Universidad ORT Uruguay
Facultad de Ingeniería
Escuela de Tecnología

OBLIGATORIO PROGRAMACIÓN 1

DOCUMENTO DE ANÁLISIS

Guillermina Costa – 222938

Virginia Rodríguez – 292711

M1D

Docente: Sergio Palay

Analista en Tecnologías de la Información

26 de mayo de 2022

Índice

1. Descripción general del problema a resolver.....5

1.1. Tipos de usuario del sistema.....5

1.2. Listado de funcionalidades.....5

2. Detalle de Funcionalidades.....7

2.1. F01 – Inicio de Sesión.....7

2.1.1. Acceso.....7

2.1.2. Descripción.....7

2.1.3. Interfaz de usuario.....8

2.1.4. Validaciones.....9

2.2. F02 – Habilitar y deshabilitar disponibilidad de reservas.....9

2.2.1. Acceso.....9

2.2.2. Descripción.....9

2.2.3. Interfaz de usuario.....10

2.2.4. Validaciones.....10

2.3. F03 – Cambiar el estado de las reservas.....10

2.3.1. Acceso.....10

2.3.2. Descripción.....10

2.3.3. Interfaz de usuario.....11

2.3.4. Validaciones.....11

2.4. F04 – Buscador dentro de las reservas.....11

2.4.1. Acceso.....11

2.4.2. Descripción.....11

2.4.3. Interfaz de usuario.....12

2.4.4. Validaciones.....12

2.5. F05 – Modificar el cupo máximo.....12

2.5.1. Acceso.....12

2.5.2. Descripción.....12

2.5.3. Interfaz de usuario 13
2.5.4. Validaciones 13
2.6. F06 – Ver porcentaje de ocupación del local 13
2.6.1. Acceso 13
2.6.2. Descripción 13
2.6.3. Interfaz de usuario 14
2.6.4. Validaciones 14
2.7. F07 – Ver el promedio de calificaciones del local 14
2.7.1. Acceso 14
2.7.2. Descripción 14
2.7.3. Interfaz de usuario 15
2.7.4. Validaciones 15
2.8. F08 – Listado de todos los locales y promedio de calificaciones de cada uno. 15
2.8.1. Acceso 15
2.8.2. Descripción 15
2.8.3. Interfaz de usuario 16
2.8.4. Validaciones 16
2.9. F09 –Ver total de reservas 16
2.9.1. Acceso 16
2.9.2. Descripción 16
2.9.3. Interfaz de usuario 17
2.9.4. Validaciones 17
2.10. F10 – Registro en la aplicación 17
2.10.1. Acceso 17
2.10.2. Descripción 17
2.10.3. Interfaz de usuario 18
2.10.4. Validaciones 18
2.11. F11 – Realizar una reserva 19
2.11.1. Acceso 19
2.11.2. Descripción 19
2.11.3. Interfaz de usuario 19
2.11.4. Validaciones 20
2.12. Cancelar una reserva 20

2.12.1. Acceso 20
2.12.2. Descripción 20
2.12.3. Interfaz de usuario 20
2.12.4. Validaciones 21
2.13. Calificar una reserva finalizada 21
2.13.1. Acceso 21
2.13.2. Descripción 21
2.13.3. Interfaz de usuario 21
2.13.4. Validaciones 22
2.14. F14 – Ver listado de reservas pendientes 22
2.14.1. Acceso 22
2.14.2. Descripción 22
2.14.3. Interfaz de usuario 22
2.14.4. Validaciones 23
2.15. F15 – Ver información sobre reservas ..... **Error! Bookmark not defined.**
2.15.1. Acceso 23
2.15.2. Descripción 23
2.15.3. Interfaz de usuario 23
2.15.4. Validaciones 24

Documento de testing

Listado de información precargada

Código js

Código HTML

CSS

1. Descripción general del problema a resolver

Vamos a crear una aplicación para reservar lugares en locales (restaurants, teatros y museos). Cualquier usuario podrá seleccionar el lugar al que desea asistir y la cantidad de personas que irán. En base a eso, la aplicación permitirá gestionar los cupos junto a otras varias funcionalidades.

1.1. Tipos de usuario del sistema

- Locales
- Personas

1.2. Listado de funcionalidades

F01 – Inicio de Sesión – Local / Persona

F02 – Habilitar y deshabilitar disponibilidad de reservas – Local

F03 – Cambiar el estado de las reservas – Local

F04 – Buscador dentro de las reservas – Local

F05 – Modificar el cupo máximo – Local

F06 – Ver porcentaje de ocupación del local – Local

F07 – Ver el promedio de calificaciones del local – Local

F08 – Listado de todos los locales y promedio de calificaciones de cada uno – Local

F09 –Ver total de reservas – Local

F10 – Registro en la aplicación – Persona

F11 – Realizar una reserva – Persona

F12 – Cancelar una reserva – Persona

F13 – Calificar una reserva finalizada – Persona

F14 – Ver listado de reservas pendientes – Persona

F15 – Ver información sobre reservas – Persona

2. Detalle de Funcionalidades

A continuación, se presenta el detalle de cada una de las funcionalidades a resolver en el obligatorio. Las funcionalidades están ordenadas por tipo de usuario.

2.1. F01 – Inicio de Sesión

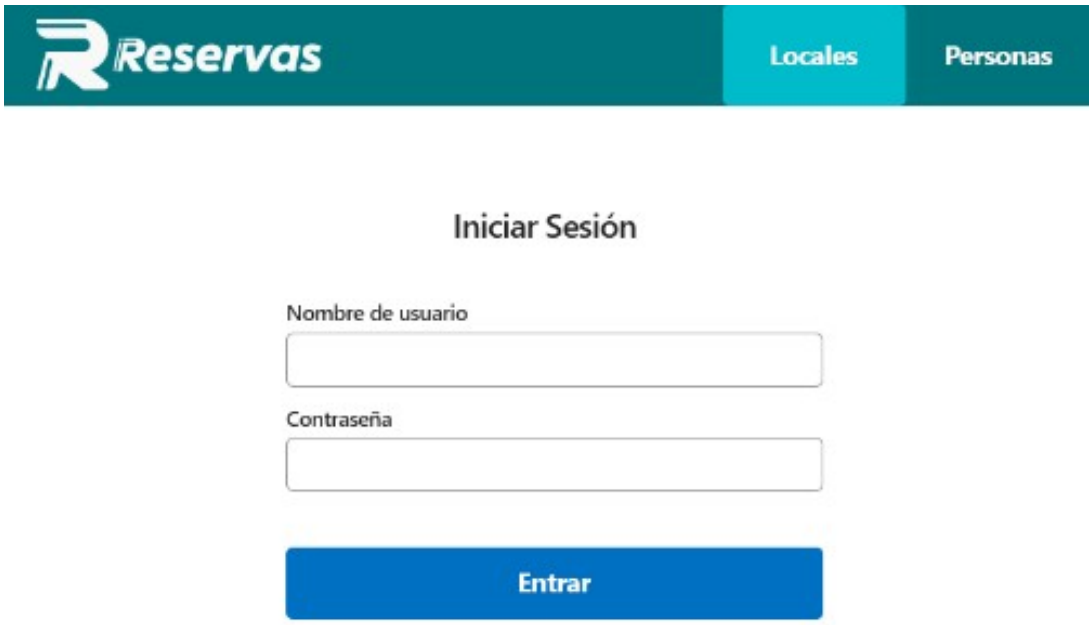
2.1.1. Acceso

Pueden ingresar ambos usuarios: local y persona.

2.1.2. Descripción

El objetivo es ingresar al sistema para hacer uso de las funcionalidades asignadas a cada tipo de usuario.

2.1.3. Interfaz de usuario

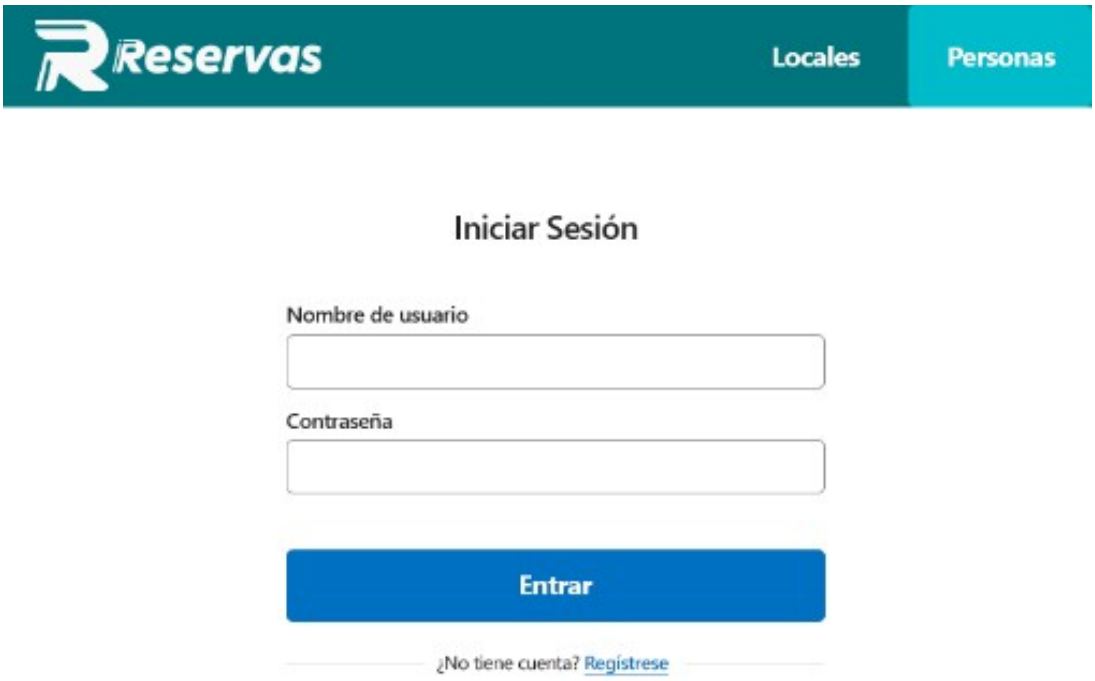


2.1.4. Validaciones

Es obligatorio ingresar todos los datos solicitados: usuario y contraseña. Hay dos opciones de perfil para seleccionar, persona o local, por lo cual no hay posibilidad de error.

Para ambos perfiles se solicitará usuario, el cual se validará que exista en el sistema. Para la contraseña, la validación es la misma para ambos tipos de usuario, se verificará que sea la contraseña asociada al usuario ingresado.

Si alguna de las credenciales de acceso fuera incorrecta, se mostrará un mensaje de error indicando dónde la validación no se cumple. Los mensajes serán: “El nombre de usuario no es correcto” o “La clave ingresada no es correcta”.



2.2. F02 – Habilitar y deshabilitar disponibilidad de reservas

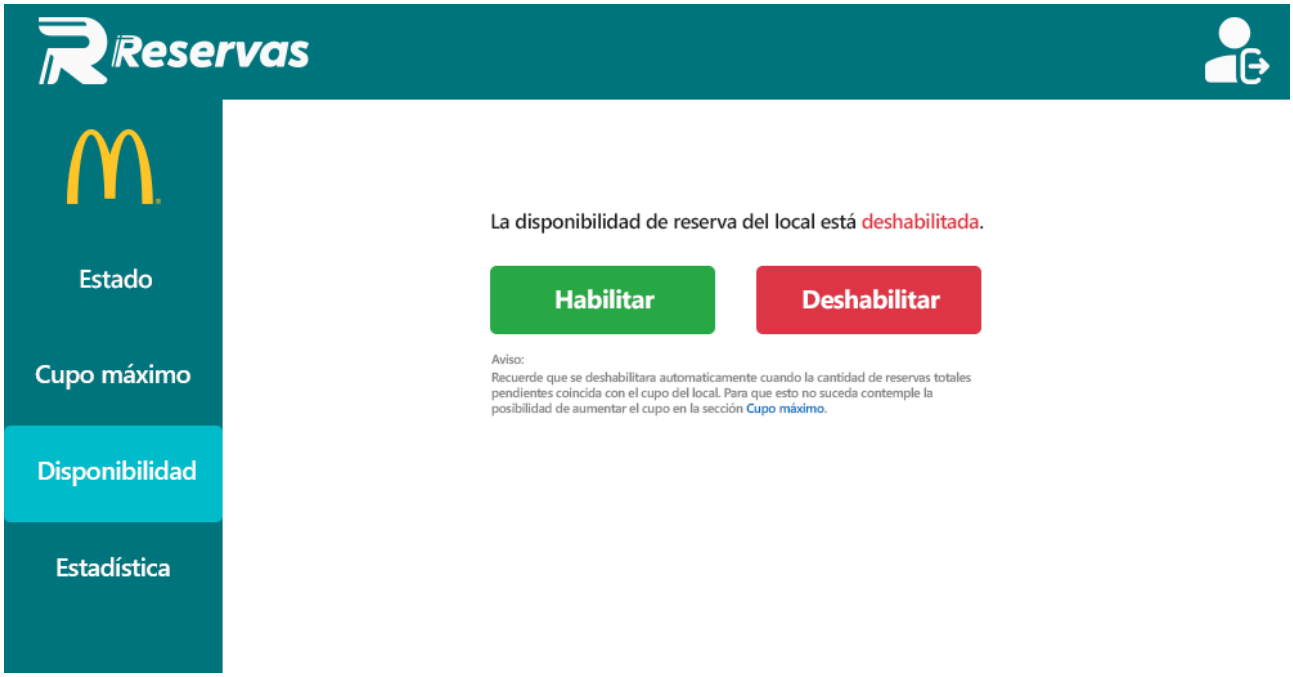
2.2.1. Acceso

Puede ingresar el usuario Local.

2.2.2. Descripción

Funcionalidad para que el local habilite o deshabilite temporalmente las reservas de su sitio.

2.2.3. Interfaz de usuario



2.2.4. Validaciones

Al momento que la cantidad de reservas totales pendientes de un local coincida con el cupo máximo del local, se deshabilitará automáticamente al local para recibir reservas.

2.3. F03 – Cambiar el estado de las reservas

2.3.1. Acceso

Puede ingresar el usuario Local.

2.3.2. Descripción

Funcionalidad para poder cambiar el estado de las reservas del local, cambiar del estado original “pendiente” a “finalizada”. Los cambios de estado serán individuales, de a una reserva, no de a múltiples simultáneamente.

2.3.3. Interfaz de usuario

2.3.4. Validaciones

La funcionalidad solo podrá seleccionar entre dos opciones, por lo tanto no hay validaciones.

2.4. F04 – Buscador dentro de las reservas

2.4.1. Acceso

Puede ingresar el usuario Local.

2.4.2. Descripción

Buscador para buscar una reserva, dentro de las pendientes, por el nombre del cliente y poder marcarla como finalizada individualmente.

2.4.3. Interfaz de usuario

2.4.4. Validaciones

En el caso de que no haya coincidencias al buscar, se mostrará un aviso de que no hay clientes con ese nombre que tengan una reserva pendiente. La búsqueda será amplia, por lo que, se buscarán substrings que coincidan con los nombres de usuario registrados.

2.5. F05 – Modificar el cupo máximo

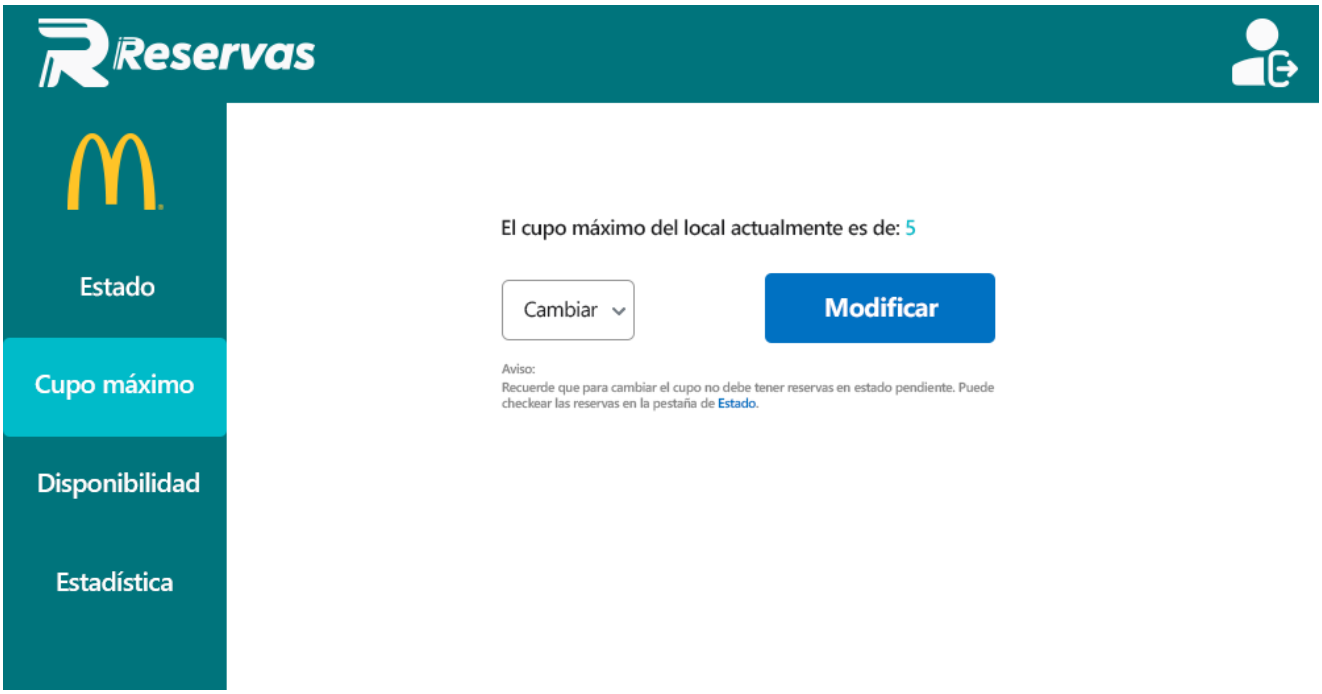
2.5.1. Acceso

Puede ingresar el usuario Local.

2.5.2. Descripción

El local podrá modificar su cupo máximo. Solo podrá realizarlo si no tiene reservas en estado pendiente.

2.5.3. Interfaz de usuario



2.5.4. Validaciones

Se verificará que el local no cuente con reservas pendientes al momento de modificar el cupo máximo, si las tuviera, se mostrará el mensaje: “No es posible realizar esta acción, su local tiene reservas pendientes.”

2.6. F06 – Ver porcentaje de ocupación del local

2.6.1. Acceso

Puede ingresar el usuario Local.

2.6.2. Descripción

Se mostrará el porcentaje de ocupación del local y se calculará a partir del total de reservas pendientes y el cupo máximo del que dispone.

2.6.3. Interfaz de usuario

2.6.4. Validaciones

Al tratarse de un cálculo no hay validaciones.

2.7. F07 – Ver el promedio de calificaciones del local

2.7.1. Acceso

Puede ingresar el usuario Local.

2.7.2. Descripción

Se mostrará el promedio de calificaciones del local.

2.7.3. Interfaz de usuario

2.7.4. Validaciones

Al tratarse de un cálculo no hay validaciones.

2.8. F08 – Listado de todos los locales y promedio de calificaciones de cada uno

2.8.1. Acceso

Puede ingresar el usuario Local.

2.8.2. Descripción

Se mostrará un listado de todos los locales y el promedio de calificaciones de cada uno.

2.8.3. Interfaz de usuario

2.8.4. Validaciones

Al tratarse de un cálculo no hay validaciones.

2.9. F09 –Ver total de reservas

2.9.1. Acceso

Puede ingresar el usuario Local.

2.9.2. Descripción

Se mostrará un listado de todas las reservas que tiene el local.

2.9.3. Interfaz de usuario

2.9.4. Validaciones

Al tratarse de un cálculo no hay validaciones.

2.10. F10 – Registro en la aplicación

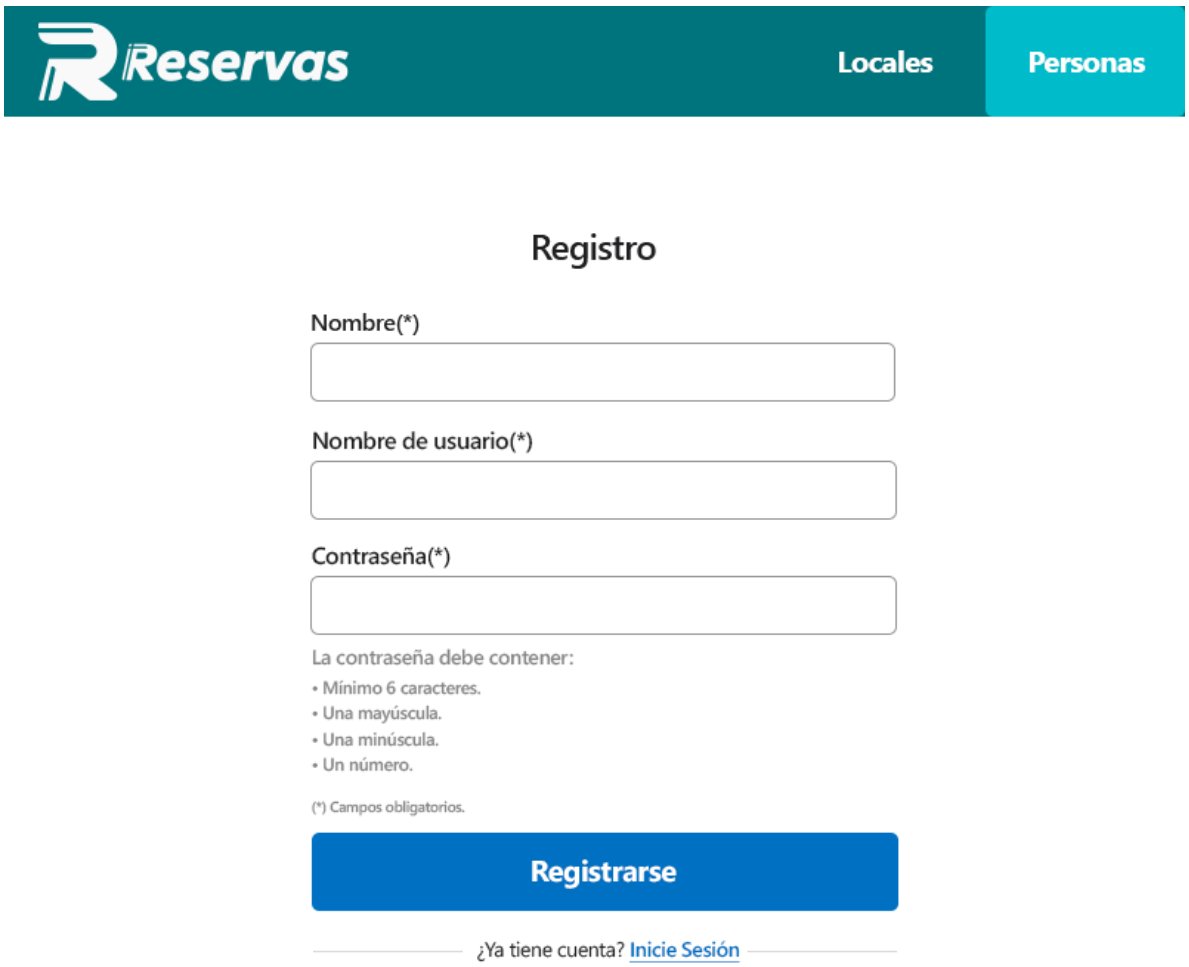
2.10.1. Acceso

Puede ingresar el usuario Persona.

2.10.2. Descripción

Los usuarios Persona deberán ingresar un nombre, nombre de usuario (será único) y contraseña será case sensitive (tendrá un mínimo de 6 caracteres, contando con al menos una mayúscula, una minúscula y un número).

2.10.3. Interfaz de usuario



2.10.4. Validaciones

Se verificará que el nombre de usuario ingresado sea distinto a los registrados previamente, en el caso de que ya estuviera registrado, se mostrará el mensaje “Debe elegir otro nombre de usuario, el que escribió ya encuentra registrado”.

Se verificará que la contraseña cumpla con los requisitos, en el caso de que no los cumpla, se mostrará el mensaje “La contraseña no cumple con los requisitos. Por favor ingrese un contraseña que contenga: mínimo 6 caracteres, al menos una letra mayúscula, una letra minúscula y un número”.

2.11. F11 – Realizar una reserva

2.11.1. Acceso

Puede ingresar el usuario Persona.

2.11.2. Descripción

Se deberá indicar el local a reservar y la cantidad de cupos a reservar. La reserva se procesará si: ese usuario no tiene una reserva pendiente en ese local, las reservas en ese local se encuentran habilitadas y si el local cuenta con cupos disponibles. Toda reserva nueva, se dará de alta automáticamente con estado “Pendiente”.

2.11.3. Interfaz de usuario

2.11.4. Validaciones

Se verificará que: el usuario no tenga reservas pendientes en el local, que el estado de las reservas del local esté habilitado y que el local cuente con suficientes cupos disponibles.

2.12. Cancelar una reserva

2.12.1. Acceso

Puede ingresar el usuario Persona.

2.12.2. Descripción

Se podrá cancelar una reserva “pendiente” propia, pasando la misma a estado “cancelada”, solamente se mostrarán las reservas con estado “pendiente” .

2.12.3. Interfaz de usuario

2.12.4. Validaciones

No se realizarán validaciones dado que no se ingresarán datos. Si no se encuentran reservas, se mostrará un mensaje que dirá “No tiene reservas pendientes”.

2.13. Calificar una reserva finalizada

2.13.1. Acceso

Puede ingresar el usuario Persona.

2.13.2. Descripción

El usuario verá sus reservas finalizadas y podrá calificar las reservas con un valor numérico del 1 a 5.

2.13.3. Interfaz de usuario

2.13.4. Validaciones

Se verificará que el puntaje ingresado sea u número entre 1 y 5. En el caso de que se ingrese un número distinto, se mostrará el mensaje “La calificación debe ser del 1 al 5”.

2.14. F14 – Ver listado de reservas pendientes

2.14.1. Acceso

Puede ingresar el usuario Persona.

2.14.2. Descripción

Al visualizar una reserva se mostrará: el nombre del local, una foto y la cantidad de cupos reservados.

2.14.3. Interfaz de usuario

2.14.4. Validaciones

No se realizarán validaciones dado que no se ingresarán datos.

2.15. F-15: Ver información estadística del porcentaje de reservas – Persona

2.15.1. Acceso

Puede ingresar el usuario Persona.

2.15.2. Descripción

Se mostrará un listado donde aparecerán: los locales donde el usuario realizó reservas que haya completado, la cantidad de reservas que realizó, el total de reservas que tiene ese local y el porcentaje de reservas que corresponden a ese usuario en ese local.

2.15.3. Interfaz de usuario

2.15.4. Validaciones

No se realizarán validaciones dado que no se ingresarán datos.

2.1. F-16: Ver información sobre el local en el que el usuario ha hecho más reservas

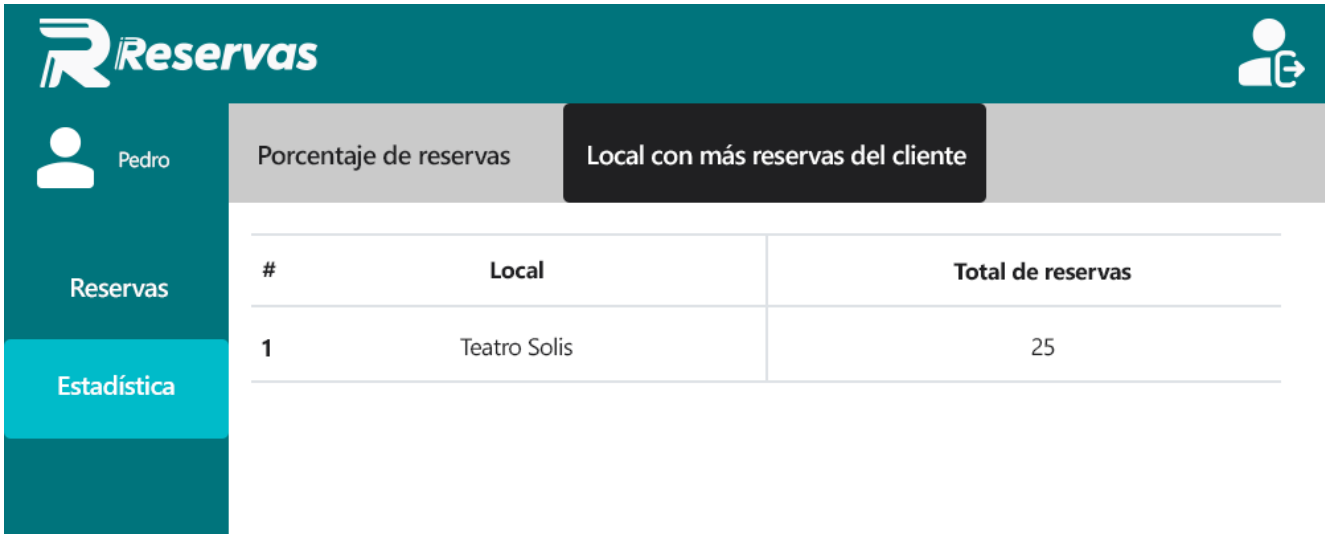
2.1.1. Acceso

Puede ingresar el usuario Persona.

2.1.2. Descripción

Se mostrará un listado donde aparecerá el o los locales donde el usuario ha hecho más reservas.

2.1.3. Interfaz de usuario



2.1.4. Validaciones

No se realizarán validaciones dado que no se ingresarán datos.

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-01: Inicio de Sesión

Precondiciones: - El usuario está autorizado en el sitio con su clave y contraseña como usuario registrado.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F01-T01	Inicio de sesión correcto persona.	1. Selecciona la opción Personas. 2. Aprieto Inicie Sesión. 3. Ingresa nombre de usuario válido. 4. Ingresa contraseña asociada. 5. Presiono botón Entrar.	Array de personas. Nombre de usuario y contraseña.	Visualizar menú de personas, y la pantalla de Solicitar reservas.	Veo menú de personas, y la pantalla de Solicitar reservas.	P
F01-T02	Inicio de sesión incorrecto persona.	1. Selecciona la opción Personas. 2.Aprieto Inicie Sesión. 3. Ingresa nombre de usuario correcto. 4. Ingresa contraseña incorrecta. 5. Presiono botón Entrar.	Array de personas. Nombre de usuario correcto y contraseña incorrecta.	Mensaje de error en la pantalla.	El usuario y/o contraseña es incorrecto.	F
F01-T03	Inicio de sesión correcto local.	1. Selecciona la opción Locales. 2. Ingresa nombre de usuario correcto. 3. Ingresa contraseña correcta. 4. Presiono botón Entrar.	Array de locales. Nombre de usuario y contraseña.	Visualizar menú de locales, y la pantalla de Estado.	Veo menú de locales, y la pantalla de Estado.	P
1F01-T04	Inicio de sesión incorrecto local.	1. Selecciona la opción Locales. 2. Ingresa nombre de usuario correcto. 3. Ingresa contraseña incorrecta. 4. Presiono botón Entrar.	Array de locales. Nombre de usuario y contraseña incorrecta.	Mensaje de error en la pantalla.	El usuario y/o contraseña es incorrecto.	F

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-02: Habilitar y deshabilitar disponibilidad de reservas – Local

Precondiciones: - El local está precargado como habilitado y tiene la opción de deshabilitar reservas.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F02-T01	Deshabilitar disponibilidad de reservas.	1. Ingreso a la app. 2. Aprieto la pestaña Disponibilidad. 3. Aprieto el botón deshabilitar.	Array de locales.	Que el estado del local sea deshabilitado.	La disponibilidad de reserva del local está: Deshabilitado.	P
F02-T02	Habilitar disponibilidad de reservas	1. Ingreso a la app. 2. Aprieto la pestaña Disponibilidad. 3. Aprieto el botón habilitar.	Array de locales.	Que el estado del local sea habilitado.	La disponibilidad de reserva del local está: Habilitado.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-03: Cambiar el estado de las reservas – Local

Precondiciones: - El local cuenta con reservas precargadas en estado pendiente.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F03-T01	Cambiar el estado de una reserva pendiente a finalizada.	1. Ingreso a la app. 2. Veo el menú local en la pestaña estado. 3. Aprieto el botón cambiar.	Array de reservas.	Que el estado de la reserva pase a finalizado y no aparezca más en la tabla.	La reserva desaparece de la tabla porque cambió a estado finalizada. Aparece el aviso No hay reservas en estado pendiente para mostrar.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-04: Buscador dentro de las reservas – Local

Precondiciones: - El local cuenta con reservas precargadas en estado pendiente.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F04-T01	Buscar un nombre de cliente con reserva pendiente.	1. Ingreso a la app. 2. Escribo parte del texto del nombre. 3. Aprieto el botón Buscar.	Array de reservas.	Que se reescriba la tabla con los datos de la búsqueda.	Veo una tabla con el estado de la reserva y con los datos de la búsqueda.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-05: Modificar el cupo máximo – Local

Precondiciones: - El local cuenta con un cupo disponible precargado.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F05-T01	El local no cuenta con reservas pendientes.	1. Ingreso a la app. 2. Veo el menú local en la pestaña estado. 3. Aprieto la pestaña Cupo Máximo. 4. Ingreso en la caja de texto el nuevo cupo máximo. 5. Aprieto el botón Modificar.	Array de locales y array de reservas.	Que aparezca el cupo actual del local y pueda modificarlo. Al ser modificado el cupo, aparezca en la pantalla el nuevo cupo máximo suplantando al anterior.	El cupo máximo del local actualmente es de (el nuevo cupo máximo ingresado).	P
F05-T02	El local cuenta con reservas pendientes.	1. Ingreso a la app. 2. Veo el menú local en la pestaña estado. 3. Aprieto la pestaña Cupo Máximo. 4. Ingreso en la caja de texto el nuevo cupo máximo. 5. Aprieto el botón Modificar.	Array de locales y array de reservas.	Que aparezca un aviso que no se puede modificar el cupo máximo.	No se puede modificar porque hay reservas pendientes.	F

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-06: Ver porcentaje de ocupación del local – Local

Precondiciones: - El local cuenta con reservas precargadas.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F06-T01	El local cuenta con reservas en todos los estados.	1. Ingreso a la app. 2. Veo el menú local en la pestaña estado. 3. Aprieto la pestaña Estadística. 4. Veo la pestaña Información.	Array de reservas y objeto local logueado.	Que aparezca una tabla con la cantidad de: reservas pendientes, reservas, finalizadas, total de reservas y porcentaje de ocupación.	Veo la tabla con la cantidad de: reservas pendientes, reservas, finalizadas, total de reservas y porcentaje de ocupación.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-07: Ver el promedio de calificaciones del local – Local
F-08: – Listado de todos los locales y promedio de calificaciones de cada uno – Local

Precondiciones: - Algunos de los locales cuentan con reservas precargadas finalizadas y con calificación.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F07-T01 F08-T01	El local cuenta con reservas finalizadas y calificadas.	1. Ingreso a la app. 2. Veo el menú local en Estado. 3. Aprieto en el menú Estadística. 4. Aprieto la pestaña Listado de locales.	Array de reservas y array de locales.	Que aparezca una tabla con el nombre de todos los locales y el promedio de calificación de cada uno.	Veo un listado de todos los locales y su promedio de calificación.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-09: Ver total de reservas – Local

Precondiciones: - El local cuenta con reservas precargadas.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F09-T01	Visualizar una tabla con la cantidad de: reservas pendientes, reservas, finalizadas, total de reservas y porcentaje de ocupación.	1. Ingreso a la app. 2. Veo el menú local en Estado. 3. Aprieto la pestaña Estadística. 4. Veo la pestaña Información.	Array de reservas y el objeto local logueado.	Que aparezca una tabla con la cantidad de: reservas pendientes, reservas, finalizadas, total de reservas y porcentaje de ocupación.	Veo una tabla con la cantidad de: reservas pendientes, reservas, finalizadas, total de reservas y porcentaje de ocupación.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-10: Registro en la aplicación – Persona

Precondiciones:

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F10-T01	El nuevo usuario persona se registra ingresando los datos solicitados correctamente.	1. Selecciono la pestaña Personas. 2. Completo los campos nombre, nombre de usuario y contraseña correctamente. La contraseña contiene: seis caracteres, una mayúscula, una minúscula y un número. 3. Aprieto el botón Registrarse.	Se agrega un nuevo objeto persona al array de personas.	Que aparezca un aviso de que se realizó correctamente el registro.	Veo un el aviso “Se registró correctamente”.	P
F10-T02	El nuevo usuario persona se registra ingresando incorrectamente los caracteres de la contraseña.	1. Selecciono la pestaña Personas. 2. Completo los campos nombre, nombre de usuario correctamente y la contraseña incorrectamente. 3. Aprieto el botón Registrarse.	Funciones: levantarDatosRegistroPersona validarContraseniaPersona	Que aparezca un aviso de que hay error.	Aparece el aviso en rojo del dato incorrecto: La contraseña debe contener: Mínimo 6 caracteres. Una mayúscula. Una minúscula. Un número.	F

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-11: Realizar una reserva – Persona

Precondiciones: Ingreso con un usuario y contraseña correctos.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F11-T01	Solicito una reserva en un local en el que el usuario no tenga reservas pendientes.	1. Ingreso a la app. 2. Veo el menú persona, en la pestaña Reservas, Solicitar reservas. 3. Despliego el select y elijo un local. 4. Ingreso cantidad de cupos. 5. Aprieto botón Solicitar reserva.	Array de personas y array de locales.	Mensaje de solicitud exitosa de reserva.	Veo el mensaje la reserva se realizó correctamente.	P
F11-T02	Solicito una reserva en un local en el que el usuario tenga reservas pendientes.	1. Ingreso a la app. 2. Veo el menú persona, en la pestaña Reservas, Solicitar reservas. 3. Despliego el select y elijo un local. 4. Ingreso cantidad de cupos. 5. Aprieto botón Solicitar reserva.	Array de personas y array de locales.	Mensaje de error, la persona no puede realizar reservas en locales con reservas pendientes.	Veo el mensaje no es posible realizar la reserva porque tiene reservas pendientes en ese local.	F
F11-T03	Solicito una reserva en un local que sobrepasa su cupo máximo.	1. Ingreso a la app. 2. Veo el menú persona, en la pestaña Reservas, Solicitar reservas. 3. Despliego el select y elijo un local. 4. Ingreso cantidad de cupos. 5. Aprieto botón Solicitar reserva.	Array de personas y array de locales.	Mensaje de error, la persona no puede realizar reservas en este local porque sobrepasa la cantidad de cupos disponibles.	Veo el mensaje no es posible realizar la reserva porque sobrepasa la cantidad de cupos disponibles del local.	F

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-12: Cancelar una reserva – Persona

Precondiciones: Ingreso con un usuario y contraseña correctos.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F12-T01	Cancelo una reserva pendiente.	1. Ingreso a la app. 2. Veo el menú persona. 3. Aprieto la pestaña Reservas pendientes. 4. Elijo la reserva a cancelar y aprieto el botón.	Array de reservas, array de locales y objeto persona logueada.	Ver en la tabla el estado de la reserva como cancelada.	Veo en la tabla el estado de la reserva Cancelada.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-13: Calificar una reserva finalizada – Persona

Precondiciones: Ingreso con un usuario y contraseña correctos.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F13-T01	Calificar una reserva finalizada.	1. Ingreso a la app. 2. Veo el menú persona. 3. Aprieto la pestaña Reservas finalizadas. 4. Elijo la reserva a cancelar y aprieto el botón. 5. Despliego el select de Calificación. 6. Elijo puntaje. 7. Aprieto el botón Calificar.	Array de reservas, array de locales y objeto persona logueada.	Que la calificación sea asignada a la reserva.	Veo en la tabla el puntaje asignado.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-14: Ver listado de reservas pendientes – Persona

Precondiciones: Ingreso con un usuario y contraseña correctos.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F14-T01	Visualizar listado de reservas pendientes con nombre del local, foto y cantidad de cupos reservados.	1. Ingreso a la app. 2. Veo el menú persona. 3. Aprieto la pestaña Reservas pendientes.	Array de reservas, arrray de locales y objeto persona logueada.	Ver una tabla con el listado de reservas pendientes, nombre del local, foto y cantidad de cupos reservados.	Veo una tabla con el listado de reservas pendientes, nombre del local, foto y cantidad de cupos reservados.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-15: Ver información estadística del porcentaje de reservas – Persona

Precondiciones: Ingreso con un usuario y contraseña correctos.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F15-T01	Visualizar una tabla con información sobre la cantidad de reservas finalizadas del cliente, el total de reservas del local y el porcentaje de reservas del cliente en ese local.	1. Ingreso a la app. 2. Veo el menú persona. 3. Aprieto en el menú Estadística.	Array de reservas, array de locales y objeto persona logueada.	Ver una tabla con información sobre la cantidad de reservas finalizadas del usuario logueado, el total de reservas del local y el porcentaje de reservas del usuario logueado en ese local.	Veo una tabla con información sobre la cantidad de reservas finalizadas del usuario logueado, el total de reservas del local y el porcentaje de reservas del usuario logueado en ese local.	P

PLANTILLA CASOS DE PRUEBA - PROGRAMACIÓN I – OBLIGATORIO

F-16: Ver información sobre el local en el que el usuario ha hecho más reservas

Precondiciones: Ingreso con un usuario y contraseña correctos.

ID	Escenario de test	Pasos	Datos utilizados	Resultado esperado	Resultado obtenido	Estado (F=Falla, P=pasa)
F16-T01	Ver información sobre el local en el que el cliente ha hecho más reservas.	1. Ingreso a la app. 2. Veo el menú persona. 3. Aprieto en el menú Estadística. 4. Aprieto la pestaña Local con más reservas del cliente.	Array de reservas, array local favorito, array nombre local calculado y objeto persona logueada.	Visualizar información sobre el local en el que el usuario logueado ha hecho más reservas.	Veo información sobre el local en el que el usuario logueado ha hecho más reservas.	P

Información precargada

Locales

```
//LOCALES

  //Restaurantes

  agregarLocalPrecargado("McDonald's", "mcdonalds", "mc123", "Restaurante", "Av. Luis Alberto de Herrera 1534", 100, 100, "../imagenesObligatorio/foto6.jpg", "Habilitado", "../img/logoMcDonalds.svg");

  agregarLocalPrecargado("Burger King", "burgerking", "bk123", "Restaurante", "Plácido Ellauri 3494", 150, 150, "../imagenesObligatorio/foto5.jpg", "Habilitado", "../img/logoBurgerKing.svg");

  agregarLocalPrecargado("Subway", "subway", "subway123", "Restaurante", "Av. Luis Alberto de Herrera 1290", 70, 70, "../imagenesObligatorio/foto4.jpg", "Habilitado", "../img/logoSubway.svg");

  //Museos

  agregarLocalPrecargado("Museo Figari", "mfigari", "mfigari123", "Museo", "Juan Carlos Gómez 1427", 200, 200, "../imagenesObligatorio/foto2.jpg", "Habilitado", "../img/logoMuseoFigari.svg");

  agregarLocalPrecargado("Museo Zorrilla", "mzorrilla", "mzorrilla123", "Museo", "José Luis Zorrilla de San Martín 96", 250, 250, "../imagenesObligatorio/foto10.jpg", "Habilitado", "../img/logoMuseoZorrilla.svg");

  //Teatros

  agregarLocalPrecargado("Teatro Solís", "tsolis", "tsolis123", "Teatro", "Buenos Aires esq. Bartolomé Miter", 300, 300, "../imagenesObligatorio/foto1.jpg", "Habilitado", "../img/logoTeatroSolis.svg");

  agregarLocalPrecargado("Teatro Sodre", "tsodre", "tsodre123", "Teatro", "Andes esquina Mercedes", 250, 250, "../imagenesObligatorio/foto3.jpg", "Habilitado", "../img/logoTeatroSodre.png");
```

Personas

```
 //PERSONAS

  agregarPersonaPrecargada("Lola", "lola23", "Lola123");

  agregarPersonaPrecargada("Yordi", "yordi", "Yordi123");

  agregarPersonaPrecargada("Roberto", "rober", "Rober123");

  agregarPersonaPrecargada("Jorge", "jorgito", "Jorgito123");

  agregarPersonaPrecargada("Marta", "martita", "Martita123");

  agregarPersonaPrecargada("Julian", "julian1", "Julian123");
```

```
agregarPersonaPrecargada("Lucas", "lukita", "Lukita123");
agregarPersonaPrecargada("Pepito", "pepe", "Pepe123");
```

Reservas

```
//RESERVAS
agregarReservaPrecargada("Lola","lola23", "McDonald's", "Pendiente", 30, 0);
agregarReservaPrecargada("Roberto","rober", "McDonald's", "Pendiente", 4, 0);
agregarReservaPrecargada("Lola","lola23", "Subway", "Finalizada", 10, 5);
agregarReservaPrecargada("Lola","lola23", "Museo Figari", "Finalizada", 6, 2);
agregarReservaPrecargada("Marta","martita", "Subway", "Finalizada", 2, 5);
agregarReservaPrecargada("Lola","lola23", "Subway", "Finalizada", 5, 1);
agregarReservaPrecargada("Roberto","rober", "Teatro Solís", "Pendiente", 35, 0);
agregarReservaPrecargada("Roberto","rober", "Teatro Sodre", "Finalizada", 35, 4);
agregarReservaPrecargada("Julian","julian1", "Burger King", "Pendiente", 1, 0);
agregarReservaPrecargada("Lucas","lukita", "Museo Figari", "Pendiente", 7, 0);
agregarReservaPrecargada("Pepito","pepe", "Museo Zorrilla", "Pendiente", 54, 0);
```

```
window.addEventListener("load", inicio);

//Variable global para guardar el local o la persona logueada (objeto)
let localLogueado = null;
let personaLogueada = null;

//OBJETOS
//Defino el id de locales en 1
let idLocal = 1;

//Defino el objeto Local
class Local {
  constructor(unNombre, unNombreUsuario, unaContrasenia, unTipoDeLocal, unaDireccion,
unCupoMaximo, unCupoRestante, unaFoto, unEstado, unLogo) {
    this.id = idLocal;
    this.nombre = unNombre;
    this.nombreUsuario = unNombreUsuario;
    this.contrasenia = unaContrasenia;
    this.tipoDeLocal = unTipoDeLocal;
    this.direccion = unaDireccion;
    this.cupoMaximo = unCupoMaximo;
    this.cuposRestantes = unCupoRestante;
    this.foto = unaFoto;
    this.estado = unEstado;
    this.logo = unLogo;
    idLocal++;
  }
}

//Defino el id de personas en 1
let idPersona = 1;

//Defino el objeto Persona
class Persona {
  constructor(unNombre, unNombreUsuario, unaContrasenia) {
    this.id = idPersona;
    this.nombre = unNombre;
    this.nombreUsuario = unNombreUsuario;
    this.contrasenia = unaContrasenia;
    this.imagenUsuario = "../img/user.svg";
    idPersona++;
  }
}

//Defino el contador de reservas en 1
let contadorReservas = 1;

//Defino el objeto Reserva
class Reserva {
  constructor(unNombreCliente, unUsuarioCliente, unNombreLocal, unEstado,
unaCantidadDeCuposAReservar,unaCalificacion) {
    this.contador = contadorReservas;
    this.nombreCliente = unNombreCliente;
    this.usuarioCliente = unUsuarioCliente;
```

```
    this.nombreLocal = unNombreLocal;
    this.estado = unEstado;
    this.cantidadDeCuposAReservar = unaCantidadDeCuposAReservar;
    this.calificacion=unaCalificacion;
    contadorReservas++;
  }
}

//Defino el objeto Local favorito como ayuda para calcular el local con más reservas del cliente
class LocalFavorito {
  constructor(unNombre, unaCantidadDeReservas) {
    this.nombre = unNombre;
    this.cantidadDeReservas = unaCantidadDeReservas;
  }
}


function inicio() {
  //Precargamos los datos (de los objetos)
  precargarDatos();

  //Ocultamos todos los divs
  ocultarTodo();

  //Mostramos la pantalla de inicio
  mostrarAlInicio();

  //Escuchamos botones (eventos de click)
  escucharBotones();
}

//Funcion que oculta todo
function ocultarTodo() {
  ocultarMenuLogIn();
  ocultarMenuLogOut();
  ocultarMenuLocales();
  ocultarMenuPersonas();
  ocultarMenusArriba();
  ocultarBuscador();
  ocultarDivInicioSesionLocales();
  ocultarDivRegistroPersonas();
  ocultarDivInicioSesionPersonas();
  ocultarDivEstadoLocal();
  ocultarDivCupoMaximoLocal();
  ocultarDivDisponibilidadLocal();
  ocultarDivEstadisticaLocalInformacion();
  ocultarDivEstadisticaLocalListadoDeLocales();
  ocultarDivReservasPersonasSolicitarReservas();
  ocultarDivReservasPersonasReservasPendientes();
  ocultarDivReservasPersonasReservasFinalizadas();
  ocultarDivEstadiscticaPersonasPorcentajeDeReservas();
  ocultarDivEstadiscticaPersonasLocalConMasReservas();
}
```

```
//Funcion que muestra la pantalla de inicio (Inicio Sesion - Locales)
function mostrarAlInicio() {
  mostrarMenuLogIn();
  mostrarDivInicioSesionLocales();
  //Añadimos la clase activo a locales para que quede seleccionado de entrada (con color celeste)
  document.querySelector("#locales").classList.add("activo");
}

//Funcion que escucha todos los botones
function escucharBotones() {
  //Inicio Sesion Locales
  mostrarInicioSesionLocales();
  //Registro Personas
  mostrarRegistroPersonas();
  //Inicio Sesion Personas
  mostrarInicioSesionPersonas();

  //LOCALES
  mostrarEstadoLocal();
  mostrarCupoMaximoLocal();
  mostrarDisponibilidadLocal();
  //Estadística Locales
  mostrarEstadisticaLocalInformacion();
  mostrarEstadisticaLocalListadoDeLocales();
  //PERSONAS
  //Reservas Personas
  mostrarReservasPersonasSolicitarReservas();
  mostrarReservasPersonasReservasPendientes();
  mostrarReservasPersonasReservasFinalizadas();
  //Estadística Personas
  mostrarEstadiscticaPersonasPorcentajeDeReservas();
  mostrarEstadisticaPersonasLocalConMasReservas();
}

//Funcion que cierra sesion
function logOut() {
  ocultarTodo();
  document.querySelector("#pErroresLoginLocales").innerHTML = "";
  document.querySelector("#pErroresRegistroPersonas").innerHTML = "";
  document.querySelector("#pErroresLoginPersonas").innerHTML = "";
  localLogueado = null;
  personaLogueada = null;
  mostrarAlInicio();
}

//-------- ESCUCHA DE CLICKS DE MOSTRAR Y SU CORRESPONDIENTE DIV --------

//Funcion que escucha los clicks para luego mostrar lo necesario en el inicio sesion de locales
function mostrarInicioSesionLocales() {
  document.querySelector("#locales").addEventListener("click", ocultarTodo);
  document.querySelector("#locales").addEventListener("click", mostrarMenuLogIn);
  document.querySelector("#locales").addEventListener("click", mostrarDivInicioSesionLocales);
  document.querySelector("#cerrarSesion").addEventListener("click", logOut);
  document.querySelector("#btnEntrarLocales").addEventListener("click", levantarDatosLoginLocales);
```

```
}

//Funcion que muestra el div inicio sesion de locales
function mostrarDivInicioSesionLocales() {
  //Muestro el div
  document.querySelector("#divInicioSesionLocales").style.display = "block";
  //Añadimos la clase activo a locales para que quede seleccionado de entrada (con color celeste)
  document.querySelector("#locales").classList.add("activo");
  //Y le quitamos (si es que tiene) la clase activo a personas
  document.querySelector("#personas").classList.remove("activo");
}

//Funcion que muestra el div menu de logueo (local o personas)
function mostrarMenuLogIn() {
  document.querySelector("#divMenuLogueoYRegistro").style.display = "block";
}

//Funcion que muestra el div menu de deslogueo (iconito para cerrar sesión)
function mostrarMenuLogOut() {
  document.querySelector("#divMenuCerrarSesion").style.display = "block";
}

//Funcion que escucha los clicks para luego mostrar lo necesario en el registro de personas
function mostrarRegistroPersonas() {
  document.querySelector("#personas").addEventListener("click", ocultarTodo);
  document.querySelector("#cuentaUsuarioRegistro").addEventListener("click", ocultarTodo);
  document.querySelector("#personas").addEventListener("click", mostrarMenuLogIn);
  document.querySelector("#cuentaUsuarioRegistro").addEventListener("click", mostrarMenuLogIn);
  document.querySelector("#personas").addEventListener("click", mostrarDivRegistroPersonas);
  document.querySelector("#cuentaUsuarioRegistro").addEventListener("click",
mostrarDivRegistroPersonas);
  document.querySelector("#btnRegistrar").addEventListener("click", levantarDatosRegistroPersonas);
}

//Funcion que muestra el div registro de personas
function mostrarDivRegistroPersonas() {
  document.querySelector("#divRegistroPersonas").style.display = "block";
  //Le quitamos (si es que tiene) la clase activo a locales
  document.querySelector("#locales").classList.remove("activo");
  //Añadimos la clase activo a personas para que quede seleccionado de entrada (con color celeste)
  document.querySelector("#personas").classList.add("activo");
}

//Funcion que escucha los clicks para luego mostrar lo necesario en el inicio sesion de personas
function mostrarInicioSesionPersonas() {
  document.querySelector("#cuentaUsuarioInicioSesion").addEventListener("click", ocultarTodo);
  document.querySelector("#cuentaUsuarioInicioSesion").addEventListener("click", mostrarMenuLogIn);
  document.querySelector("#cuentaUsuarioInicioSesion").addEventListener("click",
mostrarDivInicioSesionPersonas);
  document.querySelector("#btnEntrarPersonas").addEventListener("click", levantarDatosLoginPersonas);
}

//Funcion que muestra el div inicio de sesion de las personas
function mostrarDivInicioSesionPersonas() {
```

```
  document.querySelector("#divInicioSesionPersonas").style.display = "block";
  document.querySelector("#locales").classList.remove("activo");
  //Añadimos la clase activo a personas para que quede seleccionado de entrada (con color celeste)
  document.querySelector("#personas").classList.add("activo");
}

//Funcion que escucha los clicks para luego mostrar lo necesario en el estado del local
function mostrarEstadoLocal() {
  document.querySelector("#menuEstadoLocales").addEventListener("click", ocultarTodo);
  document.querySelector("#menuEstadoLocales").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#menuEstadoLocales").addEventListener("click", mostrarMenuLocales);
  document.querySelector("#menuEstadoLocales").addEventListener("click", mostrarBuscador);
  document.querySelector("#menuEstadoLocales").addEventListener("click", mostrarDivEstadoLocal);
  document.querySelector("#btnBuscar").addEventListener("click", buscarReserva);
}

//Funcion que muestra el div estado local
function mostrarDivEstadoLocal() {
  document.querySelector("#divEstadoLocal").style.display = "block";
  //Le quitamos (si es que tiene) la clase activo a todos los elementos del menu
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralLocales li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuEstadoLocales").classList.add("activo");
  mostrarTablaEstadoLocal();
}

//Funcion que muestra el div menu de los locales
function mostrarMenuLocales() {
  document.querySelector(".sidebar").style.display = "block";
  document.querySelector("#menuLateralLocales").style.display = "block";
  document.querySelector("#logoLocal").innerHTML = `<img src="${localLogueado.logo}">`;
  document.querySelector("#nombreLocal").innerHTML = `¡Hola ${localLogueado.nombre}!`;
}

//Funcion que escucha los clicks para luego mostrar lo necesario en el cupo maximo del local
function mostrarCupoMaximoLocal() {
  document.querySelector("#menuCupoMaximoLocales").addEventListener("click", ocultarTodo);
  document.querySelector("#menuCupoMaximoLocales").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#menuCupoMaximoLocales").addEventListener("click", mostrarMenuLocales);
  document.querySelector("#menuCupoMaximoLocales").addEventListener("click",
mostrarDivCupoMaximoLocal);
  document.querySelector("#btnModificar").addEventListener("click", cambiarCupoMaximo);
  //Le quitamos (si es que tiene) la clase activo a todos los elementos del menu
  // document.querySelector("#menuLateralLocales li").classList.remove("activo");
}

//Funcion que muestra el div de cupo maximo del local
function mostrarDivCupoMaximoLocal() {
  document.querySelector("#divCupoMaximoLocal").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralLocales li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
```

```
  }
  document.querySelector("#menuCupoMaximoLocales").classList.add("activo");

  mostrarFuncionalidadCupoMaximoLocal();
}

//Funcion que escucha los clicks para luego mostrar lo necesario en la disponibilidad del local
function mostrarDisponibilidadLocal() {
  document.querySelector("#menuDisponibilidadLocales").addEventListener("click", ocultarTodo);
  document.querySelector("#menuDisponibilidadLocales").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#menuDisponibilidadLocales").addEventListener("click", mostrarMenuLocales);
  document.querySelector("#menuDisponibilidadLocales").addEventListener("click",
mostrarDivDisponibilidadLocal);
  document.querySelector("#btnHabilitar").addEventListener("click", habilitarDisponibilidad);
  document.querySelector("#btnDeshabilitar").addEventListener("click", deshabilitarDisponibilidad);
}

//Funcion que muestra el div disponibilidad del local
function mostrarDivDisponibilidadLocal() {
  document.querySelector("#divDisponibilidadLocal").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralLocales li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuDisponibilidadLocales").classList.add("activo");

  mostrarFuncionalidadDisponibilidad();
}

//Funcion que escucha los clicks para luego mostrar lo necesario en la estadistica del local informacion
function mostrarEstadisticaLocalInformacion() {
  document.querySelector("#menuEstadisticaLocales").addEventListener("click", ocultarTodo);
  document.querySelector("#informacion").addEventListener("click", ocultarTodo);
  document.querySelector("#menuEstadisticaLocales").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#informacion").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#menuEstadisticaLocales").addEventListener("click", mostrarMenuLocales);
  document.querySelector("#informacion").addEventListener("click", mostrarMenuLocales);
  document.querySelector("#menuEstadisticaLocales").addEventListener("click",
mostrarDivEstadisticaLocalInformacion);
  document.querySelector("#informacion").addEventListener("click",
mostrarDivEstadisticaLocalInformacion);
}

//Funcion que muestra el div estadistica del local informacion
function mostrarDivEstadisticaLocalInformacion() {
  document.querySelector("#menuArribaEstadisticaLocales").style.display = "block";
  document.querySelector("#informacionEstadisticaLocal").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralLocales li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuEstadisticaLocales").classList.add("activo");
  let arrayElementosMenuArriba = document.querySelectorAll("#menuArribaEstadisticaLocales li");
  for (let elemento of arrayElementosMenuArriba) {
```

```
    elemento.classList.remove("activo2");
  }
  document.querySelector("#informacion").classList.add("activo2");

  mostrarTablaEstadisticaLocalesInformacion();
}

//Funcion que escucha los clicks para luego mostrar lo necesario en la estadistica del local listado de locales
function mostrarEstadisticaLocalListadoDeLocales() {
  document.querySelector("#listadoDeLocales").addEventListener("click", ocultarTodo);
  document.querySelector("#listadoDeLocales").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#listadoDeLocales").addEventListener("click", mostrarMenuLocales);
  document.querySelector("#listadoDeLocales").addEventListener("click",
mostrarDivEstadisticaLocalListadoDeLocales);
  document.querySelector("#listadoDeLocales").addEventListener("click",
mostrarDivEstadisticaLocalListadoDeLocales);
}

//Funcion que muestra el div estadistica del local listado de locales
function mostrarDivEstadisticaLocalListadoDeLocales() {
  document.querySelector("#menuArribaEstadisticaLocales").style.display = "block";
  document.querySelector("#listadoDeLocalesEstadisticaLocal").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralLocales li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuEstadisticaLocales").classList.add("activo");
  let arrayElementosMenuArriba = document.querySelectorAll("#menuArribaEstadisticaLocales li");
  for (let elemento of arrayElementosMenuArriba) {
    elemento.classList.remove("activo2");
  }
  document.querySelector("#listadoDeLocales").classList.add("activo2");

  mostrarTablaEstadisticaLocalListadoDeLocales();
}

//RESERVAS PERSONAS
//Funcion que escucha los clicks para luego mostrar lo necesario en las reservas de personas solicitar
reservas
function mostrarReservasPersonasSolicitarReservas() {
  document.querySelector("#menuReservaPersonas").addEventListener("click", ocultarTodo);
  document.querySelector("#solicitarReservas").addEventListener("click", ocultarTodo);
  document.querySelector("#menuReservaPersonas").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#solicitarReservas").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#menuReservaPersonas").addEventListener("click", mostrarMenuPersonas);
  document.querySelector("#solicitarReservas").addEventListener("click", mostrarMenuPersonas);
  document.querySelector("#menuReservaPersonas").addEventListener("click",
mostrarDivReservasPersonasSolicitarReservas);
  document.querySelector("#solicitarReservas").addEventListener("click",
mostrarDivReservasPersonasSolicitarReservas);
  document.querySelector("#btnSolicitarReserva").addEventListener("click", nuevaReserva);
}

//Funcion que muestra el div reservas personas solicitar reservas
```

```
function mostrarDivReservasPersonasSolicitarReservas() {
  document.querySelector("#menuArribaReservasPersonas").style.display = "block";
  document.querySelector("#solicitarReservasPersonas").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralPersonas li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuReservaPersonas").classList.add("activo");
  let arrayElementosMenuArriba = document.querySelectorAll("#menuArribaReservasPersonas li");
  for (let elemento of arrayElementosMenuArriba) {
    elemento.classList.remove("activo2");
  }
  document.querySelector("#solicitarReservas").classList.add("activo2");

  cargarSelectSolicitarReservaPersonas();
}

//Funcion que escucha los clicks para luego mostrar lo necesario en las reservas de personas reservas
pendientes
function mostrarReservasPersonasReservasPendientes() {
  document.querySelector("#reservasPendientes").addEventListener("click", ocultarTodo);
  document.querySelector("#reservasPendientes").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#reservasPendientes").addEventListener("click", mostrarMenuPersonas);
  document.querySelector("#reservasPendientes").addEventListener("click",
mostrarDivReservasPersonasReservasPendientes);
}

//Funcion que muestra el div reservas personas reservas pendientes
function mostrarDivReservasPersonasReservasPendientes() {
  document.querySelector("#menuArribaReservasPersonas").style.display = "block";
  document.querySelector("#reservasPendientesPersonas").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralPersonas li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuReservaPersonas").classList.add("activo");
  let arrayElementosMenuArriba = document.querySelectorAll("#menuArribaReservasPersonas li");
  for (let elemento of arrayElementosMenuArriba) {
    elemento.classList.remove("activo2");
  }
  document.querySelector("#reservasPendientes").classList.add("activo2");

  mostrarTablaReservasPendientesPersonas();
}

//Funcion que escucha los clicks para luego mostrar lo necesario en las reservas de personas reservas
finalizadas
function mostrarReservasPersonasReservasFinalizadas() {
  document.querySelector("#reservasFinalizadas").addEventListener("click", ocultarTodo);
  document.querySelector("#reservasFinalizadas").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#reservasFinalizadas").addEventListener("click", mostrarMenuPersonas);
  document.querySelector("#reservasFinalizadas").addEventListener("click",
mostrarDivReservasPersonasReservasFinalizadas);
}
```

```
//Funcion que muestra el div reservas personas reservas finalizadas
function mostrarDivReservasPersonasReservasFinalizadas() {
  document.querySelector("#menuArribaReservasPersonas").style.display = "block";
  document.querySelector("#reservasFinalizadasPersonas").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralPersonas li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuReservaPersonas").classList.add("activo");
  let arrayElementosMenuArriba = document.querySelectorAll("#menuArribaReservasPersonas li");
  for (let elemento of arrayElementosMenuArriba) {
    elemento.classList.remove("activo2");
  }
  document.querySelector("#reservasFinalizadas").classList.add("activo2");

  mostrarTablaReservasFinalizadasPersonas();
}

//ESTADISTICA PERSONAS
//Funcion que escucha los clicks para luego mostrar lo necesario en las estadisticas de personas
//por porcentaje de reservas
function mostrarEstadiscticaPersonasPorcentajeDeReservas() {
  document.querySelector("#menuEstadisticaPersonas").addEventListener("click", ocultarTodo);
  document.querySelector("#porcentajeReservas").addEventListener("click", ocultarTodo);
  document.querySelector("#menuEstadisticaPersonas").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#porcentajeReservas").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#menuEstadisticaPersonas").addEventListener("click", mostrarMenuPersonas);
  document.querySelector("#porcentajeReservas").addEventListener("click", mostrarMenuPersonas);
  document.querySelector("#menuEstadisticaPersonas").addEventListener("click",
mostrarDivEstadiscticaPersonasPorcentajeDeReservas);
  document.querySelector("#porcentajeReservas").addEventListener("click",
mostrarDivEstadiscticaPersonasPorcentajeDeReservas);
}

//Funcion que muestra el div estadistica personas por porcentaje de reservas
function mostrarDivEstadiscticaPersonasPorcentajeDeReservas() {
  document.querySelector("#menuArribaEstadisticasPersonas").style.display = "block";
  document.querySelector("#porcentajeDeReservasPersonas").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralPersonas li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuEstadisticaPersonas").classList.add("activo");
  let arrayElementosMenuArriba = document.querySelectorAll("#menuArribaEstadisticasPersonas li");
  for (let elemento of arrayElementosMenuArriba) {
    elemento.classList.remove("activo2");
  }
  document.querySelector("#porcentajeReservas").classList.add("activo2");

  mostrarTablaEstadisticaPersonasPorcentajeDeReservas();
}
```

```
//Funcion que escucha los clicks para luego mostrar lo necesario en las estadisticas de personas
//local con mas reservas
function mostrarEstadisticaPersonasLocalConMasReservas() {
  document.querySelector("#localConMasReservasCliente").addEventListener("click", ocultarTodo);
  document.querySelector("#localConMasReservasCliente").addEventListener("click", mostrarMenuLogOut);
  document.querySelector("#localConMasReservasCliente").addEventListener("click",
mostrarMenuPersonas);
  document.querySelector("#localConMasReservasCliente").addEventListener("click",
mostrarDivEstadisticaPersonasLocalConMasReservas);
}

//Funcion que muestra el div estadistica personas local con mas reservas
function mostrarDivEstadisticaPersonasLocalConMasReservas() {
  document.querySelector("#menuArribaEstadisticasPersonas").style.display = "block";
  document.querySelector("#localConMasReservasPersonas").style.display = "block";
  let arrayElementosMenuLateral = document.querySelectorAll("#menuLateralPersonas li");
  for (let elemento of arrayElementosMenuLateral) {
    elemento.classList.remove("activo");
  }
  document.querySelector("#menuEstadisticaPersonas").classList.add("activo");
  let arrayElementosMenuArriba = document.querySelectorAll("#menuArribaEstadisticasPersonas li");
  for (let elemento of arrayElementosMenuArriba) {
    elemento.classList.remove("activo2");
  }
  document.querySelector("#localConMasReservasCliente").classList.add("activo2");

  mostrarTablaEstadisticaPersonasLocalConMasReservasDelCliente();
}

//Funcion mostrar menu de personas
function mostrarMenuPersonas() {
  document.querySelector(".sidebar").style.display = "block";
  document.querySelector("#menuLateralPersonas").style.display = "block";
  document.querySelector("#logoPersona").innerHTML = `<img src="$
{personaLogueada.imagenUsuario}">`;
  document.querySelector("#nombrePersona").innerHTML = `¡Hola ${personaLogueada.nombre}!`;
}

//Funcion que muestra el div buscador
function mostrarBuscador() {
  document.querySelector("#buscador").style.display = "block";
}
//-------- HASTA ACA ESCUCHA LOS CLICKS DE MOSTRAR Y SU CORRESPONDIENTE DIV --------

//-------- OCULTA LOS DIVS = display:none --------

//Funcion que oculta el div menu de logueo (local o personas)
function ocultarMenuLogIn() {
  document.querySelector("#divMenuLogueoYRegistro").style.display = "none";
}

//Funcion que oculta el div menu de deslogueo (iconito para cerrar sesión)
function ocultarMenuLogOut() {
  document.querySelector("#divMenuCerrarSesion").style.display = "none";
```

```
}

//Funcion que oculta el div menu de los locales
function ocultarMenuLocales() {
  document.querySelector(".sidebar").style.display = "none";
  document.querySelector("#menuLateralLocales").style.display = "none";
}

//Funcion que oculta el div menu de las personas
function ocultarMenuPersonas() {
  document.querySelector(".sidebar").style.display = "none";
  document.querySelector("#menuLateralPersonas").style.display = "none";
}

//Funcion que oculta el div menu de arriba de ciertas secciones
function ocultarMenusArriba() {
  //Uso document.querySelectorAll porque me devuelve un array con TODOS los elementos con la
clase .menuArriba
  //si usara document.querySelector solo me devolvería el primer objeto que encuentre con esa clase
  let arrayMenusArriba = document.querySelectorAll(".menuArriba");

  for (let menus of arrayMenusArriba) {
    menus.style.display = "none";
  }
}

//Funcion que oculta el div buscador
function ocultarBuscador() {
  document.querySelector("#buscador").style.display = "none";
}

//Funcion que oculta el div inicio sesion de locales
function ocultarDivInicioSesionLocales() {
  document.querySelector("#divInicioSesionLocales").style.display = "none";
}

//Funcion que oculta el div registro de personas
function ocultarDivRegistroPersonas() {
  document.querySelector("#divRegistroPersonas").style.display = "none";
}

//Funcion que oculta el div inicio de sesión de personas
function ocultarDivInicioSesionPersonas() {
  document.querySelector("#divInicioSesionPersonas").style.display = "none";
}

//Funcion que oculta el div estado local
function ocultarDivEstadoLocal() {
  document.querySelector("#divEstadoLocal").style.display = "none";
}

//Funcion que oculta el div cupo maximo del local
function ocultarDivCupoMaximoLocal() {
  document.querySelector("#divCupoMaximoLocal").style.display = "none";
```

```
}

//Funcion que oculta el div disponibilidad del local
function ocultarDivDisponibilidadLocal() {
  document.querySelector("#divDisponibilidadLocal").style.display = "none";
}

//Funcion que oculta el div estadistica del local
function ocultarDivEstadisticaLocalInformacion() {
  document.querySelector("#informacionEstadisticaLocal").style.display = "none";
}

//Funcion que oculta el div estadistica del local
function ocultarDivEstadisticaLocalListadoDeLocales() {
  document.querySelector("#listadoDeLocalesEstadisticaLocal").style.display = "none";
}

//Funcion que oculta el div reservas de las personas
function ocultarDivReservasPersonasSolicitarReservas() {
  document.querySelector("#solicitarReservasPersonas").style.display = "none";
}

//Funcion que oculta el div reservas de las personas
function ocultarDivReservasPersonasReservasPendientes() {
  document.querySelector("#reservasPendientesPersonas").style.display = "none";
}

//Funcion que oculta el div reservas de las personas
function ocultarDivReservasPersonasReservasFinalizadas() {
  document.querySelector("#reservasFinalizadasPersonas").style.display = "none";
}

function ocultarDivEstadiscticaPersonasPorcentajeDeReservas() {
  document.querySelector("#porcentajeDeReservasPersonas").style.display = "none";
}

function ocultarDivEstadiscticaPersonasLocalConMasReservas() {
  document.querySelector("#localConMasReservasPersonas").style.display = "none";
}
//-------- HASTA ACA OCULTA LOS DIVS --------

//------ FUNCION QUE PRECARGA LOS DATOS ------
function precargarDatos() {
  //LOCALES
  //Restaurantes
  agregarLocalPrecargado("McDonald's", "mcdonalds", "mc123", "Restaurante", "Av. Luis Alberto de Herrera
1534", 100, 100, "../imagenesObligatorio/foto6.jpg", "Habilitado", "../img/logoMcDonalds.svg");
  agregarLocalPrecargado("Burger King", "burgerking", "bk123", "Restaurante", "Plácido Ellauri 3494", 150,
150, "../imagenesObligatorio/foto5.jpg", "Habilitado", "../img/logoBurgerKing.svg");
  agregarLocalPrecargado("Subway", "subway", "subway123", "Restaurante", "Av. Luis Alberto de Herrera
1290", 70, 70, "../imagenesObligatorio/foto4.jpg", "Habilitado", "../img/logoSubway.svg");
  //Museos
  agregarLocalPrecargado("Museo Figari", "mfigari", "mfigari123", "Museo", "Juan Carlos Gómez 1427", 200,
200, "../imagenesObligatorio/foto2.jpg", "Habilitado", "../img/logoMuseoFigari.svg");
```

```
  agregarLocalPrecargado("Museo Zorrilla", "mzorrilla", "mzorrilla123", "Museo", "José Luis Zorrilla de San
Martín 96", 250, 250, "../imagenesObligatorio/foto10.jpg", "Habilitado", "../img/logoMuseoZorrilla.svg");
  //Teatros
  agregarLocalPrecargado("Teatro Solís", "tsolis", "tsolis123", "Teatro", "Buenos Aires esq. Bartolomé Miter",
300, 300, "../imagenesObligatorio/foto1.jpg", "Habilitado", "../img/logoTeatroSolis.svg");
  agregarLocalPrecargado("Teatro Sodre", "tsodre", "tsodre123", "Teatro", "Andes esquina Mercedes", 250,
250, "../imagenesObligatorio/foto3.jpg", "Habilitado", "../img/logoTeatroSodre.png");

  //PERSONAS
  agregarPersonaPrecargada("Lola", "lola23", "Lola123");
  agregarPersonaPrecargada("Yordi", "yordi", "Yordi123");
  agregarPersonaPrecargada("Roberto", "rober", "Rober123");
  agregarPersonaPrecargada("Jorge", "jorgito", "Jorgito123");
  agregarPersonaPrecargada("Marta", "martita", "Martita123");
  agregarPersonaPrecargada("Julian", "julian1", "Julian123");
  agregarPersonaPrecargada("Lucas", "lukita", "Lukita123");
  agregarPersonaPrecargada("Pepito", "pepe", "Pepe123");

  //RESERVAS
  agregarReservaPrecargada("Lola","lola23", "McDonald's", "Pendiente", 30, 0);
  agregarReservaPrecargada("Roberto","rober", "McDonald's", "Pendiente", 4, 0);
  agregarReservaPrecargada("Lola","lola23", "Subway", "Finalizada", 10, 5);
  agregarReservaPrecargada("Lola","lola23", "Museo Figari", "Finalizada", 6, 2);
  agregarReservaPrecargada("Marta","martita", "Subway", "Finalizada", 2, 5);
  agregarReservaPrecargada("Lola","lola23", "Subway", "Finalizada", 5, 1);
  agregarReservaPrecargada("Roberto","rober", "Teatro Solís", "Pendiente", 35, 0);
  agregarReservaPrecargada("Roberto","rober", "Teatro Sodre", "Finalizada", 35, 4);
  agregarReservaPrecargada("Julian","julian1", "Burger King", "Pendiente", 1, 0);
  agregarReservaPrecargada("Lucas","lukita", "Museo Figari", "Pendiente", 7, 0);
  agregarReservaPrecargada("Pepito","pepe", "Museo Zorrilla", "Cancelada", 54, 0);
}

//ARRAYS
//Array de Locales
let listaLocales = new Array();

//Array de Personas
let listaPersonas = new Array();

//Array de Reservas
let listaReservas = new Array();

console.log(listaLocales)

// Funcion agregar locales precargados
function agregarLocalPrecargado(nombreLocal, nombreUsuarioLocal, contraseniaLocal, tipoDeLocal,
direccionLocal, cupoMaximoLocal, cupoRestante, fotoLocal, estadoLocal, logoLocal) {
  let unLocal = new Local(nombreLocal, nombreUsuarioLocal, contraseniaLocal, tipoDeLocal, direccionLocal,
cupoMaximoLocal, cupoRestante, fotoLocal, estadoLocal, logoLocal);
  // Agregamos el objeto local al Array de locales
  listaLocales.push(unLocal);
}

// Funcion agregar personas precargadas
```

```
function agregarPersonaPrecargada(nombrePersona, nombreUsuarioPersona, contraseniaPersona) {
  let unaPersona = new Persona(nombrePersona, nombreUsuarioPersona, contraseniaPersona);
  // Agregamos el objeto persona al Array de personas
  listaPersonas.push(unaPersona);
}

// Funcion agregar reservas precargadas
function agregarReservaPrecargada(nombrePersona, nombreUsuarioPersona, nombreLocal, estado,
cantidadDeCuposAReservar, calificacion) {
  let unaReserva = new Reserva(nombrePersona, nombreUsuarioPersona, nombreLocal, estado,
cantidadDeCuposAReservar, calificacion);
  // Agregamos el objeto reservas al Array de reservas
  listaReservas.push(unaReserva);
}

//LOGIN LOCAL
//Funcion que levanta los datos del login de locales
function levantarDatosLoginLocales() {
  //Levanto datos del formulario de Locales
  let usernameLocal = document.querySelector("#txtNombreUsuario").value;
  let passwordLocal = document.querySelector("#txtContrasenia").value;

  //Paso la contraseña a minúscula para que coincida siempre me la ingrese en mayus o no
  passwordLocal = passwordLocal.toLowerCase();

  //Voy a la funcion que valida los datos
  validarDatosLoginLocal(usernameLocal, passwordLocal);
}

function validarDatosLoginLocal(username, password) {
  //Para cada local del array de locales
  for (let locales of listaLocales) {
    if (username == locales.nombreUsuario && password == locales.contrasenia) {
      //Guardo el objeto local que se logueó
      localLogueado = locales;
      //Oculto todo y muestro la pantalla principal --> Estado Local
      ocultarTodo();
      mostrarMenuLogOut();
      mostrarMenuLocales();
      mostrarBuscador();
      mostrarDivEstadoLocal();
    } else {
      document.querySelector("#pErroresLoginLocales").innerHTML = "El usuario y/o contraseña es
incorrecto";
    }
  }
}

//REGISTRO PERSONA
//Funcion que levanta los datos del registro de personas
function levantarDatosRegistroPersonas() {
  //Levanto datos del formulario de Registro Personas
  let namePersona = document.querySelector("#txtNombrePersona").value;
  let usernamePersona = document.querySelector("#txtNombreUsuarioPersonasRegistro").value;
```

```
  let passwordPersona = document.querySelector("#txtContraseniaPersonasRegistro").value;

  //Paso el usuario de la persona a minúsculas
  usernamePersona = usernamePersona.toLowerCase();

  let contraseniaNoEsValida = validarContraseniaPersona(passwordPersona);

  let datosPersonaCorrectos = validarDatosRegistroPersonas(usernamePersona, contraseniaNoEsValida,
passwordPersona);

  if (datosPersonaCorrectos) {
    agregarNuevaPersona(namePersona, usernamePersona, passwordPersona);
  }
}

//Función validar datos de la persona
function validarDatosRegistroPersonas(usernamePersona, contraseniaNoEsValida, contraseniaPersona) {
  let datosCorrectos = false;
  let existe = false;

  //Para cada persona del array de Personas
  for (let personas of listaPersonas) {
    //Paso a minúscula y comparo para ver si existe
    if (personas.nombreUsuario.toLowerCase() == usernamePersona.toLowerCase()) {
      existe = true;
      document.querySelector("#pErroresRegistroPersonas").innerHTML = "Ya está registrado";
      alert(existe);
    }
  }
  //Si no existe el usuario y la contraseña cumple con los requisitos, tira mensaje de registro correcto
  if (!existe && contraseniaNoEsValida == false) {
    limpiaColoresDeErrorContrasenia();
    document.querySelector("#pErroresRegistroPersonas").innerHTML = "Se registró correctamente";
    datosCorrectos = true;
    //Si existe el usuario y la contraseña no es válida, tira mensaje de error
  } else if (!existe && contraseniaNoEsValida) {
    mostrarErroresContrasenia(contraseniaPersona);
    document.querySelector("#pErroresRegistroPersonas").innerHTML = "La contraseña debe cumplir con los
requisitos";
  }

  return datosCorrectos;
}

// Funcion que valida la contraseña ingresada
function validarContraseniaPersona(contrasenia) {
  if (tieneNumeros(contrasenia) == true && tieneMayuscula(contrasenia) == true &&
tieneMinuscula(contrasenia) == true && cantidadDeCaracteresOK(contrasenia) == true) {
    return false;
  } else {
    return true;
  }
}
```

```
//Busco en la tabla ASCII si tiene numeros
function tieneNumeros(contrasenia) {
  for (i = 0; i < contrasenia.length; i++) {
    if (contrasenia.charCodeAt(i) >= 48 && contrasenia.charCodeAt(i) <= 57) {
      return true;
    }
  }
  return false;
}

//Busco en la tabla ASCII si tiene mayusculas
function tieneMayuscula(contrasenia) {
  for (i = 0; i < contrasenia.length; i++) {
    if (contrasenia.charCodeAt(i) >= 65 && contrasenia.charCodeAt(i) <= 90) {
      return true;
    }
  }
  return false;
}

//Busco en la tabla ASCII si tiene minusculas
function tieneMinuscula(contrasenia) {
  for (i = 0; i < contrasenia.length; i++) {
    if (contrasenia.charCodeAt(i) >= 97 && contrasenia.charCodeAt(i) <= 122) {
      return true;
    }
  }
  return false;
}

//Ver si tiene 6 caracteres o más
function cantidadDeCaracteresOK(contrasenia) {
  if (contrasenia.length >= 6) {
    return true;
  }
  return false;
}

// Funcion agregar personas nuevas (con el registro)
function agregarNuevaPersona(nombrePersona, nombreUsuarioPersona, contraseniaPersona) {
  let unaPersona = new Persona(nombrePersona, nombreUsuarioPersona, contraseniaPersona);
  // Agregamos el objeto persona al Array de personas
  listaPersonas.push(unaPersona);

  limpiaColoresDeErrorContrasenia();
}

// Funcion que muestra los errores (si es que los tiene) de la contraseña
function mostrarErroresContrasenia(contrasenia) {
  limpiaColoresDeErrorContrasenia();

  //Muestra los errores en rojo
  if (tieneNumeros(contrasenia) == false) {
    document.querySelector("#liUnNumero").style.color = "#ff0000";
```

```
  }
  if (tieneMayuscula(contrasenia) == false) {
    document.querySelector("#liUnaMayus").style.color = "#ff0000";
  }
  if (tieneMinuscula(contrasenia) == false) {
    document.querySelector("#liUnaMinus").style.color = "#ff0000";
  }
  if (cantidadDeCaracteresOK(contrasenia) == false) {
    document.querySelector("#liSeisCaracteres").style.color = "#ff0000";
  }
}

// Funcion que vuelve los colores de los errores de la contraseña al color original
function limpiaColoresDeErrorContrasenia() {
  let lista = document.querySelector("#listaRequisitos");
  let colorListaOriginal = lista.style.color;

  //Le saco el atributo style para luego darles el color que tenian
  document.querySelector("#liSeisCaracteres").removeAttribute("style");
  document.querySelector("#liUnNumero").removeAttribute("style");
  document.querySelector("#liUnaMayus").removeAttribute("style");
  document.querySelector("#liUnaMinus").removeAttribute("style");
  //Devolver el color original a la lista (que no quede en rojo)
  document.querySelector("#liSeisCaracteres").style.color = colorListaOriginal;
  document.querySelector("#liUnNumero").style.color = colorListaOriginal;
  document.querySelector("#liUnaMayus").style.color = colorListaOriginal;
  document.querySelector("#liUnaMinus").style.color = colorListaOriginal;
}

function levantarDatosLoginPersonas() {
  //Levanto datos del formulario de Personas
  let usernamePersona = document.querySelector("#nombreUsuarioPersonas").value;
  let passwordPersona = document.querySelector("#contraseniaPersonas").value;

  //Paso el usuario de la persona a minúsculas
  usernamePersona = usernamePersona.toLowerCase();

  //Voy a la funcion que valida los datos
  validarDatosLoginPersonas(usernamePersona, passwordPersona);
}

function validarDatosLoginPersonas(username, password) {
  //Para cada persona del array de personas
  for (let personas of listaPersonas) {
    if (username == personas.nombreUsuario && password == personas.contrasenia) {
      personaLogueada = personas;
      //Oculto todo y muestro la pantalla principal --> Reservas Personas
      ocultarTodo();
      mostrarMenuLogOut();
      mostrarMenuPersonas();
      mostrarDivReservasPersonasSolicitarReservas();
    } else {
      document.querySelector("#pErroresLoginPersonas").innerHTML = "El usuario y/o contraseña es
incorrecto";
```

```
    }
  }
}

//FUNCIONES DE LOCALES
//Estado del local
function mostrarTablaEstadoLocal() {
  let contadorReservasPorLocal = 0;
  //Cabezal de la tabla
  let miTabla = `<thead>
                <tr>
                <th>#</th>
                <th>Cliente</th>
                <th>Estado de la reserva</th>
                <th>Cambiar estado</th>
                </tr>
              <thead>`;

    //Hago un for común para poder guardarme la posición
    for (let pos=0; pos<listaReservas.length; pos++){
      let objReserva = listaReservas[pos];
      //si el local logueado es igual al nombre del local en la lista de locales, y el estado es pendiente
      //sumo 1 al contar de reservas por local
      if (localLogueado.nombre == objReserva.nombreLocal && objReserva.estado == "Pendiente") {
        contadorReservasPorLocal++;

        //armo el cuerpo de la tabla
        miTabla+=`<tbody>
                  <tr>
                  <td>${contadorReservasPorLocal}</td>
                  <td>${objReserva.nombreCliente}</td>
                  <td>${objReserva.estado}</td>
                  <td> <input type="button" value="Cambiar" class="redondear botonAccionCambiar" id="boton$
{pos}"> </td>
                  </tr>
                </tbody>`
      }

    }
    //muestro la tabla
    document.querySelector("#tablaEstadoLocal").innerHTML = miTabla;

    //TIRA EL ERROR SI NO HAY RESERVAS PENDIENTES
    if (contadorReservasPorLocal == 0) {
      document.querySelector("#tablaEstadoLocal").innerHTML = "";
      document.querySelector("#pMensajesEstadoLocal").innerHTML = "No hay reservas en estado pendiente
para mostrar";
    }

    //ASIGNACIÓN DE EVENTOS A TODOS LOS BOTONES QUE TENGAN LA CLASE botonAccionCambiar
    let listaInputs = document.getElementsByClassName("botonAccionCambiar");

    //cambia a estado finalizada la reserva en la que hago click en el botón cambiar
    for (let unInput of listaInputs){
```

```
    unInput.addEventListener("click", accionBotonCambiarEstado);
   }
}

//Función para cambiar el estado de una reserva a finalizada
function accionBotonCambiarEstado(){
  let cualFueElIdDelBoton = this.id; //me guardo el id del boton clickeado
  let posicion = Number(cualFueElIdDelBoton.substring(5)); //me quedo solo con el numero del texto y lo
paso a numero
  let objetoReserva = listaReservas[posicion]; //me traigo el objeto auto según posicion
  objetoReserva.estado="Finalizada";

  mostrarTablaEstadoLocal();
}

//FUNCION BUSCADOR DE RESERVAS - LOCAL
function buscarReserva(){
  // recojo el texto ingresado por el usuario
  let filtro=document.querySelector("#txtBuscador").value;

  // se limpia la tabla
  let miTabla="";

  // armo el cabezal de la tabla
  miTabla += `<thead>
            <th> # </th>
            <th> Cliente </th>
            <th> Estado de la reserva </th>
            <th> Cambiar estado </th>
           </thead>`;
  // recorro la lista de reservas
  for (let reservas of listaReservas){
     // paso el filtro y el nombre del cliente en el objeto a minuscula
     let nombreClienteMinuscula= reservas.nombreCliente.toLowerCase();
     let filtroMinuscula= filtro.toLowerCase();

     //si dentro del filtro esta incluido el nombre del cliente y ademas la reserva esta en estado pendiente
     //y ademas la reserva esta hecha en el local logueado entonces genero la tabla con el cliente encontrado
     if (nombreClienteMinuscula.includes(filtroMinuscula) && reservas.estado == "Pendiente" &&
reservas.nombreLocal == localLogueado.nombre){
       //limpio los mensajes y el input del buscador
       document.querySelector("#pMensajesEstadoLocal").innerHTML = "";
       document.querySelector("#txtBuscador").value = "";

       //cuerpo de la tabla
       miTabla+=  `<tbody>
                <tr>
                <td> ${reservas.contador}</td>
                <td> ${reservas.nombreCliente}</td>
                <td> ${reservas.estado}</td>
                <td> <input type="button" value="Cambiar" class="redondear" onclick=cambiarEstado($
{reservas.id})> </td>
                </tr>
               </tbody>`
```

```
    } //si dejo el buscador vacio que me tire todas las reservas de vuelta
    else if (filtroMinuscula == "" && reservas.estado == "Pendiente" && reservas.nombreLocal ==
localLogueado.nombre) {
      document.querySelector("#pMensajesEstadoLocal").innerHTML = "";
      document.querySelector("#txtBuscador").value = "";

      miTabla+= `<tbody>
                <tr>
                <td> ${reservas.contador}</td>
                <td> ${reservas.nombreCliente}</td>
                <td> ${reservas.estado}</td>
                <td> <input type="button" value="Cambiar" class="redondear" onclick=cambiarEstado($
{reservas.id})> </td>
                </tr>
               </tbody>`

      //document.querySelector("#pMensajesEstadoLocal").innerHTML = "No hay reservas en estado
pendiente para mostrar";
    }// y sino tiro un mensaje de error de que no se encontraron reservas pendientes con esa busqueda
    else {
      document.querySelector("#pMensajesEstadoLocal").innerHTML = "";
      document.querySelector("#txtBuscador").value = "";
      //document.querySelector("#pMensajesEstadoLocal").innerHTML = "No hay reservas en estado
pendiente para mostrar";
    }

  }

  // muestro la tabla
  document.querySelector("#tablaEstadoLocal").innerHTML=miTabla;
}

//FUNCION CAMBIAR ESTADO LOCAL
function cambiarEstado(unID){
  //recorro la lista de reservas y guardo el id
  for (let reservas of listaReservas){
    if (reservas.id == unID){
      //si el estado de la reserva es pendiente, lo cambio a finalizado
        if (reservas.estado =="Pendiente"){
          reservas.estado="Finalizado"
        }
    }

  }
  buscarReserva();
}

//FUNCION QUE MUESTRA EL PARRAFO CUPO LOCAL
//Cupo Maximo local
function mostrarFuncionalidadCupoMaximoLocal() {
  //recorro el array de locales
  for (let locales of listaLocales) {
    //si el nombre del local logueado matchea con uno de la lista de locales, muestra su cupo máximo actual
```

```
    if (localLogueado.nombre == locales.nombre) {
      document.querySelector("#pCupoMaximo").innerHTML = `El cupo máximo del local actualmente es de:
${locales.cupoMaximo}`;
    }
  }
}

//FUNCION QUE CAMBIA EL CUPO MAXIMO LOCAL
function cambiarCupoMaximo() {
  //Capturo datos de formulario
  let cupoMaximoIngresado = document.querySelector("#txtCupoMaximo").value;
  let existenReservasPendientes = false;

  //recorro el array de reservas
  for (let reservas of listaReservas) {
    //encuentra el nombre del local logueado en el array de reservas y busca si tiene reservas pendientes
    if (reservas.nombreLocal == localLogueado.nombre && reservas.estado == "Pendiente") {
      existenReservasPendientes = true;
    }
  }

  //recorro array de locales, si no existen reservas pendientes es posible modificar el
  //cupo maximo por el valor ingresado por el usuario
  for (let locales of listaLocales) {
    if (!existenReservasPendientes) {
      if (localLogueado.nombre == locales.nombre) {
        locales.cupoMaximo = cupoMaximoIngresado;
        document.querySelector("#pCupoMaximo").innerHTML = `El cupo máximo del local actualmente es de:
${locales.cupoMaximo}`;
      }
      // si el local tiene reservas pendientes no puede modificarlo y muestra un aviso
    } else {
      document.querySelector("#pCupoMaximo").innerHTML = `No se puede modificar porque hay reservas
pendientes`;
    }

  }
}

//FUNCION QUE MUESTRA EL PARRAFO DE DISPONIBILIDAD DEL LOCAL
//Disponibilidad de locales
function mostrarFuncionalidadDisponibilidad() {
  for (let locales of listaLocales) {
    if (locales.id == localLogueado.id) {
      document.querySelector("#pDisponibilidad").innerHTML = `La disponibilidad de reserva del local está: $
{locales.estado}`;
    }
  }
}

//FUNCION QUE DESHABILITA LA DISPONIBILIDAD DEL LOCAL
function deshabilitarDisponibilidad() {
  let disponibilidad = "Deshabilitado";
  for (let locales of listaLocales) {
```

```
    if (locales.id == localLogueado.id) {
      locales.estado = disponibilidad;
      document.querySelector("#pDisponibilidad").innerHTML = `La disponibilidad de reserva del local está: $
{locales.estado}`;
    }
  }
}

//FUNCION QUE HABILITA LA DISPONIBILIDAD DEL LOCAL
function habilitarDisponibilidad() {
  let disponibilidad = "Habilitado";
  for (let locales of listaLocales) {
    if (locales.id == localLogueado.id) {
      locales.estado = disponibilidad;
      document.querySelector("#pDisponibilidad").innerHTML = `La disponibilidad de reserva del local está: $
{locales.estado}`;
    }
  }
}

// TABLA ESTADISTICA LOCALES - INFORMACION
function mostrarTablaEstadisticaLocalesInformacion() {
  let miTabla = "";

  //armo el cabezal
  miTabla += `<thead>
              <th>Reservas pendientes</th>
              <th>Reservas finalizadas</th>
              <th>Total de reservas</th>
              <th>Porcentaje de ocupación</th>
            </thead>`;

  //defino contadores para las reservas pendientes, reservas finalizadas, y el total de reservas (para hacer el
porcentaje)
  let reservasPendientes = 0;
  let reservasFinalizadas = 0;
  let totalReservas = 0;
  let porcentajeDeOcupacion = 0;

  //recorremos las reservas
  for (let reservas of listaReservas) {
    //si la reserva esta en estado pendiente y ademas es del local logueado
    if(reservas.estado == "Pendiente" && reservas.nombreLocal == localLogueado.nombre ){
      //sumamos una al contador de reservas pendientes
      reservasPendientes++
    }
    //si la reserva esta en estado finalizada y ademas es del local logueado
    if(reservas.estado == "Finalizada" && reservas.nombreLocal == localLogueado.nombre ){
      //sumamos una al contador de reservas finalizadas
      reservasFinalizadas++
    }
    //sumamos las pendientes y las finalizadas para obtener el total de reservas
    totalReservas = reservasFinalizadas + reservasPendientes;
```

```
  //sacamos el porcentaje de ocupacion del local en base al total de reservas
  porcentajeDeOcupacion = Math.round((100*reservasPendientes)/localLogueado.cupoMaximo);
 }

 //armo el cuerpo de la tabla
 miTabla += `<tbody>
            <tr>
            <td>${reservasPendientes}</td>
            <td>${reservasFinalizadas}</td>
            <td>${totalReservas}</td>
            <td>${porcentajeDeOcupacion}%</td>
            </tr>
           <tbody>`

 //muestro la tabla
 document.querySelector("#tablaInformacionEstadisticaLocal").innerHTML = miTabla;
}

// TABLA ESTADISTICA LOCALES - LISTADO DE LOCALES
function mostrarTablaEstadisticaLocalListadoDeLocales() {
 let miTabla = "";

 //armo el cabezal
 miTabla += `<thead>
            <th>Nombre</th>
            <th>Promedio de calificaciones</th>
           </thead>`;

 //armo el cuerpo de la tabla
 //recorro array de locales
 for (let locales of listaLocales) {
  /* defino contador para la cantidad de calificaciones, una variable para calcular la calificacion promedio,
  y la calificacion total (para hacer el promedio) */
  let calificacionTotal = 0;
  let cantCalificaciones = 0;
  let calificacionPromedio = 0;

  /* dentro del for que recorre los locales recorro array de reservas
  para que por cada local vaya recorriendo todas las reservas */
  for (let reservas of listaReservas) {
   /* si el nombre del local guardado en reservas coincide con el local siendo recorrido y ademas la reserva
   esta en estado finalizado */

   // si el nombre del local coincide con el nombre de un local reservado y el estado es finalizada
   //guardo la calificacion y sumo 1 al contador de cantidad de calificaciones
   if (reservas.nombreLocal == locales.nombre && reservas.estado == "Finalizada") {
    console.log(reservas.estado)
    // la calificacion total es igual a la calificacion total + la calificacion de la reserva
    calificacionTotal = calificacionTotal+reservas.calificacion;
    //sumo 1 al contador de calificaciones
    cantCalificaciones++;
   }
  }
  //si la cantidad de calificaciones es mayor a 0 y ademas la calificacion total es distinta de 0
```

```
    if (cantCalificaciones>0 && calificacionTotal!=0 ){
      //si la cantidad de calificaciones es >0 y la sumatoria de calificaciones no es cero
      //calculo el promedio y lo redondeo

      /* calculo el promedio haciendo la calificacion total, dividido la cantidad de calificaciones
      ademas le dejo un numero despues de la coma */
      calificacionPromedio = (calificacionTotal/cantCalificaciones).toFixed(1);
    }

    //armo el cuerpo de la tabla con el nombre del local y el promedio de calificación
    miTabla += `<tbody>
                <td>${locales.nombre}</td>
                <td>${calificacionPromedio}</td>
               </tbody>`
    }

    //muestro la tabla
    document.querySelector("#tablaListadoDeLocales").innerHTML = miTabla;
}

//Funcion que checkea si el local ya existe en el array auxiliar, y si no existe lo agrega
function estaEnAux (local,arrayAux){
  for (let loc of arrayAux){
    if (loc == local){
      return true
    }
  }
  arrayAux.push(local)
  return false
}

//Función para contar la cantidad de reservas que tiene el usuario logueado en un local
function contarReservasDeUsarioLocal(unLocal){
  let contador = 0;
  for (let reservas of listaReservas){
    if(reservas.nombreLocal == unLocal && reservas.usuarioCliente == personaLogueada.nombreUsuario){
      contador++
    }
  }
  return contador
}

//Función para contar el total de reservas que tiene un local
function contadorTotalResrevasEnUnLocal(unLocal){
  let contador = 0;
  for (let reservas of listaReservas){
    if(reservas.nombreLocal == unLocal){
      contador++
    }
  }
  return contador
}

//Función agregar nueva reserva("nombre","nombre de usuario", "nombre del local", "estado", cupo);
```

```
function nuevaReserva() {
  //Asignamos el valor del select (el id de los locales) a la variable localId
  let localId = document.querySelector("#slcSolicitarReservaPersonas").value;
  //Tomamos la cantidad de cupos ingresada por el usuario
  let cupos = document.querySelector("#cuposAReservarPersonas").value;
  let calificacion = 0;

  //Recorro la lista de locales para ver encontrar el local seleccionado
  let localAReservar;
  for (let locales of listaLocales){
    if (locales.id == localId){
      localAReservar = locales;
    }
  }

  //Se verifica si existen reservas pendientes de la persona logueada en ese local
  let existenReservas = false;
  for (let reservas of listaReservas){
    if (reservas.estado == "Pendiente" && reservas.nombreLocal == localAReservar.nombre &&
reservas.nombreCliente == personaLogueada.nombre){
      console.log(reservas.estado)
      console.log(reservas.nombreLocal)
      console.log(localAReservar.nombre)
      existenReservas = true;
    }
  }

  //Se verifica si es posible reservar la cantidad de cupos que el usuario ingresó
  let hayCuposDisponibles = true;
  if (localAReservar.cuposRestantes < cupos){
    hayCuposDisponibles = false;
  }

  //Se verifica si el local está habilitado a recibir reservas
  let localHabilitado = true;
  if (localAReservar.estado == "Deshabilitado") {
    localHabilitado = false;
  }

  // Se verifica que no se deje vacío el campo de cupos
  if (cupos == ""){
    document.querySelector("#pErroresReservasPersonas").innerHTML = "Falta ingresar datos";
    cupos = parseInt(cupos);

   //Se avisa a la persona si tiene reservas pendientes
  } else if (existenReservas == true){
    document.querySelector("#pErroresReservasPersonas").innerHTML = "Ya hay reservas pendientes en ese
local";

   //Se avisa si no hay cupos disponibles
  } else if (hayCuposDisponibles == false){
    document.querySelector("#pErroresReservasPersonas").innerHTML = "No hay cupos disponibles en ese
local";
```

```
   //Se avisa si el local está deshabilitado
  } else if (localHabilitado == false) {
    document.querySelector("#pErroresReservasPersonas").innerHTML = "El local está deshabilitado";

  //Si está todo bien, se crea reserva y se restan los cupos solicitados por el usuario de los cupos disponibles
del local
  } else {
    localAReservar.cuposRestantes = localAReservar.cuposRestantes - cupos;
    //crear reserva
    let unaReserva = new Reserva(personaLogueada.nombre, personaLogueada.nombreUsuario,
localAReservar.nombre, "Pendiente", cupos, calificacion);
    //pushear, agregar la reserva al array
    listaReservas.push(unaReserva);
    document.querySelector("#pErroresReservasPersonas").innerHTML = "La reserva se ingresó
correctamente";
    if (localAReservar.cuposRestantes == 0){
      localAReservar.estado = "Deshabilitado";
    }
  }

  console.log(listaReservas);

}

// Funcion agregar reservas
function agregarReserva(nombrePersona, nombreUsuarioPersona, nombreLocal, estado,
cantidadDeCuposAReservar, calificacion) {
  //Creo la variable para que me muestre true si puedo agregar la reserva y false si no puedo
  let agregoReserva;
  //Llamamos a la funcion que checkea si el usuario ya tiene una reserva pendiente en ese local
  let yaExisteReserva = existeReserva(nombreUsuarioPersona, nombreLocal);
  if (yaExisteReserva) {
    agregoReserva = false; // No la puedo agregar porque ya existe
    document.querySelector("#pErroresReservasPersonas").innerHTML = "Ya tiene una reserva pendiente en
este local."
  }
  else {
    console.log("tercero: "+nombreUsuarioPersona);
    let unaReserva = new Reserva(nombrePersona, nombreUsuarioPersona, nombreLocal, estado,
cantidadDeCuposAReservar, calificacion);
    // Agregamos la reserva al array de reservas
    listaReservas.push(unaReserva);
    //Se avisa que se ingresó correctamente la reserva
    document.querySelector("#pErroresReservasPersonas").innerHTML = `Se agregó la reserva
correctamente en el local ${nombreLocal} con ${cantidadDeCuposAReservar} cupos`

    agregoReserva = true;
  }

  return agregoReserva;
}

// Funcion que checkea si el usuario ya tiene una reserva pendiente en ese local
function existeReserva(nombreUsuarioPersona, nombreDelLocal) {
```

```
  let existe = false;
  for (let i = 0; i < listaReservas.length && !existe; i++) {
    let objReserva = listaReservas[i];
    if (objReserva.usuarioCliente.toLowerCase() == nombreUsuarioPersona.toLowerCase() &&
objReserva.nombreLocal.toLowerCase() == nombreDelLocal.toLowerCase()) {
      existe = true;
    }
  }
  return existe;
}

// TABLA ESTADISTICA PERSONAS - LOCAL CON MÁS RESERVAS DEL CLIENTE
function mostrarTablaEstadisticaPersonasLocalConMasReservasDelCliente() {
  let miTabla = "";

  //armo el cabezal
  miTabla = `<thead>
            <th>Local</th>
            <th>Total de reservas</th>
          </thead>`;

  //armo el cuerpo de la tabla
  let localFavorito = new Array();
  let nombreLocalCalculado = new Array();
  let misReservas = 0;

  //Recorro la lista de reservas y la lista de locales
  for (let reservas of listaReservas) {
    let localYaCalculado = false;
    for (let calculados of nombreLocalCalculado) {
      if (calculados.nombre == reservas.nombreLocal) {
        localYaCalculado = true;
      }
    }

    if (localYaCalculado == false) {

      for (let i=0;i<listaReservas.length;i++) {
        if (reservas.nombreLocal == listaReservas[i].nombreLocal && reservas.usuarioCliente ==
personaLogueada.nombreUsuario && listaReservas[i].usuarioCliente == personaLogueada.nombreUsuario)
{
          misReservas++;
        }
      }
      let datos = new LocalFavorito(reservas.nombreLocal,misReservas);
      nombreLocalCalculado.push(datos);
      misReservas = 0;
    }
  }

  for (let calculados of nombreLocalCalculado) {

    if (localFavorito.length == 0) {
      let datos2 = new LocalFavorito (calculados.nombre,calculados.cantidadDeReservas);
```

```
      localFavorito.push(datos2);

    } else if (calculados.cantidadDeReservas == localFavorito[0].cantidadDeReservas) {
      let datos2 = new LocalFavorito (calculados.nombre,calculados.cantidadDeReservas);
      localFavorito.push(datos2);

    } else if (calculados.cantidadDeReservas > localFavorito[0].cantidadDeReservas) {

      while (localFavorito.length) {
        localFavorito.pop();
      }

      let datos2 = new LocalFavorito (calculados.nombre,calculados.cantidadDeReservas);
      localFavorito.push(datos2);

    }

  }

  while (nombreLocalCalculado.length) {
    nombreLocalCalculado.pop();
  }

  console.log("-----------------------------");
  console.log(localFavorito);
  console.log("-----------------------------");

  document.querySelector("#tablaLocalConMasReservasPersonas").innerHTML = miTabla;


    for(let favoritos of localFavorito){

      document.querySelector("#tablaLocalConMasReservasPersonas").innerHTML += `<tr><td>$
{favoritos.nombre}</td><td>${favoritos.cantidadDeReservas}</td></tr>`

    }

}

//SELECT SOLICITAR RESERVA PERSONAS
function cargarSelectSolicitarReservaPersonas() {
  let combo = document.querySelector("#slcSolicitarReservaPersonas");
  combo.innerHTML="";
  for (let locales of listaLocales) {
    combo.innerHTML += `<option value=${locales.id}>
                        ${locales.id} ${locales.nombre}
                      </option>`

  }
}

//TABLA RESERVAS PENDIENTES PERSONAS
function mostrarTablaReservasPendientesPersonas() {
  //armo el cabezal
```

```
 let miTabla = `<thead>
            <tr>
            <th>#</th>
            <th>Nombre</th>
            <th>Cupos Reservados</th>
            <th>Foto</th>
            <th>Estado</th>
            <th>Cancelar reserva</th>
            </tr>
           </thead>`;

 //armo el cuerpo de la tabla
 //Recorro el la lista de reservas y me guardo la posición
 for (let i = 0; i < listaReservas.length; i++) {
   let objetoReserva = listaReservas[i];
   //Recorro el array de lcoales
   for (let locales of listaLocales) {
     //Guardo la ruta de la foto del local
     if (locales.nombre == objetoReserva.nombreLocal) {
       let rutaFoto = locales.foto;
       //Si la reserva en la posición i es de la persona logueada y está en estado pendiente o cancelada, la
muestro en la tabla junto con la info del local
       if (objetoReserva.usuarioCliente == personaLogueada.nombreUsuario && objetoReserva.estado ==
"Pendiente" || objetoReserva.estado == "Cancelada") {
        miTabla += `<tbody>
                <tr>
                <td>${objetoReserva.contador}</td>
                <td>${objetoReserva.nombreLocal}</td>
                <td>${objetoReserva.cantidadDeCuposAReservar}</td>
                <td><img class="imgLocales" src="${rutaFoto}"</td>
                <td>${objetoReserva.estado}</td>
                <td><input type ="button" value ="Cancelar" class="botonAccionCancelar redondear"
id="boton${i}"></td>
                </tr>
               </tbody>`
       }
     }

   }

 }
 document.querySelector("#tablaReservasPendientesPersonas").innerHTML = miTabla;

 //ASIGNACIÓN DE EVENTOS A TODOS LOS BOTONES QUE TENGAN LA CLASE botonAccion
 let arrayBotones = document.getElementsByClassName("botonAccionCancelar");
//getElementsByClassName devuelve un array x eso lo guardo en una variable arrayBotones

 //Para cada boton del array de botones le asigno un evento de click que llama a la funcion accionBoton
 for (let unBoton of arrayBotones) {
   unBoton.addEventListener("click", accionBotonReservasPendientes);
 }
}

//Funcion acción boton que guarda el id del boton clickeado y cambia el estado de esa posición
```

```
function accionBotonReservasPendientes() {
  let cualFueElIdDelBoton = this.id; //me guardo el id del botón clickeado
  //let id = cualFueElBoton.id; //guardo el id del boton clickeado (boton0, boton1..)
  let posicion = Number(cualFueElIdDelBoton.substring(5)); //me quedo solo con el numero del texto y lo
paso a numero
  let objetoReserva = listaReservas[posicion]; //me traigo el objeto según posicion
  if (objetoReserva.estado == "Pendiente") {
    objetoReserva.estado = "Cancelada";
    console.log(listaReservas);
  }
  mostrarTablaReservasPendientesPersonas();
}

//TABLA RESERVAS FINALIZADAS PERSONAS
function mostrarTablaReservasFinalizadasPersonas() {
  //armo el cabezal
  miTabla = `<thead>
            <tr>
            <th>#</th>
            <th>Nombre local</th>
            <th>Cliente</th>
            <th>Cupos reservados</th>
            <th>Estado reserva</th>
            <th>Calificación</th>
            <tr>
          </thead>`;

  //armo el cuerpo de la tabla
  //Recorro la lista de reservas y busco las reservas finalizadas de la persona logueada para mostrar en la
tabla
  for (let unaReserva of listaReservas) {
    if (unaReserva.usuarioCliente == personaLogueada.nombreUsuario && unaReserva.estado ==
"Finalizada") {
      miTabla += `<tbody>
                <tr>
                  <td>${unaReserva.contador}</td>
                  <td>${unaReserva.nombreLocal}</td>
                  <td>${unaReserva.usuarioCliente}</td>
                  <td>${unaReserva.cantidadDeCuposAReservar}</td>
                  <td>${unaReserva.estado}</td>
                  <td><select id="slcCalificacion${unaReserva.id}">
                       <option value=1>Elija calificación</option>
                       <option value=1>1</option>
                       <option value=2>2</option>
                       <option value=3>3</option>
                       <option value=4>4</option>
                       <option value=5>5</option>
                     </select>
                  </td>
                  <td><input type="button" class="redondear" value="Calificar" onClick=calificarReserva($
{unaReserva.id})></td>
                </tr>
              </tbody>`;
    }
```

```
  }

  document.querySelector("#tablaReservasFinalizadasPersonas").innerHTML = miTabla;
}

//Funcion para calificar
function calificarReserva(unaReservaID) {
  //Tomo el valor de la calificación seleccionado por la persona
  let calificacion = parseInt(document.querySelector(`#slcCalificacion${unaReservaID}`).value);

  //Se toma el valor de la calificación si la reserva está finalizada
  for (let reserva of listaReservas) {
    if (reserva.id == unaReservaID && reserva.estado == "Finalizada") {
      reserva.calificacion = calificacion;
      //Se avisa que se ingresó correctamente la calificación
      document.querySelector("#pErrorCalificacion").innerHTML = "La calificación se ingresó correctamente";
    }


  }
  console.log(listaReservas);
}


// TABLA ESTADISTICA PERSONAS - PORCENTAJE DE RESERVAS
//Defino un array auxiliar
let arrayAuxMostrado= new Array()
function mostrarTablaEstadisticaPersonasPorcentajeDeReservas() {
  //Miestras el array tenga elementos, elimine uno por uno
  while(arrayAuxMostrado.length){
    arrayAuxMostrado.pop();
    }
  let miTabla = "";
  //armo el cabezal
  miTabla += `<th>Local</th>
            <th>Cantidad de reservas completadas del cliente</th>
            <th>Total de reservas del local</th>
            <th>Porcentaje de reservas del cliente</th>
            `;

  //armo el cuerpo de la tabla
  let local = "";
  let promedio = 0;
  console.log(arrayAuxMostrado);
  /* Recorro el array de reservas si la reserva está finalizada y es del usuario logueado,
  guardo en una variale el nombre del local en el que la reserva está finalizada */
  for (let unaReserva of listaReservas) {
    console.log(unaReserva)
    if (unaReserva.estado == "Finalizada" && unaReserva.usuarioCliente ==
personaLogueada.nombreUsuario){
      local = unaReserva.nombreLocal
      //Con la función estaEnAux, compruebo si el local no está en el array auxiliar,
      //para contar la cantidad de reservas que tiene el usuario logueado
      if (!estaEnAux(local,arrayAuxMostrado)){
```

```
      console.log("Entro bien x2")
      let misResevas= contarReservasDeUsarioLocal(local)
      let totalReservas = contadorTotalResrevasEnUnLocal(local)
      //Si el total de reservas es mayor a cero, calculo el promedio, y lo redondeo
      if (totalReservas >0){
        console.log("Entro bien x3")
        promedio = Math.round((misResevas / totalReservas) * 100)
        miTabla += `<tr>
                <td>${local}</td>
                <td>${misResevas}</td>
                <td>${totalReservas}</td>
                <td>${promedio}%</td>
                </tr>`
      }

     }

    }

  }
  //Muestro la tabla
  document.querySelector("#tablaPorcentajeReservasPersonas").innerHTML = miTabla;
}
```

```html
<!DOCTYPE html>
<html lang="es">

<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">
    <!--Estilos-->
    <link href="css/estilos.css" rel="stylesheet" type="text/css" />

    <!-- El favicon -->
    <link rel="icon" type="image/x-icon" href="img/favicon.png">

    <!-- Codigo JS -->
    <script type="text/javascript" src="js/codigo.js"></script>
    <title>Reservas</title>
</head>

<body>
    <!-- Encabezado -->
    <div id="fijadorMenuPrincipal">
        <header>
            <nav class="" id="menuPrincipal">
                <a href="index.html"><img src="img/Reservas.svg" alt="Logo" /></a>
                <div id="divMenuLogueoYRegistro">
                    <ul>
                        <li class="redondear" id="locales"><a href="#">Locales</a></li>
                        <li class="redondear" id="personas"><a href="#">Personas</a></li>
                    </ul>
```

```
            </div>
            <div id="divMenuCerrarSesion">
              <ul>
                <li id="imgLogOut"><a href="#" id="cerrarSesion"><img src="img/logout.svg" alt="Cerrar
Sesión"></a></li>
              </ul>
            </div>


          </nav>
        <!-- Para arreglar flotacion CSS -->
        <div class="clearfix"></div>
        </header>
      </div>


      <!-- Menu Lateral -->
      <!-- Solo se muestra cuando inicio sesion -->
      <div class="sidebar">
        <nav>
          <!-- Solo se muestra en locales -->
          <ul class="menuLateral" id="menuLateralLocales">
            <li id="logoLocal"></li>
            <li class="redondear" id="menuEstadoLocales"><a href="#">Estado</a></li>
            <li class="redondear" id="menuCupoMaximoLocales"><a href="#">Cupo Máximo</a></li>
            <li class="redondear" id="menuDisponibilidadLocales"><a href="#">Disponibilidad</a></li>
            <li class="redondear" id="menuEstadisticaLocales"><a href="#">Estadística</a></li>
          </ul>
          <!-- Solo se muestra en personas -->
          <ul class="menuLateral" id="menuLateralPersonas">
            <li id="logoPersona"></li>
```

```
            <span id="nombrePersona"></span>
            <li class="redondear" id="menuReservaPersonas"><a href="#">Reservas</a></li>
            <li class="redondear" id="menuEstadisticaPersonas"><a href="#">Estadística</a></li>
        </ul>
    </nav>
    <!-- Para arreglar flotacion CSS -->
    <div class="clearfix"></div>
</div>

<!-- LOCALES -->
<!-- Buscador que solo se muestra en estado de local -->
<div id="buscador" class="fijadorMenusArriba">
    <input type="text" id="txtBuscador" class="redondear" placeholder="Buscar reservas">
    <input type="button" id="btnBuscar" class="redondear" value="Buscar">
</div>

<!-- Cuerpo -->
<section class="containerLogIn">
    <!-- Solo se muestra al iniciar sesion de locales -->
    <div id="divInicioSesionLocales">
        <h1 class="textoCentrado" id="inicioSesion">Iniciar Sesión</h1>
        <article class="login">
            <label for="txtNombreUsuario">Nombre de usuario</label>
            <input type="text" id="txtNombreUsuario" class="redondear">
            <label for="txtContrasenia">Contraseña</label>
            <input type="text" id="txtContrasenia" class="redondear">
            <input type="button" id="btnEntrarLocales" class="redondear" value="Entrar">
            <p class="error" id="pErroresLoginLocales"></p>
            <!-- <p class="error" id="pErrorUsuario"></p>
```

```
        <p class="error" id="pErrorContraseña"></p> -->
    </article>
</div>
<!-- Solo se muestra al registrar personas -->
<div id="divRegistroPersonas">
    <h1 class="textoCentrado" id="registro">Registro</h1>
    <article id="registro">
        <label for="txtNombrePersona">Nombre(*)</label>
        <input type="text" id="txtNombrePersona" class="redondear">
        <label for="txtNombreUsuarioPersonasRegistro">Nombre de usuario(*)</label>
        <input type="text" id="txtNombreUsuarioPersonasRegistro" class="redondear">
        <label for="txtContraseniaPersonasRegistro">Contraseña(*)</label>
        <input type="password" id="txtContraseniaPersonasRegistro" class="redondear">
        <p>La contraseña debe contener:</p>
        <ul id="listaRequisitos">
            <li id="liSeisCaracteres">Mínimo 6 caracteres.</li>
            <li id="liUnaMayus">Una mayúscula.</li>
            <li id="liUnaMinus">Una minúscula.</li>
            <li id="liUnNumero">Un número.</li>
        </ul>
        <span id="camposObligatorios">(*)Campos obligatorios</span>
        <input type="button" id="btnRegistrar" class="redondear" value="Registrarse">
        <p class="error" id="pErroresRegistroPersonas"></p>
        <span id="cuentaUsuarioInicioSesion">¿Ya tiene cuenta? <a href="#">Inicie Sesión</a></span>
    </article>
</div>

<!-- Solo se muestra al iniciar sesion personas -->
<div id="divInicioSesionPersonas">
```

```
        <h1 class="textoCentrado" id="inicioSesion">Iniciar Sesión</h1>
        <article class="login">
            <label for="nombreUsuarioPersonas">Nombre de usuario</label>
            <input type="text" id="nombreUsuarioPersonas" class="redondear">
            <label for="contraseniaPersonas">Contraseña</label>
            <input type="text" id="contraseniaPersonas" class="redondear">
            <input type="button" id="btnEntrarPersonas" class="redondear" value="Entrar">
            <p class="error" id="pErroresLoginPersonas"></p>
            <span id="cuentaUsuarioRegistro">¿No tiene cuenta? <a href="#">Regístrese</a></span>
        </article>
    </div>
  </section>

  <section class="container">
    <!-- Estado Local -->
    <article class="contenidoDeSeccionesConMenuArriba" id="divEstadoLocal">
        <table id="tablaEstadoLocal"></table>
        <p id="pMensajesEstadoLocal"></p>
    </article>

    <!-- Cupo Maximo -->
    <article id="divCupoMaximoLocal">
        <p id="pCupoMaximo"></p>
        <input type="number" id="txtCupoMaximo" class="redondear" placeholder="Ingrese el cupo
máximo">
        <input type="button" id="btnModificar" class="redondear" value="Modificar">
        <span>Aviso: <br>
            Recuerde que para cambiar el cupo no debe tener reservas en estado pendiente. <br>
            Puede checkear las reservas en la pestaña Estado.</span>
```

```
    </article>

    <!-- Disponibilidad -->
    <article id="divDisponibilidadLocal">
        <p id="pDisponibilidad">---</p>
        <input type="button" id="btnHabilitar" class="redondear" value="Habilitar">
        <input type="button" id="btnDeshabilitar" class="redondear" value="Deshabilitar">
        <br>
        <span>Aviso: <br>
            Recuerde que se deshabilitará automáticamente cuando la cantidad <br>
            de reservas totales pendientes coincida con el cupo del local. <br>
            Para que esto no suceda, contemple la posibilidad de aumentar el <br>
            cupo de la sección Cupo máximo.
        </span>
    </article>
</section>

<!-- Estadística Locales -->
<!-- Menu -->
<div class="fijadorMenusArriba">
    <nav class="menuArriba" id="menuArribaEstadisticaLocales">
        <div>
            <ul>
                <li class="redondear" id="informacion"><a href="#">Información</a></li>
                <li class="redondear" id="listadoDeLocales"><a href="#">Listado de locales</a></li>
            </ul>
        </div>
    </nav>
</div>
```

```
<!-- Cuerpo -->
<section class="container">
  <article class="contenidoDeSeccionesConMenuArriba" id="divEstadisticaLocal">
    <!-- Informacion -->
    <div id="informacionEstadisticaLocal">
      <table id="tablaInformacionEstadisticaLocal"></table>
    </div>
    <!-- Listado de locales -->
    <div id="listadoDeLocalesEstadisticaLocal">
      <table id="tablaListadoDeLocales"></table>
    </div>
  </article>
</section>

<!-- PERSONAS -->
<!-- Menu -->
<div class="fijadorMenusArriba">
  <nav class="menuArriba" id="menuArribaReservasPersonas">
    <div>
      <ul>
        <li class="redondear" id="solicitarReservas"><a href="#">Solicitar reservas</a></li>
        <li class="redondear" id="reservasPendientes"><a href="#">Reservas pendientes</a></li>
        <li class="redondear" id="reservasFinalizadas"><a href="#">Reservas finalizadas</a></li>
      </ul>
    </div>
  </nav>
</div>
```

```
<!-- Reserva Personas -->
<!-- Cuerpo -->
<section class="container">
  <article class="contenidoDeSeccionesConMenuArriba" id="divReservasPersonas">
    <!-- Solicitar Reservas -->
    <div id="solicitarReservasPersonas">
      <select name="selectReservasPersonas" id="slcSolicitarReservaPersonas"></select>
      <p id="pInfoLocalSeleccionado"></p>
      <label for="cuposAReservarPersonas">Cupos a reservar</label>
      <input type="text" id="cuposAReservarPersonas" class="redondear">
      <input type="button" id="btnSolicitarReserva" class="redondear" value="Solicitar">
      <p class="error" id="pErroresReservasPersonas"></p>
    </div>
    <!-- Reservas Pendientes -->
    <div id="reservasPendientesPersonas">
      <table id="tablaReservasPendientesPersonas"></table>
    </div>
    <!-- Reservas Finalizadas -->
    <div id="reservasFinalizadasPersonas">
      <table id="tablaReservasFinalizadasPersonas"></table>
      <p id="pErrorCalificacion"></p>
    </div>
  </article>
</section>

<!-- Estadística Personas -->
<!-- Menu -->
<div class="fijadorMenusArriba">
  <nav class="menuArriba" id="menuArribaEstadisticasPersonas">
```

```
        <div>
            <ul>
                <li class="redondear" id="porcentajeReservas"><a href="#">Porcentaje de reservas</a></li>
                <li class="redondear" id="localConMasReservasCliente"><a href="#">Local con más reservas
del cliente</a></li>
            </ul>
        </div>
    </nav>
  </div>

  <!-- Cuerpo -->
  <section class="container">
    <article class="contenidoDeSeccionesConMenuArriba" id="divEstadisticaPersonas">
        <!-- Porcentaje de reservas -->
        <div id="porcentajeDeReservasPersonas">
            <table id="tablaPorcentajeReservasPersonas"></table>
        </div>
        <!-- Local con más reservas del cliente -->
        <div id="localConMasReservasPersonas">
            <table id="tablaLocalConMasReservasPersonas"></table>
        </div>
    </article>
  </section>

</body>

</html>
```

```
* {
   margin: 0;
   padding: 0;
}

.container {
   max-width: 1155px;
   /* margin: 0 auto; */
   /* margin-top: 90px;
   margin-left: 285px;
   margin-right: 100px; */
   margin: 190px auto 0 435px;
}

.containerLogIn {
   margin-top: 140px;
}

.contenidoDeSeccionesConMenuArriba {
   margin-top: 190px;
}

.segundoContainer {
   width: 600px;
   margin: auto;
}

.tercerContainer {
   margin: 10px 255px;
}

.clearfix {
   clear: both;
}

.textoCentrado {
   /* margin: 0 auto; */
   text-align: center;
}

.redondear {
   border-radius: 5px;
   border-width: 1px;
}

.activo {
   background-color: #00bbc9;
}

.activo2 {
   background-color: #202022;
}

header {
```

```
    padding: 0 20px;
}
header, div.sidebar {
    /* height: 50px;  */

    background-color: #00747c;
}

header nav a img {
    /* display: block; */
    float: left;
    width: 180px;
    /* margin: 0 auto; */
    padding: 10px 0;
}

header nav ul {
    float: right;
}

nav ul li {
    list-style-type: none;
    padding: 25px 30px;
    transition: background-color .30s ease-in;
}

nav ul li.redondear {
    display: inline-block;
}

nav ul li:hover {
    cursor: pointer;
    background-color: rgba(0, 0, 0, 0.5);
}

nav ul li a {
    width: 100%;
    display: block;
    text-decoration: none;
    font-weight: 700;
    color: white;
}

nav.menuArriba ul li a {
    color: #202022;
}

/* nav.menuArriba ul li:hover {
    background-color: rgba(0, 0, 0, 0.5);
} */

nav.menuArriba ul li:hover a {
    color: white;
}
```

```
nav.menuArriba ul li.activo2 a{
   color: white;
}

input[type=button] {
   font-weight: 400;
   font-size: 14px;
}

header nav ul li a#cerrarSesion {
   padding: 0;
}

header nav ul li a#cerrarSesion img {
   width: 40px;
}

div.sidebar {
   width: 220px;
   height: 100%;
   position: fixed;
   top: 63px;
   left: 0;
   z-index: 1;
   overflow-x: hidden;
}

div.sidebar nav ul.menuLateral {
   /* margin-top: 125px; */
   text-align: center;
}

div.sidebar nav ul.menuLateral li {
   width: 100%;
   padding: 25px 0;
   font-size: 20px;
}

div.sidebar nav ul.menuLateral li a {
   text-align: center;
   display: inline;
   /* padding: 50px 0; */
}

body {
   font-family: 'Segoe UI', sans-serif;
}

h1 {
   font-weight: 700;
   font-size: 22px;
}
```

```
h1#inicioSesion {
   margin: 60px 0 25px 0;
}

h1#registro {
   margin: 30px 0 20px 0;
}

span {
   display: block;
}

select {
   display: block;
   width: 100%;
   padding-top: 0.5rem;
   padding-bottom: 0.5rem;
   padding-left: 1rem;
   margin-bottom: 5px;
   font-size: 1rem;
   font-weight: 400;
   line-height: 1.5;
   color: #212529;
   background-color: #fff;
   background-image: url("../img/dropdownmenu.svg");
   background-repeat: no-repeat;
   background-position: right 0.75rem center;
   background-size: 16px 12px;
   border: 1px solid #ced4da;
   border-radius: 0.25rem;
   transition: border-color .15s ease-in-out,box-shadow .15s ease-in-out;
   appearance: none;
   word-wrap: normal;
   text-transform: none;
}

select::after, select::before {
   box-sizing: border-box;
}

select option::before, select option::after {
   box-sizing: border-box;
}

section article.login, section article#registro {
   width: 400px;
   margin: 0 auto;
}

section article.login label, section article#registro label {
   display: block;
   font-size: 16px;
   font-weight: 500;
   margin: 15px 0;
```

```
}

section article.login input, section article#registro input, div#buscador input {
   display: block;
   width: 100%;
   padding: 0 10px;
   font-size: 16px;
   height: 35px;
}

input[type=text], input[type=password], input[type=number], section article.login input[type=text], section
article#registro input[type=text],
section article.login input[type=password], section article#registro input[type=password] {
   width: 380px;
   color: #495057;
   /* color: #878787; */
   background-color: white;
   background-clip: padding-box;
   border: 1px solid #ced4da;
   border-radius: 0.25rem;
   transition: border-color .15s ease-in-out, box-shadow .15s ease-in-out;
}

input[type=text]:focus, input[type=password]:focus, input[type=number]:focus, section article.login
input[type=text]:focus, section article#registro input[type=text]:focus,
section article.login input[type=password]:focus, section article#registro input[type=password]:focus,
select:focus {
   color: #495057;
   /* color: #878787; */
   background-color: white;
   border-color: #80bdff;
   outline: 0;
   box-shadow: 0 0 0 0.2rem rgb(0 123 255 / .25);
}

input[type=button], section article.login input[type=button], section article#registro input[type=button],
div#buscador input[type=button] {
   height: 45px;
   margin-top: 20px;
   color: white;
   font-weight: 700;
   border-color: transparent;
   background-color: #0071c2;
   transition: background-color .20s ease-in-out;
}

section.container article.segundoContainer input[type=number] {
   height: 30px;
   padding: 5px 10px;
   margin-right: 25px;
}

section.container article.segundoContainer input[type=button] {
   padding: 5px 10px;
```

```
}
/*
section.container article.segundoContainer span {
   margin-top: ;
} */

table {
   width: 100%;
   max-width: 100%;
   margin-bottom: 1rem;
   background-color: transparent;
   border-collapse: collapse;
}

table#tablaLocalConMasReservasPersonas {
   width: 40%;
   margin: auto;
}

/* thead {
   vertical-align: middle;
   border-color: inherit;
} */

thead::after, thead::before {
   box-sizing: border-box;
}

th {
   text-align: inherit;
}

table td, table th {
   padding: 0.75rem;
   vertical-align: top;
   border-top: 1px solid #dee2e6;
}

table thead th {
   vertical-align: bottom;
   border-bottom: 2px solid #dee2e6;
}

tbody::after, tbody::before {
   box-sizing: border-box;
}

table input[type=button] {
   height: 40px;
   color: white;
   padding: 5px 10px;
   margin-top: 0;
   font-size: 14px;
   font-weight: 700;
```

```
    border-color: transparent;
    background-color: #0071c2;
    transition: background-color .20s ease-in-out;
}

table input[type=button]:hover {
    cursor: pointer;
    background-color: #004e85;
}

table img {
    width: 300px;
}

section article#registro input[type=button]#btnRegistrar {
    margin-top: 15px;
}

section article.login input[type=button]:hover, section article#registro input[type=button]:hover,
div#buscador input[type=button]:hover {
    cursor: pointer;
    background-color: #004e85;
}

section article.login p, section article#registro p {
    font-size: 14px;
    margin-top: 15px;
    margin-bottom: 5px;
}

section article.login p, section article#registro p, section article#registro ul,
section article#registro span#camposObligatorios, section.container article.segundoContainer span {
    color: #878787;
}

section article#registro ul {
    font-size: 12px;
    list-style-position: inside;
}

section article.login span, section article#registro span, section.container article.segundoContainer span {
    display: block;
    font-size: 14px;
    margin-top: 20px;
}

section article#registro span#camposObligatorios {
    font-size: 10px;
    margin-top: 5px;
}

section article.login span#cuentaUsuarioRegistro, section article#registro span#cuentaUsuarioInicioSesion {
    text-align: center;
}
```

```
/* Para hacer las líneas al lado de ¿Ya tiene cuenta? */
section article.login span#cuentaUsuarioRegistro:before, section article#registro
span#cuentaUsuarioInicioSesion:before,
section article.login span#cuentaUsuarioRegistro:after, section article#registro
span#cuentaUsuarioInicioSesion:after {
   content: "";
   display: inline-block;
   width: 25%;
   margin: 0 .5em 0 -55%;
   vertical-align: middle;
   border-bottom: 1px solid #878787;
}

section article.login span#cuentaUsuarioRegistro:after, section article#registro
span#cuentaUsuarioInicioSesion:after {
   margin: 0 -55% 0 .5em;
}

section article.login span#cuentaUsuarioRegistro a, section article#registro span#cuentaUsuarioInicioSesion
a {
   font-weight: 500;
   color: #0071c2;
}

/* article nav {
   background-color: #cacaca
} */

#cuposAReservarPersonas {
   width: 300px;
   padding-top: 0.5rem;
   padding-bottom: 0.5rem;
   padding-left: 1rem;
   font-size: 1rem;
   font-weight: 400;
}

div#buscador {
   position: fixed;
}

div#buscador, nav.menuArriba {
   background-color: #cacaca;
   text-align: center;
   padding: 15px 100px;
   /* margin-left: 215px; */
   /* margin-bottom: 300px; */
}
nav.menuArriba {
   /* text-align: left; */
   margin-left: 105px;
   padding: 10px 0;
   /* margin-bottom: 300px; */
```

```
}

nav.menuArriba div.contenidoDeSeccionesConMenuArriba ul li {
   color: #202022;
   transition: all .20s ease-in-out;
}

nav.menuArriba div.contenidoDeSeccionesConMenuArriba ul li:hover {
   color: white;
   background-color: #202022;
}

/* nav.menuArriba ul li a{
   color: #202022;
} */

/* nav.menuArriba ul li a:hover {
   color: white;
} */

article div input[type=button] {
   padding: 5px 10px;
   height: 37px;
}

#fijadorMenuPrincipal {
   width: 100%;
   top: 0;
   left: 0;
   position: fixed;
}

.fijadorMenusArriba {
   width: 100%;
   top: 63px;
   left: 0;
   position: fixed;
}

#btnHabilitar {
   background-color: #28a745;
}

#btnDeshabilitar {
   background-color: #dc3545;
}

#imgLogOut, #logoLocal, #logoPersona, #logoLocal {
   padding-top: 0;
   transition: none;
}

#imgLogOut:hover, #logoLocal:hover, #logoPersona:hover {
   padding-top: 0;
```

```
    transition: none;
    background-color: #00747c;
}

#logoLocal:hover, #logoPersona:hover {
    cursor: auto;
}

div nav ul#menuLateralPersonas div#logoYNombrePersona li#logoPersona, div nav ul#menuLateralLocales
div#logoYNombreLocal li#logoLocal {
    padding-bottom: 5px;
}

div nav ul#menuLateralPersonas span#nombrePersona, div nav ul#menuLateralLocales span#nombreLocal {
    display: block;
    margin-bottom: 40px;
    color: white;
}

#logoYNombreLocal, #logoYNombrePersona {
    margin-top: 25px;
    margin-bottom: 100px;
}

#logoPersona img, #logoLocal img{
    /* width: 70px; */
    width: 130px;
    margin: auto;
}

div#buscador input {
    display: inline-block;

    /* margin: 0 auto; */
}

div#buscador input[type=button] {
    width: 100px;
    margin-top: 0;
    transition: background-color .20s ease-in-out;
}

div#buscador input[type=text] {
    width: 300px;
    color: #495057;
    background-color: white;
    background-clip: padding-box;
    border: 1px solid #ced4da;
    border-radius: 0.25rem;
    margin-right: 30px;
    padding: 5px 10px;
    transition: border-color .15s ease-in-out, box-shadow .15s ease-in-out;
}
```

```
div#buscador input[type=text]:focus {
   color: #495057;
   /* color: #878787; */
   background-color: white;
   border-color: #80bdff;
   outline: 0;
   box-shadow: 0 0 0 0.2rem rgb(0 123 255 / .25);
}
```