia(id) {

    if (!esNumeroEnteroValido(id)) {

        return false;

    }

}

```

```

    let instancia = this.buscarInstanciaPorID(id);

    let alquiler = this.buscarAlquiler(true, id); // debe haber un alquiler activo
para encender la maquina.

    if (instancia === null || alquiler === null) {

        return false;

    }

    let encenderOk = instancia.encender();

    if (!encenderOk) {

        return false;

    }

    alquiler.contadorEncendido++;

    return true;

}

/**
 * Intenta apagar una maquina virtual por su ID.
 *
 * @param {number} id
 *
 * @returns {boolean}
 *
 */
apagarInstancia(id) {

```

```

if (!esNumeroEnteroValido(id)) {

    return false;

}

let instancia = this.buscarInstanciaPorID(id);

if (instancia === null) {

    return false;

}

return instancia.apagar();

}

/**
 * Recaba información de todos los alquileres realizados y devuelve los ingresos
totales para instancias de un tipo dado.
 * @param {string} tipo
 * @returns {number}
 */
ingresosPorTipoDeInstancia(tipo) {

    let ingresos = 0;

    for (let i = 0; i < this.arrayAlquileres.length; i++) {

        let alquiler = this.arrayAlquileres[i];

```

```

    let instancia = this.buscarInstanciaPorID(alquiler.idInstancia);

    if (instancia.tipo.tipo === tipo) {

        ingresos +=

            instancia.tipo.costoAlquiler +

            (alquiler.contadorEncendido - 1) * instancia.tipo.costoEncendido;

    }

}

return ingresos;

}

/**
 * @param {string} pTipo
 * @returns {?TipoInstancia}
 */
buscarTipo(pTipo) {

    for (let i = 0; i < this.arrayTiposInstancia.length; i++) {

        let tipoI = this.arrayTiposInstancia[i];

        if (tipoI.tipo === pTipo) {

            return tipoI;

        }

    }

}

```

```

    }

    return null;
}

/**
 * @param {string} categoria
 * @returns {string[]}
 */
buscarTiposDeCategoria(categoria) {
    let tipos = [];

    for (let i = 0; i < this.arrayTiposInstancia.length; i++) {
        let tipoI = this.arrayTiposInstancia[i];

        if (tipoI.categoria === categoria) {
            tipos.push(tipoI.tipo);
        }
    }

    return tipos;
}

/**

```

```

* @param {string} tipo

* @returns {number}

*/

cantidadAlquileresPorTipo(tipo) {

    let cantidad = 0;

    for (let i = 0; i < this.arrayAlquileres.length; i++) {

        let alquiler = this.arrayAlquileres[i];

        let instancia = this.buscarInstanciaPorID(alquiler.idInstancia);

        if (instancia.tipo.tipo === tipo) {

            cantidad++;

        }

    }

    return cantidad;

}

/**

* @param {string} pCategoria

* @returns {boolean}

*/

esCategoriaValida(pCategoria) {

```



```

for (let i = 0; i < this.categorias.length; i++) {

    if (this.categorias[i] === pCategoria) {

        return true;

    }

}

return false;

}

/**
 * Agrega un nuevo TipoInstancia a this.arrayTiposInstancia.
 * @param {string} pTipo el tipo a agregar
 * @param {number} pCostoAlquiler costo a cobrar al clickear en "Alquilar"
 * @param {number} pCostoEncendido costo al hacer click en "Encender"
 * @param {string} pCategoria categoria a la que corresponde este tipo.
 */
agregarTipoInstancia(pTipo, pCostoAlquiler, pCostoEncendido, pCategoria) {

    if (

        !hayCaracteres(pTipo) ||

        isNaN(pCostoAlquiler) ||

        isNaN(pCostoEncendido) ||

        !this.esCategoriaValida(pCategoria)

    ) {

        console.log("datos para agregar tipo de instancia invalidos");
    }
}

```

```

        return false;
    }

    let tipoI = new TipoInstancia(
        pTipo,
        pCostoAlquiler,
        pCostoEncendido,
        pCategoria
    );

    this.arrayTiposInstancia.push(tipoI);
}

precargaDeDatos() {
    // precarga de usuarios administradores - 5

    let admin1 = new UsuarioAdministrador("gaston", "1234La");
    let admin2 = new UsuarioAdministrador("mateo", "1234La");
    let admin3 = new UsuarioAdministrador("matt1", "1234La");
    let admin4 = new UsuarioAdministrador("lean1", "1234La");
    let admin5 = new UsuarioAdministrador("admin", "1234La");

    this.arrayUsuariosAdmin.push(admin1, admin2, admin3, admin4, admin5);

    // precarga de usuarios comunes - 5

    this.registrarUsuario(

```

```
"mateo",

"carriqui",

"matteo",

"1234La",

"4539-6253-0847-8250",

"323"

);

this.registrarUsuario(

    "leandro",

    "guzman",

    "lean.g",

    "1234La",

    "4539-6253-0847-8250",

    "323"

);

this.registrarUsuario(

    "Gaston",

    "Sanguinetti",

    "g.sangui",

    "1234La",

    "4539-6253-0847-8250",

    "323"

);

this.registrarUsuario(
```

```

        "Mateo",

        "Sanguinetti",

        "m.sangui",

        "1234La",

        "4539-6253-0847-8250",

        "323"

    );

    this.registrarUsuario(

        "Soy",

        "Yo",

        "soy.yo",

        "1234La",

        "4539-6253-0847-8250",

        "323"

    );

    for (let i = 0; i < this.arrayUsuariosComunes.length; i++) {

        this.activarUsuario(this.arrayUsuariosComunes[i].nombreUsuario);

    }

    this.registrarUsuario(

        "Bruno",

        "Perez",

        "Pru.Per1",

        "1234La",

```

```

        "4539-6253-0847-8250",

        "323"

    );

    this.registrarUsuario(

        "Gaston",

        "Sanguinetti",

        "Gas.Ton1",

        "1234La",

        "4539-6253-0847-8250",

        "323"

    );

    // precarga de tipos de instancia

    this.agregarTipoInstancia("c7small", 20, 2.5, "Optimizadas para computo");
    this.agregarTipoInstancia("c7medium", 30, 3.5, "Optimizadas para computo");
    this.agregarTipoInstancia("c7large", 50, 6.0, "Optimizadas para computo");
    this.agregarTipoInstancia("r7small", 35, 4.0, "Optimizadas para memoria");
    this.agregarTipoInstancia("r7medium", 50, 6.5, "Optimizadas para memoria");
    this.agregarTipoInstancia("r7large", 60, 7.0, "Optimizadas para memoria");
    this.agregarTipoInstancia(

        "i7medium",

        30,

        3.5,

```

```

        "Optimizadas para almacenamiento"

    );

    this.agregarTipoInstancia(

        "i7large",

        50,

        6.5,

        "Optimizadas para almacenamiento"

    );


    // precarga de stock por tipo de instancia

    this.agregarInstancias("c7small", 3);

    this.agregarInstancias("c7medium", 5);

    this.agregarInstancias("c7large", 3);

    this.agregarInstancias("r7small", 7);

    this.agregarInstancias("r7medium", 5);

    this.agregarInstancias("r7large", 2);

    this.agregarInstancias("i7medium", 3);

    this.agregarInstancias("i7large", 3);

    /**

     * ALQUILARLE MAQUINAS A USUARIOS - 10 Alquileres.

    */

    this.alquilarInstancia(

        "c7small",

        this.arrayUsuariosComunes[0].nombreUsuario

```

```
);

this.alquilarInstancia(

    "r7small",

    this.arrayUsuariosComunes[0].nombreUsuario

);

this.alquilarInstancia(

    "c7small",

    this.arrayUsuariosComunes[1].nombreUsuario

);

this.alquilarInstancia(

    "c7medium",

    this.arrayUsuariosComunes[1].nombreUsuario

);

this.alquilarInstancia(

    "c7large",

    this.arrayUsuariosComunes[2].nombreUsuario

);

this.alquilarInstancia(

    "r7small",

    this.arrayUsuariosComunes[2].nombreUsuario

);

this.alquilarInstancia(

    "r7medium",

    this.arrayUsuariosComunes[3].nombreUsuario
```

```

);

this.alquilarInstancia(

    "r7medium",

    this.arrayUsuariosComunes[3].nombreUsuario

);

this.alquilarInstancia(

    "i7medium",

    this.arrayUsuariosComunes[4].nombreUsuario

);

this.alquilarInstancia(

    "i7medium",

    this.arrayUsuariosComunes[4].nombreUsuario

);

}

}

```

3.4. ui..js

```

window.addEventListener("load", inicio);

```



```

/**
 * @type {Sistema}
 * @global
 */
let sistema = new Sistema();

/**
 * @type {?UsuarioComun}
 * @global
 */
let usuarioLogeado = null;

/**
 * Filtro a aplicar para la vista de instancias alquiladas de los usuarios.
 * @type {"APAGADA" | "ENCENDIDA" | "todas"}
 * @global
 */
let filtroInstanciasUsuario = "todas";

/**
 * Filtro a aplicar para la vista de alquileres, para los usuarios.
 * @type {INSTANCIA_CATEGORIA}
 * @global
 */

```

```

let filtroInstanciasAlquilar = "Optimizadas para computo";

function inicio() {

    // USUARIO UI - Usuarios se registran

    document

        .querySelector("#btnRegistrarse")

        .addEventListener("click", mostrarPantallaRegistro);

    // USUARIO UI - Desde la pantalla de registro, se puede volver al login.

    document

        .querySelector("#btnVolverLogin")

        .addEventListener("click", mostrarPantallaLogin);

    // USUARIO UI - Desde la pantalla de login, se puede ir al registro.

    document

        .querySelector("#btnCrearUsuario")

        .addEventListener("click", registroUsuarioUI);

    // USUARIO Y ADMIN - Ambos se pueden logear desde la misma pantalla

    document

        .querySelector("#btnIniciarSesion")

        .addEventListener("click", iniciarSesionUI);

    // USUARIO Y ADMIN - Ambos pueden cerrar sesion

    document

        .querySelector("#aCerrarSesion")

        .addEventListener("click", cerrarSesionUI);

```

```
// ADMIN UI - Administrador puede ver y operar sobre los usuarios
```

```
document
```

```
.querySelector("#aListadoUsuarios")
```

```
.addEventListener("click", verListadoDeUsuarios);
```

```
// ADMIN UI - Administrador puede modificar el stock de instancias
```

```
document
```

```
.querySelector("#aGestionarInstancias")
```

```
.addEventListener("click", verListadoDeInstanciasAdmin);
```

```
// ADMIN UI - Administrador puede ver un reporte de los ingresos del sistema
```

```
document
```

```
.querySelector("#aReportes")
```

```
.addEventListener("click", verReportesAdmin);
```

```
// USUARIO UI - Usuario puede ver y operar sobre sus instancias
```

```
document
```

```
.querySelector("#aMisInstancias")
```

```
.addEventListener("click", verMisInstanciasUsuario);
```

```
// USUARIO UI - Usuario puede ver sus costos
```

```
document.querySelector("#aMisCostos").addEventListener("click", verMisCostos);
```

```

// USUARIO UI - Usuario puede ver maquinas para alquilar

document

    .querySelector("#aAlquilar")

    .addEventListener("click", verOpcionesAlquilerUsuario);

// USUARIO UI - Usuario puede filtrar maquinas para alquilar

document

    .querySelector("#slcOpcionesAlquiler")

    .addEventListener("change", cambiarFiltroAlquilerUsuario);

// La pantalla por defecto es el login - no landing page (bajo presupuesto).

mostrarPantallaLogin();

}

function mostrarPantallaLogin() {

    mostrarElemento("divLogin");

    ocultarElemento("divRegistroUsuario");

    ocultarElemento("divContenidoAdministrador");

    ocultarElemento("divContenidoUsuario");

    ocultarElemento("cabezal");

}

function iniciarSesionUI() {

    ocultarNavAdmin();

```

```

ocultarNavUsuario();

ocultarElemento("divContenidoAdministrador");

ocultarElemento("divContenidoUsuario");


let nomUsuario = obtenerValorDeUnElementoHTML("txtNomUsu");

let contrasena = obtenerValorDeUnElementoHTML("txtPassword");


if (sistema.validarLogin(nomUsuario, contrasena)) {

    let esAdmin = sistema.esAdmin(nomUsuario);


    if (esAdmin) {

        document.querySelector("#divContenidoAdministrador").style.display =

            "block";

        mostrarNavAdmin();

        mostrarElemento("divContenidoAdministrador");

        verListadoDeUsuarios(); // por defecto mostrarle la lista de usuarios al
admin.

    } else {

        let usuario = sistema.buscarUsuarioObjeto(nomUsuario);

        if (usuario.estado === "bloqueado") {

            imprimirEnHtml("pMensajesLogin", "Su usuario fue bloqueado.");

            return;

        } else if (usuario.estado === "pendiente") {

```

```

        imprimirEnHtml(

            "pMensajesLogin",

            "Su registro se encuentra pendiente de aprobación."

        );

        return;

    }

    usuarioLogeado = usuario;

    filtroInstanciasUsuario = "todas"; // resetear el filtro.

    filtroInstanciasAlquilar = "Optimizadas para computo"; // resetear el filtro.

    mostrarNavUsuario();

    mostrarElemento("divContenidoUsuario");

    verMisInstanciasUsuario(); // por defecto mostrarle sus instancias al usuario

}

mostrarElemento("cabezal");

ocultarElemento("divLogin");

limpiarUnCampoDeTexto("txtNomUsu");

limpiarUnCampoDeTexto("txtPassword");

limpiarUnElemento("pMensajesLogin");

} else {

    imprimirEnHtml("pMensajesLogin", "Usuario y/o contraseña incorrecto");

}

}

```

```

function registroUsuarioUI() {

    let nombre = obtenerValorDeUnElementoHTML("txtNombreUsuarioRegistro");

    let apellido = obtenerValorDeUnElementoHTML("txtApellidoRegistro");

    let nombreUsuario =

        obtenerValorDeUnElementoHTML("txtNomUsuRegistro").toLowerCase(); // case
insensitive

    let contrasena = obtenerValorDeUnElementoHTML("txtContrasenaRegistro");

    let tarjeta = obtenerValorDeUnElementoHTML("txtTarjeta");

    let cvcTarjeta = obtenerValorDeUnElementoHTML("txtCvcTarjeta");


    let erroresValidacion = validarDatosRegistroUsuario(

        nombre,

        apellido,

        nombreUsuario,

        contrasena,

        tarjeta,

        cvcTarjeta

    );


    if (!arrayTieneElementos(erroresValidacion)) {

        if (sistema.buscarUsuarioObjeto(nombreUsuario) !== null) {

            imprimirEnHtml(

                "pInfoRegistroUsuario",

```

```

        "El nombre de usuario ya existe en el sistema"

    );

    return;
}

let registroOk = sistema.registrarUsuario(

    nombre,

    apellido,

    nombreUsuario,

    contrasena,

    tarjeta,

    cvcTarjeta

);

if (registroOk) {

    limpiarUnCampoDeTexto("txtNombreUsuarioRegistro");

    limpiarUnCampoDeTexto("txtApellidoRegistro");

    limpiarUnCampoDeTexto("txtNomUsuRegistro");

    limpiarUnCampoDeTexto("txtContrasenaRegistro");

    limpiarUnCampoDeTexto("txtTarjeta");

    limpiarUnCampoDeTexto("txtCvcTarjeta");

    imprimirEnHtml(

        "pInfoRegistroUsuario",

```



```

        "Usuario creado exitosamente. Pendiente de aprobación"

    );

} else {

    imprimirEnHtml("pInfoRegistroUsuario", "Error al guardar el usuario");

}

} else {

    let erroresMostrar = "";

    for (let i = 0; i < erroresValidacion.length; i++) {

        erroresMostrar += erroresValidacion[i] + "<br>";

    }

    imprimirEnHtml("pInfoRegistroUsuario", erroresMostrar);

}

}

function validarDatosRegistroUsuario(

    pNombre,

    pApellido,

    pNomUsu,

    pContrasena,

    pNroTjt,

    pCvcTjt

) {

    let arrayErrores = [];

    if (!hayCaracteres(pNombre)) {

```

```

    arrayErrores.push("Debe ingresar nombre.");

}

if (!hayCaracteres(pApellido)) {

    arrayErrores.push("Debe ingresar apellido.");

}

if (!hayCaracteres(pNomUsu)) {

    arrayErrores.push("Debe ingresar nombre de usuario.");

}


if (!sistema.validarNombreUsuario(pNomUsu)) {

    arrayErrores.push(

        "Nombre de usuario inválido. Solo se admiten números, letras, puntos y guiones
bajos"

    );

}

if (!hayCaracteres(pContrasena) || !sistema.validarContrasena(pContrasena)) {

    arrayErrores.push(

        "Debe ingresar una contraseña válida. Mínimo de 5 caracteres, contando con al
menos una mayúscula, una minúscula y un número."

    );

}

if (!hayCaracteres(pNroTjt) || !sistema.validarTarjeta(pNroTjt)) {

    arrayErrores.push("Tarjeta invalida.");

}

if (!hayCaracteres(pCvcTjt) || pCvcTjt.length !== 3 || isNaN(pCvcTjt)) {

```

```

        arrayErrores.push("CVC invalido.");
    }

    return arrayErrores;
}

function cerrarSesionUI() {

    usuarioLogeado = null;

    filtroInstanciasAlquilar = "Optimizadas para computo";

    document.querySelector("#slcOpcionesAlquiler").value =

        filtroInstanciasAlquilar;

    filtroInstanciasUsuario = "todas";

    mostrarPantallaLogin();
}

function mostrarPantallaRegistro() {

    ocultarElemento("divLogin");

    mostrarElemento("divRegistroUsuario");
}

function verListadoDeUsuarios() {

    ocultarElemento("divAdminStockInstancias");

    ocultarElemento("divAdminReportes");
}

```

```
let tabla = `  
  
    <h2>Gestión de Usuarios del Sistema</h2>  
  
    <table>  
  
        <tr>  
  
            <th>  
  
                Nombre  
  
            </th>  
  
            <th>  
  
                Apellido  
  
            </th>  
  
            <th>  
  
                Nombre usuario  
  
            </th>  
  
            <th>  
  
                Estado  
  
            </th>  
  
            <th>  
  
                Operar  
  
            </th>  
  
        </tr>  
  
    `;  

```

```

for (let i = 0; i < sistema.arrayUsuariosComunes.length; i++) {

    let usuario = sistema.arrayUsuariosComunes[i];

    let textoAccion = "Activar"; // a mostrar si el usuario esta pendiente

    let clase = "activacionUsuarios"; // clase a dar si esta pendiente

    if (usuario.estado === "activo") {

        textoAccion = "Bloquear"; // a mostrar si el usuario ya fue activado

        clase = "bloqueoUsuarios"; // a mostrar si el usuario ya fue activado

    }

    let botonAccion = `

<td>

    <input

        usuario-nombre="${usuario.nombreUsuario}"

        class="${clase}" id="btnActivarUsuario${usuario.ID}"

        type="button" value="${textoAccion}"

    >

</td>`;

    if (usuario.estado === "bloqueado") {

        // una vez bloqueado el usuario, no hay ninguna accion

        // para ofrecerle al administrador sobre este usuario.

```

```

        botonAccion = "";

    }

    tabla += `

        <tr>

            <td>${usuario.nombre}</td>

            <td>${usuario.apellido}</td>

            <td>${usuario.nombreUsuario}</td>

            <td>${usuario.estado}</td>

            ${botonAccion}

        </tr>

    `;

}

tabla += `</table>`;

imprimirEnHtml("divAdminAdministrarUsuario", tabla);

let botonesDeActivacion = document.querySelectorAll(".activacionUsuarios");
for (let i = 0; i < botonesDeActivacion.length; i++) {

    let idDelBoton = botonesDeActivacion[i].id;

    document

        .querySelector(`#${idDelBoton}`)

        .addEventListener("click", activarUsuarioUI);

}

```

```

let botonesDeBloqueo = document.querySelectorAll(".bloqueoUsuarios");

for (let i = 0; i < botonesDeBloqueo.length; i++) {

    let idDelBoton = botonesDeBloqueo[i].id;

    document

        .querySelector(`#${idDelBoton}`)

        .addEventListener("click", bloquearUsuarioUI);

}

document.querySelector("#divAdminAdministrarUsuario").style.display = "block";
}

function activarUsuarioUI() {

    let nombreUsuario = this.getAttribute("usuario-nombre");

    sistema.activarUsuario(nombreUsuario);

    verListadoDeUsuarios();

}

function bloquearUsuarioUI() {

    let nombreUsuario = this.getAttribute("usuario-nombre");

    sistema.bloquearUsuario(nombreUsuario);

    verListadoDeUsuarios();

```

```

}

function verListadoDeInstanciasAdmin() {

    ocultarElemento("divAdminAdministrarUsuario");

    ocultarElemento("divAdminReportes");


    let html = `

    <h2>Gestión de Instancias</h2>

    <p id="pErroresGIAdmin" style="color: red; font-weight: 500"></p>

    `;


    for (let i = 0; i < sistema.categorias.length; i++) {

        let tabla = generarTablaParaCategoriaInstanciaAdmin(sistema.categorias[i]);

        html += tabla;

    }


    imprimirEnHtml("divAdminStockInstancias", html);


    for (let i = 0; i < sistema.arrayTiposInstancia.length; i++) {

        let tipo = sistema.arrayTiposInstancia[i].tipo;


        document

        .querySelector(`#btnStockGuardar${tipo}`)

```



```

        .addEventListener("click", guardarCambioStockTipoAdmin);

    }

    mostrarElemento("divAdminStockInstancias");
}

/**
 * Genera el html para una `table` que contiene la información
 * de los distintos tipos de instancias de una categoría dada.
 * @param {string} categoria
 */
function generarTablaParaCategoriaInstanciaAdmin(categoria) {

    // buscar todas las instancias para la categoría dada
    // usaremos este arreglo pre-computado como argumento
    // en la llamada a `sistema.buscarInstanciasPorTipo`.
    // logrando optimizar el tiempo de ejecución de la misma.

    let arrInstancias = sistema.buscarInstanciasPorCategoria(categoria);

    // buscar todos los tipos posibles para esta categoría
    // (ej: "c7.small", "c7.medium", etc.)

    let tipos = sistema.buscarTiposDeCategoria(categoria);

    let tabla = `

        <h3>${categoria}</h3>

        <table>

```

```

        <tr>

            <th>

                Tipo

            </th>

            <th>

                Unidades Totales

            </th>

            <th>

                Unidades Alquiladas

            </th>

            <th>

                Unidades Disponibles

            </th>

            <th>

                Modificar Disponibles

            </th>

            <th>

                Confirmar

            </th>

        </tr>

    `;

```

```

// generar una fila para cada tipo en esta categoria.

```

```

for (let i = 0; i < tipos.length; i++) {

```

```

    let tipo = tipos[i];

    let instancias = sistema.buscarInstanciasPorTipo(tipo, arrInstancias);

    tabla += generarFilaParaTipoInstanciaAdmin(tipo, instancias.length);
}

tabla += "</table>";

return tabla;
}

/**
 * Genera el html para una fila de una tabla que contiene la información de un tipo
de instancia dado.
 *
 * @param {string} tipo
 * @param {number} stock
 * @returns
 */
function generarFilaParaTipoInstanciaAdmin(tipo, stock) {

    let libres = sistema.buscarInstanciasLibresPorTipo(tipo);

    /**
     * `stock - libres.length` = cantidad que están alquiladas actualmente
     */

```

```

return `
<tr>

  <td>${formatearTipoUI(tipo)}</ts>

  <td>${stock}</td>

  <td>${stock - libres.length}</td>

  <td>${libres.length}</td>

  <td>

    <input

      type="number"

      min="0"

      id="numStockModificar${tipo}"

      value="${libres.length}"

    />

  </td>

  <td>

    <input

      type="button"

      id="btnStockGuardar${tipo}"

      tipo-instancia="${tipo}"

      value="Guardar"

    />

  </td>

</tr>

`;

```

```

}

/**
 * bindeada al evento "click" en los botones de "Guardar" de la pantalla `Gestión
de Instancias`.
 */

function guardarCambioStockTipoAdmin() {

  let tipo = this.getAttribute("tipo-instancia");

  let valorActual = sistema.buscarInstanciasLibresPorTipo(tipo).length;

  let valorNuevo = document.querySelector(`#numStockModificar${tipo}`).value;

  let ok = false;

  if (esNumeroEnteroValido(valorNuevo) && esUnNumeroRazonable(valorNuevo)) {

    valorNuevoNumerico = Number(valorNuevo);

    if (valorNuevoNumerico > valorActual) {

      let cantidadAgregar = valorNuevoNumerico - valorActual;

      ok = sistema.agregarInstancias(tipo, cantidadAgregar);

    } else if (valorNuevoNumerico < valorActual) {

      let cantidadAReducir = valorActual - valorNuevoNumerico;

      ok = sistema.reducirStockDisponible(tipo, cantidadAReducir);

    } else {

```

```

        ok = true; // mejora la UX, si el admin puso mal un valor, el mismo se
        resettea al valor actual, y si vuelve a hacer click en guardar con dicho valor
        correcto, de esta forma se remueve el mensaje de error previo.

    }

}

verListadoDeInstanciasAdmin(); // mostramos primero el HTML

if (!ok) {

    // Y luego mostramos el mensaje de error, si corresponde.

    // No se puede hacer al reves. De otra forma, el mensaje de error se pierde al
    refrescar la tabla.

    imprimirEnHtml("pErroresGIAdmin", "Valor inválido.");

}

}

/**
 * Callback para el evento "change" del select de la pantalla `Alquilar`.
 * Regenera la tabla de opciones de alquiler según la categoría seleccionada.
 * @see {@link Sistema.buscarTiposDeCategoria}
 */
function cambiarFiltroAlquilerUsuario() {

    let filtro = this.value;

    filtroInstanciasAlquilar = filtro;

    verOpcionesAlquilerUsuario();

```

```

}

/**
 * Muestra todas las instancias que se pueden alquilar
 * para una categoria previamente seleccionada.
 * @default "Optimizadas para computo"
 */

function verOpcionesAlquilerUsuario() {

  ocultarElemento("divUsuarioMisInstancias");

  ocultarElemento("divUsuarioMisCostos");

  let tiposAMostrar = sistema.buscarTiposDeCategoria(filtroInstanciasAlquilar);

  let tabla = `

    <p id="pMensajeAlquiler"></p>

    <table>

      <tr>

        <th>

          Instancia

        </th>

        <th>

          Costo Alquiler

        </th>

        <th>

```

```

        Costo por Encendido

    </th>

    <th>

        Disponible

    </th>

    <th>

        Operar

    </th>

</tr>

`;

for (let i = 0; i < tiposAMostrar.length; i++) {

    let tipo = tiposAMostrar[i];

    let instancias = sistema.buscarInstanciasLibresPorTipo(tipo);

    let textoDisponible = "No";

    let clase = "alquilarInstancia";

    if (instancias.length > 0) {

        textoDisponible = "Si";

    }

    let botonAccion = `

```



```

<td>

    <input

        instancia-tipo="${tipo}"

        class="${clase}" id="btnAlquilarInstancia${tipo}"

        type="button" value="Alquilar"

    >

</td>`;

let tipoI = sistema.buscarTipo(tipo);

tabla += `

    <tr>

        <td>${formatearTipoUI(tipo)}</td>

        <td>US$ ${tipoI.costosAlquiler}</td>

        <td>US$ ${tipoI.costosEncendido}</td>

        <td>${textoDisponible}</td>

        ${botonAccion}

    </tr>

`;

}

tabla += `</table>`;

imprimirEnHtml("divUsuarioAlquilarContenedorTabla", tabla);

```

```

let botonesDeAlquiler = document.querySelectorAll(".alquilarInstancia");

for (let i = 0; i < botonesDeAlquiler.length; i++) {

    let idDelBoton = botonesDeAlquiler[i].id;

    document

        .querySelector(`#${idDelBoton}`)

        .addEventListener("click", alquilerInstanciaUI);

}

mostrarElemento("divUsuarioAlquilar"); // refrescar la tabla
}

/**
 * Callback para el evento "click" de los botones de "Alquilar" de la pantalla
`Alquilar Instancias`.
 * Refresca la tabla de opciones de alquiler luego de ejecutar la operación.
 */
function alquilerInstanciaUI() {

    let tipo = this.getAttribute("instancia-tipo");

    let alquilerOk = sistema.alquilarInstancia(

        tipo,

        usuarioLogeado.nombreUsuario

```

```

);

if (alquilerOk) {

    verOpcionesAlquilerUsuario();

    // NOTE: El mensaje de exito debería mostrarse luego de que se refresque la
tabla.

    // de otra forma, se pierde el mensaje al refrescar la tabla posteriormente.

    imprimirEnHtml(

        "pMensajeAlquiler",

        `La máquina ${formatearTipoUI(tipo)} fue alquilada correctamente.`

    );

} else {

    let el = document.querySelector("#pMensajeAlquiler");

    el.innerHTML = `La máquina ${formatearTipoUI(

        tipo

    )} no tiene stock disponible por el momento. Intente de nuevo más tarde.`;

    el.style.color = "red";

    el.style["font-weight"] = 500;

}

}

```

```

/**
 * Muestra las instancias que el usuario tiene alquiladas.
 * Genera una tabla con las instancias, y permite filtrarlas por estado.
 *
 * Habilita botones para encender y apagar las instancias, acorde a su estado
actual.
 * @default "todas"
 */

function verMisInstanciasUsuario() {

  ocultarElemento("divUsuarioMisCostos");

  ocultarElemento("divUsuarioAlquilar");

  mostrarElemento("divUsuarioMisInstancias");


  let misInstancias = sistema.buscarInstanciasDeUsuario(

    usuarioLogeado.nombreUsuario

  );


  let tabla = `

    <h2>Mis Instancias</h2>


    <select id="slcFiltroMisInstancias" style="width: 100%;">


    </select>


    <table>

```

```

        <tr>

            <th>

                ID

            </th>

            <th>

                Tipo

            </th>

            <th>

                Estado

            </th>

            <th>

                Veces iniciada

            </th>

            <th>

                Operar

            </th>

        </tr>

    `;

    for (let i = 0; i < misInstancias.length; i++) {

        let instancia = misInstancias[i];

        // si el filtro no es "todas", y el estado de esta instancia no coincide con el
        filtro, no la mostramos.

```

```

if (

    filtroInstanciasUsuario !== "todas" &&

    instancia.estado !== filtroInstanciasUsuario

) {

    continue;

}

let alquiler = sistema.buscarAlquiler(

    true,

    instancia.ID,

    usuarioLogeado.nombreUsuario

);

let textoAccion = "Encender";

let clase = "encenderInstancia";

if (instancia.estado === "ENCENDIDA") {

    textoAccion = "Apagar";

    clase = "apagarInstancia";

}

let botonAccion = `

<td>

    <input

```

```

        instancia-id="${instancia.ID}"

        class="${clase}" id="btnOperarInstancia${instancia.ID}"

        type="button" value="${textoAccion}"

    >

</td>`;

let textoVeces = "veces";

if (alquiler.contadorEncendido === 1) {

    textoVeces = "vez";

}

tabla += `

    <tr>

        <td>INSTANCE_ID_${instancia.ID}</td>

        <td>${formatearTipoUI(instancia.tipo.tipo)}</td>

        <td>${instancia.estado}</td>

        <td>${alquiler.contadorEncendido} ${textoVeces}</td>

        ${botonAccion}

    </tr>

`;

}

tabla += `</table>`;

```

```

// agregar la tabla

imprimirEnHtml("divUsuarioMisInstancias", tabla);


// poblar de opciones el select de filtros

imprimirEnHtml(

    "slcFiltroMisInstancias",

    `

    <option id="todas" value="todas">Todas las instancias</option>

    <option id="ENCENDIDA" value="ENCENDIDA">Encendidas</option>

    <option id="APAGADA" value="APAGADA">Apagadas</option>

    `

);


// buscar la opcion que corresponde al filtro actual y marcarla como seleccionada
document

    .querySelector(`#${filtroInstanciasUsuario}`)

    .setAttribute("selected", true);


// bindear el evento change del select a la funcion de filtrado
document

    .querySelector("#slcFiltroMisInstancias")

    .addEventListener("change", changeFiltroInstancias);

```



```

// bindear los eventos click de los botones de encendido

let botonesDeEncendido = document.querySelectorAll(".encenderInstancia");

for (let i = 0; i < botonesDeEncendido.length; i++) {

    let idDelBoton = botonesDeEncendido[i].id;

    document

        .querySelector(`#${idDelBoton}`)

        .addEventListener("click", encenderInstanciaUI);

}

// bindear los eventos click de los botones de apagado

let botonesDeApagado = document.querySelectorAll(".apagarInstancia");

for (let i = 0; i < botonesDeApagado.length; i++) {

    let idDelBoton = botonesDeApagado[i].id;

    document

        .querySelector(`#${idDelBoton}`)

        .addEventListener("click", apagarInstanciaUI);

}

}

function encenderInstanciaUI() {

    let idInstancia = this.getAttribute("instancia-id");

    sistema.encenderInstancia(Number(idInstancia));

    verMisInstanciasUsuario();

```

```

}

function apagarInstanciaUI() {

    let idInstancia = this.getAttribute("instancia-id");

    sistema.apagarInstancia(Number(idInstancia));

    verMisInstanciasUsuario();

}

function changeFiltroInstancias() {

    let filtro = this.value;

    filtrarMisInstanciasUsuario(filtro);

}

/**
 * Muestra las instancias que el usuario tiene alquiladas.
 *
 * Opcionalmente, se puede filtrar por un estado dado.
 *
 * @param {"APAGADA" | "ENCENDIDA" | "todas"} filtro
 * @default "todas"
 */
function filtrarMisInstanciasUsuario(filtro = "todas") {

    filtroInstanciasUsuario = filtro;

```

```

verMisInstanciasUsuario();

}

/**
 * Muestra los costos de cada tipo de instancia que el usuario tiene en alquiler.
 */

function verMisCostos() {

    ocultarElemento("divUsuarioMisInstancias");

    ocultarElemento("divUsuarioAlquilar");

    mostrarElemento("divUsuarioMisCostos");

    let tabla = `

        <h2>Mis Costos</h2>

        <table>

            <tr>

                <th>

                    Tipo de Instancia

                </th>

                <th>

                    Costo por Encendido

                </th>

                <th>

                    Total de veces encendida

                </th>

```

```

        <th>

            Costo total

        </th>

    </tr>

    `;

let todasMisInstancias = sistema.buscarInstanciasDeUsuario(

    usuarioLogeado.nombreUsuario

);

for (let i = 0; i < sistema.arrayTiposInstancia.length; i++) {

    let tipo = sistema.arrayTiposInstancia[i].tipo;

    let instancias = sistema.buscarInstanciasPorTipo(tipo, todasMisInstancias);

    tabla += crearFilaCostosUsuario(instancias);

}

tabla += "</table>";

imprimirEnHtml("divUsuarioMisCostos", tabla);

mostrarElemento("divUsuarioMisCostos");

}

```

```

/**
 * Crea una fila que irá en la tabla
 * @param {MaquinaVirtual[]} arrInstancias
 */
function crearFilaCostosUsuario(arrInstancias) {
    if (!arrayTieneElementos(arrInstancias)) {
        return "";
    }

    let tipo = arrInstancias[0].tipo;
    let costoEncendido = tipo.costoEncendido;
    let costoTotal = 0;
    let totalEncendidos = 0;

    for (let i = 0; i < arrInstancias.length; i++) {
        let alquiler = sistema.buscarAlquiler(
            true,
            arrInstancias[i].ID,
            usuarioLogeado.nombreUsuario
        );

        costoTotal += tipo.calcularCostos(alquiler.contadorEncendido);
        totalEncendidos += alquiler.contadorEncendido;
    }
}

```

```

return `

<tr>

  <td>${formatearTipoUI(tipo.tipo)}</td>

  <td>US$ ${costoEncendido}</td>

  <td>${totalEncendidos}</td>

  <td>US$ ${costoTotal}</td>

</tr>

`;
}

function verReportesAdmin() {

  ocultarElemento("divAdminAdministrarUsuario");

  ocultarElemento("divAdminStockInstancias");

  let tabla = `

    <h2>Reportes</h2>

    <table>

      <tr>

        <th>

          Tipo de Instancia

        </th>

        <th>

          Alquileres efectuados


```

```

        </th>

        <th>

            Ingresos

        </th>

    </tr>

    `;

let totalCantidadAlquileres = 0;

let totalIngresos = 0;

for (let i = 0; i < sistema.arrayTiposInstancia.length; i++) {

    let tipo = sistema.arrayTiposInstancia[i].tipo;

    let cantidadPorTipo = sistema.cantidadAlquileresPorTipo(tipo);

    let ingresosPorTipo = sistema.ingresosPorTipoDeInstancia(tipo);

    tabla += `

        <tr>

            <td>${formatearTipoUI(tipo)}</td>

            <td>${cantidadPorTipo}</td>

            <td>US$ ${ingresosPorTipo}</td>

        </tr>

    `;

```

```

        totalCantidadAlquileres += cantidadPorTipo;

        totalIngresos += ingresosPorTipo;
    }

    tabla += `
        <tr style="outline: thin solid; font-weight: bold">
            <td>TOTAL</td>
            <td>${totalCantidadAlquileres}</td>
            <td>US$ ${totalIngresos}</td>
        </tr>
    `;

    tabla += "</table>";

    imprimirEnHtml("divAdminReportes", tabla);

    mostrarElemento("divAdminReportes");
}

```


3.5. Libreria.js

```
function hayCaracteres(pUnTexto) {

    let tengoCaracteres = false;

    let textoSinEspacios = pUnTexto.trim();

    if (textoSinEspacios.length > 0) {

        tengoCaracteres = true;

    }

    return tengoCaracteres;

}

function arrayTieneElementos(pArray) {

    let tieneElementos = false;

    if (pArray.length > 0) {

        tieneElementos = true;

    }

    return tieneElementos;

}

function limpiarUnCampoDeTexto(idDelCampoDeTexto) {

    document.querySelector(`#${idDelCampoDeTexto}`).value = "";

}
```

```

function limpiarUnElemento(idDelElemento) {

    document.querySelector(`#${idDelElemento}`).innerHTML = "";

}

function obtenerValorDeUnElementoHTML(idDelElemento) {

    return document.querySelector(`#${idDelElemento}`).value;

}

function imprimirEnHtml(pIdElemento, pLoQueImprimo) {

    document.querySelector(`#${pIdElemento}`).innerHTML = pLoQueImprimo;

}

function mostrarNavAdmin() {

    let elementos = document.querySelectorAll(".nav-item-admin");

    for (let i = 0; i < elementos.length; i++) {

        elementos[i].style.display = "block";

    }

}

function ocultarNavAdmin() {

    let elementos = document.querySelectorAll(".nav-item-admin");

```

```

for (let i = 0; i < elementos.length; i++) {

    elementos[i].style.display = "none";

}

}

function mostrarNavUsuario() {

    let elementos = document.querySelectorAll(".nav-item-usuario");

    for (let i = 0; i < elementos.length; i++) {

        elementos[i].style.display = "block";

    }

}

function ocultarNavUsuario() {

    let elementos = document.querySelectorAll(".nav-item-usuario");

    for (let i = 0; i < elementos.length; i++) {

        elementos[i].style.display = "none";

    }

}

/**
 * Dada una id de un elemento, lo muestra.
 *
 * @param {string} pIdElemento

```

```

*/

function mostrarElemento(pIdElemento) {

    document.querySelector(`#${pIdElemento}`).style.display = "block";

}

/**
 * Dada una id de un elemento, lo oculta.
 *
 * @param {string} pIdElemento
 */

function ocultarElemento(pIdElemento) {

    document.querySelector(`#${pIdElemento}`).style.display = "none";

}

/**
 * Recibe un tipo de instancia y lo devuelve formateado para mostrar en la interfaz
de usuario.
 *
 * Ejemplo: "c7small" -> "c7.small"
 *
 * @param {string} tipo
 */

function formatearTipoUI(tipo) {

    let pre = tipo.substring(0, 2);

    let post = tipo.substring(2, tipo.length);

    return pre + "." + post;
}

```

```

}

/**
 * Verifica si un `valor` es un numero entero positivo.
 *
 * Ejemplo: "123" -> `true`
 *
 * Ejemplo: "123.5" -> `false`
 *
 * Ejemplo: "-123" -> `false`
 *
 * @param {number|string} valor
 * @returns {boolean}
 */
function esNumeroEnteroValido(valor) {
    return (
        !isNaN(valor) &&
        // valor !== " " &&
        // valor !== "." &&
        // valor !== "," &&
        // valor !== "-" &&
        // valor !== "+" &&
        // valor !== "" &&
        String(valor).indexOf(".") === -1 && // no acepta numeros decimales
        String(valor).length > 0
    )
}

```

```
);  
  
}  
  
function esUnNumeroRazonable(valor) {  
    return valor <= 1000000  
}
```

Anexo 1 Documento de análisis Original

1. Descripción general del problema a resolver

El objetivo es desarrollar un servicio de alquiler de máquinas virtuales.

1.1. Tipos de Usuarios del Sistema

- Usuarios
- Administradores

1.2. Listado de funcionalidades

- F01 - Registro de Usuarios (Usuario)
- F02 - Ingreso de Usuarios (Usuario,Administrador)
- F03 - Alquiler de Máquinas Virtuales (Usuario)
- F04 - Visualización de Máquinas Alquiladas (Usuario)
- F05 - Encender / Apagar Máquinas Alquiladas (Usuario)
- F06 - Visualización del Costo Total de Alquiler (Usuario)
- F07 - Aprobación de Registros de Usuarios (Administrador)
- F08 - Bloqueo de Usuarios (Administrador)
- F09 - Modificación del Stock de Máquinas Virtuales (Administrador)
- F10 - Visualización de Informe de Máquinas Virtuales (Administrador)

2. Detalle de funcionalidades

A continuación, se presenta el detalle de cada una de las funcionalidades a resolver en el obligatorio.

2.1. F01 - Registro de usuarios

2.1.1. Acceso

Tipo de usuario que puede acceder: Usuario

2.1.2. Descripción

Permitir a los usuarios registrarse en la aplicación proporcionando información personal.

2.1.3. Interfaz de Usuario

La interfaz de usuario para el registro debe incluir un formulario que solicita los siguientes datos:

- Nombre
- Apellido
- Nombre de usuario (formato alfanumérico, case insensitive)
- Contraseña (debe cumplir con ciertos requisitos)
- Número de tarjeta de crédito (formato xxxx-xxxx-xxxx-xxxx)
- CVC (código de verificación, formato numérico de 3 dígitos)

Contiene al final un botón con el título “Registrarse”, el cual permitirá ejecutar la acción indicada.

Registro de Usuarios

Nombre	<input type="text" value="Nombre"/>	Apellido	<input type="text" value="Apellido"/>
Nombre de usuario	<input type="text" value="Nombre de usuario"/>	Contraseña	<input type="text" value="Contraseña"/>
Número de tarjeta de crédito	<input type="text" value="xxxx-xxxx-xxxx-xxxx"/>	CVC	<input type="text" value="Cód, verificación, (3 dígitos)"/>
<input type="button" value="Registrarse"/>			

2.1.4. Validaciones

- Todos los campos del formulario deben ser completados.
- El nombre de usuario debe cumplir con un formato alfanumérico y debe ser case insensitive.
- La contraseña debe cumplir con los siguientes requisitos:
 - Al menos 5 caracteres
 - Al menos una mayúscula
 - Al menos una minúscula
 - Al menos un número
- La tarjeta de crédito debe cumplir con los siguientes requisitos:
 - Cumplir el formato xxxx-xxxx-xxxx-xxxx
 - Tener 16 caracteres sin contar los guiones
 - Debe cumplir con el algoritmo de verificación de tarjetas
- El CVC debe ser un número de 3 dígitos.
- No pueden existir dos usuarios con el mismo nombre de usuario

2.2. F02 - Ingreso de usuarios

2.2.1. Acceso

Tipo de usuario que puede acceder: Usuario, Administrador

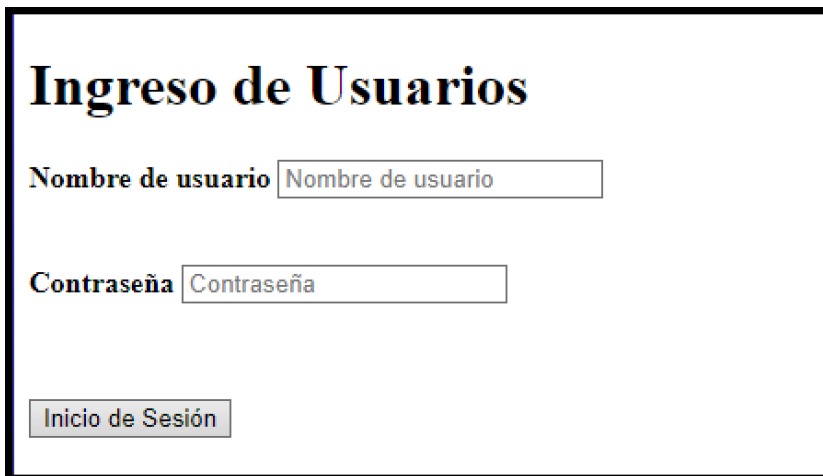
2.2.2. Descripción

Permitir a los usuarios registrados iniciar sesión.

2.2.3. Interfaz de Usuario

La interfaz de usuario para el ingreso contará con:

- Nombre de Usuario: Campo de entrada de texto
- Contraseña: Campo de entrada de texto
- Botón de Inicio de Sesión



The screenshot shows a login form with the title "Ingreso de Usuarios" in a large, bold, black serif font. Below the title, there are two input fields. The first is labeled "Nombre de usuario" in a black serif font, followed by a text box containing the placeholder text "Nombre de usuario" in a light blue sans-serif font. The second is labeled "Contraseña" in a black serif font, followed by a text box containing the placeholder text "Contraseña" in a light purple sans-serif font. At the bottom left of the form, there is a button labeled "Inicio de Sesión" in a black serif font, with a light gray background and a thin black border.

2.2.4. Validaciones

- Las credenciales ingresadas deben corresponder con las de un usuario ya registrado previamente.
- En caso de que las credenciales proporcionadas sean incorrectas, el sistema mostrará una notificación de error para informar al usuario sobre la falla en el inicio de sesión.
- El usuario debe estar habilitado para usar el sistema (no bloqueado)

2.3. F03 - Alquiler de Máquinas Virtuales

2.3.1. Acceso

Tipo de usuario que puede acceder: Usuario

2.3.2. Descripción

Permitir a los usuarios alquilar máquinas virtuales temporales para su uso. Al alquilar una máquina virtual, esta se pone en estado "encendido" automáticamente y, en esa primera ocasión, no se cobra el encendido, solo el alquiler.

2.3.3. Interfaz de Usuario

La interfaz de usuario para el alquiler contará con:

- Tipo de Instancia: Combo desplegable con opciones cargadas dinámicamente, deben corresponderse con los tipos de instancia ya definidas.
- Notificación de Disponibilidad: Área de texto o etiqueta que muestra si la instancia seleccionada está disponible o no.
- Información del Costo de Alquiler: Área de texto o etiqueta que muestra el costo del alquiler.
- Información del Costo por Encendido: Área de texto o etiqueta que muestra el costo del encendido.
- Botón de Alquiler: Botón con el texto "Alquilar"

Optimizadas para computo		v		
Instancia	costo	Estado	Costo por encendido	
c7.small	U\$S 20	Disponible	USD 2.50	Alquilar
c7.medium	U\$S 30	Disponible	USD 3.50	Alquilar
c7.large	U\$S 50	Disponible	USD 6.00	Alquilar

2.3.4. Validaciones

- El usuario debe estar autorizado a usar el sistema (no bloqueado)

- El tipo de instancia seleccionada debe ser válida (corresponde a las opciones del combobox)
- El tipo de instancia seleccionada debe tener stock disponible
- Solo se puede alquilar una instancia a la vez, por lo que si se desean alquilar múltiples instancias, se debe reiniciar el flujo de esta funcionalidad.

2.4. F04 - Visualización de máquinas alquiladas

2.4.1. Acceso

Tipo de usuario que puede acceder: Usuario

2.4.2. Descripción

Permitir a los usuarios ver una lista de las máquinas virtuales que han alquilado, así como información sobre el estado de cada instancia.

2.4.3. Interfaz de Usuario

La interfaz de usuario para el ingreso contará con:

- Tipo de Instancia: Combo desplegable con opciones cargadas dinámicamente, deben corresponderse con los tipos de instancia ya definidas, incluyendo “Todas las instancias”
- Tabla de instancias: Tabla que mostrará las instancias alquiladas por el usuario, contiene los siguientes datos:
 - Instancia - nombre de la instancia
 - Estado - indicando si la instancia está encendida o apagada
 - Veces Iniciada - cantidad de veces que se prendió la instancia
 - Acción - contiene un botón que puede tener el título “Apagar” o “Encender” acorde al estado actual de la instancia.

Todas las instancias			v
Instancia	Estado	Veces iniciada	
c7.large	Encendida	3 veces	Apagar
c7.large	Encendida	1 vez	Apagar
i7.medium	Apagada	2 veces	Encender

2.4.4. Validaciones

- El usuario debe estar autorizado a usar el sistema (no bloqueado)
- El usuario debe estar logueado

2.5. F05 - Encender / Apagar Máquinas Alquiladas

2.5.1. Acceso

Tipo de usuario que puede acceder: Usuario

2.5.2. Descripción

Permitir a los usuarios encender o apagar las Instancias.

2.5.3. Interfaz de Usuario

Misma Interfaz que en el punto 2.4 (F04).

Todas las instancias			v
Instancia	Estado	Veces iniciada	
c7.large	Encendida	3 veces	Apagar
c7.large	Encendida	1 vez	Apagar
i7.medium	Apagada	2 veces	Encender

2.5.4. Validaciones

- El usuario debe estar logueado
- El usuario debe estar habilitado para operar (no bloqueado)

2.6. F06 - Visualización del Costo Total de Alquiler

2.6.1. Acceso

Tipo de usuario que puede acceder: Usuario

2.6.2. Descripción

Permitir a los usuarios visualizar el costo total de los alquileres de máquinas virtuales que han realizado. Esto incluye la agrupación de gastos por tipo de instancia.

2.6.3. Interfaz de Usuario

Tabla con los siguientes datos:

- Tipo de Instancia
- Costo por encendido
- Total de veces iniciada
- Costo total

Tipo de Instancia	Costo por encendido	Total de veces encendidas	Costo total
c7.large	U\$S 6	4	U\$S 112
i7.medium	U\$S 3.50	2	U\$S 33.50

2.6.4. Validaciones

- El usuario debe estar habilitado a operar el sistema (no bloqueado)
- El usuario debe estar logueado

2.7. F07 - Aprobación de Registros de Usuarios

2.7.1. Acceso

Tipo de usuario que puede acceder: Administrador

2.7.2. Descripción

Permitir a los administradores ver la lista de usuarios registrados que están en estado "pendiente" y aprobarlos para que puedan iniciar sesión en la aplicación.

2.7.3. Interfaz de Usuario

Lista de Usuarios Pendientes de Aprobación: Una tabla que muestra la lista de usuarios que están en estado "pendiente" con las siguientes columnas:

- Nombre
- Apellido
- Nombre de Usuario
- Acción - Botón con el texto "Aprobar"

Notificación de Acción Exitosa: Área de texto o etiqueta que muestra un mensaje de confirmación cuando se realiza una acción de aprobación o bloqueo.

Usuarios pendientes a aprobacion			
Nombre	Apellido	Nombre de Usuario	
Leandro	Guzman	L guzman95	<input type="button" value="Aprobar"/>
Mateo	Carriqui	MCarri321268	<input type="button" value="Aprobar"/>

2.7.4. Validaciones

- El administrador debe estar logueado
- El administrador que opere debe tener efectivamente el rol de Administrador. No puede ser operado por un usuario corriente.

2.8. F08 - Bloqueo de Usuarios

2.8.1. Acceso

Tipo de usuario que puede acceder: Administrador

2.8.2. Descripción

Permitir a los administradores bloquear usuarios activos en la aplicación. El bloqueo de usuarios implica que el usuario no podrá iniciar sesión, y todas las máquinas virtuales alquiladas por el usuario volverán al stock, eliminando sus alquileres

2.8.3. Interfaz de Usuario

Lista de Usuarios Activos: Una tabla que muestra la lista de usuarios activos con las siguientes columnas:

- Nombre
- Apellido
- Nombre de Usuario
- Acción - Botón con el texto "Bloquear"

Notificación de Bloqueo Exitoso: Área de texto o etiqueta que muestra un mensaje de confirmación cuando se bloquea un usuario

Lista de Usuarios Activos			
Nombre	Apellido	Nombre de Usuario	
Leandro	Guzman	L guzman95	<input type="button" value="Bloquear"/>
Mateo	Carriquí	MCarri321268	<input type="button" value="Bloquear"/>

2.8.4. Validaciones

- Estar logueado como administrador
- El administrador que opere debe tener efectivamente el rol de Administrador. No puede ser operado por un usuario corriente.

2.9. F09 - Modificación del Stock de Máquinas Virtuales

2.9.1. Acceso

Tipo de usuario que puede acceder: Administrador

2.9.2. Descripción

Permitir a los administradores modificar el stock de máquinas virtuales disponibles por tipo de instancia.

2.9.3. Interfaz de Usuario

Lista de Tipos de Instancia: Una tabla que muestra la lista de tipos de instancia con las siguientes columnas:

- Tipo de Instancia
- Stock Actual
- Cantidad a Modificar (campo de entrada numérico)
- Botón "Modificar Stock" (para cada tipo de instancia)

Notificación de Modificación Exitosa: Área de texto o etiqueta que muestra un mensaje de confirmación cuando se realiza una modificación exitosa

Tipo de Instancia	Stock Actual	Cantidad a Modificar	
c7.small	3	<input type="text" value="ingrese cantidad a modificar"/>	<button>Modificar Stock</button>
c7.medium	1	<input type="text" value="ingrese cantidad a modificar"/>	<button>Modificar Stock</button>
c7.large	2	<input type="text" value="ingrese cantidad a modificar"/>	<button>Modificar Stock</button>
r7.small	3	<input type="text" value="ingrese cantidad a modificar"/>	<button>Modificar Stock</button>
r7.medium	1	<input type="text" value="ingrese cantidad a modificar"/>	<button>Modificar Stock</button>
r7.large	2	<input type="text" value="ingrese cantidad a modificar"/>	<button>Modificar Stock</button>
i7.medium	3	<input type="text" value="ingrese cantidad a modificar"/>	<button>Modificar Stock</button>
i7.large	1	<input type="text" value="ingrese cantidad a modificar"/>	<button>Modificar Stock</button>

2.9.4. Validaciones

- Estar logueado como administrador
- Solo los administradores deben tener acceso para modificar el stock de máquinas virtuales.
- La cantidad a modificar debe ser numérica.
- La cantidad a modificar no debe ser negativa.
- La cantidad a modificar tiene que ser igual o mayor al número de máquinas alquiladas actual.

2.10. F010 - Visualización de Informe de Máquinas Virtuales

2.10.1. Acceso

Tipo de usuario que puede acceder: Administrador

2.10.2. Descripción

Permitir a los administradores acceder a un informe que muestre información sobre los tipos de instancia en alquiler, la cantidad de cada tipo de instancia en alquiler y su ingreso total actual por cada tipo de instancia. Además, se debe mostrar el ingreso total global de la empresa.

2.10.3. Interfaz de Usuario

Informe de Máquinas Virtuales: Una tabla que muestra el informe con las siguientes columnas:

- Tipo de Instancia
- Cantidad en Alquiler

Tabla de Instancias : Ingreso Total por Tipo de Instancia con las columnas

- Optimizadas para computo
- Optimizadas para memoria
- Optimizadas para almacenamiento
- Ingreso Total Global: Una etiqueta que muestra el Ingreso Total Global de la empresa

Informe de Máquinas Virtuales

Tipo de Instancia	Cantidad en Alquiler
c7.small	3
c7.medium	1
c7.large	2
r7.small	3
r7.medium	1
r7.large	2
i7.medium	3
i7.large	1

Instancias

Optimizadas para computo	Optimizadas para memoria	Optimizadas para almacenamiento	Ingreso Total Global
6	6	4	16

2.10.4. Validaciones

- Solo los administradores deben tener acceso para ver el informe de máquinas virtuales.
- El administrador debe estar logueado

3. Casos de prueba

3.1. F01 - Registro de usuarios

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F01-T01	Registro exitoso	El usuario ingresa un nombre, apellido, nombre de usuario, contraseña, número de tarjeta de crédito, CVC. El usuario solicita el registro.	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> LeaGuz95 <u>Contraseña:</u> Tarea1 <u>Número de Tarjeta De Crédito:</u> 1234-5678-9076-543 <u>CVC:</u> 123	1.El sistema registra al usuario como "pendiente" y muestra un mensaje indicando que su registro será aprobado por un administrador. 2.El sistema registra al usuario con éxito en caso de ser aprobado por un administrador y muestra un mensaje de confirmación		
F01-T02	Intento de registro con una contraseña débil.	1.El usuario ingresa una contraseña que no cumple con los requisitos (por ejemplo, una contraseña de 3 caracteres sin mayúsculas ni números). 2.El usuario solicita el registro	<u>Nombre:</u> Leandro <u>Apellido:</u> Guzmán <u>Nombre De Usuario:</u> LeaGuz95 <u>Contraseña:</u> Tar <u>Número de Tarjeta De Crédito:</u> 1234-5678-9076-543 <u>CVC:</u> 123	El sistema muestra un mensaje de error indicando que la contraseña no cumple con los requisitos.		

F01-T03	Intento de registro con una tarjeta de crédito inválida.	<p>1.El usuario ingresa un número de tarjeta de crédito que no tiene 16 dígitos o no pasa la validación del algoritmo de tarjetas.</p> <p>2.El usuario solicita el registro.</p>	<p><u>Nombre:</u> Leandro</p> <p><u>Apellido:</u> Guzmán</p> <p><u>Nombre De Usuario:</u> LeaGuz95</p> <p><u>Contraseña:</u> Tarea1</p> <p><u>Número de Tarjeta De Crédito:</u>1234-56</p> <p><u>CVC:</u> 123</p>	El sistema muestra un mensaje de error indicando que el número de tarjeta de crédito es inválido.		
F01-T04	Intento de registro con un nombre de usuario ya en uso.	<p>1.El usuario ingresa un nombre de usuario que ya está en uso por otro usuario.</p> <p>2.El usuario solicita el registro.</p>	<p><u>Nombre:</u> Leandro</p> <p><u>Apellido:</u> Guzmán</p> <p><u>Nombre De Usuario:</u> LeaGuz95</p> <p><u>Contraseña:</u> Tarea1</p> <p><u>Número de Tarjeta De Crédito:</u>1234-5678-9076-543</p> <p><u>CVC:</u> 123</p>	El sistema muestra un mensaje de error indicando que el nombre de usuario ya está en uso.		
F01-T05	Registro exitoso con aprobación pendiente	<p>El usuario ingresa datos válidos.</p> <p>El usuario solicita el registro.</p>	<p><u>Nombre:</u> Leandro</p> <p><u>Apellido:</u> Guzmán</p> <p><u>Nombre De Usuario:</u> LeaGuz95</p> <p><u>Contraseña:</u> Tarea1</p> <p><u>Número de Tarjeta De Crédito:</u>1234-5678-9076-543</p> <p><u>CVC:</u> 123</p>	El sistema registra al usuario como "pendiente" y muestra un mensaje indicando que su registro será aprobado por un administrador.		

3.2. F02 - Ingreso de usuarios

Precondiciones :El usuario cuenta con un usuario creado.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Fall a, P=pasa)
F02-T01	Ingreso exitoso con credenciales válidas.	El usuario ingresa su nombre de usuario y contraseña correctos. El usuario solicita el ingreso.	<u>Nombre De Usuario:</u> LeaGuz95 <u>Contraseña:</u> Tarea1	El sistema permite el ingreso		
F02-T02	Intento de ingreso con un nombre de usuario incorrecto.	El usuario ingresa un nombre de usuario incorrecto. El usuario ingresa la contraseña correcta. El usuario solicita el ingreso.	<u>Nombre De Usuario:</u> LeaGuz98 <u>Contraseña:</u> Tarea1	El sistema muestra un mensaje de error sin especificar si existe el usuario		
F02-T03	Intento de ingreso con una contraseña incorrecta.	El usuario ingresa su nombre de usuario correcto. El usuario ingresa una contraseña incorrecta. El usuario solicita el ingreso.	<u>Nombre De Usuario:</u> LeaGuz95 <u>Contraseña:</u> Tarea2	El sistema muestra un mensaje de error sin especificar si existe el usuario		

3.3. F03 - Alquiler de Máquinas Virtuales

Precondiciones :El usuario ingreso al sistema

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F03-T01	Alquiler exitoso de una máquina virtual.	El usuario elige el tipo de instancia a alquilar. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para computo: c7.small	El sistema registra el alquiler con éxito, cambia el estado de la máquina a "encendida" (sin cobrar por el encendido) y muestra un mensaje de confirmación.		
F03-T02	Intento de alquiler de una máquina virtual no disponible.	El usuario elige un tipo de instancia que no está disponible en el stock. El usuario solicita el alquiler.	Tipo de instancia: Optimizadas para computo: c7.small(en este caso no disponible)	El sistema muestra un mensaje de error indicando que el tipo de instancia no está disponible.		

3.4. F04 - Visualización de máquinas alquiladas

Precondiciones :El usuario ingreso al sistema

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado o obtenido	Estado (F=Falla, P=pasa)
F04-T01	Visualización exitosa de las máquinas alquiladas.	El usuario ingresa a la pantalla de visualización de máquinas alquiladas.		El sistema muestra la lista de máquinas alquiladas por el usuario, incluyendo el tipo de instancia, estado, y el número de veces que se ha iniciado.		
F04-T02	Usuario sin máquinas alquiladas.	El usuario ingresa a la pantalla de visualización de máquinas alquiladas.		El sistema muestra la lista de máquinas alquiladas vacía		

3.5. F05 - Encender / Apagar Máquinas Alquiladas

Precondiciones :El usuario ingreso al sistema y cuenta con máquinas alquiladas.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F05-T01	Encendido exitoso de una máquina alquilada	<p>El usuario ingresa a la pantalla de visualización de máquinas alquiladas.</p> <p>El usuario selecciona una máquina alquilada que está apagada.</p> <p>El usuario solicita encender la máquina.</p>		El sistema enciende la máquina alquilada, cobra el costo correspondiente y muestra un mensaje de confirmación.		
F05-T02	Apagado exitoso de una máquina alquilada.	<p>El usuario ingresa a la pantalla de visualización de máquinas alquiladas.</p> <p>El usuario selecciona una máquina alquilada que está encendida.</p> <p>El usuario solicita apagar la máquina.</p>		El sistema apaga la máquina alquilada con éxito y muestra un mensaje de confirmación.		

3.6. F06 - Visualización del Costo Total de Alquiler

Precondiciones :El usuario ingreso al sistema.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F06-T01	Visualización exitosa del costo total de alquiler por usuario.	El usuario ingresa a la pantalla de visualización del costo total de alquiler.		El sistema muestra el costo total de alquiler por usuario, agrupado por tipo de instancia y sin incluir el costo de encendido inicial.		
F06-T02	Usuario no tiene máquinas	El usuario ingresa a la pantalla de visualización del costo total de alquiler.		La tabla se muestra vacía		

3.7. F07 - Aprobación de Registros de Usuarios

Precondiciones :El usuario ingresa al sistema como administrador.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F07-T01	Aprobación exitosa de un registro de usuario pendiente.	El administrador ingresa a la pantalla de aprobación de registros de usuarios. El administrador selecciona un registro de usuario pendiente. El administrador aprueba el registro.		El sistema aprueba el registro de usuario pendiente y cambia su estado a "activo". El usuario puede iniciar sesión en la aplicación.		
F07-T02	Sin Usuarios Pendientes	El administrador ingresa a la pantalla de aprobación de registros de usuarios.		Lista de usuarios por aprobar vacía.		

3.8. F08 - Bloqueo de Usuarios

Precondiciones :El usuario ingresa al sistema como administrador.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F08-T01	Bloqueo exitoso de un usuario activo por parte de un administrador.	El administrador ingresa a la pantalla de bloqueo de usuarios. El administrador selecciona un usuario activo. El administrador bloquea al usuario.		El sistema bloquea al usuario, cambia su estado a "bloqueado" y todas las máquinas virtuales alquiladas por el usuario vuelven al stock, eliminando sus alquileres.		
F08-T02	Lista de Usuarios para bloquear vacía	El administrador ingresa a la pantalla de bloqueo de usuarios.		Lista de usuarios por bloquear vacía.		

3.9. F09 - Modificación del Stock de Máquinas Virtuales

Precondiciones :El usuario ingresa al sistema como administrador.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F09-T0 1	Modificación exitosa del stock de máquinas virtuales disponibles por tipo de instancia.	<p>El administrador ingresa a la pantalla de modificación del stock de máquinas virtuales.</p> <p>El administrador selecciona un tipo de instancia.</p> <p>El administrador modifica la cantidad de máquinas virtuales disponibles.</p>	Cantidad a modificar: 5	El sistema permite al administrador modificar la cantidad de máquinas virtuales disponibles y actualiza el stock con el nuevo valor.		
F09-T0 2	Intento de modificar el stock a una cantidad negativa.	<p>El administrador ingresa a la pantalla de modificación del stock de máquinas virtuales.</p> <p>El administrador selecciona un tipo de instancia.</p> <p>El administrador intenta modificar la cantidad de máquinas virtuales disponibles a un valor negativo.</p>	Cantidad a modificar: -5	El sistema muestra un mensaje de error indicando que no se puede establecer un stock negativo para las máquinas virtuales.		

F09-T0 3	Modificación del stock de máquinas virtuales a un número menor que la cantidad de alquileres	<p>El administrador ingresa a la pantalla de modificación del stock de máquinas virtuales.</p> <p>El administrador selecciona un tipo de instancia.</p> <p>El administrador modifica la cantidad de máquinas virtuales disponibles</p>	<p>Cantidad a modificar: 5</p> <p>(en alquiler actual :7)</p>	<p>El sistema muestra un mensaje de error indicando que no se puede establecer en stock un valor menor a la cantidad de alquileres actuales.</p>		
-------------	--	--	---	--	--	--

3.10. F010 - Visualización de Informe de Máquinas Virtuales

Precondiciones :El usuario ingresa al sistema como administrador.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F010-T01	Visualización exitosa del informe de máquinas virtuales.	El administrador ingresa a la pantalla de visualización del informe de máquinas virtuales. El administrador revisa el informe con la información de tipos de instancia, cantidad y su ingreso total actual por cada tipo de instancia.		El sistema muestra el informe de máquinas virtuales con la información actualizada de tipos de instancia, cantidad y su ingreso total por cada tipo de instancia.		
F010-T02	Visualización del informe cuando no hay máquinas virtuales en el sistema.	El administrador ingresa a la pantalla de visualización del informe de máquinas virtuales.		El sistema muestra el informe vacío, indicando que no hay máquinas virtuales alquiladas en el sistema.		

