

# Manual de instalación y despliegue del software del Sistema de información multiplataforma para el acompañamiento de padres en el proceso de posparto

---

A continuación se dará información básica necesaria para el ejecutar de forma local y desplegar el software desarrollado en el proyecto. Esto se hará separando cada modulo del software por separado y especificando su despliegue individual.

## Backend

---

Este es un backend / Maven / Spring Boot (versión 2.7.15) aplicación que es un microservicio que contiene un REST API que responde a las necesidades de los frontend del software.

### Definiendo parámetros de aplicación

#### Base de datos

Este proyecto usa un servicio de datos MySQL, donde para definir el tipo de base de datos específico (Postgres, MySQL Server, etc) es necesario modificar el archivo pom.xml y añadir la dependencia requerida para su conexión.

Para el caso de MySQL Server se tiene lo siguiente:

```
<dependency>
  <groupId>com.mysql</groupId>
  <artifactId>mysql-connector-j</artifactId>
  <scope>runtime</scope>
</dependency>
```

Una vez hecho esto, se pueden definir los parámetros de conexión a la base de datos en el archivo `src/main/resources/application.properties`. El archivo debe lucir así:

```
spring.datasource.url={{url del servicio de base de datos}}
misdevsc_amacom?sessionVariables=sql_mode=''
spring.datasource.username={{nombre de usuario para acceder a la base de datos}}
spring.datasource.password={{contraseña para acceder a la base de datos}}

server.servlet.context-path=/api
spring.jpa.show-sql=true

#spring.datasource.driver-class-name=org.mariadb.jdbc.Driver

# ENABLE SQL DDL AUTO FOR DATABASE MIGRATIONS
spring.jpa.hibernate.ddl-auto=update
```

```
spring.jpa.properties.hibernate.format_sql=true
#spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MariaDBDialect
server.error.include-stacktrace=never

spring.datasource.hikari.pool-name=HikariPoolMariaDB
```

## Credenciales para envío de correos electrónicos

Para el envío de correos electrónicos usado en diferentes módulos de la aplicación se deben definir los parámetros de conexión al servicio de correo en el archivo

`src/main/resources/application.properties`. El archivo debe lucir así:

```
spring.mail.host={{smtp usado por el servicio}}
spring.mail.port={{puerto para el servicio de smtp}}
spring.mail.username={{nombre de usuario (correo)}}
spring.mail.password={{contraseña}}
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
spring.mail.properties.mail.smtp.starttls.required=true

spring.mvc.pathmatch.matching-strategy=ant_path_matcher
```

## Cómo ejecutar

Esta aplicación está empaquetada como un jar. Se ejecuta usando el comando ``java -jar``.

- Clona este repositorio
- Asegúrate de que estás usando Java 11 y Maven 3.x
- Puedes construir el proyecto y ejecutar las pruebas ejecutando `mvn clean package`.
- Una vez construido con éxito, puede ejecutar el servicio por uno de estos dos métodos:

```
$ java -jar -Dspring.config.location=src/main/resources/application.properties
{{ la ruta hacía el archivo jar generado}}
```

o

```
$ mvn spring-boot:run -
Drun.arguments="spring.config.location=src/main/resources/application.properties"
```

Una vez que la aplicación se ejecuta debería ver algo como esto

```
2024-02-28 14:05:06.747 INFO 14568 --- [ restartedMain]
.d.s.w.r.o.CachingOperationNameGenerator : Generating unique operation named:
updateUsingPUT_34
```

```
2024-02-28 14:05:06.760 INFO 14568 --- [ restartedMain]
.d.s.w.r.o.CachingOperationNameGenerator : Generating unique operation named:
findPageableUsingGET_24
2024-02-28 14:05:06.884 INFO 14568 --- [ restartedMain]
com.amacom.amacom.AmacomApplication      : Started AmacomApplication in 30.54
seconds (JVM running for 31.534)
```

## Acerca del servicio

El servicio es un simple servicio REST de revisión de hoteles. Utiliza una base de datos relacional como MySQL o PostgreSQL. Si tus propiedades de conexión a la base de datos funcionan, puedes llamar a algunos endpoints REST definidos en `com.amacom.amacom` en el **puerto 8080** bajo la ruta `/api`.

De forma detallada se tiene la documentación de cada ruta y subruta junto con información de cada endpoint en la siguiente ruta:

```
{{url del servicio backend}}/api/swagger-ui.html
```

## Aplicativo WEB

---

### 1. Introducción

Le damos la bienvenida al Manual de Despliegue, una guía esencial que ofrece instrucciones detalladas para la implementación efectiva y la puesta en marcha de la aplicación. Se aconseja revisar y preparar los requisitos previos necesarios, así como seguir cuidadosamente el procedimiento paso a paso proporcionado para asegurar una integración exitosa.

### 2. Requisitos Previos

Programas instalados:

#### Editor de código:

Se recomienda instalar [Visual Studio Code](#).

#### NodeJS y NPM (Node Package Manager)

Para la instalación de NodeJS se recomienda seguir la [guía oficial](#) y instalar una versión LTS (Long Term Support) y instalar el NPM en las opciones de instalación.

### 3. Configuraciones:

Primero

El siguiente paso es instalar la herramienta Angular CLI. Para llevar a cabo esta instalación de manera global en su sistema, abra la terminal y ejecute el comando que se proporciona a continuación:

```
$ npm install -g @angular/cli
```

Este procedimiento instalará la versión más reciente y estable de Angular CLI, garantizando su disponibilidad para su uso en todo el sistema.

## Segundo

Una vez que haya completado los pasos previos, estará en condiciones de acceder al [repositorio web de la aplicación](#).

Al ingresar al enlace, diríjase a la sección 'Código' y copie la URL proporcionada para poder clonar el proyecto en su sistema.

Después de copiar el enlace, el siguiente paso es crear una carpeta específica en nuestro sistema para alojar el proyecto clonado. Una vez ubicados en esta carpeta, abrimos una terminal y ejecutamos el comando git clone, seguido de la URL que hemos copiado anteriormente, para iniciar el proceso de clonación del proyecto:

```
$ git clone https://github.com/AmacomV2/amacom-web.git
```

Una vez finalizado el proceso de clonación del proyecto web se necesita viajar a la raíz del proyecto, donde es posible cambiar entre las diferentes ramas del proyecto. Para cambiar entre una rama u otra se ejecuta el comando:

```
$ git checkout [[desired-branch]]
```

## Tercero

Para compilar el proyecto en modo desarrollo, navegue hasta la siguiente ruta específica: `src\app` y ejecute alguno de los siguientes comandos:

```
src/app$ ng serve
```

```
src/app$ npm start
```

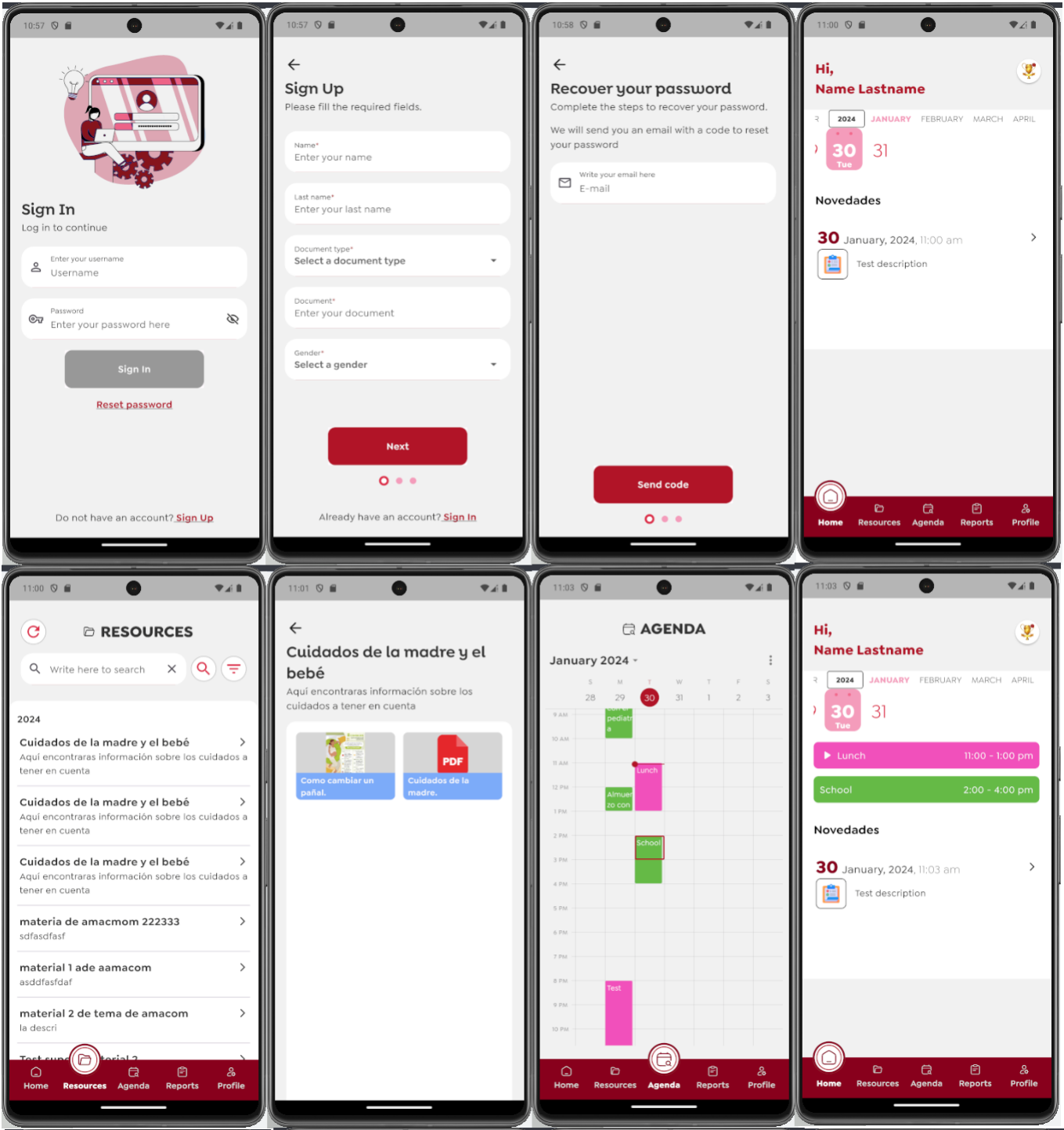
Este proceso creará un servidor web de desarrollo. Mientras el servidor esté en funcionamiento, abra el navegador de su preferencia y diríjase a la dirección del servidor local: `http://localhost:4200/`. Donde podrá visualizar e interactuar con la interfaz web de la aplicación.

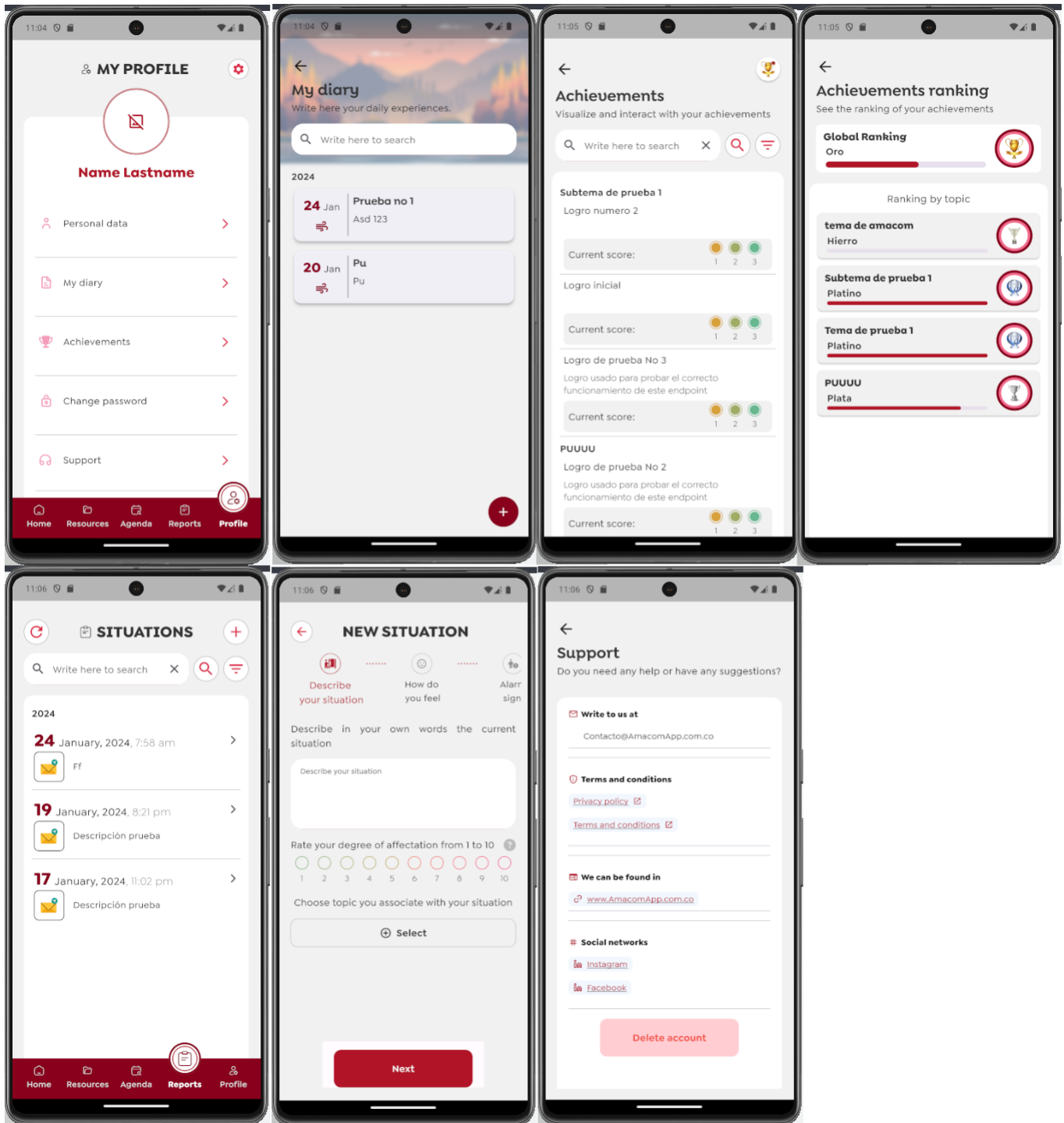
## Cuarto

Finalmente, si se quiere navegar o explorar el proyecto, es crucial comprender su estructura organizativa. La aplicación web se organiza de acuerdo con roles de usuario, los cuales incluye: 1. administrador, 2. enfermería y 3. usuario final, cada uno con su carpeta respectiva. Además, se han asignado carpetas específicas para componentes clave, como la 4. autenticación de usuarios y el 5. calendario, facilitando así la navegación y comprensión del aplicativo.

# Mobile APP

Una aplicación de salud hecha en Flutter.





## Primeros Pasos

Este proyecto es un punto de partida para una aplicación Flutter.

Algunos recursos para empezar si este es tu primer proyecto Flutter:

- [Lab: Write your first Flutter app](#)
- [Cookbook: Useful Flutter samples](#)

Si necesitas ayuda para empezar a desarrollar en Flutter, consulta la [documentación en línea](#), que ofrece tutoriales, ejemplos, orientación sobre desarrollo móvil y una referencia completa de la API.

### 1. Información de versión

- Flutter Version: 3.10.2
- FVM Version: 2.4.1

Asegúrese de utilizar las versiones especificadas de Flutter y FVM para garantizar la compatibilidad con este proyecto.

Para configurar Flutter con la versión correcta, puede utilizar FVM (Flutter Version Management). Siga los pasos que se indican a continuación:

1. Instala FVM ejecutando el siguiente comando en tu terminal:

```
$ dart pub global activate fvm
```

2. Inicializa FVM en tu proyecto ejecutando el siguiente comando en el directorio raíz de tu proyecto:

```
$ fvm init
```

3. Establece la versión de Flutter para este proyecto:

```
$ fvm use 3.10.2
```

Ahora ya tienes Flutter configurado con la versión correcta para este proyecto usando FVM. Puedes continuar con el desarrollo de Flutter para esta aplicación.

Para más información sobre FVM y su uso, consulta la [documentación de FVM](#).

## 2. Ejecuta la Aplicación en modo Development, Release or Profile

Para que la aplicación funcione sin problemas debes especificar la url HOST para enviar las peticiones HTTP. Para ello, debes definir en entorno de compilación de la aplicación. Si estás usando VSCode, añade o modifica el archivo `.vscode/launch.json` y para los modos de lanzamiento deseados añade las siguientes líneas con tu url HOST.

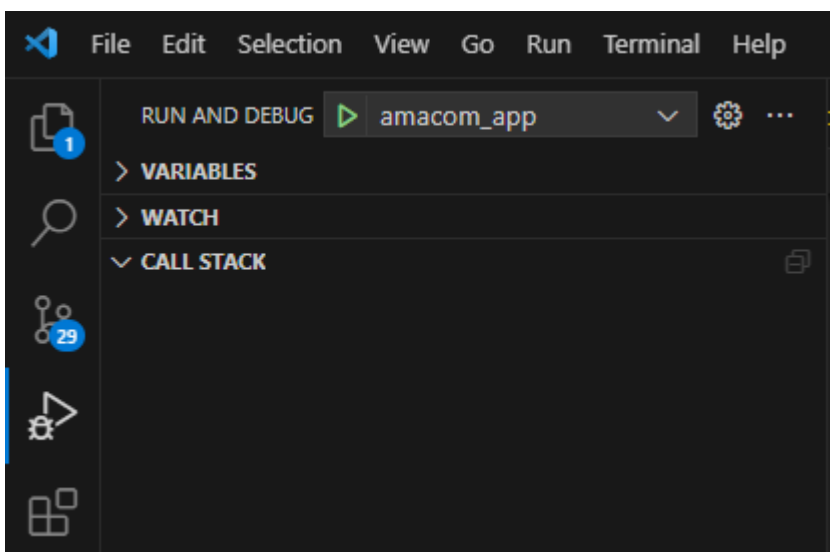
```
{
  "name": "amacom_app",
  "request": "launch",
  "type": "dart",
  "flutterMode": "{{modo deseado}}",
  "toolArgs": [
    "--dart-define",
    "HOST={{su url de host aquí}}",
  ],
  "args": [
    "--dart-define=HOST={{su url de host aquí}}",
  ]
}
```

```
],  
  "env": {  
    "HOST": "",  
  }  
}
```

Con esto ya puedes lanzar la aplicación sin problemas en tu dispositivo o emulador ejecutando el siguiente comando en el directorio raíz de la app:

```
$ flutter run
```

O simplemente lanzándola directamente desde VSCode



### 3. Compilar la aplicación para su distribución

Antes de cualquier operación debemos definir la url del HOST en la compilación de la aplicación. Si compila directamente desde el terminal, sólo tiene que añadir `--dart-define=HOST={{su url de host}}` en su comando de compilación.

Como ejemplo, aquí es cómo debe ser similar para android apk construir:

```
$ flutter build apk --release --dart-define=HOST={{su url de host}}
```

Para compilar la aplicación para su despliegue en plataformas soportadas por flutter debes seguir la guía oficial de flutter [online documentation for deployment](#).