

Solución Prueba técnica para desarrollador backend.

1. Estrategia de branching.



Para el desarrollo se optó por una variante de la estrategia **GitFlow**, la cual consiste en tener varias ramas (master, develop, feature) las cuales tiene las siguientes características:

- En la rama *develop* se llevara todo el desarrollo del código.
- En la rama *feature* se **crea** cuando se necesite desarrollar una nueva característica/microservicio/etc. Y cuando este ya esté terminado se unirá con la rama *develop* integrando la nueva funcionalidad y también se eliminará después de realizar este proceso. Esta rama se llamó según el modulo que se estuviera trabajando.
- La rama *master* se aloja todo el código de producción.

Se eligió esta ya que permite llevar un orden según cada funcionalidad.

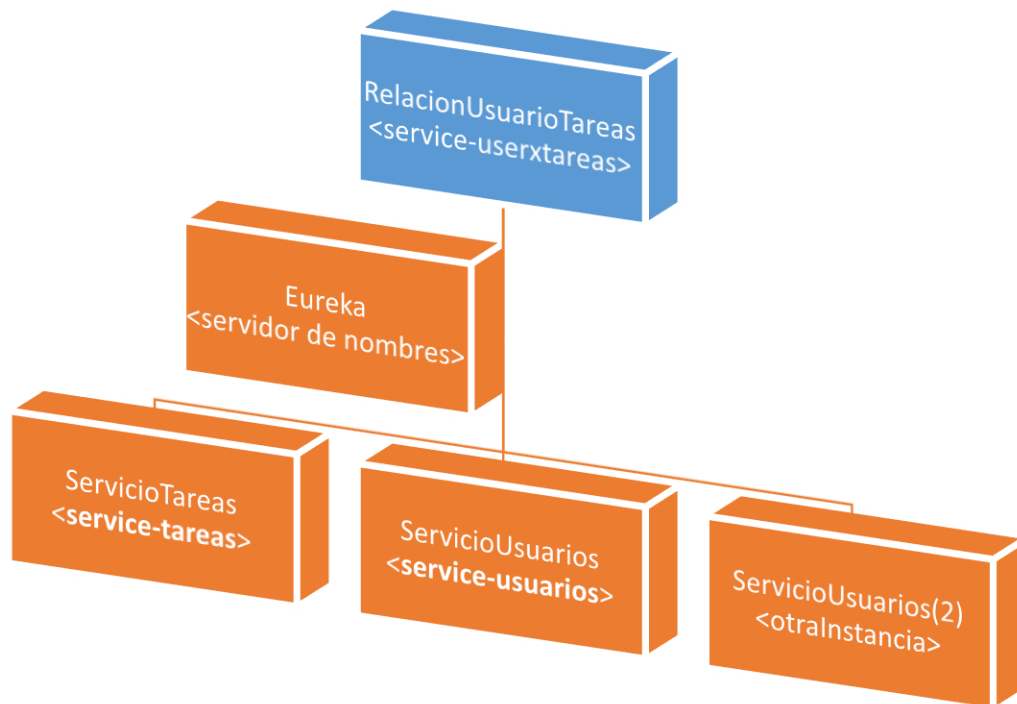
A continuación, se presenta el historial de la solución:

Autor	Registro	Mensaje
Emanuel henao giraldo	38098a2	MERGED Merge branch 'develop'
Emanuel henao giraldo	9cdc5f0	Docs[]: creacion de Swagger
Emanuel henao giraldo	cb19988	Feat[DEV-EU]:implemntacion de Eureka
Emanuel henao giraldo	cad5111	MERGED Merge branch 'feature/moduloUserTask' into develop
Emanuel henao giraldo	fc75b7c	Feat[DEV-M3] EndPoint relaciona Tareas con usuarios
Emanuel henao giraldo	707efe7	Feat[DEV-M3]: moduloUserTask
Emanuel henao giraldo	85478e1	MERGED Merge branch 'feature/moduloTarea' into develop
Emanuel henao giraldo	da5cae6	Feat[DEV-M2] se agrego un filtro por idUsuario
Emanuel henao giraldo	8ec2ce9	Feat[DEV-M2] modulo de tareas
Emanuel henao giraldo	e93944c	MERGED Merge branch 'feature/moduloUsuario' into develop
Emanuel henao giraldo	f4ecb40	Feat[DEV-M1]: crud servicio
Emanuel henao giraldo	411a0ab	Feat[DEV-M1]: modulo servicio usuario
Emanuel henao giraldo	e310f70	MERGED Merge branch 'develop'
Emanuel henao giraldo	782aa73	Feat[INIT]: initialMerge
Emanuel henao giraldo	5ed9490	Feat[DEV-INIT]: Inicio de Solucion
Emanuel henao giraldo	e886ab3	Initial commit

Nota: cada uno de los mensajes de los commits tiene un formato de envío según que se esté modificando, por ejemplo, para la documentación se utilizó “Docs”.

2. Arquitectura

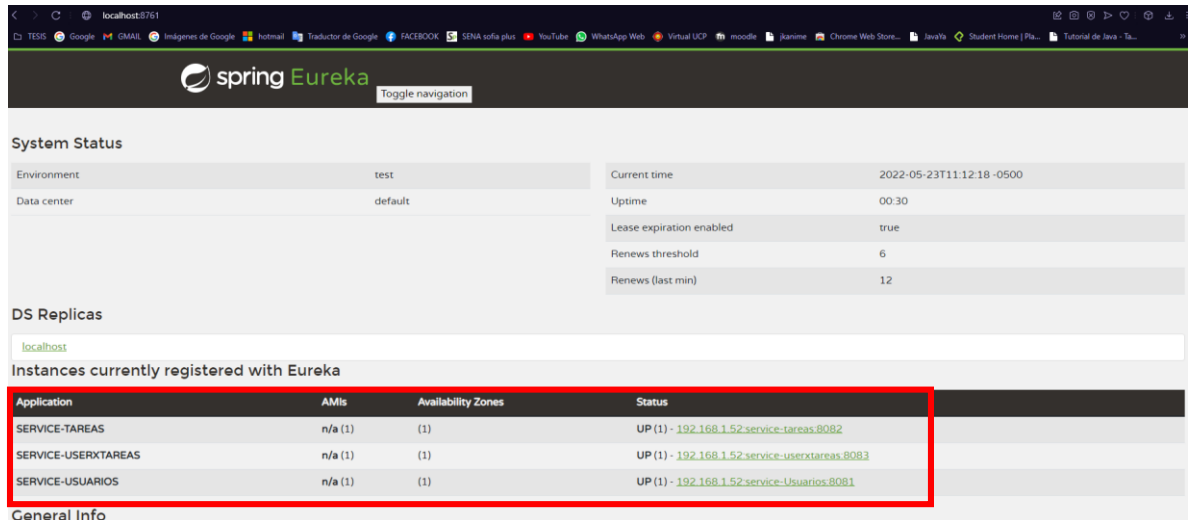
Para la solución se planteó la siguiente arquitectura:



Se crearon **3 microservicios** los cuales tiene bases de datos separadas y interactúan entre ellos:

- **servicioTareas:** Se encarga de gestionar todas las solicitudes respecto a *tareas*, tiene varios endpoints que hacen referencia a CRUD y algunos filtros de búsqueda.
- **servicioUsuarios:** Se encarga de gestionar todas las solicitudes respecto a *Usuarios*, tiene varios endpoints que hacen referencia a CRUD y algunos filtros de búsqueda.
- **servicioRelacionUsuarioTareas:** Posee un único endpoint que se encarga de relacionar a cada uno de los usuarios con sus tareas, para esto les realiza solicitudes a los otros dos servicios anteriores.

- **ServidorEureka:** Es un servidor de nombres, que permite relacionar la dirección con el nombre que le dimos a cada servicio, para esto cuando cada uno de los servicios se despliega se registran en este servidor de nombres. Como se muestra a continuación.



The screenshot shows the Spring Eureka web interface. At the top, there's a navigation bar with the Spring Eureka logo and a 'Toggle navigation' button. Below this, the 'System Status' section displays various system metrics in a table format. The 'DS Replicas' section shows the current state of the data center. The 'Instances currently registered with Eureka' section is highlighted with a red box and contains a table listing registered services.

Application	AMIs	Availability Zones	Status
SERVICE-TAREAS	n/a (1)	(1)	UP (1) - 192.168.1.52-service-tareas.8082
SERVICE-USERTAREAS	n/a (1)	(1)	UP (1) - 192.168.1.52-service-userxtareas.8083
SERVICE-USUARIOS	n/a (1)	(1)	UP (1) - 192.168.1.52-service-Usuarios.8081

Below the table, there's a 'General Info' section.

Este servidor nos otorga facilidad al momento de:

- Levantar varias instancias de un mismo servicio en diferente puerto, ya que este nos agrupa según el servicio.
- consumir otros servicios, ya que no se debe cambiar de dirección de consumo cada vez que se levante unas nuevas instancias de un mismo servicio.
- Balancear la carga según la demanda del servicio.

3. Datos

A nivel de datos se utilizó JPA para persistir los datos, así como también JpaRepository para facilitar las consultas.

Es importante resaltar que cada uno de los servicios tiene su propio BD en memoria, para lo cual se utilizó H2.

Las entidades creadas fueron:

- Para el ServiceTarea

```
Tarea {
  categoria      string
  completa       boolean
  descripcion     string
  fechaCreacion  string($date)
  id             integer($int64)
  idUsuario      integer($int64)
  tiempoEstimadoMin integer($int32)
}
```

- Para el ServiceUsuario

```
Usuario {
  apellido      string
  cargo         string
  fechaNacimiento string($date)
  id            integer($int64)
  nombre       string
  telefono     string
}
```

Consola de H2

User: ema

contraseña:123

English Preferences Tools Help

Login

Saved Settings: Generic H2 (Embedded) ▾

Setting Name: Generic H2 (Embedded) Save Remove

Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:mem:prueba

User Name: ema

Password: ***

Connect Test Connection

Datos por Default

- Para el ServiceUsuario

The screenshot shows a web application running on localhost:8081. The browser address bar displays 'localhost:8081/app/h2-console/login.do'. The application interface includes a sidebar with a tree view showing the database structure: 'jdbc:h2:mem:prueba' > 'USUARIO' > 'ID_USUARIO', 'APELLIDO', 'CARGO', 'FECHA_NACIMIENTO', 'NOMBRE', 'TELEFONO', 'Indexes', 'INFORMATION_SCHEMA', 'Sequences', 'Users', and 'H2 2.1.212 (2022-04-09)'. The main area displays the SQL statement 'SELECT * FROM USUARIO;' and its results in a table. The table has 6 columns: 'ID_USUARIO', 'APELLIDO', 'CARGO', 'FECHA_NACIMIENTO', 'NOMBRE', and 'TELEFONO'. There are 5 rows of data. Below the table, it indicates '(5 rows, 4 ms)' and an 'Edit' button.

ID_USUARIO	APELLIDO	CARGO	FECHA_NACIMIENTO	NOMBRE	TELEFONO
1	Perez	Psicologo	1998-03-13	John	123456
2	london	Soporte	1970-12-08	Marco	987654
3	lopez	Desarrollador	2000-09-08	Daniela	321654
4	ibarra	Frontend	1980-06-18	Marisol	147258
5	zarai	Backend	2005-01-01	Karen	369258

- Para el ServiceTarea

The screenshot shows the same web application running on localhost:8082. The browser address bar displays 'localhost:8082/app/h2-console/login.do'. The sidebar tree view shows: 'jdbc:h2:mem:prueba' > 'TAREA' > 'INFORMATION_SCHEMA', 'Sequences', 'Users', and 'H2 2.1.212 (2022-04-09)'. The main area displays the SQL statement 'SELECT * FROM TAREA;' and its results in a table. The table has 7 columns: 'ID_TAREA', 'CATEGORIA', 'DESCRIPCION', 'FECHA_CREACION', 'ID_USUARIO', 'IS_COMPLETA', and 'TIEMPO_ESTIMADO_MIN'. There are 6 rows of data. Below the table, it indicates '(6 rows, 2 ms)' and an 'Edit' button.

ID_TAREA	CATEGORIA	DESCRIPCION	FECHA_CREACION	ID_USUARIO	IS_COMPLETA	TIEMPO_ESTIMADO_MIN
1	Psicologo	Cita con antonio	2022-03-13	1	FALSE	30
2	Soporte	resolver dudas del usuario	2022-12-08	2	FALSE	30
3	Desarrollador	integrar al nuevo compañero con los demas	2022-09-08	3	FALSE	30
4	Frontend	realizar diseño de pagina web	2022-06-18	4	FALSE	300
5	Backend	hacer conexion con la bd test	2022-01-01	5	TRUE	30
6	Backend	Desarrollar Crud	2022-02-01	5	FALSE	60

4. Documentación

Para la documentación de cada Service se utilizó Swagger.

