DIARIO DE DESARROLLO

JavaBNB

**Alberto José Castillo Toma & Cristina Martínez Toledo**

**Github Alberto:** [**https://github.com/AlbeTroll**](https://github.com/AlbeTroll)

**Github Cristina:** [**https://github.com/Cris30158**](https://github.com/Cris30158)

**Documentación | Día 1: 14/04/2024 |**

Hoy hemos comenzado este nuevo proyecto, llamado “JavaBNB”, el cual trata sobre crear una aplicación parecida al “AirBNB” que conocemos ya, que permite el alquiler de inmuebles vacacionales.

Para ello, seguiremos unos requisitos definidos en un pdf llamado “PL POO 2024”.

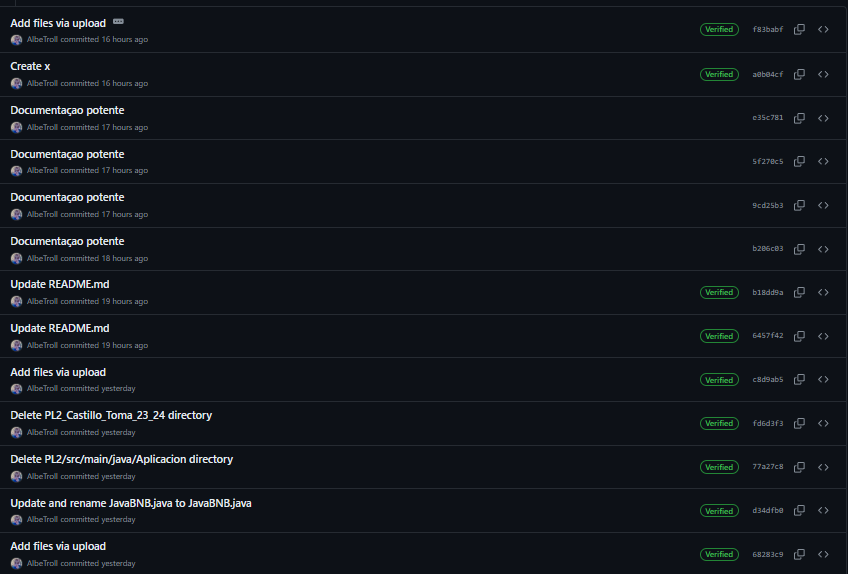
Este día creamos las clases “Administrador, Anfitrion, Cliente, DatosInmueble, Direccion, Inmueble, Main (main class), Particular, Tarjeta”

En cada clase hemos definido los atributos, el constructor, y algún que otro método, pero nada importante aún.

Además, creamos una nueva clase llamada JavaBNB, donde vamos a incorporar todas las funcionalidades relacionadas con la aplicación como tal. Por ejemplo: un método para buscar inmuebles disponibles según la ciudad y fechas de entrada y salida, otro método para ordenar los inmuebles por precio de menor a mayor, otro para calcular precio en total…

Ha sido día de inicialización, más adelante nos podremos con las funcionalidades.

**Imágenes | Día 1: 14/04/2024 |**



**Documentación | Día 2: 15/04/2024 |**

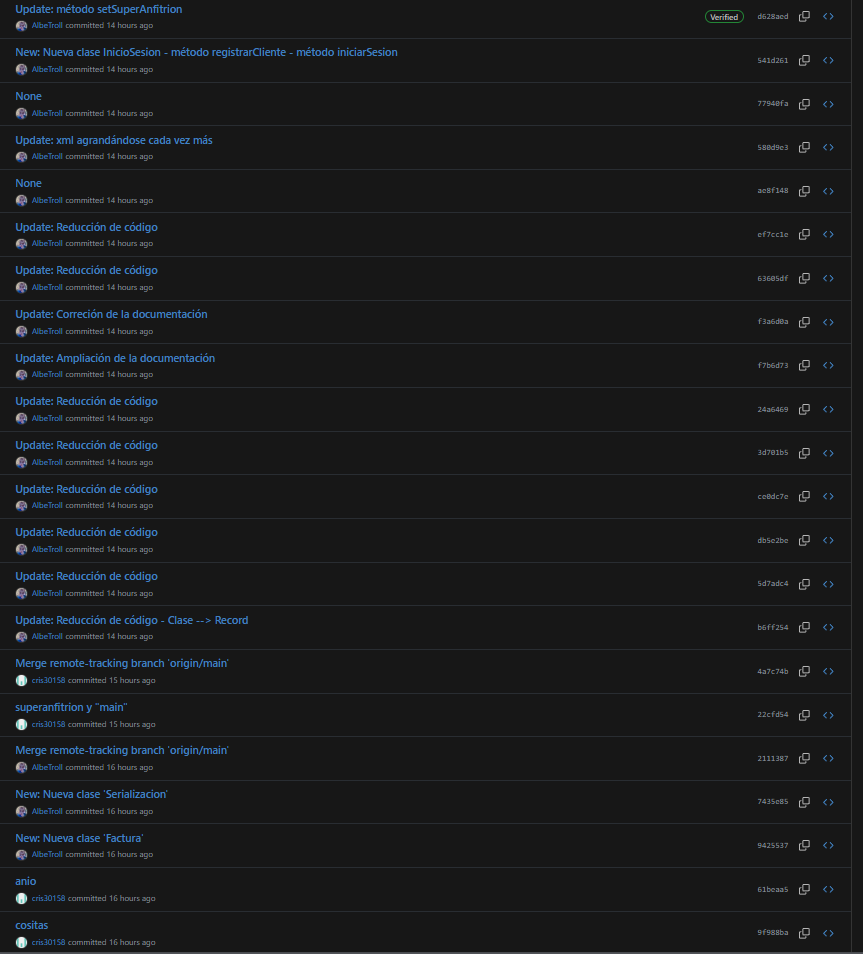
Hoy ha sido un buen día, ya que hemos avanzado bastante.

Lo primero que hemos hecho ha sido hacer una documentación más profesional, con el objetivo de poder generar un ‘Javadoc’ más adelante sin mucho problema. En vez de decir solamente qué hace cada bloque, hemos indicado los parámetros y qué se devuelve (si se devuelve algo).

En cuanto a las clases, hemos creado tres nuevas clases: “Factura (para generar una factura más adelante), Serializacion (con el objetivo de no perder los datos al cerrar la aplicación), InicioSesion (qué incluye los métodos registrarCliente e iniciarSesion)”

Además, hemos arreglado toda la documentación y hemos reducido algo de código que no hacía falta, como por ejemplo los comentarios que se añaden al crear una nueva clase en Java, o los setters en la clase ‘Administrador’ (ya que el usuario y la clave los hemos hecho finales, porque no van a cambiar).

Hemos agregado funcionalidades en la clase ‘Anfitrion’: método isSuperAnfitrion(), addInmuebles().

**Imágenes | Día 2: 15/04/2024 |**

**Documentación | Día 3: 16/04/2024 |**

Hoy ha sido día de dejar todo un poco ordenado y con un lavado de cara, ya que hemos seguido arreglando la documentación y los nombres de las variables, métodos…

Alguno de esos arreglos ha sido eliminar ‘imports’ que no utilizamos de momento, corregir mayúsculas y minúsculas (para seguir la sintaxis apropiada)...

Hemos creado un nuevo paquete llamado UI/UX, en el que irá todo lo relacionado con el diseño de interfaces realizado con Swing.

Hemos movido la clase Factura al paquete UI/UX, ya que nuestra primera idea es crear esas facturas utilizando Swing.

Los arreglos comentados anteriormente se pueden ver en el método gestionarReservas(), el método setSuperAnfitrión() [mayúsculas y minúsculas], documentación relacionada con los constructores…

En la clase ‘InicioSesion’, hemos:

* Añadido 3 nuevos atributos (Particular particular, ArrayList<Anfitrion> anfitriones, ArrayList<Particular> particulares)
* Arreglado la documentación del constructor.
* Añadido los métodos comprobarExistenciaCliente() [para ver si ya está registrado], registrarAnfitrión(), registrarParticular().
* Eliminado el método registrarCliente() [ya que hemos preferido añadir los tipos de clientes, para ser más preciso].

En la clase inmueble:

* setCalificacion() controla la excepción ‘IllegalArgumentException’ si no se cumple que 0 <= calificacion <= 5.

En la clase JavaBNB:

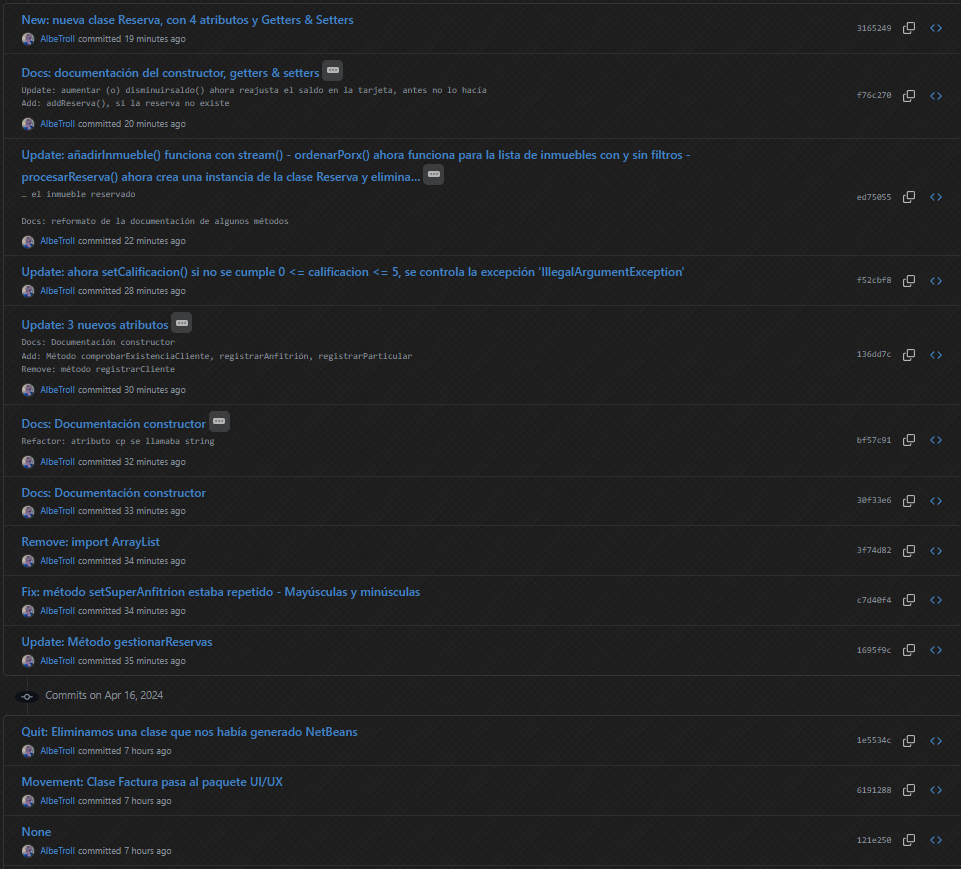
* Update:
* añadirInmueble() funciona con stream().
* ordenarPorx() ahora funciona para la lista de inmuebles con y sin filtros.
* procesarReserva() ahora crea una instancia de la clase Reserva y elimina el inmueble reservado.
* Docs:
* Reformato de la documentación de algunos métodos.

En la clase Particular:

* Docs: documentación del constructor, getters & setters.
* Update: aumentar (o) disminuirsaldo() ahora reajusta el saldo en la tarjeta, antes no lo hacía.
* Add: addReserva(), si la reserva no existe.

Por último, hemos creado una nueva clase Reserva, que tiene 4 atributos (Inmueble inmueble, Particular particular, LocalDate fechaInicio, LocalDate fechaFin), y sus correspondientes Getters & Setters.

**Imágenes | Día 3: 16/04/2024 |**



**Documentación | Día 4: 17/04/2024 |**

**Imágenes | Día 4: 17/04/2024 |**

**Documentación | Día 4: 17/04/2024 |**

**Imágenes | Día 4: 17/04/2024 |**

**Documentación | Día 5: 18/04/2024 |**

**Imágenes | Día 5: 18/04/2024 |**

**Documentación | Día 6: 19/04/2024 |**

**Imágenes | Día 6: 19/04/2024 |**