

TRABAJO DE FIN DE GRADO

IVÁN GÓMEZ CALVO

CURSO 2022-2023 2º DAW

BURJ AL ARAB, SERVICIO WEB

Índice	Páginas
<u>Portada.</u>	1
<u>Índice.</u>	2 - 3
<u>1. Definición del proyecto.</u>	4
1.1. Tipos de empresas / Sectores productivos.	4
1.2. Descripción del problema.	4
1.3. Herramientas de diseño.	5
1.4. Estudio de viabilidad.	5
<u>2. Planificación del proyecto.</u>	6
2.1. Definición de objetivos, FrontEnd.	6
2.2. Definición de objetivos, BackEnd.	7
2.2.1. Definición de objetivos en la base de datos.	7
2.2.1. Definición de objetivos en la API.	8
2.3. Diseño de estilos.	9
2.4. Codificación.	9
2.5. Documentación.	9
2.6. Planificación de tiempo.	9
<u>3. Análisis del proyecto.</u>	10
3.1. Casos de uso.	10
3.2. Esquema Entidad-Relación.	11
3.3. Esquema Relacional.	12
3.4. Normalización del modelo relacional.	13 - 14
3.4.1. Nuevo esquema Entidad-Relación.	15
3.4.2. Nuevo esquema Relacional.	16
3.5. Diccionario de datos.	17 - 18
3.6. Bloque de creación de tablas.	19 - 21

Índice	Páginas
<u>4. Diseño del proyecto.</u>	22
4.1. Diseño de la estructura de clases FrontEnd.	22 - 24
4.2. Diseño de la estructura de clases BackEnd, API.	24 - 25
4.2.1. Diagrama de clases..	26
4.3. Diseño de estilos del FrontEnd.	27
4.3.1. Sketch de cada página.	27 - 38
4.3.2. WireFrame de cada página.	39 - 52
4.3.3. Mockup.	53 - 54
<u>5. Instalación del proyecto y manuales.</u>	
5.1. Archivos necesarios.	
5.2. Manual de instalación.	
5.3. Manual de uso para el usuario.	
<u>6. Conclusiones.</u>	
6.1. Autoevaluación del proyecto.	
6.2. Propuestas de mejora.	

1. Definición del proyecto.

Mi proyecto supone un servicio web para el hotel de 5 estrellas *Burj Al Arab*, ubicado en Dubái, Emiratos Árabes Unidos. Este servicio web se compone de: una página web donde los usuarios pueden reservar una habitación y contratar servicios adicionales, una API ajustada a la arquitectura RESTful para gestionar las peticiones de los clientes mediante *endpoints* y una base de datos relacional que guarda información del hotel (habitaciones, camas, servicios, clientes...) y de los usuarios registrados.

1.1. Tipos de empresas / Sectores productivos.

Las aplicaciones web están a la orden del día, empresas como Google, Microsoft y Facebook lideran la forma en la que se programa con sus frameworks (o espacios de trabajo) Angular, JQuery y React, respectivamente. Debemos utilizar sus tecnologías aquellos que queramos abrirnos un hueco en el mercado.

Angular es el framework de Google, escrito en TypeScript, una ampliación de JavaScript orientada a objetos que facilita la reutilización y mantenimiento del código con la implementación de un “tipado estricto”, y el que voy a utilizar.

1.2. Descripción del problema.

La población está cada vez más conectada a la red y sus niveles de exigencia no paran de aumentar, necesitan información útil e inmediata, por lo que es de suma importancia mostrar un contenido útil para retener la mayor cantidad de usuarios posible. Tampoco se pueden permitir el lujo de perder tiempo, quieren que la navegación sea rápida y fiable. Para los usuarios que busquen un hotel fuera de lo común, necesitan una página web eficiente, rápida y segura.

1.3. Herramientas de diseño.

A continuación, enumeraré las herramientas que voy a utilizar a lo largo del proyecto:

Cliente / FrontEnd:

- Framework Angular CLI 16.
- Entorno de ejecución Node 18.15.0.
- Gestor de paquetes npm 8.5.5.
- Lenguaje Typescript.
- Estilos con Bootstrap.
- Componentes de Angular Material.
- Visual Studio Code 1.67.2.
- Sistema operativo Windows 10 x64 bits.
- Git, programa de control de versiones para desarrollo de aplicaciones.

Servidor / BackEnd:

- Framework ASP .Net Core.
- Lenguaje C#.
- Visual Studio 2022.
- Microsoft SQL Server Management Studio 17.
- Git, programa de control de versiones para desarrollo de aplicaciones.

Documentación:

- DIA, programa que usaré para generar esquemas entidad-relación, esquemas relacionales y esquemas de clases UML.
- Paint, programa que usaré para generar los diseños de estilos del FrontEnd.
- Microsoft Word, programa que usaré para escribir la documentación del proyecto.
- Microsoft Power Point, programa que usaré para realizar la presentación del proyecto.

1.4. Estudio de viabilidad.

- Viabilidad económica: El proyecto es rentable pues es utilizado como un medio de promoción que va a generar expectativa y ventas.
- Viabilidad técnica: Los lenguajes de programación utilizados están en el ranking de los lenguajes más utilizados actualmente, destacados por la modernidad y por una amplia comunidad de usuarios programadores. Su mantenibilidad de cara al futuro está más que garantizada, pues nunca va a faltar un programador que no sepa TypeScript, C# ni SQL.

2. Planificación del proyecto.

Mi servicio web estará compuesto por un FrontEnd escrito en TypeScript, usando el framework Angular, y un BackEnd compuesto por una Web API RESTful y una base de datos en SQL Server.

2.1. Definición de objetivos, FrontEnd.

En la página web quiero que los usuarios puedan hacer estas tareas:

- Los usuarios pueden registrarse en la página.
- Los usuarios pueden iniciar sesión con su cuenta registrada.
- Los usuarios podrán buscar habitaciones por categoría.
- Los usuarios podrán buscar habitaciones por disponibilidad entre fechas.
- Los usuarios registrados, y con la sesión iniciada, podrán crear reservas de cero, una, o varias habitaciones.
- Los usuarios registrados, y con la sesión iniciada, podrán crear reservas de cero, uno, o varios servicios.
- Los usuarios registrados, y con la sesión iniciada, podrán visualizar, modificar y cancelar sus reservas activas en un panel de usuario.
- Los usuarios registrados, y con la sesión iniciada, podrán visualizar, escribir comentarios, y borrar sus propios comentarios en un foro para los usuarios.
- Habrá un usuario administrador que pueda gestionar todas las reservas, usuarios y comentarios de los clientes. Podrá generar una reserva, modificarla, darla de baja y eliminarla. Tendrá un panel donde añada, modifique, dé de baja y elimine tipos de habitaciones y servicios.
- Diseño web *responsive*. Los usuarios podrán acceder desde móviles, tabletas y ordenadores y que el contenido sea el mismo pero esté adaptado y ordenado para estos tipos de dispositivos. El sistema operativo no será importante siempre y cuando tengan un navegador de internet.

2.2. Definición de objetivos, BackEnd.

2.2.1. Definición de objetivos en la base de datos.

Tener una base de datos que contenga las siguientes tablas y vistas:

	TABLAS	DESCRIPCIÓN
1	CLIENTES	Datos personales de los clientes.
2	HABITACIONES	Registrar cuántas habitaciones hay, el tipo de habitación y si están disponibles o no (reservadas o disponibles).
3	RESERVAS_DE_HABITACIONES	Registrar las reservas, el cliente, la habitación, las fechas de inicio y fin y si la reserva está activa o no.
4	RESERVAS_DE_SERVICIOS	Registrar las reservas de servicio, el cliente, el servicio que reserva y si la reserva está activa o no.
5	TIPOS_DE_CAMA	Tipos de cama que existen.
6	TIPOS_DE_HABITACIONES	Tipos de habitaciones que existen.
7	TIPOS_DE_SERVICIOS	Tipos de servicios que se ofertan.
8	USUARIOS	Datos de los usuarios registrados en el sistema.

	VISTAS	DESCRIPCIÓN
1	DATOS_DE_HABITACIONES_DISPONIBLES	Datos de los tipos de habitaciones y cuántas habitaciones de ese tipo hay disponibles para reservar.

2.2.2. Definición de objetivos en la API.

Una Web API RESTful que contenga los siguientes Endpoints:

- Crear reservas de habitaciones.
- Borrar reservas de habitaciones.
- Actualizar reservas de habitaciones.
- Registrar usuarios.
- Iniciar la sesión de usuarios.
- Añadir servicios a reservas.
- Quitar servicios a reservas.

Aspectos para tener en cuenta:

- Solo los usuarios registrados y con la sesión iniciada pueden reservar habitaciones.
- Solo los usuarios registrados y con la sesión iniciada pueden reservar servicios.
- Los usuarios registrados pueden reservar cero, una o varias habitaciones sin contratar servicios adicionales.
- Los usuarios registrados pueden contratar cero, uno o varios servicios sin reservar habitación.
- No se puede reservar una habitación entre las fechas X e Y mientras la reserva de esta misma habitación esté comprendida entre X e Y, séase antes de X y/o después de Y.

2.3. Diseño de estilos.

En este punto voy a definir qué apariencia general quiero que tenga la página. Crearé los *mockups* que considere oportunos y mostraré más adelante.

Quiero que las páginas tengan un diseño minimalista y elegante, sin sobrecargarlas en exceso. Voy a darle más prioridad a mostrar la información de forma clara y concisa en vez de dársela a cómo de bonita la muestro. Los botones y menús también serán intuitivos para el usuario.

Selecciono Bootstrap como la librería de estilos que voy a utilizar porque cumple con los requisitos que me he autoimpuesto.

2.4. Codificación.

Voy a darle uso a las tecnologías Angular para crear la lógica del FrontEnd, para el BackEnd he seleccionado el marco de trabajo ASP.NET Core para crear una Web API que reciba peticiones del FrontEnd, las procese y responda y SQL Server para generar una base de datos y guardar información del hotel y usuarios.

El FrontEnd es la página web que visitará el cliente y operará en ella. Contará con estilos propios mezclados con elementos de Bootstrap para hacerla *responsive* y ajustada a los estándares que propongo en el anterior apartado.

El BackEnd es el conjunto de programas que interactuarán con el usuario, la API gestionará las peticiones HTTP recibidas del FrontEnd y la base de datos donde se reflejarán los cambios generados por los administradores y usuarios del sistema.

2.5. Documentación.

Mientras programo, voy a ir apuntando puntos interesantes mientras codifico nuevas implementaciones para luego recogerlos, ordenarlos, modificarlos y escribirlos en apartados posteriores en este documento.

2.6. Planificación de tiempo.

El primer mes y medio lo voy a dedicar a aprender las tecnologías Angular y ASP.NET CORE, ver los cursos online proporcionados por la empresa ICP, donde estoy haciendo las prácticas, en la web Udemy.

Dedicaré la segunda mitad de mi estancia aquí a crear mi proyecto y poner mis conocimientos en práctica. Avanzaré en el FrontEnd y en el BackEnd simultáneamente para no saturarme más de una parte que de otra. Como no tengo experiencia suficiente en crear proyectos grandes, voy a tener que modificar muchas cosas sobre la marcha y, siendo realista, seguramente sea lo que más tiempo me consuma, los cambios imprevistos.

3. Análisis del proyecto.

3.1. Casos de uso.

Un cliente puede visualizar los siguientes componentes:

- Home.
- Reservas.
 - Habitaciones en concreto.
- Habitaciones.
 - Habitaciones en concreto.
- Actividades.
- Foro.

Un cliente puede realizar las siguientes acciones:

- Iniciar sesión.
- Registrarse.

Un usuario registrado, y con la sesión iniciada, puede visualizar los siguientes componentes:

- Home.
- Reservas.
 - Filtrar habitaciones por fechas.
- Habitaciones.
 - Filtrar habitaciones por tipo de habitación.
- Actividades.
- Foro.
- Sus datos de usuario.
- Sus reservas.

Un usuario registrado, y con la sesión iniciada, puede realizar las siguientes acciones:

- Reservar habitaciones.
- Reservar servicios.
- Escribir comentarios en el foro.
- Eliminar sus comentarios en el foro.
- Modificar sus reservas.
- Cancelar sus reservas.

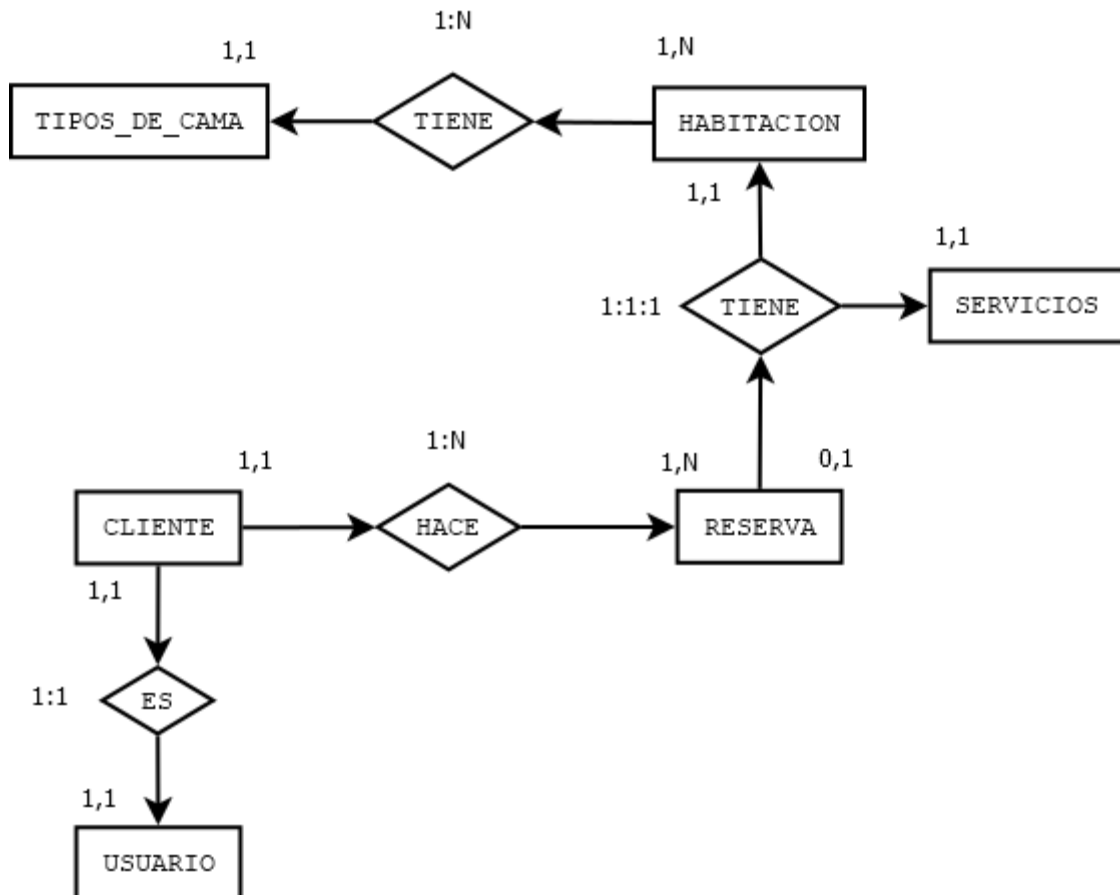
Un usuario administrador puede ver todo lo anterior.

Un usuario administrador puede modificar lo siguiente:

- Ver, añadir, modificar, dar de baja y borrar reservas de habitaciones.
- Ver, añadir, modificar, dar de baja y borrar reservas de servicios.
- Ver, añadir, modificar, deshabilitar y borrar habitaciones.
- Ver, añadir, modificar, deshabilitar y borrar servicios.
- Ver, añadir, modificar, dar de baja y borrar usuarios del sistema.

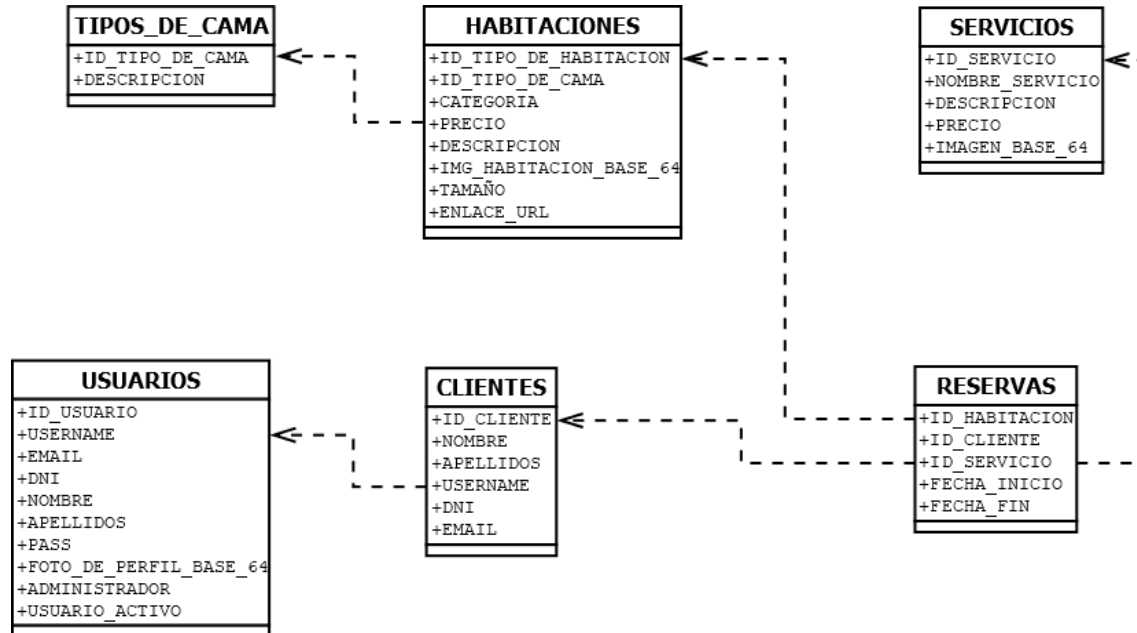
3.2. Esquema Entidad-Relación.

Aquí reflejo un esquema Entidad-Relación de lo que supondría la base de datos en una primera instancia. Quiero aclarar que este esquema va a cambiar más adelante cuando haya hecho el ejercicio de normalización y haya insertado datos por mi cuenta mientras he ido desarrollando la aplicación.



3.3. Esquema Relacional.

Aquí reflejo la transición del primer esquema Entidad-Relación al modelo Relacional. Aclaro nuevamente que este esquema es provisional, está sujeto a cambios.



3.4. Normalización del modelo relacional.

Siendo una primera instancia de ambos modelos Entidad-Relación y Relacional, no he tenido en cuenta la duplicidad de datos ni filas con muchos valores nulos. Es por eso que voy a modificar las siguientes tablas:

- HABITACIONES: Se convertirá en 2 tablas, una llamada TIPOS_DE_HABITACIONES, que guardará datos descriptivos y únicos para cada tipo de habitación, y otra llamada HABITACIONES, que guardará el ID de la habitación, el tipo y un campo que registre si la habitación está reservada o no.
- RESERVAS: Se convertirá en 2 tablas, una llamada RESERVAS_DE_HABITACIONES y otra llamada RESERVAS_DE_SERVICIOS. Una guardará un ID de una reserva asociada al ID de la habitación reservada, la fecha de inicio y fin de la reserva y un valor que indique si la reserva está activa o no. La otra guardará un ID de una reserva de servicio asociada al ID del cliente que efectúa esa reserva, un ID del servicio reservado y un campo que registre si la reserva está activa o no.
- USUARIOS: Se convertirá en 3 tablas, una llamada USUARIOS, contendrá los datos de cada usuario, otra tabla llamada USUARIOS_ADMINISTRADORES, contendrá el ID del usuario que sí sea administrador, y otra tabla llamada USUARIOS_INACTIVOS, contendrá el ID del usuario que sea inactivo.

De esta manera, cumpla con la primera forma normal 1FN, pues no existen filas repetidas en las tablas.

No cumpla con la segunda forma normal 2FN porque, con los cambios realizados anteriormente, surge un nuevo problema en la tabla RESERVAS_DE_SERVICIOS: tiene duplicidad de datos (columna ID_CLIENTE), por lo que tendría que dividirla en 3 tablas distintas:

- RESERVAS_DE_SERVICIOS, donde guardaré el ID de la reserva de servicio y el campo booleano RESERVA_ACTIVA.
- RESERVA_DE_SERVICIOS_DE_CLIENTES, donde guardaré el ID de la reserva del servicio y el ID del cliente.
- RESERVAS_DE_SERVICIOS_TIPOS, donde guardaré el ID de la reserva del servicio y el ID del servicio.

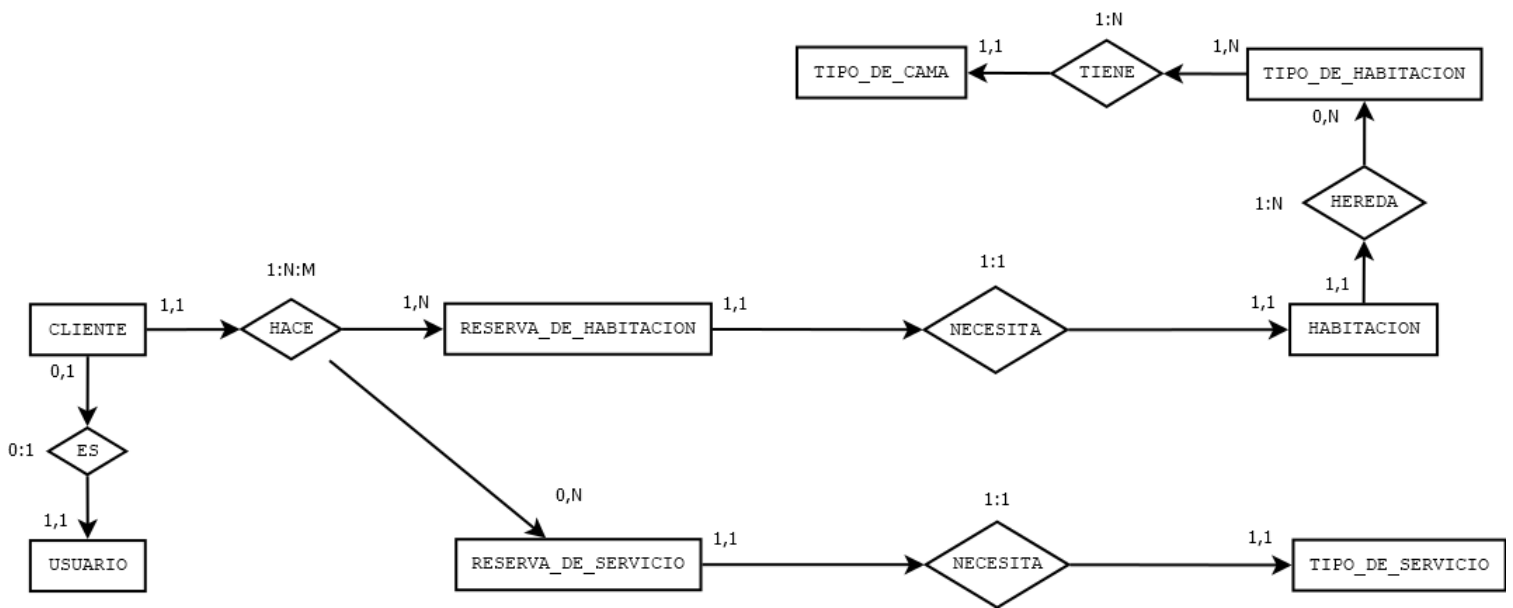
No cumpla con la tercera forma normal 3FN porque tengo dependencias funcionales en las tablas TIPOS_DE_HABITACIONES y HABITACIONES.

- TIPOS_DE_HABITACIONES: Dependencia con el campo TIPO_DE_CAMA, habría que dividir la tabla en dos tablas nuevas. Una para guardar la relación entre una habitación y su tipo de cama y otra tabla con el tipo de habitación y sus datos adicionales.
- HABITACIONES: Dependencia con los campos ID_TIPO_DE_HABITACION y HABITACION_DISPONIBLE_ACTUALMENTE, habría que separarla en dos tablas nuevas. Una para guardar la relación entre el ID de la habitación y el tipo de habitación y otra para guardar el ID de la habitación y el campo HABITACION_DISPONIBLE_ACTUALMENTE.

Por cuestiones de tiempo no voy a poder cumplir con las formas normales segunda y tercera, pero al menos cuento con las soluciones a este problema.

Concluyo este apartado indicando los nuevos esquemas Entidad-Relación y esquema Relacional que se encontrarán en los 2 puntos posteriores a este.

3.4.1. Nuevo esquema Entidad-Relación.



1 Cliente está asociado a 1 Usuario.

1 Usuario puede no ser cliente.

1 Cliente, para ser cliente, hace:

1 o N Reservas de habitaciones.

0 o N Reservas de servicios.

Para hacer 1 Reserva de habitación, se necesita que exista 1 Habitación.

1 o N Habitaciones hereda de 1 tipo de habitación.

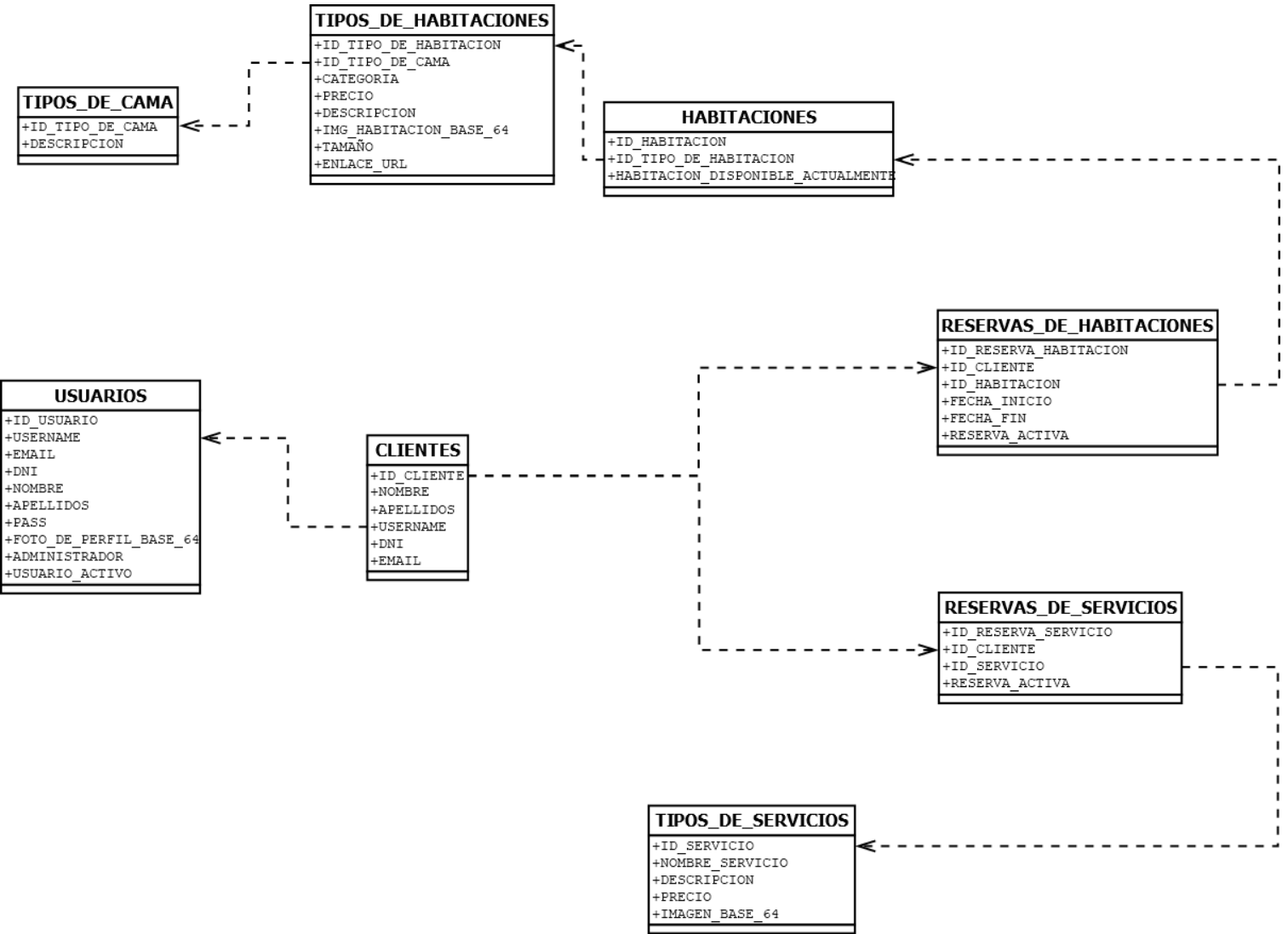
1 Tipo de habitación define 1 o N Habitaciones.

1 Tipo de habitación tiene 1 Tipo de cama.

1 Tipo de cama puede estar asociada a 0 o N Tipos de habitaciones.

Para hacer 1 Reserva de servicio, se necesita que exista 1 Tipo de servicio.

3.4.2. Nuevo esquema Relacional.



3.5. Diccionario de datos.

CLIENTES		
COLUMNA	TIPO DE DATO	LONGITUD
ID_CLIENTE	INT	4
NOMBRE	VARCHAR	30
APELLIDOS	VARCHAR	60
USERNAME	VARCHAR	60
DNI	VARCHAR	9
EDAD	INT	4
EMAIL	VARCHAR	60

HABITACIONES		
COLUMNA	TIPO DE DATO	LONGITUD
ID_HABITACION	INT	4
ID_TIPO_DE_HABITACION	INT	4
HABITACION_DISPONIBLE_ACTUALMENTE	INT	4

RESERVAS_DE_HABITACIONES		
COLUMNA	TIPO DE DATO	LONGITUD
ID_RESERVA_HABITACION	INT	4
ID_CLIENTE	INT	4
ID_HABITACION	INT	4
FECHA_INICIO	DATE	3
FECHA_FIN	DATE	3
RESERVA_ACTIVA	INT	4

RESERVAS_DE_SERVICIOS		
COLUMNA	TIPO DE DATO	LONGITUD
ID_RESERVA_SERVICIO	INT	4
ID_CLIENTE	INT	4
ID_SERVICIO	INT	4
RESERVA_ACTIVA	INT	4

TIPOS_DE_CAMAS		
COLUMNA	TIPO DE DATO	LONGITUD
ID_TIPO_DE_CAMA	INT	4
DESCRIPCION	VARCHAR	60

TIPOS_DE_HABITACIONES		
COLUMNA	TIPO DE DATO	LONGITUD
ID_TIPO_DE_HABITACION	INT	4
ID_TIPO_DE_CAMA	INT	4
CATEGORIA	VARCHAR	60
PRECIO	DECIMAL	9
DESCRIPCION	VARCHAR	512
IMG_HABITACION_BASE_64	VARCHAR	-1
TAMAÑO	INT	4
ENLACE_URL	VARCHAR	-1

TIPOS_DE_SERVICIOS		
COLUMNA	TIPO DE DATO	LONGITUD
ID_SERVICIO	INT	4
NOMBRE_SERVICIO	VARCHAR	120
PRECIO	DECIMAL	9
DESCRIPCION	VARCHAR	512
IMAGEN_BASE_64	VARCHAR	-1

USUARIOS		
COLUMNA	TIPO DE DATO	LONGITUD
ID_USUARIO	INT	4
USERNAME	VARCHAR	60
EMAIL	VARCHAR	60
DNI	VARCHAR	9
NOMBRE	VARCHAR	60
APELLIDOS	VARCHAR	60
PASS	VARCHAR	120
FOTO_DE_PERFIL_BASE_64	VARCHAR	-1
ADMINISTRADOR	BIT	1
USUARIO_ACTIVO	BIT	1

NOTA: Las columnas cuya longitud sea igual a -1 indican que la longitud de ese campo es variable.

3.6. Bloque de creación de tablas.

```
CREATE TABLE TIPOS_DE_CAMAS(  
    ID_TIPO_DE_CAMA      INT          NOT NULL      IDENTITY(1,1)      PRIMARY KEY,  
    DESCRIPCION          VARCHAR(60)      DEFAULT 'NULL_STRING',  
);
```

```
CREATE TABLE TIPOS_DE_HABITACIONES(  
    ID_TIPO_DE_HABITACION INT          NOT NULL      PRIMARY KEY,  
    ID_TIPO_DE_CAMA      INT          NOT NULL      FOREIGN KEY REFERENCES  
TIPOS_DE_CAMAS(ID_TIPO_DE_CAMA),  
    CATEGORIA            VARCHAR(60)      DEFAULT 'NULL_STRING',  
    PRECIO                DECIMAL          DEFAULT 1,  
    DESCRIPCION          VARCHAR(512)      DEFAULT 'NULL_STRING',  
    IMG_HABITACION_BASE_64 VARCHAR(MAX) NOT NULL      DEFAULT  
'IMAGEN_SERIALIZADA_BASE_64',  
    TAMAÑO                INT              DEFAULT 1,  
    ENLACE_URL            VARCHAR(MAX)     NOT NULL      DEFAULT 'NULL_URL',  
  
    CONSTRAINT CONST_CH_TAMAÑO CHECK (TAMAÑO > 0),  
    CONSTRAINT CONST_CH_PRECIO CHECK (PRECIO > 0),  
);
```

```
CREATE TABLE HABITACIONES(  
    ID_HABITACION          INT          NOT NULL      PRIMARY KEY,  
    ID_TIPO_DE_HABITACION INT          NOT NULL      FOREIGN KEY REFERENCES  
TIPOS_DE_HABITACIONES(ID_TIPO_DE_HABITACION),  
    HABITACION_DISPONIBLE_ACTUALMENTE INT NOT NULL      DEFAULT 1,  
);
```

```
CREATE TABLE CLIENTES(  
    ID_CLIENTE            INT          NOT NULL      IDENTITY(1,1)      PRIMARY KEY,  
    NOMBRE                VARCHAR(30)   NOT NULL,  
    APELLIDOS             VARCHAR(60)   NOT NULL,
```

```
USERNAME    VARCHAR(60)          DEFAULT 'NULL_STRING',
DNI          VARCHAR(9)    NOT NULL    UNIQUE,
EMAIL        VARCHAR(60)    NOT NULL    UNIQUE
);
```

```

CREATE TABLE RESERVAS_DE_HABITACIONES(
    ID_RESERVA_HABITACION    INT            NOT NULL        IDENTITY(1,1)    PRIMARY KEY,
    ID_CLIENTE                INT            NOT NULL                                FOREIGN KEY
REFERENCES CLIENTES(ID_CLIENTE),
    ID_HABITACION            INT            NOT NULL                                FOREIGN KEY
REFERENCES HABITACIONES(ID_HABITACION),
    FECHA_INICIO              DATE           NOT NULL,
    FECHA_FIN                 DATE           NOT NULL,
    RESERVA_ACTIVA            INT            DEFAULT 1,

    CONSTRAINT CONST_CH_FECHA_INICIO_LOWER_EQUAL_THAN_FECHA_FIN CHECK(FECHA_INICIO <=
FECHA_FIN),
    CONSTRAINT CONST_CH_FECHA_INICIO_BIGGER_EQUAL_THAN_TODAYS_DATE CHECK( FECHA_INICIO >=
CONVERT(date, GETDATE()) ),
    CONSTRAINT CONST_CH_FECHA_FIN_BIGGER_EQUAL_THAN_TODAYS_DATE CHECK( FECHA_FIN >=
CONVERT(date, GETDATE()) ),
);

CREATE TABLE TIPOS_DE_SERVICIOS(
    ID_SERVICIO                INT            NOT NULL        IDENTITY(1,1)    PRIMARY KEY,
    NOMBRE_SERVICIO            VARCHAR(120)    NOT NULL,
    DESCRIPCION                VARCHAR(512)      DEFAULT 'NULL_STRING',
    PRECIO                     DECIMAL          NOT NULL,
    IMAGEN_BASE_64             VARCHAR(MAX)      DEFAULT 'IMAGEN_SERIALIZADA_BASE_64',
);

CREATE TABLE RESERVAS_DE_SERVICIOS(
    ID_RESERVA_SERVICIO        INT            IDENTITY(1,1)    PRIMARY KEY,
    ID_CLIENTE                 INT            FOREIGN KEY REFERENCES
CLIENTES(ID_CLIENTE),
    ID_SERVICIO                INT            FOREIGN KEY REFERENCES
TIPOS_DE_SERVICIOS(ID_SERVICIO),
    RESERVA_ACTIVA             INT            DEFAULT 1,
);

```

```
CREATE TABLE USUARIOS(  
    ID_USUARIO          INT          NOT NULL          IDENTITY(1,1) PRIMARY KEY,  
    USERNAME            VARCHAR(60)   NOT NULL          UNIQUE,  
    EMAIL               VARCHAR(60)   NOT NULL          UNIQUE,  
    DNI                 VARCHAR(9)    NOT NULL          UNIQUE,  
    NOMBRE              VARCHAR(60)   NOT NULL,  
    APELLIDOS           VARCHAR(60)   NOT NULL,  
    PASS               VARCHAR(120)   NOT NULL,  
    FOTO_DE_PERFIL_BASE_64 VARCHAR(MAX) NOT NULL          DEFAULT 'NULL_VALUE',  
    ADMINISTRADOR        BIT          NOT NULL          DEFAULT 0,  
    USUARIO_ACTIVO       BIT          NOT NULL          DEFAULT 1,  
  
    CONSTRAINT CONST_CH_USERNAME_LENGTH CHECK ( LEN(USERNAME) > 0 ),  
    CONSTRAINT CONST_CH_PASSWORD_LENGTH CHECK ( LEN(PASS) > 8 ),  
    CONSTRAINT CONST_CH_EMAIL_LENGTH CHECK ( LEN(EMAIL) > 0 ),  
);
```

4. Diseño del proyecto.

4.1. Diseño de la estructura de clases FrontEnd.

Listo el árbol de directorios src y todos sus subdirectorios y ficheros:

Nota: no voy a mostrar los ficheros que terminen .component.html y .component.css (exceptuando app.component.html) ni los ficheros .component.ts de los directorios que únicamente tengan 3 ficheros (.component.html, .component.ts y .component.css) porque considero que harían más largo el documento de manera injusta.

```
|   favicon.ico
|   index.html
|   main.ts
|
+---app
|   |   app-routing.module.ts
|   |   app.component.html
|   |   app.component.ts
|   |   app.module.ts
|   |
|   +---backend
|   |       backend.service.ts
|   |
|   +---core
|   |   |   constantes.ts
|   |   |
|   |   +---guards
|   |   |       auth.guard.ts
|   |   |
|   |   \---interfaces
|   |       card-habitacion.interface.ts
|   |       card.interface.ts
|   |       datos-de-habitacion-disponible.interface.ts
|   |       fecha-inicio-fin.interface.ts
|   |       navbarLinks.interface.ts
|   |       user-logged.interface.ts
|   |       user-register-form.interface.ts
|   |
```



```
|   +---pages
|   |   |   gestionar-imagenes.service.ts
|   |   |   pages-routing.module.ts
|   |   |   pages.component.ts
|   |   |   pages.module.ts
|   |   |
|   |   +---actividades
|   |   |
|   |   +---habitaciones
|   |   |   |   habitaciones-routing.module.ts
|   |   |   |   habitaciones.component.ts
|   |   |   |   habitaciones.module.ts
|   |   |   |   habitaciones.service.ts
|   |   |   |
|   |   |   +---habitacion-ejecutivos
|   |   |   |
|   |   |   +---habitacion-individual
|   |   |   |
|   |   |   +---habitacion-lujo
|   |   |   |
|   |   |   +---habitacion-parejas
|   |   |   |
|   |   |   \---habitacion-parejas-azotea
|   |   |
|   |   +---home
|   |   |   |   home.component.ts
|   |   |   |   home.module.ts
|   |   |   |
|   |   |   \---components
|   |   |       \---carousel
|   |   |
|   |   +---panel-administrador
|   |   |
|   |   +---reservas
|   |   |   |   reservas-routing.module.ts
|   |   |   |   reservas.component.ts
```

```
| | | | reservas.module.ts
| | | | reservas.service.ts
| | | |
| | | \---components
| | |     \---caja-reservar-mediante-fechas
| | |
| | +---nuestro-foro
| | |     nuestro-foro.component.ts
| | |
| | +---user-login
| | |     user-login.component.ts
| | |
| | \---user-register
| |     user-register.component.ts
| |
| \---shared
| |     shared.module.ts
| |
| +---button-go-home
| |
| +---button-login
| |
| +---button-register
| |
| +---caja-reservar-mediante-fechas
| |
| +---footer
| |
| +---material
| |
| \---navbar
|
+---assets
| |     .gitkeep
| |
| \---images
```



4.2. Diseño de la estructura de clases BackEnd, API.

Listo todos los directorios, ficheros y subdirectorios que cuelgan de la raíz del proyecto:

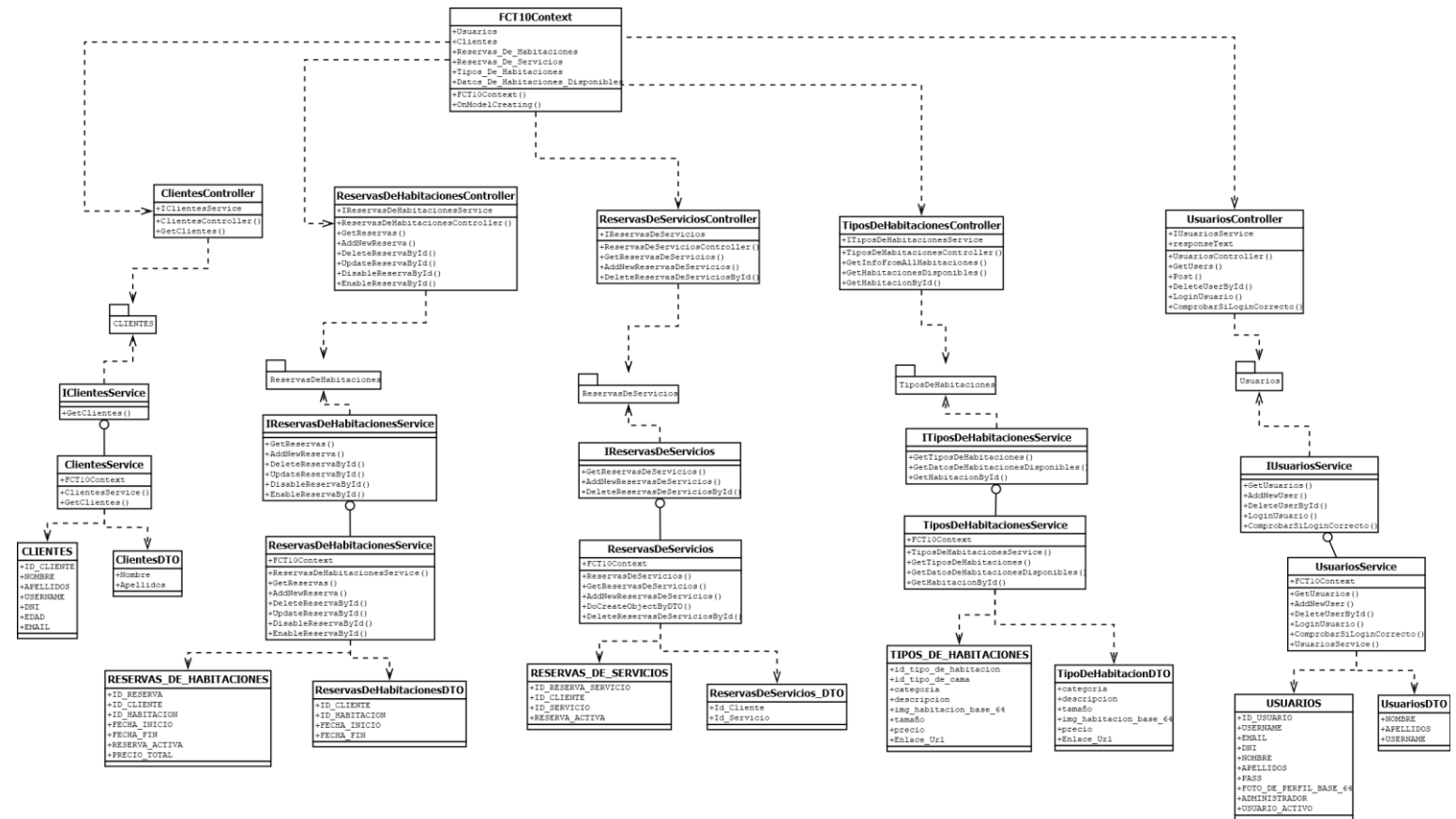
Nota: Descarto los directorios bin/* y obj/* pues no contienen ficheros que haya programado yo sino ficheros que crea el IDE Visual Studio para la ejecución del proyecto y porque considero que harían más largo el documento de manera injusta como indiqué en el apartado anterior.

```
| appsettings.Development.json
| appsettings.json
| Program.cs
| TFGHotel.csproj
|
+---Context
|     FCT10Context.cs
|
+---Controllers
|     ClientesController.cs
|     ReservasDeHabitacionesController.cs
|     ReservasDeServiciosController.cs
|     TiposDeHabitacionesController.cs
|     UsuariosController.cs
|
+---DTO
|     ClientesDTO.cs
|     FechaInicioFinDTO.cs
|     ReservasDeHabitacionesDTO.cs
|     ReservasDeServicios_DTO.cs
|     TipoDeHabitacionDTO.cs
|     UsuarioLoginDTO.cs
|     UsuariosDTO.cs
|
+---Entities
|     CLIENTES.cs
```

```
|      DATOS_DE_HABITACIONES_DISPONIBLES.cs
|      RESERVAS_DE_HABITACIONES.cs
|      RESERVAS_DE_SERVICIOS.cs
|      TIPOS_DE_HABITACIONES.cs
|      USUARIOS.cs
|
+---Properties
|      launchSettings.json
|
\---Services
    +---Clientes
    |      ClientesService.cs
    |      IClientesService.cs
    |
    +---ReservasDeHabitaciones
    |      IReservasDeHabitacionesService.cs
    |      ReservasDeHabitacionesService.cs
    |
    +---ReservasDeServicios
    |      IReservasDeServicios.cs
    |      ReservasDeServicios.cs
    |
    +---TiposDeHabitaciones
    |      ITiposDeHabitacionesService.cs
    |      TiposDeHabitacionesService.cs
    |
    \---Usuarios
        IUserariosService.cs
        UsuariosService.cs
```

4.2.1. Diagrama de clases.

A continuación, muestro el diagrama de clases UML

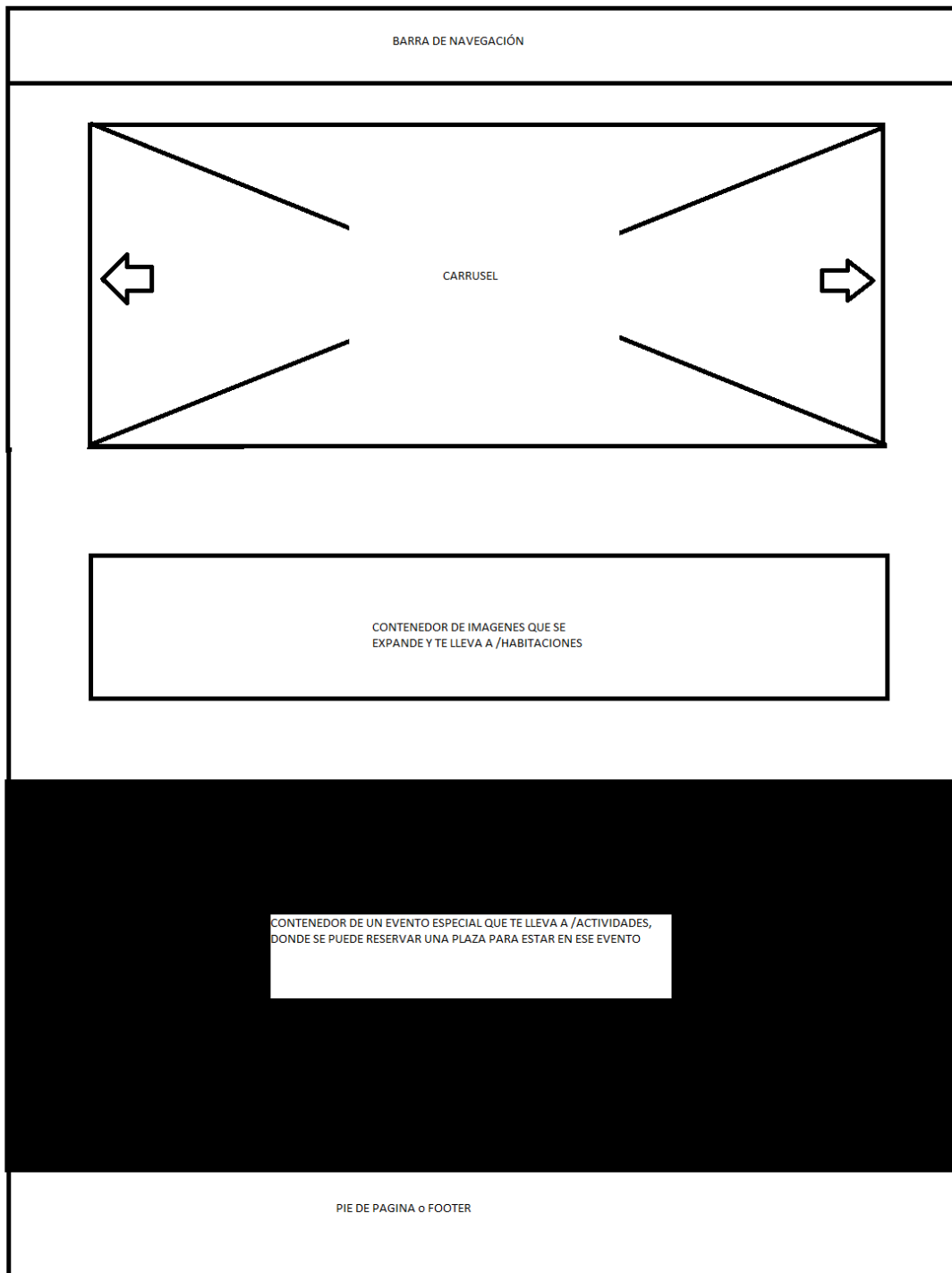


4.3. Diseño de estilos del FrontEnd.

4.3.1. Sketch de cada página

Página /home

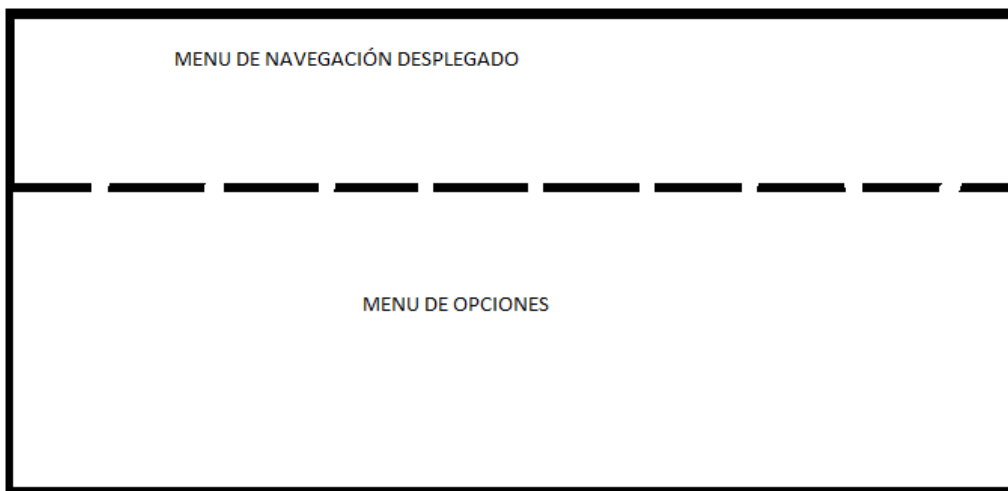
Ordenadores:



Página /home

Móviles y tablets:

Nota: Las páginas son iguales, siguen la misma estructura solo que, la barra de navegación se convierte en una barra desplegable con las opciones del menú.



Página /reservas

Ordenadores:

MENU DE NAVEGACION

FORMULARIO

CARD

CARD

CARD

CARD

CARD

FOOTER o PIE DE PÁGINA

Página /reservas

Nota: La única diferencia es cómo se muestran las Tarjetas.

Móviles	Tablets	
<div>CARD</div> <div>CARD</div> <div>CARD</div>	<div>CARD</div> <div>CARD</div> <div>CARD</div> <div>CARD</div>	<div>CARD</div> <div>CARD</div> <div>CARD</div>

Página /habitaciones

Ordenadores:

MENU DE NAVEGACION

TEXT0

FORMULARIO

CARD

CARD

CARD

CARD

CARD

FOOTER o PIE DE PÁGINA

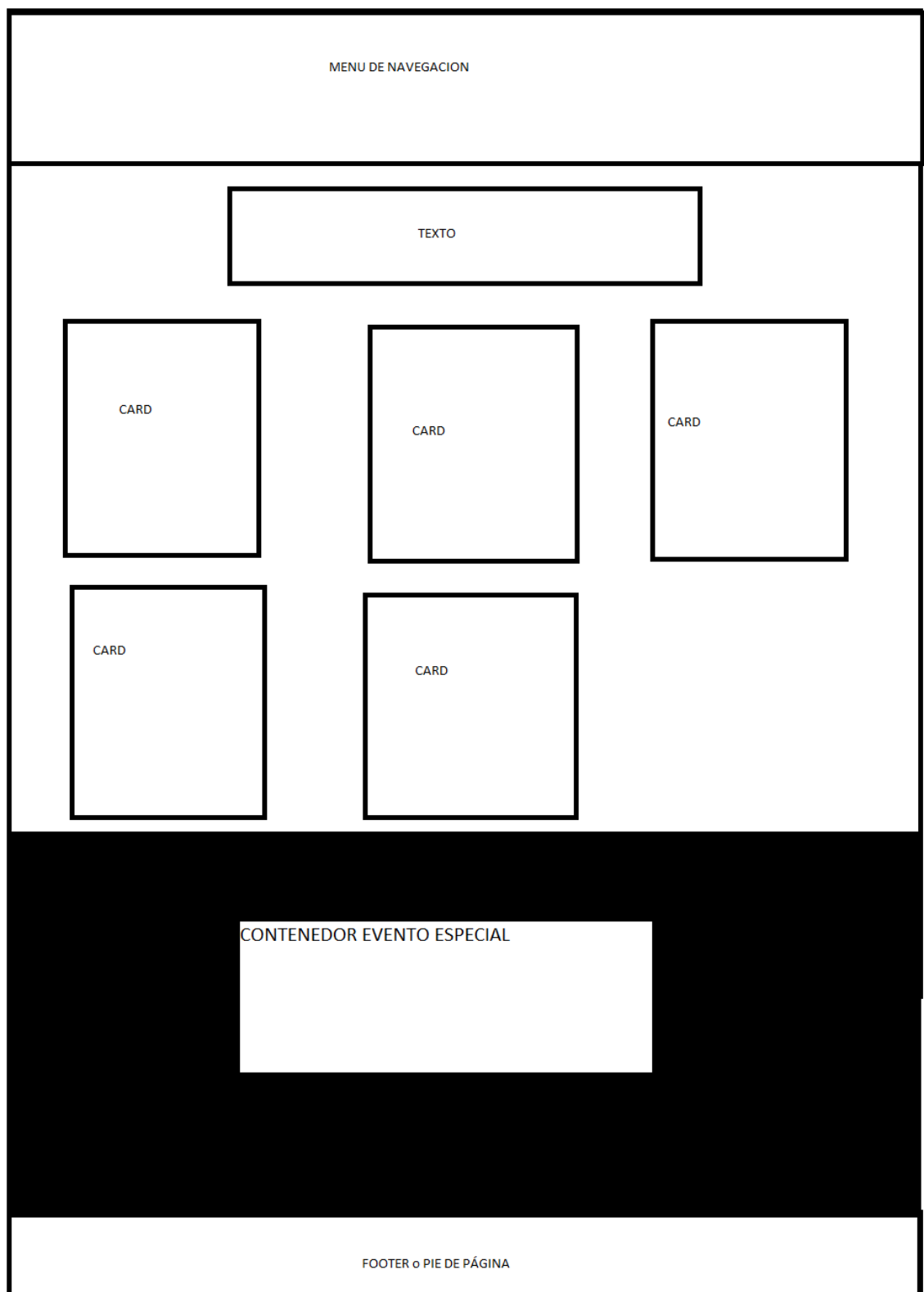
Página /habitaciones

Nota: La única diferencia es cómo se muestran las Tarjetas.

Móviles	Tablets	
<div>CARD</div>	<div>CARD</div>	<div>CARD</div>
<div>CARD</div>	<div>CARD</div>	<div>CARD</div>
<div>CARD</div>	<div>CARD</div>	

Página /reservas

Ordenadores:



Nota: La única diferencia es cómo se muestran las Tarjetas.

Móviles	Tablets	
<div>CARD</div>	<div>CARD</div>	<div>CARD</div>
<div>CARD</div>	<div>CARD</div>	<div>CARD</div>
<div>CARD</div>	<div>CARD</div>	

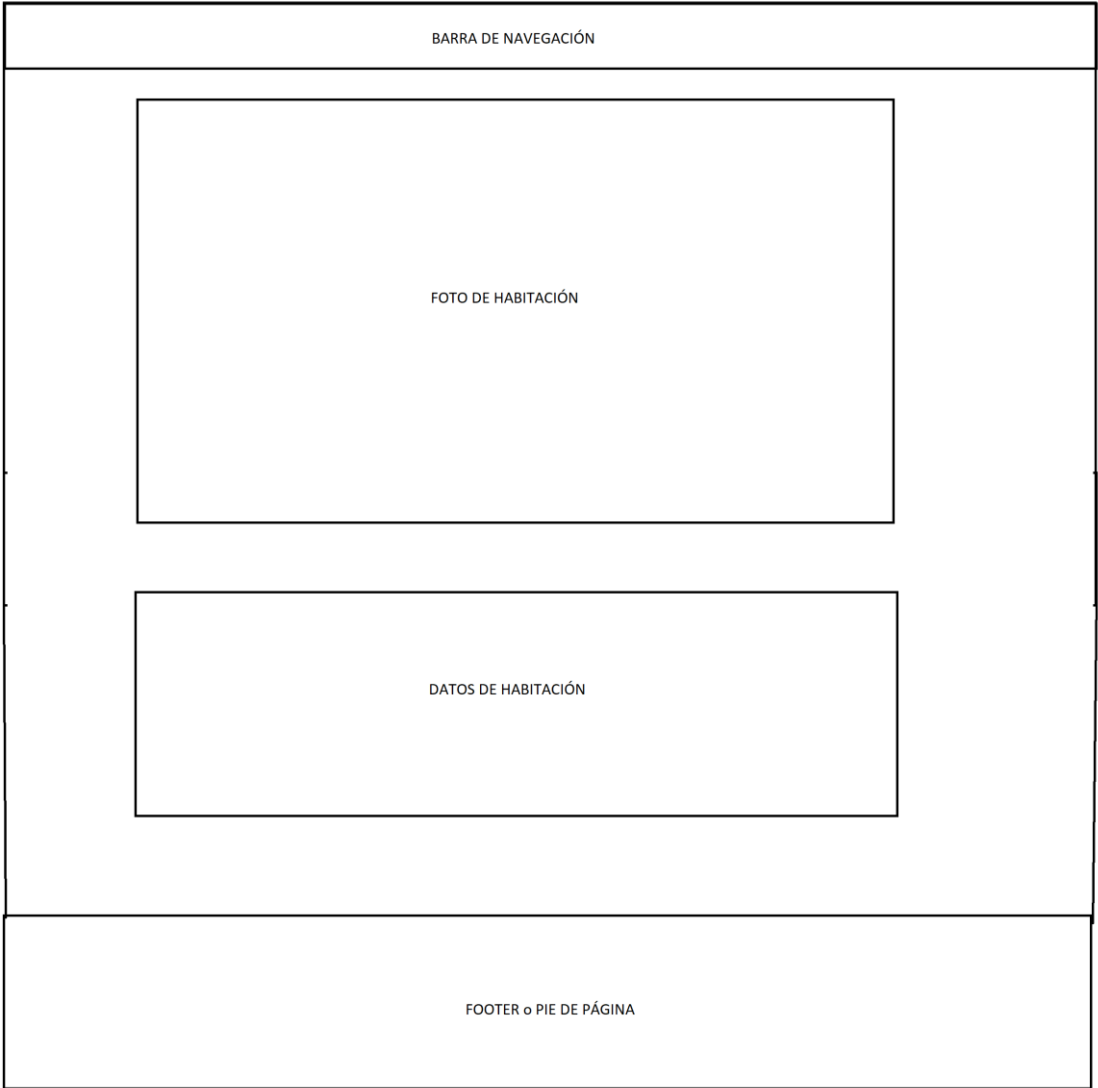
Página /habitaciones/habitación

Ordenador:

BARRA DE NAVEGACIÓN	
IMAGEN DE HABITACIÓN	DATOS DE HABITACIÓN
PIE DE PÁGINA o FOOTER	

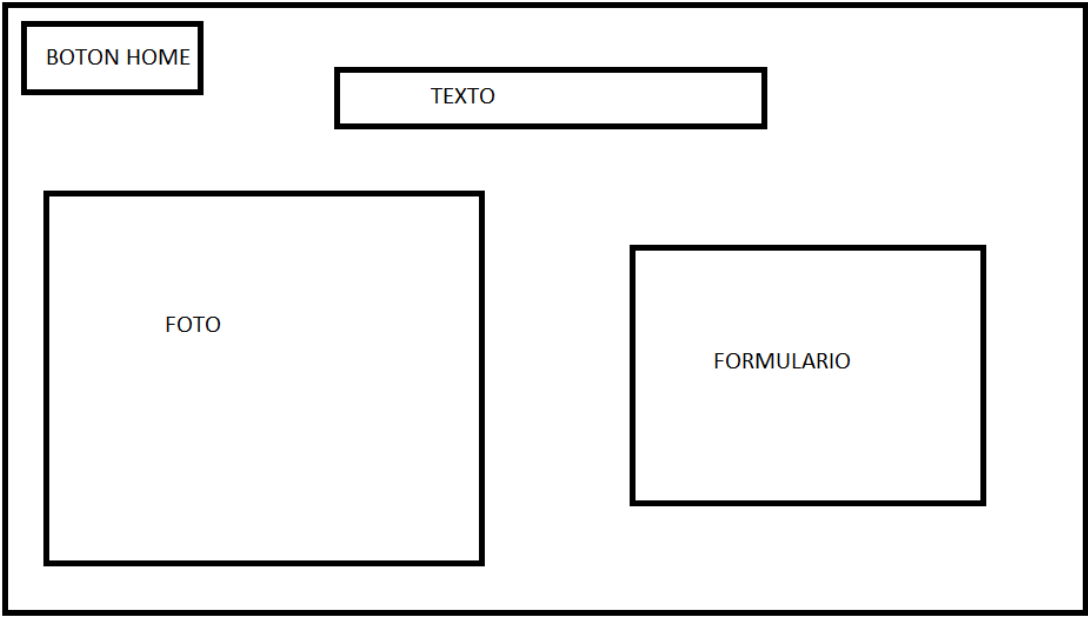
Página /habitaciones/habitación

Móviles y tablets:



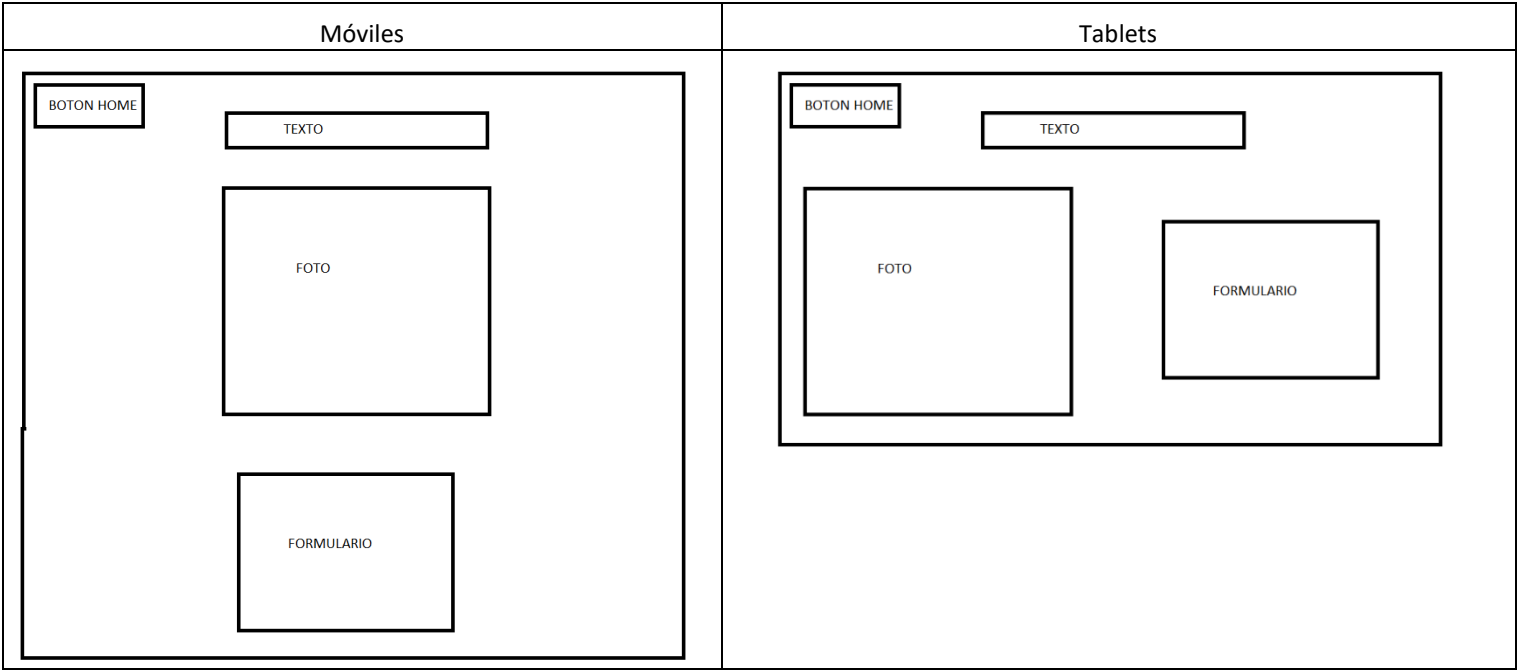
Página /login

Ordenadores:



Página /login

Nota: La única diferencia es cómo se muestra la imagen y el formulario.



Página /register

Ordenadores, móviles y tablets.

BOTON HOME

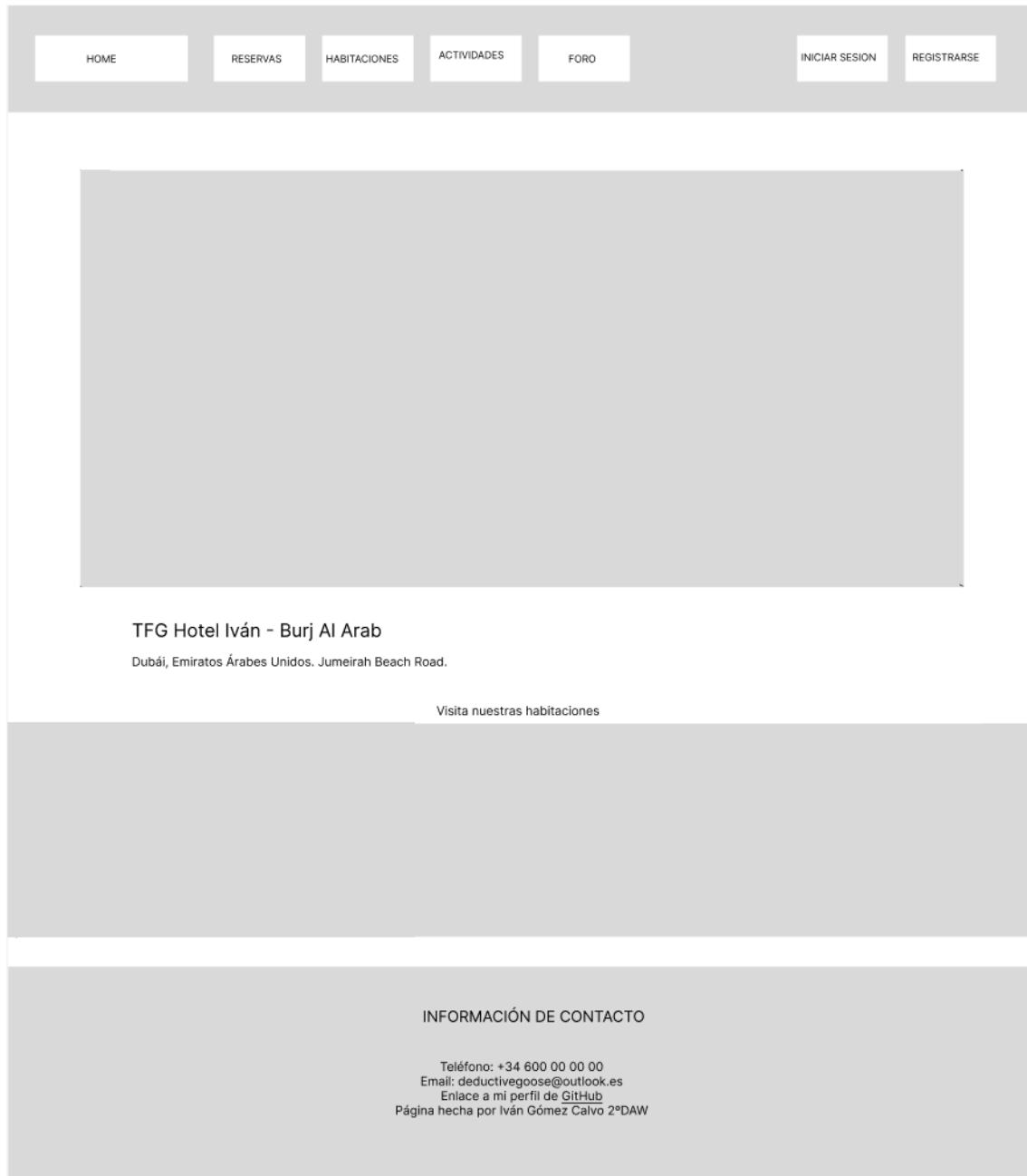
TEXT0

FORMULARIO

4.3.2. WireFrame de cada página

Página /home

Ordenadores:



Página /home

Los cambios de dispositivos para la página home únicamente afectan a cómo se muestra la barra de navegación. La colocación de elementos del resto de la página no cambia. A continuación muestro cómo se vería esta barra de navegación plegada en móviles y desplegada en Tablets.

Móviles	Tablets
<div><div>TFG Hotel Iván - Burj Al Arab</div><div>▼</div></div>	<div><div>HOME</div><div>HABITACIONES</div><div>RESERVAS</div><div>ACTIVIDADES</div><div>FORO</div><div>INICIAR SESION</div><div>REGISTRARSE</div></div>

Página /reservas

Ordenadores:

HOME

RESERVAS

HABITACIONES

ACTIVIDADES

FORO

INICIAR SESION

REGISTRARSE

TEXTO

FORMULARIO

FORMULARIO

BOTON

TITULO

DESCRIPCION

Ver

Reservar

TITULO

DESCRIPCION

Ver

Reservar

TITULO

DESCRIPCION

Ver

Reservar

INFORMACIÓN DE CONTACTO







Teléfono: +34 600 00 00 00

Email: deductivegoose@outlook.es

Enlace a mi perfil de [GitHub](#)

Página hecha por Iván Gómez Calvo 2ºDAW

Los cambios en la colocación de los elementos para móviles y tablets se ven reflejados, únicamente, en la manera en que se organizan las tarjetas.

Móviles	Tablets
<div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div></div></div>	<div><div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div></div><div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div></div></div></div></div></div></div>

Página /habitaciones

Ordenadores:

HOME

RESERVAS

HABITACIONES

ACTIVIDADES

FORO

INICIAR SESION

REGISTRARSE

Permite que la habitación te elija a ti

Duerme en la habitación de tus sueños. Date el placer de dormir el dormitorio que se ajuste a tu estilo de vida. Puedes tocar las nubes, dormir bajo el agua, tener la arena de la playa en la puerta.

¡ El límite es tu imaginación !

— Filtra tu habitación —

Reiniciar Filtro

TITULO

DESCRIPCION

Ver

Reservar

TITULO

DESCRIPCION

Ver

Reservar

TITULO

DESCRIPCION

Ver







Reservar

INFORMACIÓN DE CONTACTO

Teléfono: +34 600 00 00 00
Email: deductivegoose@outlook.es
[Enlace a mi perfil de GitHub](#)
Página hecha por Iván Gómez Calvo 2ºDAW

Página /habitaciones

Los cambios en la colocación de los elementos para móviles y tablets se ven reflejados, únicamente, en la manera en que se organizan las tarjetas.

Móviles	Tablets
<div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div></div></div>	<div><div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div></div><div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div><div><div><p>TITULO</p><p>DESCRIPCION</p><div><div>Ver</div><div>Reservar</div></div></div></div></div></div></div></div></div>

Página /habitaciones/habitación

Ordenador:

HOME

RESERVAS

HABITACIONES

ACTIVIDADES

FORO

INICIAR SESION

REGISTRARSE

TITULO

PRECIO €

DESCRIPCION

BOTON

INFORMACIÓN DE CONTACTO

Teléfono: +34 600 00 00 00
Email: deductivegoose@outlook.es
Enlace a mi perfil de [GitHub](#)
Página hecha por Iván Gómez Calvo 2ºDAW

Página /habitaciones/habitación

Móviles y tablets

HOME

RESERVAS

HABITACIONES

ACTIVIDADES

FORO

INICIAR SESION

REGISTRARSE

TITULO

PRECIO €

DESCRIPCION

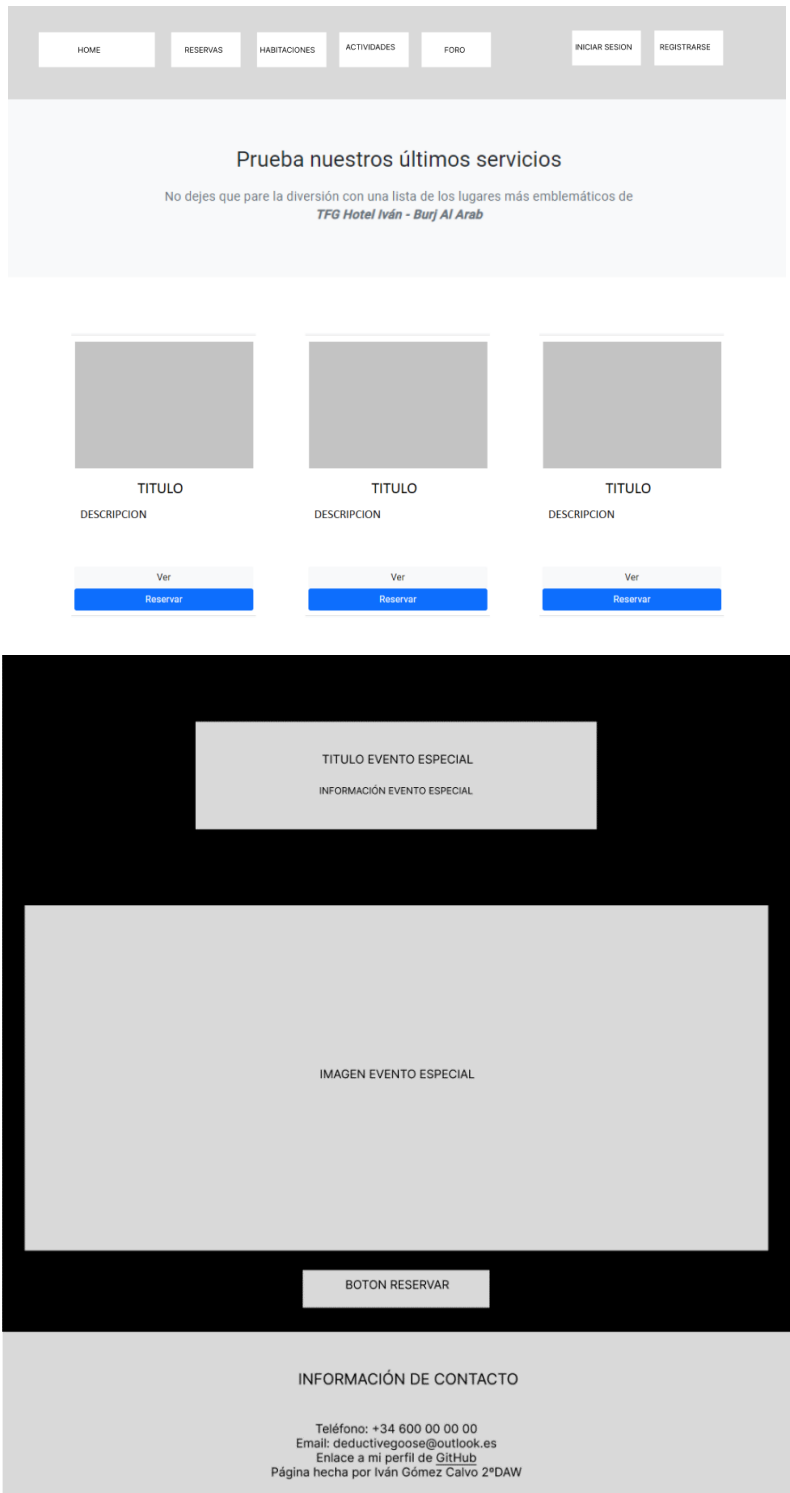
BOTON

INFORMACIÓN DE CONTACTO

Teléfono: +34 600 00 00 00
Email: deductivegoose@outlook.es
Enlace a mi perfil de [GitHub](#)
Página hecha por Iván Gómez Calvo 2ºDAW

Página /actividades

Ordenadores:



Página /actividades

Los cambios en la colocación de los elementos para móviles y tablets se ven reflejados, únicamente, en la manera en que se organizan las tarjetas.

Móviles	Tablets	
<div><div></div><div>TITULO</div><div>DESCRIPCION</div><div>Ver</div><div>Reservar</div></div> <div><div></div><div>TITULO</div><div>DESCRIPCION</div><div>Ver</div><div>Reservar</div></div> <div><div></div><div>TITULO</div><div>DESCRIPCION</div><div>Ver</div><div>Reservar</div></div>	<div><div></div><div>TITULO</div><div>DESCRIPCION</div><div>Ver</div><div>Reservar</div></div>	<div><div></div><div>TITULO</div><div>DESCRIPCION</div><div>Ver</div><div>Reservar</div></div>

Página /login

Ordenador:

BOTON



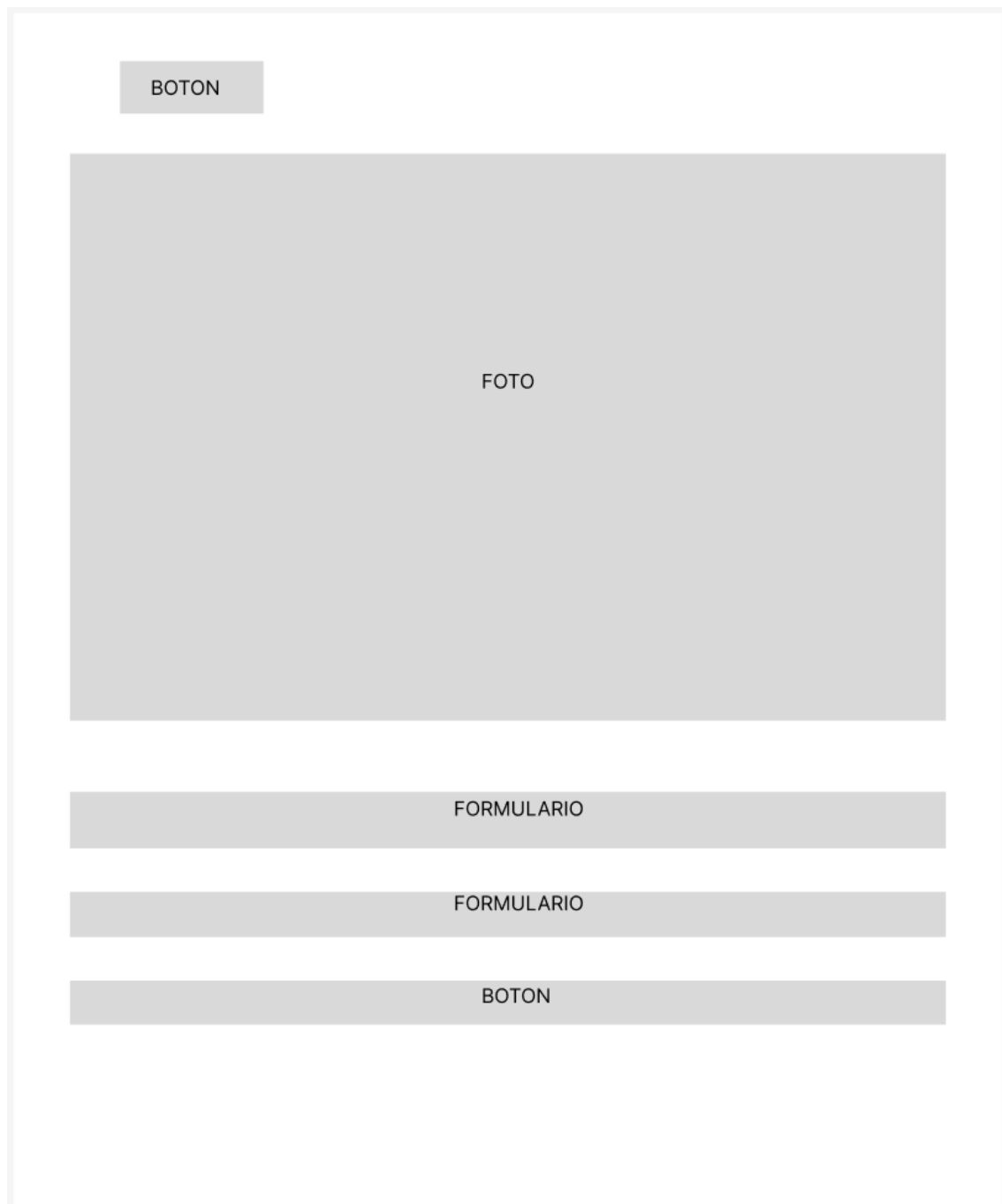
FORMULARIO

FORMULARIO

BOTON

Página /login

Móviles y tablets:



Página /register

Ordenadores:

BOTON

TEXTO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

BOTON

Página /register

Tablets:

BOTON

TEXTO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

BOTON

Página /register

Móviles:

BOTON

TEXTO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

FORMULARIO

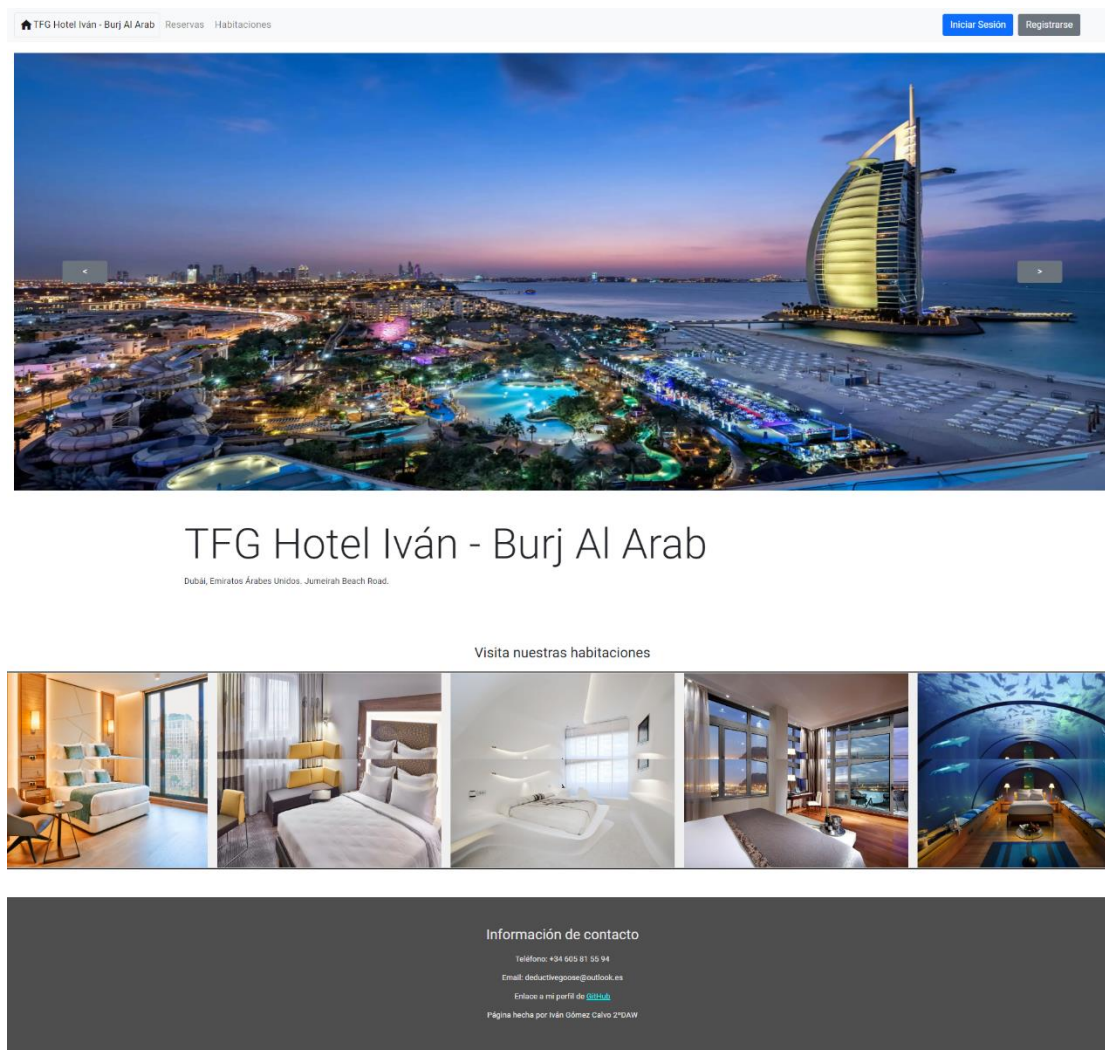
BOTON

4.3.3. Mockup de cada página.

Por cuestiones de espacio, no voy a incluir el Mockup de cada página para cada dispositivo porque son muchas capturas de pantalla y muy largas. Incluyo 2 páginas para que se vea un ejemplo del resultado final.

Página /home

Ordenadores:



Página /reservas

Ordenadores:

TFG Hotel Iván - Burj Al Arab

Reservas

Habitaciones

Iniciar Sesión

Registrarse

Elige tu periodo de vacaciones

Busca las habitaciones disponibles entre las fechas que desees disfrutar.

Inicio

dd/mm/aaaa

Fin

dd/mm/aaaa

Visualizar habitaciones disponibles

Habitación individual

Habitación para los amantes de la soledad. Selección perfecta para desconectar de las personas. Disfruta de las vistas a la ciudad mientras tomas tu bebida favorita

Ver

Reservar

Habitación para ejecutivos

Habitación individual. Selección perfecta para hospedarte en un lugar que permita el descanso y el trabajo al mismo tiempo.

Ver

Reservar

Habitación para parejas

Habitación para parejas. Disfrute de las vistas al mar y a la playa desde lo alto del hotel. Su disfrute está insonorizado para no tener ni un ápice de ruido del exterior.

Ver

Reservar

Habitación para parejas en azotea

Habitación para parejas. Disfrute de las vistas a la ciudad y al mar desde lo alto del edificio en una terraza perfectamente iluminada con luz natural.

Ver

Reservar

Habitación de lujo

Habitación para los clientes más exigentes. Disfrute de las vistas al mar desde lo más bajo del edificio en un espacio habilitado debajo del agua.

Ver

Reservar

Información de contacto

Teléfono: +34 605 81 55 94

Email: deductivegoose@outlook.es

Enlace a mi perfil de [GitHub](#)

Página hecha por Iván Gómez Calvo 2ºDAW

Página 57 de X

5. Instalación del proyecto y manuales.

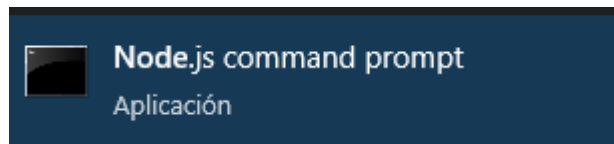
5.1. Archivos necesarios.

Los archivos necesarios, Front, Back y documentación se encuentran en un repositorio público de mi GitHub en la siguiente url: <https://github.com/deduc/TFG-Hotel.git>

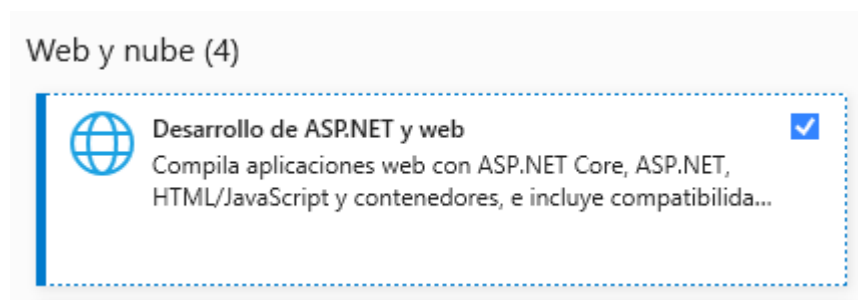
5.2. Manual de instalación.

Este manual de instalación es para dispositivos que utilicen el sistema operativo Windows 10 porque utilizo tecnologías de Microsoft como lo son Visual Studio y SQL Server. Voy a utilizar una máquina virtual de Windows 10 de 64 bits y partiré de una instalación limpia: sistema operativo instalado y actualizado.

1. Descargar e instalar la versión 18.16.0 de Node.js de este enlace: <https://nodejs.org/es>. Lo más difícil del proceso de instalación es contar las veces que hay que pulsar "Siguiente". Hay que abrir la consola de Node.js, para ello hay que abrir el menú de Windows y escribir "Node".

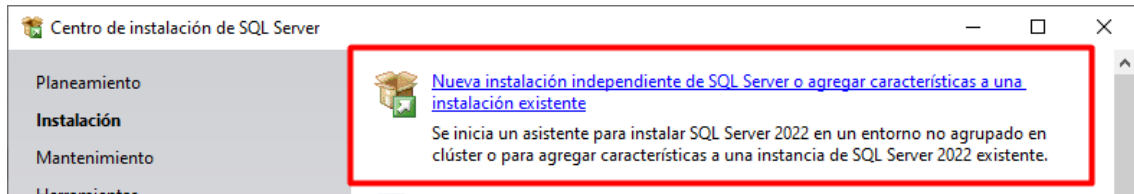


2. Instalar el gestor de paquetes npm con este comando: `npm install -g npm`
3. Instalar Angular CLI con este comando: `npm install -g @angular/cli`
4. Descargar el instalador de Visual Studio de este enlace: <https://visualstudio.microsoft.com/es/thank-you-downloading-visual-studio/?sku=Community&channel=Release&version=VS2022&source=VSLandingPage&cid=2030&workload=dotnet-dotnetwebcloud&passive=false#dotnet>. La instalación también consiste en apretar el botón de "Siguiente". Y, en un apartado donde te pregunta qué tecnologías quieres instalar, selecciona la mostrada en la siguiente foto:



5. Descargar el instalador de Microsoft SQL Server, edición Express, en este enlace (yo he usado la versión en inglés y no he tenido ningún problema): <https://www.microsoft.com/en-us/sql-server/sql-server-downloads>. Esta instalación es más compleja, por lo que procedo a detallarla a continuación:

- 5.1. Ejecutar el instalador. Seleccionar el tipo de instalación Custom.
- 5.2. Seleccionar la ruta de instalación. En mi caso, se ubicará en "C:\SQL2022"
- 5.3. Una vez haya terminado de cargar, selecciona la opción que se ve en la siguiente captura de pantalla:

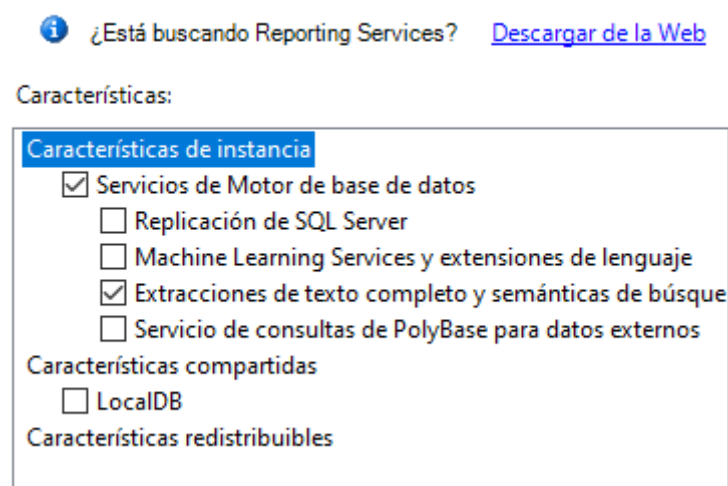


- 5.4. Acepta los términos de licencia, marcar esta casilla y clicar en siguiente:

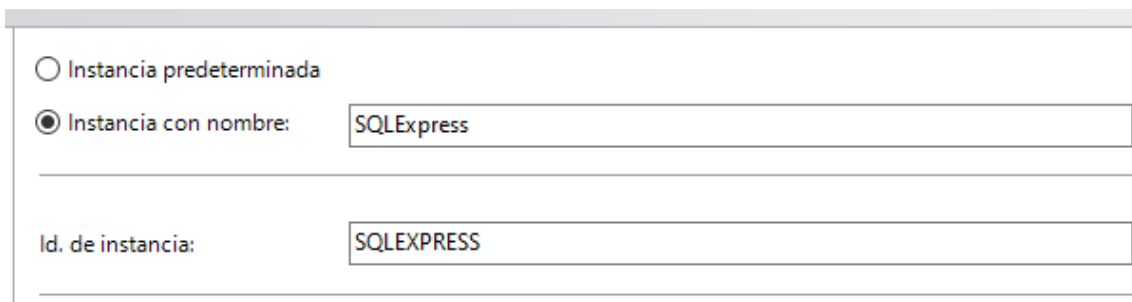
Microsoft Update proporciona actualizaciones de seguridad y otras actualizaciones importantes para Windows y otro software de Microsoft, como SQL Server 2022. Las actualizaciones se proporcionan mediante actualizaciones automáticas o a través del sitio web de Microsoft Update.

☒ Usar Microsoft Update para comprobar las actualizaciones (recomendado)

- 5.5. Darle a siguiente, esperar un momento a que cargue y volver a darle. A continuación, marca las siguientes casillas:



5.6. Selecciona el nombre de tu instancia y clicla en siguiente.



The screenshot shows a window with two radio buttons. The first is 'Instancia predeterminada' (unselected). The second is 'Instancia con nombre:' (selected). To the right of the second radio button is a text box containing 'SQLExpress'. Below this, there is a label 'Id. de instancia:' followed by a text box containing 'SQLEXPRESS'.

5.7. Selecciona tu método favorito de ordenación de textos. Yo lo he dejado por defecto y he clicado en siguiente.

5.8. Te van a salir un nuevo menú que permite configurar algunas rutas interesantes para gestionar la base de datos. Yo las he dejado por defecto y he clicado en siguiente. Ahora el programa continuará instalando nuevos paquetes sql.

5.9. Clicla en esta opción y descarga las SQL Server Management Studio y ejecuta el instalador:



[Instalar las herramientas de administración de SQL Server](#)

Se inicia una página de descarga que proporciona un vínculo para instalar SQL Server Management Studio, las utilidades de la línea de comandos de SQL Server (SQLCMD y BCP), el proveedor SQL Server PowerShell, SQL Server Profiler y Asesor de optimización de base de datos. Se necesita una conexión a Internet para instalar estas herramientas.

Descargar SSMS

↓ [Descarga gratuita de SQL Server Management Studio \(SSMS\) 19.1](#)

5.10. Tras haber instalado el programa, solo hay que abrir la aplicación SQL Server Management, insertar tus credenciales de acceso, navegar por el panel izquierdo hasta encontrar "Databases", crear una base de datos, yo la voy a llamar FCT_10. A continuación, has de abrir el directorio "BDD - my sql files" y ejecutar en tu base de datos los ficheros "0. resetear bdd.sql", "0.1 crear vistas.sql" y "0.2 insertar imagenes base 64.sql" para crear las tablas e insertar los datos.

6. Descargar Git en este enlace: <https://gitforwindows.org/>. La instalación consiste en apretar muchas veces "Siguiente".
7. Descargar el repositorio de Git con el siguiente comando: `git clone https://github.com/deduc/TFG-Hotel.git`.
8. Para ejecutar todo el servicio web:

8.1. En primer lugar, hay que abrir la consola de Node.js, navegar hasta el directorio descargado, ir a esta carpeta: "TFG-Hotel\Front - TFG Hotel" y escribir los siguientes comandos: `npm install` y `ng serve -o`.

8.2. En segundo lugar, hay que ejecutar la aplicación SQL Server Express.

8.3. En tercer lugar, hay que abrir el fichero TFGHotel.sln que se ubica en la ruta "Back - TFG Hotel\TFGHotel" con el programa Visual Studio, entrar en el fichero "TFGHotel/appsettings.json" y escribir lo mostrado a continuación. Es importante que, en el campo defaultConnection escribas el nombre de tu servidor y la base de datos donde quieras operar.

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "defaultConnection": "Server=PORTATIL-IVAN\\SQLEXPRESS;Database=FCT_10;Trusted_Connection=true;MultipleActiveResultSets=True; Encrypt=False; TrustServerCertificate=False"
  }
}
```

6. Conclusiones.

Concluyo la documentación del proyecto repasando los aspectos más importantes del desarrollo de mi servicio web para el hotel *Burj Al Arab*. El servicio está compuesto, principalmente, por 3 elementos generales: *FrontEnd*, *BackEnd* y una base de datos.

Angular es el *framework* que he elegido para estructurar, maquetar, dotar de lógica y separar la información que está a disposición del usuario en componentes.

ASP.NET Core Web API es el *framework* que he escogido para gestionar las solicitudes HTTP que vienen del FrontEnd. Es una aplicación que he diseñado para recibir llamadas HTTP, gestionarlas mediante controladores y devolver datos o ejecutar procedimientos en la base de datos.

Microsoft SQL Server es la aplicación que he utilizado para crear toda la base de datos y almacenar información sobre el hotel y los clientes.

Debido a mi falta de experiencia, la etapa de análisis y diseño de la aplicación tenía muchas debilidades como el ejercicio de normalización de las tablas, fallos de seguridad en el login de usuarios y problemas con el entendimiento de conceptos al ser 2 tecnologías que aprendía prácticamente desde cero.

6.1. Autoevaluación del proyecto.

He tenido aproximadamente tres meses para realizar este proyecto de los cuales uno y medio se han esfumado viendo los cursos de Angular y ASP.NET Core en la plataforma Udeemy (los cursos han sido proporcionados por la empresa ICP) y la otra mitad del tiempo ha estado orientada a realizar el proyecto.

Termino con una sensación agri dulce; es la primera vez que curso clases online con unos profesores a los que no he podido preguntar dudas y es algo a lo que no estoy acostumbrado, entonces me he visto solo en muchas ocasiones y frustrado por no entender qué falla o dónde está el error.

El framework ASP.NET Core, con el lenguaje C#, es el primer espacio de trabajo donde he programado con un lenguaje orientado a objetos (no es lo mismo que un lenguaje que la implementa) y con un tipado estricto. Aprender TypeScript ha sido una tarea menos exigente porque es una ampliación de JavaScript, lenguaje que he aprendido en este curso, 2º DAW, entonces ya contaba con una base sólida que me ha permitido evolucionar rápidamente.

Angular ha sido lo que me ha dado quebraderos de cabeza con las tecnologías que usa como el sistema de enrutamiento de componentes, los componentes que son importados, exportados y declarados por los módulos.

Por otro lado, estoy muy contento de haber aprendido a usar dos frameworks desde cero y haber creado una aplicación también desde cero. He conseguido ampliar mucho mis conocimientos en el ámbito de la programación y el mundo empresarial.

6.2. Propuestas de mejora.

Para la próxima voy a hacer mucho más énfasis en las fases de planificación y diseño. A medida que he ido programando me he ido dando cuenta de cuán débil es la estructura de una aplicación cuando ocurren esos imprevistos puntuales que te hacen replantearte tu vida desde el inicio de los tiempos.

Programar como un loco siempre ha sido mi hábito porque es más divertido que pararse a pensar y analizar la estructura de un programa, pensar en casos de error, cómo mejorar la estructura, plantearse si hay formas más eficientes o si voy a tener buenos hábitos a la hora de programar.

6.3. Objetivos no alcanzados.

No he conseguido crear un panel de administrador donde pueda gestionar los usuarios, ni las reservas de las habitaciones ni servicios por falta de tiempo. Tampoco he conseguido implementar un panel de comentarios de usuarios en las habitaciones ni valoraciones.