Documento de solución técnica

weather-app

Mercado Libre

|  |  |
| --- | --- |
| Proyecto | WEATHER-APP |
| Realizado Por | Damián Campana |
| Fecha de Generación | 24/09/2019 |
| Versión | 1.0 |

Tabla de contenido

[Objetivos 3](#_Toc20310176)

[Descripción de la solución. 4](#_Toc20310177)

[1 – Descripción del desarrollo 4](#_Toc20310178)

[2 – Representación de los planetas. 4](#_Toc20310179)

[3 – Repositorio DAO y DTO. 4](#_Toc20310180)

[4 – Capa de servicios. 4](#_Toc20310181)

[5 – Capa de Controladores. 5](#_Toc20310182)

[6 – Jobs y Clase Init. 5](#_Toc20310183)

Documento de solución técnica

# Objetivos

El objetivo de este documento es explicar paso a paso la solución técnica tomada para el desarrollo de la aplicación.

# Descripción de la solución.

**Tecnologías y herramientas**:

* Base de datos MySql.
* Java 8.
* Spring boot.
* Maven.
* Heroku.

## 1 – Descripción del desarrollo

Se consideró realizar un aplicativo web utilizando los siguientes patrones:

MVC, DTO, DAO, BUILDER, STRATEGY.

## 2 – Representación de los planetas.

Para la creación de los planetas se decidió utilizar el patrón creacional Builder. Generando Planet Builder y para cada planeta su concreteBuilder, obteniendo como producto final un Planeta con sus atributos ya definidos. Velocidad, distancia y position.

Para la posición de los planetas, se creó una clase Point, representando los puntos de los ejes cartesianos.

El sol se representó por una clase llamada SUN, con sus propios puntos.

Adicionalmente se creó una clase Prediction para representar la predicción con los

siguientes atributos:

- Día: Indica el día del pronostico

- Perímetro: Perímetro que forman los planetas

- Estado: LLUVIA, NORMAL, SEQUIA, CONDICIONES\_OPTIMAS

## 3 – Repositorio DAO y DTO.

Una vez realizado el modelo, se procedió a realizar un repositorio utilizando JPA y su interface JpaRepository. También se confeccionó un Dao custom para poder realizar otras operaciones adicionales a las definidas por JpaRepository.

Se agregó una capa DTO para la transferencia de datos hacia los servicios rest y la web.

## 4 – Capa de servicios.

Se creó una capa de servicios para interactuar entre los controladores y los repositorios.

Adicionalmente utilizó el patrón strategy para el desarrollo de la solución, para ir creando la predicción correcta según las condiciones climáticas que se iban calculando.

Para la resolución de los cálculos, se fue tomando la posición de los planetas por el valor de los ejes y calculando el área para comprobar si formaban un triángulo. De ser así se calculó el perímetro del mismo, para comparar a que día de predicción correspondía y adicionalmente si se trataba de un día de LLUVIA persistir el valor del perímetro para obtener el día de máxima lluvia.

## 5 – Capa de Controladores.

En esta capa se creación dos controladores uno para los servicios rest y otro para la web.  
Generando dos servicios rest, uno para la obtención de la predicción por día, y otro para obtener el día de máxima lluvia.

## 6 – Jobs y Clase Init.

Para la carga de datos inicial, se calcularon la cantidad de días que hay en 10 años.

Toma como fecha inicial la fecha en que la aplicación es deployada.

Se creó un Jobs para que se ejecute todos los días por la noche para ir agregando los pronósticos por día.