**Documento de Arquitectura de Software**

**Proyecto MentalSync**

**Versión 1.0**

**Historial de Revisiones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 03/10/2024 | 1.0 | Arquitectura de Software | Jeremy Andrew Rosillo Ramirez |
|  |  |  |  |
|  |  |  |  |

**Tabla de Contenido**

[1. Introducción 3](#_Toc179409677)

[1.1. Propósito 3](#_Toc179409678)

[1.2. Alcance 3](#_Toc179409679)

[1.3. Definiciones, siglas y abreviaturas 3](#_Toc179409680)

[1.4. Vista Global 4](#_Toc179409681)

[2. Macro Arquitectura 4](#_Toc179409682)

[3. Vista Física 4](#_Toc179409683)

[4. Vista Lógica 5](#_Toc179409684)

[5. Vista de Despliegue 6](#_Toc179409685)

# Introducción

## Propósito

El propósito de este documento es definir y dar a conocer la Arquitectura de Software, la cual va a ser utilizada para el desarrollo del proyecto “MentalSync”. Se utiliza la vista física, para describir la estructura y funcionalidad

Este documento va dirigido al equipo de desarrollo, con el fin de proporcionar las bases arquitecturales, además de una visión global y comprensible del diseño, sobre las cuales el proyecto se define para ser llevado a cabo, con el documento podrán comprender la organización de herramientas utilizadas en la creación del sistema.

## Alcance

Este documento describe cómo se va a desarrollar el sistema que permitirá a los usuarios comunes y psicólogos interactuar de una manera más eficiente, pudiendo reservar y gestionar sus respectivas citas. La aplicación es Web y por tal motivo funcionaria en computadores que tengan un navegador Web.

## Definiciones, siglas y abreviaturas

**DAS:** Documento de Arquitectura de Software.

**Front-End:** término utilizado en desarrollo de aplicaciones para determinar la parte del software que interactúa con los usuarios.

**Back-End:** término utilizado en desarrollo de aplicaciones para determinar la parte del software que procesa la información del Front-End.

**JavaScript (JS):** Lenguaje de programación interpretado por el navegador web

**NodeJS:** Entorno de ejecución de JavaScript

**React:** Biblioteca de JavaScript diseñada para crear interfaces de usuario.

**ExpressJS:** Marco de trabajo de NodeJS, diseñado para desarrollar apps web y APIs

**API: P**ieza de código que permite a dos aplicaciones comunicarse entre sí para compartir información y funcionalidades.

**DB:** base de datos del sistema.

**DBMS:** Es un software de sistema para crear y administrar bases de datos**.**

**MySQL:** Sistema de gestión de bases de datos relacional

**SQL:** Lenguajediseñado para administrar, y recuperar información de sistemas de gestión de bases de datos relacionales

## Vista Global

A diagram of a service

Description automatically generated

# Macro Arquitectura

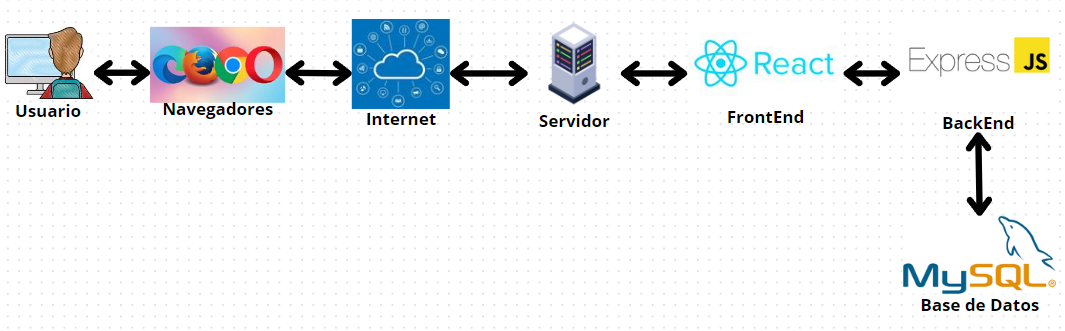
La macro arquitectura que comprende el sistema actual se basa en 2 vistas claves, la vista física y lógica que comprenderán los siguientes diagramas:

* **Vista Física:** Diagrama informal, el cual permite mostrar la interacción física necesaria y usada entre los dispositivos que incorpora el proyecto.
* **Vista Lógica:** Diagrama de componentes, debido a la facilidad que tiene junto con su estructura el permitir un análisis completo y a fondo de las diversas etapas de desarrollo de un aplicativo software, expresando las relaciones que se obtendrán las distintas dependencias de un componente. Diagrama de despliegue, ya que muestra la arquitectura del sistema desde el punto de vista de red de los clientes a los servidores.

# Vista Física

**Diagrama informal de la vista física:** Es un diagrama que describe las características físicas de los elementos que componen la arquitectura del proyecto.

El diagrama contempla el uso de la aplicación por parte del usuario que inicialmente ingresa al navegador desde un dispositivo, ya sea un computador de escritorio o dispositivos móviles. El usuario ingresará a la página web de la plataforma que está alojada en un servidor de aplicaciones Node.js. De igual forma, el servidor tiene como gestor de bases de datos MySQL.



# Vista Lógica

**Diagrama de componentes:** Este diagrama representa el marco de trabajo de la relación entre las diferentes dependencias del stack tecnológico (React, Express, MySQL) y los componentes que se encuentran dentro de esta arquitectura para la funcionalidad de MentalSync. En este diagrama se ve reflejado que el sistema es multicapa representado por la capa de presentación, capa de lógica del negocio, y capa de acceso a datos, donde al ser un sistema multicapa nos permite realizar de manera óptima y eficiente la mantenibilidad ya que se puede dar soporte o atención a un punto específico sin necesidad de tocar el resto del sistema, la modificabilidad del sistema permitiendo incluir componentes en una línea de tiempo sin que afecte lo que ya existe, como también permite mayor funcionalidad.

En la capa de presentación se utiliza el framework de diseño React para construir una interfaz de usuario interactiva y dinámica. React permite hacer un diseño web responsivo y portable para diferentes plataformas de computadores y que tengan un navegador Web.

En la capa de lógica de negocio se utiliza Express, un marco de aplicación web para Node.js, que proporciona un conjunto robusto de características para las aplicaciones web y móviles. Express maneja las rutas y redirecciona a un controlador donde éste a su vez tiene unas funciones específicas de acuerdo con la ruta invocada. Dentro del controlador tenemos funciones propias del marco de trabajo y la lógica de procesos que ejecuta el sistema.

En la capa de acceso a datos se utiliza MySQL, un sistema de gestión de bases de datos relacional de código abierto, que, con ayuda de MySQLWorkbench nos permitirá tener un GUI para gestionar la base de datos de la aplicación.

A diagram of a computer

Description automatically generated

# Vista de Despliegue

Este diagrama describe las configuraciones de redes físicas, sobre las cuales el software será desarrollado. Se indican los nodos físicos que interactúan con la aplicación web tales como un nodo para el dispositivo donde se accede la aplicación web en un PC con acceso a Internet y un navegador. Éste envía las peticiones por HTTP para mayor seguridad de información accediendo al nodo del servidor web donde se aloja el servidor Express.js y la base de datos MySQL.

Dentro del servidor Express.js, tenemos la funcionalidad de la aplicación dividida por capas tales como la capa de presentación (React), lógica del negocio (Express), y acceso a datos (MySQL), permitiéndonos tener mejor mantenibilidad, modificabilidad y funcionalidad.

Por otra parte, la lógica de negocios integra las interfaces de APIs la cual sería el backend de la lógica del negocio creada por microservicios. En este caso, Express.js actúa como el microservicio que maneja las solicitudes HTTP y proporciona las APIs para la aplicación.

A diagram of a server

Description automatically generated