uniMedic v0.1

Asistente Virtual Médico

Plan de Despliegue

Índice

1. Introducción

El propósito del presente documento es establecer cómo se llevará a cabo la instalación de la solución propuesta en su ámbito de producción.

1.1. Propósito

El software de consultas médicas permitirá a los usuarios resolver sus dudas sin la necesidad de concurrir a un centro médico especializado a menos que sea necesario. De esta manera el usuario se evita la exposición a situaciones de riesgo, o también como solución a la falta de tiempo.

1.2. Alcance

Este documento es de interés para el cliente y para los involucrados en el desarrollo y puesta en producción del producto. Se refiere a la primera puesta en marcha de la solución en determinadas situaciones a determinar por el cliente, es decir, en entornos reales.

2. Planificación del Despliegue

Se proveerá como producto final un instalador APK que permitirá hacer funcionar al producto en un dispositivo android; alternativamente el producto también será accesible desde un navegador web. Se harán pruebas antes de la entrega final al cliente en ambientes similares. No habrá una capacitación especializada para el usuario final, no se considera necesario. Se habilitará un canal de soporte vía correo electrónico para evacuar dudas o reportar problemas que puedan surgir.

2.1. Responsabilidades

Es responsabilidad del cliente que el dispositivo android cuente con espacio de almacenamiento disponible y una buena conexión a internet. El cliente será quien instalará en esos dispositivos el software, no será responsabilidad del proveedor hacerlo.

3. Recursos

3.1. Software de soporte

Se utilizará Visual Studio Code, con algunos plug-ins, para desarrollar la solución, haciendo uso de los lenguajes de programación Javascript, JSX y Python. Para las interfaces gráficas se utilizarán React y React Native. El equipo utilizará un repositorio github[1] para el mantenimiento de los diferentes módulos de la aplicación. La base de datos a utilizar para el almacenamiento del historial de consultas será DynamoDB.

- **Visual Studio Code**

  Es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

  Sitio: https://code.visualstudio.com/

[1] El repositorio esta disponible en https://github.com/uniMedic

- **Dynamo DB**

Es un servicio de base de datos noSQL ofrecido por Amazon como parte de Amazon Web Services.

Dynamo almacenará el historial de cada paciente.

Sitio: https://aws.amazon.com/es/dynamodb/

- **Amazon Cognito**

Ofrece a los clientes la flexibilidad de usar proveedores de identidades existentes, empresariales o de redes sociales. Además, usted ahorra tiempo con las configuraciones sencillas para federar proveedores de identidades.

Cognito será el encargado del login y registro de los usuarios.

Sitio: https://aws.amazon.com/es/cognito/

- **Amazon Amplify**

Ofrece un servicio completamente administrado para implementar y alojar aplicaciones web estáticas, con flujos de trabajo de integración y entrega continuas integrados que aceleran el ciclo de lanzamiento de la aplicación. Solo debe conectar el repositorio de código de la aplicación en la consola de Amplify, y las modificaciones que se realicen en el frontend y el backend se implementarán en un único flujo de trabajo en cada confirmación de código.

Amplify será el encargado de hostear la aplicación móvil.

Sitio: https://aws.amazon.com/es/amplify/hosting/

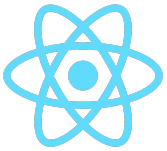
- **FastAPI**

FastAPI es un framework web moderno y rápido (de alto rendimiento) para crear API's con Python 3.

FastAPI se encargará se servir los modelos de IA para recibir y procesar las consultas provenientes del frontend.

Sitio: https://fastapi.tiangolo.com/

- **React Native**



Framework de desarrollo móvil. Se utiliza para desarrollar aplicaciones para Android, Android TV, iOS y macOS al permitir a los desarrolladores utilizar el marco de React junto con la plataforma nativa.

React Native será el encargado de proporcionar una interfaz amigable tipo chat para el usuario.

Sitio: https://reactnative.dev/

4. Capacitación

Los usuarios finales del producto no serán altamente capacitados. Como único tutorial de ayuda se proporcionará un vídeo demostrativo del uso de la aplicación. Además el software contendrá ayudas contextuales y será los más simple posible para evitar confundir a estos usuarios.

5. Configuración de la plataforma de desarrollo

A continuación se detallan los pasos para que los desarrolladores involucrados en el proyecto puedan construir la aplicación.

5.1. Configuración de FastAPI

Una vez que ya se tengan los modelos de IA completamente entrenados y listos para poder realizar inferencia, usaremos FastAPI para poder servirlos. Los pasos a continuación se ejecutan desde el módulo **Core**, por lo que tendrá que abrir el terminal desde dicha ubicación.

- Instale todas las dependencias de Python necesarias, en un entorno virtual (ya sea conda o virtualenv). Ejecute el siguiente comando desde consola:

```
#!/bin/bash
pip install -r requirements.txt
```

- Una vez instalada las dependencias previas, proceda a lanzar el servidor, para ello posicione el terminal en la carpeta **core** (dentro del módulo *Core*). Posteriormente ejecute el siguiente comando:

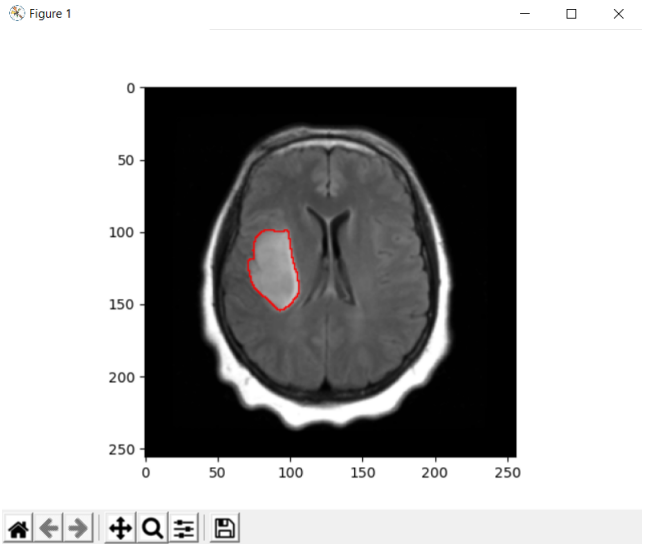
```
uvicorn main:app --host 0.0.0.0 --port 8000
```

  Todas las peticiones que hagamos a partir de ahora por el puerto 8000 serán recibidas y respondidas por los modelos de IA.

- Podemos hacer una prueba. Para ello posicione el terminal en la carpeta **test** (dentro de *core*). En dicha carpeta se encuentran 4 archivos de prueba. En caso se desee probar la segmentación cerebral, ejecute el siguiente comando:

```
python model_brain_segmentation_sanity.py
```

  Lo que abrirá automáticamente una ventana con el resultado, como se muestra a continuación.



5.2. Configuración Amplify

- Para configurar las herramientas de Amazon Web Services que usaremos para la aplicación, necesitamos instalar Amplify. Para ello posicionamos la terminal en la carpeta del módulo **App** y ejecutamos los siguientes comandos:

```
npm i -g @aws-amplify/cli
npm install aws-amplify @aws-amplify/ui-react
```

- Posteriormente procedemos a configurar Amplify con nuestras credenciales de AWS ejecuntando el siguiente comando:

  ```
  amplify configure
  ```

- Una vez validada nuestra credenciales, creamos una instancia de Amplify para nuestro proyecto. Para ello ejecutamos el siguiente comando en la misma ruta:

  ```
  amplify init
  ```

- Posteriormente para tener una instancia de DynamoDB para nuestra aplicación (para almacenar el historial) ejecutamos desde terminal (en la misma ruta) el siguiente comando:

  ```
  amplify add api
  ```

  El comando anterior creará una tabla donde guardará los historiales y será accesible desde la interfaz web de DynamoDB en AWS.

- Posteriormente para tener una instancia de Cognito para nuestra aplicación (para logear y registrar a los usuarios) ejecutamos desde terminal (en la misma ruta) el siguiente comando:

  ```
  amplify add auth
  ```

  El comando anterior creará un registro donde guardará las credenciales de los usuarios y será accesible desde la interfaz web de Cognito en AWS.

  Cabe destacar que podemos crear usuarios IAM dentro de AWS, los cuales pueden encargarse de supervisar ciertas tareas en específico (administrar la base de datos, administrar los login's, etc).

5.3. Configuración App Android

- Antes de lanzar la aplicación necesitamos instalar todos los paquetes necesarios mediante Node.js y su manejador de paqueterías *npm*. Para ello posicionamos la terminal en la carpeta del módulo **App** y ejecutamos el siguiente comando:

  ```
  npm i
  ```

- También necesitamos instalar Android Studio junto al SDK. Debemos configurar las variables de entorno según este tutorial.

- Una vez que se tenga la interfaz creada, y conectada con el backend (modelos IA, DynamoDB y Cognito), podemos ejecutar la aplicación desde el emulador de Android Studio. Para ello procedemos e ejecutar el siguiente comando en la misma ruta:

  ```
  yarn start --reset-cache
  ```

- Mientras el proceso anterior se ejecuta (aparece el ícono de React Native en terminal), procedemos a abrir una nueva terminal en la misma ruta y ejecutamos el siguiente comando:

```
npx react-native run-android
```

El comando anterior conectará con el emulador android y mostrará la interfaz móvil de la aplicación. Podemos ejecutar los mismos pasos conectando un dispositivo android mediante cable USB en modo depuración.