HACKATHON

VUELING

Travel search- Backend Challenge

Índex

[1 INFO DE LA HACKATHON 3](#_Toc126936671)

[2 X HACER 4](#_Toc126936672)

[2.1 INVESTIGAR 4](#_Toc126936673)

[2.2 CONSIDERACIONES DURANTE LA EJECUCIÓN 4](#_Toc126936674)

[3 OBJETIVOS 5](#_Toc126936675)

[3.1 PRIMARIOS 5](#_Toc126936676)

[4 DISEÑO 6](#_Toc126936677)

[4.1 VIEW 7](#_Toc126936678)

[4.1.1 Diseño general de las pantallas 8](#_Toc126936679)

[4.1.2 Esquema de las pantallas 9](#_Toc126936680)

[4.1.3 Welcome 10](#_Toc126936681)

[4.1.4 Main Menu 10](#_Toc126936682)

[4.1.5 Search trip 10](#_Toc126936683)

[4.1.6 Show Results 11](#_Toc126936684)

[4.1.7 Configuration 11](#_Toc126936685)

[4.1.8 App Configuration 12](#_Toc126936686)

[4.1.9 Add Hotel 12](#_Toc126936687)

[4.1.10 Add Flight 13](#_Toc126936688)

[4.1.11 Add City 13](#_Toc126936689)

[4.1.12 Add Trip 13](#_Toc126936690)

[4.1.13 Exit 14](#_Toc126936691)

[4.2 BBDD 16](#_Toc126936692)

[4.2.1 Tables 17](#_Toc126936693)

[4.2.2 Views 20](#_Toc126936694)

[4.2.3 User Manage 21](#_Toc126936695)

[4.3 JAVA ARCHITECTURE 22](#_Toc126936696)

[4.3.1 Consideraciones 22](#_Toc126936697)

[4.3.2 Estrucutras de datos 22](#_Toc126936698)

[4.3.3 Clases necearias 25](#_Toc126936699)

[4.3.4 Esquema 27](#_Toc126936700)

[4.4 Temas para hacer 28](#_Toc126936701)

# INFO DE LA HACKATHON

* Entrega: 8-2-2023
* sdf

# X HACER

## INVESTIGAR

* ¿Cómo gestiona JAVA caracteres árabes i chinos?
* ¿Las BBDD aceptan caracteres árabes i chinos?
* “TERMINAL”
  + Imprimir en la misma línea de la terminal => OK
  + Marcar la línea de entrada de datos => false
* ¿Cómo buscar mientras entran información?
  + ¿Threds?
  + No se puede
* Gestión de idiomas
  + ¿Tiene Java la opción de Android Studio de gestión de idiomas?
* sdf

## CONSIDERACIONES DURANTE LA EJECUCIÓN

* Crear excepciones propias
* sdf

# OBJETIVOS

## PRIMARIOS

Los objetivos que piden en la hackathon

* REQUERIMIENTOS FUNCIONALES
  + Para realizar la búsqueda se necesitan mínimo 3 caracteres
  + Hay 2 tipos de viajes
    - “land trip”
    - “air trip”
  + Debe aceptar caracteres árabes y asiáticos
  + El resultado debe devolver los siguientes campos
    - Trip name
    - Type of trip
    - Duration
    - Specific details
      * For hotels
        + Name
        + Category
      * For flights
        + Deperture city
        + Time
* TECHINCAL REQUERIMENTS
  + El sistema está creciendo y las restricciones pueden cambiar
  + El sistema debe poder agregar nuevas funcionalidades
  + No utilizar frameworks como Symphonu, Larabel, etc…
  + El formato de salida…

Con una búsqueda por <<Mad>>…

.- Europe Popular Capitals; Land Trip; 3 days; Madrid, Paris, London; Hotel Best Madrid 3\*, Hotel Worst Paris 4\*

.- Europe laying out cities; Land Trip; 3 days; Berlín, Madrid; Hotel Best Madrid 3\*, Hotel Worst Berlin 4\*

.- Infinite Egipt; Air Trip; 4 days; Madrid, القاهرة‎, الأقصر‎; Madrid 09:00, الأقصر‎ 10:00

.- Incredible China; Air Trip; 10 dies; Barcelona, Madrid, 广州市, 北京市; Madrid 09:00, 北京市 10:00

# DISEÑO

**¡NOTA!** Debido a la Info recibida en el e-mail de NUWE, diciendo que no es necesario la utilización de BBDD, se trabajará como si estuviesen, pero la Info se guardara en unas clases que simularan las tablas.

Tres partes en el diseño…

* BBDD
* JAVA ARCHITECTURE
* VIEW

## VIEW

Empezamos por el diseño de la VIEW.

Las pantallas necesarias en la versión inicial són…

* Bienvenida
* Menú principal
  + Muestra las opciones principales
    1. Búsqueda
    2. Configuración
    3. Salida
* Búsqueda
  + Donde el cliente entra que quiere buscar
* Salida resultados
  + Donde se muestran los resultados de la búsqueda
* Configuración
  + Muestra el menú de configuración, para poder agregar nuevas funcionalidades
  + Opciones…
    1. Add hotel
    2. Add Flight
    3. Add city
    4. Add Type of trip
       - Solo si hay tiempo
    5. Add Trip
* Salida
  + Para no acabar el programa
  + Mensaje de despedida
* Bienvenida
  + Se muestra durante unos pocos segundos
  + Mientras carga información de BBDD
* Menú principal
  + Muestra las opciones principales
    1. Búsqueda
    2. Configuración
    3. Salida
* Búsqueda
  + Donde el cliente entra que quiere buscar
* Salida resultados
  + Donde se muestran los resultados de la búsqueda
* Configuración
  + Muestra el menú de configuración, para poder agregar nuevas funcionalidades
  + Opciones…
    1. Add hotel
    2. Add Flight
    3. Add city
    4. Add Type of trip
       - Solo si hay tiempo
    5. Add Trip
* Error
  + Para mostrar que ha habido algún error y la solución
* Salida
  + Para no acabar el programa
  + Mensaje de despedida

### Diseño general de las pantallas

Todas las pantallas tendrán el siguiente diseño

* **TÍTULO**
  + Línea de algún carácter especial
  + Línea con el título de la pantalla y al final la fecha y la hora
  + El texto estará en mayusculas
* Línea de separación
* **CUERPO**
  + Donde se muestra la información de la pantalla
* Línea de separación
* **ENTRADA DE DATOS**
  + Línea pidiendo al usuario que debe entrar

### Esquema de las pantallas

El esquema del flujo del programa por las diferentes pantallas…

Diagrama

Descripción generada automáticamente

Tendremos 2 partes diferenciadas, una para realizar las búsquedas y otra para poder añadir nuevos viajes al programa.

### Welcome

* La primera que se muestra
* Unos pocos segundos
* ¿Se aprovecha mientras se carga algo de información?
* Mensaje…
  + “Welcome OTA travel agency”
  + Puede haber una primera línea con Welcome en diferentes idiomas y una segunda línea con el resto
* Calcular el tiempo necesario para leer el contenido y será el tiempo que se mostrará

### Main Menu

* **TITULO**
  + OPA MAIN MENU
* **CUERPO**
  + Muestra las siguientes opciones

1. Search trip

Salta a la pantalla “Search Trip”

1. Configuration

Salta a la pantalla “Configuration”

1. Exit

Salta a la pantalla “Exit”

* **ENTRADA DE DATOS**
  + Please, select an option
* **CONSIDERACIONES**
  + En el caso que se entre un valor incorrecto…
    1. Salta a la pantalla erro
    2. Con tipo 1
* sdf

### Search trip

* **TITULO**
  + Search trip
* **CUERPO**
  + Sin cuerpo
* **ENTRADA DE DATOS**
  + Mensaje
    1. Enter a city destination (0 to back)
* **CONSIDERACIONES**

### Show Results

* **TITULO**
  + Viajes encontrados
* **CUERPO**
  + El resultado de la busqueda
* **ENTRADA DE DATOS**
  + Función pulsar cualquier tecla para continuar
  + SDF
* **CONSIDERACIONES**
  + No hay resultados
    1. Mostrar mensaje en la línea siguiente
       - No matches for this city
    2. Mensaje push any key
    3. Vuelve a la pantalla de busqueda

### Configuration

* **TITULO**
  + APP CONFIGURATION
* **CUERPO**
  + Muestra las siguientes opciones…

1. ~~App Configuration~~
2. Add Hotel
3. Add Flight
4. Add City
5. ~~Add Type of trip~~
6. Add Trip
7. Back
   * SDF

* **ENTRADA DE DATOS**
  + Mensaje
    1. Please, select an option
* **CONSIDERACIONES**
  + En el caso que entre un valor incorrecto
    1. Salta a la pantalla de error

### App Configuration

* **TITULO**
  + sdf
* **CUERPO**
  + sdf
* **ENTRADA DE DATOS**
  + sdf

### Add Hotel

* **TITULO**
  + CONFIGURATION (Add Hotel)
* **CUERPO**
  + Sin cuerpo
* **ENTRADA DE DATOS**
  + Tiene 2 pasos

1. Entrar el nombre del hotel

“Please enter hotel name”

1. Seleccionar una categoría

“Please select hotel’s category (1 \*, 2 \*\*, 3 \*\*\*, 4\*\*\*\*, 5 \*\*\*\*\*, 6 \*\*\*\*\*LUX)

Se puede hacer una pantalla propia si hay tiempo

1. Selecciona la ciudad del hotel
2. Mensaje si todo bien

“Hotel ‘%s %s’ added correctly” nombre y categoría

* **CONSIDERACIONES**
  + Ya existe el hotel
    1. Salto a la pantalla error tipo 3
  + Para salir sin guardar entrar 0 sin nada más

### Add Flight

* **TITULO**
  + CONFIGURATION (Add Flight)
* **CUERPO**
  + Tiene 2 opciones
    1. Selección de ciudad
    2. Entrada de la hora de salida
* **ENTRADA DE DATOS**
  + Tiene 2 pasos

1. Selección ciudad de salida
2. Selección departure time
3. sdf

* **CONSIDERACIONES**
  + Ya existe el hotel
    1. Salto a la pantalla error tipo 4 vuelo ya existe
  + Para salir sin guardar entrar 0 sin nada más
* sdf

### Add City

* **TITULO**
  + CONFIGURATION (Add City)
* **CUERPO**
  + Tiene 2 opciones

1. Sdf
2. Sdf
   * sdf

* **ENTRADA DE DATOS**
  + Tiene 2 pasos

1. Entrada de nombre de la ciudad

“Enter acity’s name”

1. Mensaje si todo ha ido bien

“New city to travel”

* **CONSIDERACIONES**
  + sdf

### Add Trip

* **TITULO**
  + CONFIGURATION (Add Trip)
* **CUERPO**
  + En la parte superior se va mostrando el viaje
  + Tiene diferentes pasos
    1. Entrada de nombre

Sin mostrar nada

* + 1. Tipo de viaje

Como si fuera un menú

* + 1. Duración del viaje
    2. Selección de ciudades

Lista de ciudades en forma de menú

* + 1. Si es land trip
       - Pide para añadir hoteles a la lista
    2. Si es Air Trip
       - Pide para añadir ciudades de salida
    3. Final
* **ENTRADA DE DATOS**
  + Tiene varios pasos

1. Entrada nombre

“Please, enter Trip name”

1. Selección tipo de viaje

“Select the Trip Type”

1. Duración del viaje

“Enter the days of trip”

1. Selección de ciudades

“Select city”

1. Si es Land Trip
   1. Selección de hoteles de cada ciudad
2. Si es Air Trip
   1. Selección de vuelos
3. Final

“¿Es correcto el viaje?”

* + Sdf
* **CONSIDERACIONES**
  + Para salir sin guardar entrar 0 sin nada más

### Exit

* **TITULO**
  + OPA travel agency
* **CUERPO**
  + Mensaje
    1. Thanks for your visit
* **ENTRADA DE DATOS**
  + Sin salida de datos

## BBDD

Información que se debe guardar en la BBDD…

* Tipo de viaje
* Nombre de los viajes
* Duración de los viajes
* Nombres de las ciudades
* Vuelos
* Horarios de salida
* Nombres de los hoteles
* Categorías
* sdf

### Tables

Tablas necesarias…

#### AIR TRIPS

* Id
  + Short
* Name
  + String
* Days
  + Short

#### CITIES VISITED – LAND TRIP

* Id Land Trip
  + Id Land Trip Table
  + Short
* Id City
  + Id Air Trip Table
  + Short

#### CITIES

* Id
  + Short
* Nombre
  + String

#### FLIGHTS

* Id
  + Short
* Id City Departure
  + Id Cities Table
  + Short
* Time Departure
  + Time

#### FLIGHTS AIR TRIP

* Id Air Trip
  + Id Air Trip Table
  + Short
* Id Flight
  + Id Flights Table
  + Short

#### LAND TRIP

* Id
  + Short
* Trip Name
  + String
* Days
  + Short

#### HOTEL

* Id
  + Short
* Id City
  + Short
* Name
  + String
* Id Categoría
  + Byte

#### HOTELS – LAND TRIP

* Id Land Trip
  + Short
* Id Hotel
  + Short

#### CITIES VISITE – AIR TRIP

* Id Air Trip
  + Id Air Trip Table
  + Short
* Id ciudad
  + Id tabla Ciudades
  + Short

#### TIPO VIAJES

Esto no hace falta que esté en la BBDD.

* Id
  + Byte
* Nombre
  + String

### Views

Las vistas que abran son…

* Flight Complet
  + Para obtener la información de las tablas flight y ciudad
* Hotel Complet
  + Para obtener la información de las tablas hotel y ciudad
* Air Trip Cities
  + Para obtener la lista de las ciudades de cada Air Trip
* Air Trip Flights
  + Para obtener la lista de los vuelos de cada Air Trip
* Land Trip Cities
  + Para obtener la lista de ciudades de cada Land Trip
* Land Trip Hotels
  + Para obtener la lista de los hoteles de cada Land Trip

### User Manage

* Habrá 2 tipos de usuario, básico y administrador
  + Básico
    - Solo podrá realizar consultas
    - Name: opabasic
    - Pass: OPABaUnoDosTres
  + Administrador
    - Podrá tanto leer como escribir
    - Name: opaadministrador
    - Pass: OPAAdCincoSeis

## JAVA ARCHITECTURE

Con el diseño de la VIEW y de la BBDD, toca realizar el diseño de la arquitectura del programa

### Consideraciones

* La estructura del programa seguirá el pattern MVC (Model View Control)
* Se utilizarán patterns adecuados para el diseño de las clases
* Las diferentes capas del programa se dividen en 3 partes diferentes…
  + Search

Estará en esta parte, todo lo relacionado con búsqueda

* + Config

Estará en esta parte todo lo relacionado con la configuración

* + Others

Todo que no esté directamente en alguna de las otras dos partes, se pondrá en esta

De esta manera varios miembros del equipo puede trabajar en distintas partes a la vez.

* Sdf

### Estrucutras de datos

Las estructuras de datos necesarias son…

* TypeTrip
  + Id
    - Boolean
  + NOTA Será una enum
  + NOTA 2 false será Air Trip y true Land Trip
* Category
  + Id
    - Byte
  + NOTA Será una enum
* City
  + Id
    - Short
    - NOTA En el mundo hay 10.000 ciudades aprox. [link](https://proyectoviajero.com/cuantas-ciudades-mundo/)
  + Name
    - String
* Hotel
  + Id
    - Int
  + Name
    - String
  + cityName
    - String
  + idCategory
    - byte
* Flight
  + Id
    - Short
  + cityDeparture
    - String
  + departureTime
    - Time
* LandTrip
  + Id
    - Short
  + name
    - String
  + tripDays
    - Short
    - NOTA Por si un loco quiere un viaje de 1 año
  + citiesVisited
    - List<Citiy>
  + Hotels
    - List<Hotel>
* AirTrip
  + Id
    - Short
  + name
    - String
  + tripDays
    - short
  + flights
    - List<Flight>
  + citiesVisited
    - List<City>

### Clases necearias

Primera definición de las clases necesarias divididas por capas

#### Capa MODEL

* ESTRUCUTRA DE DATOS
  + AirTrip
  + Citiy
  + Flight
  + LandTrip
  + Hotel
* SearchCls

Esta clase se encargará de controlar la parte de búsqueda de viajes con el nombre de la ciudad que haya introducido el usuario.

Los métodos necesarios son

* + Check trips with city name
  + Get air trips
  + Get land trips
* ConfigCls

En esta clase, se encargará de la gestión de añadir elementos a las diferentes tablas.

Los métodos necesarios en la clase son…

* + Add City
  + Add Hotel
  + Add Flight
  + Add Air Trip
  + Add Land Trip
* ToolsCls

Los métodos que puedan ser utilizados en más de una clase

* + Check city exist
* SDF

#### Capa VIEW

La VIEW tendrá

* Main
* SearchCls
* ConfigCls
* OthersCls
* sdf

#### Capa CONTROL

Habrá una clase para cada una de las diferentes partes en que se ha dividido el programa

* SearchCls
* ConfigCls

### Esquema

Una primera idea del esquema de clases…

Diagrama

Descripción generada automáticamente

Las líneas representan des de que clase se puede acceder a la clase.

Los package ‘EnumPckg’ y ‘DataStrucutrePckg’ todo y estar en el package ‘ModelPckg’ serán accesibles des de todas las clases del programa.

## Temas para hacer

* sdf