

Documento de Desarrollo del Proyecto Baymax

1. Introducción

Resumen del Documento: Este documento describe la fase de desarrollo de Baymax, una inteligencia artificial orientada a la consulta médica y apoyo psicológico. Se presenta la estructura del código, la implementación de funcionalidades clave y los casos de prueba necesarios para validar el sistema.

Objetivo del Documento: Definir los detalles de implementación y pruebas de Baymax, documentando el proceso y resultados para confirmar que el sistema cumple con los requerimientos funcionales y no funcionales definidos previamente.

2. Estructura del Código

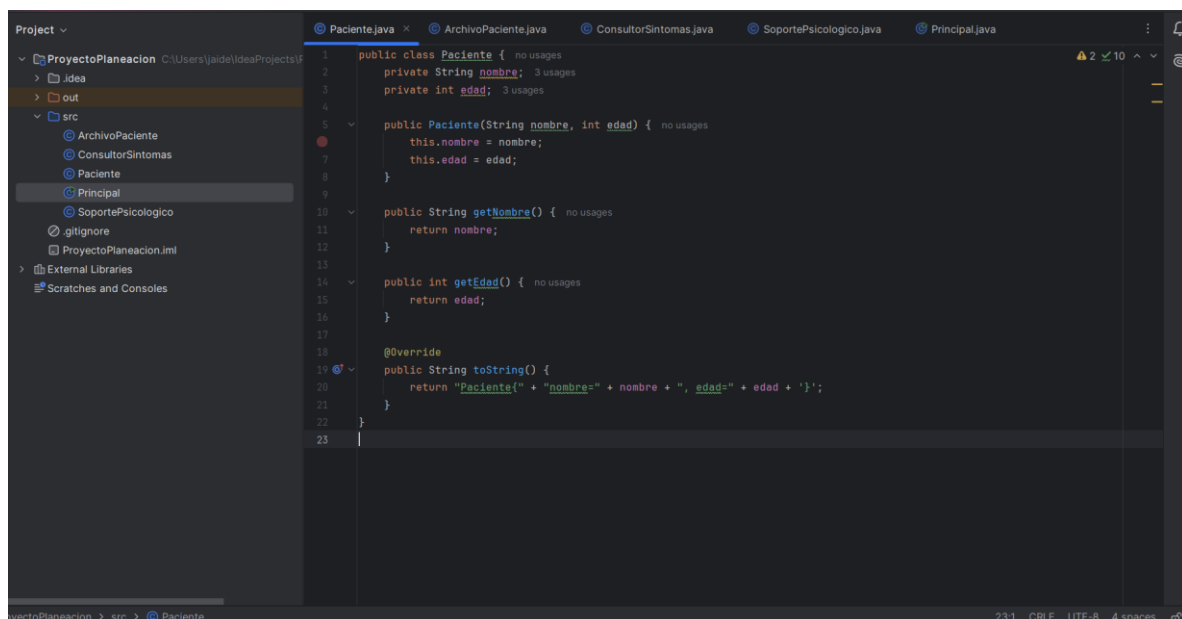
Organización del Proyecto: Baymax está desarrollado en Java utilizando el IDE IntelliJ IDEA, con una estructura organizada en directorios:

- **src/:** Contiene el código fuente principal.
- **test/:** Clases para pruebas unitarias e integradas.
- **resources/:** Archivos de configuración y recursos estáticos (como guías de síntomas y mensajes de apoyo).
- **database/:** Archivos de persistencia de datos, como pacientes.txt, manejado por la clase ArchivoPaciente.

3. Estructura de Clases y Componentes

Clase Paciente

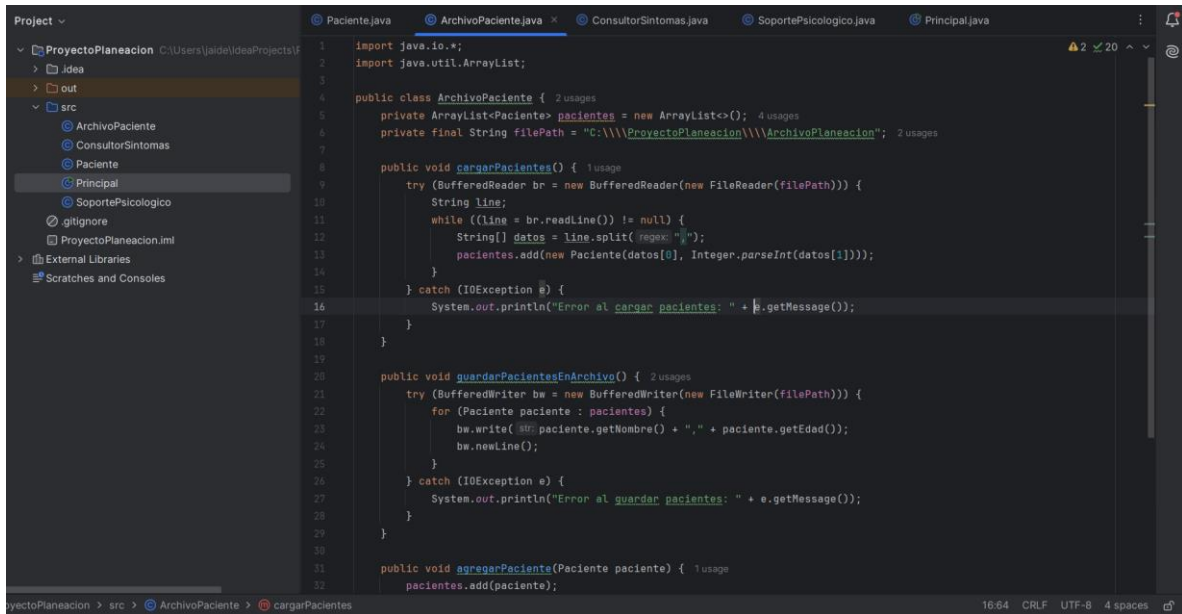
Define un paciente, con atributos básicos como nombre y edad.



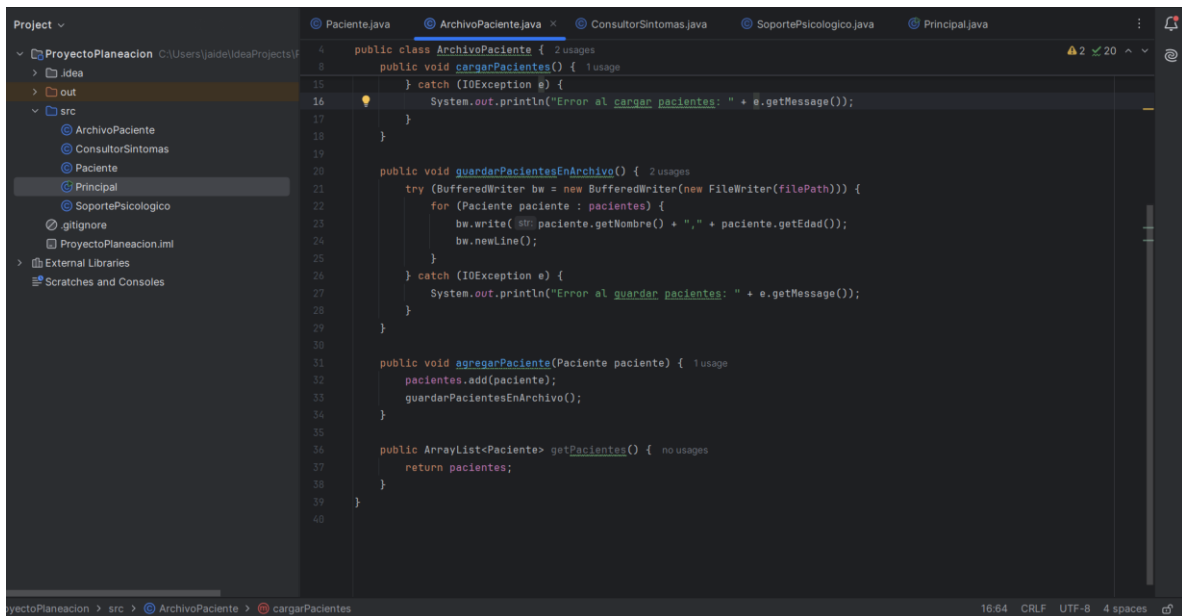
```
1 public class Paciente {
2     private String nombre;
3     private int edad;
4
5     public Paciente(String nombre, int edad) {
6         this.nombre = nombre;
7         this.edad = edad;
8     }
9
10    public String getNombre() {
11        return nombre;
12    }
13
14    public int getEdad() {
15        return edad;
16    }
17
18    @Override
19    public String toString() {
20        return "Paciente{" + "nombre=" + nombre + ", edad=" + edad + '}';
21    }
22 }
23
```

Clase ArchivoPaciente

Se encarga de la persistencia de los datos de los pacientes, cargándolos desde pacientes.txt y guardando cambios cuando se actualiza la información.



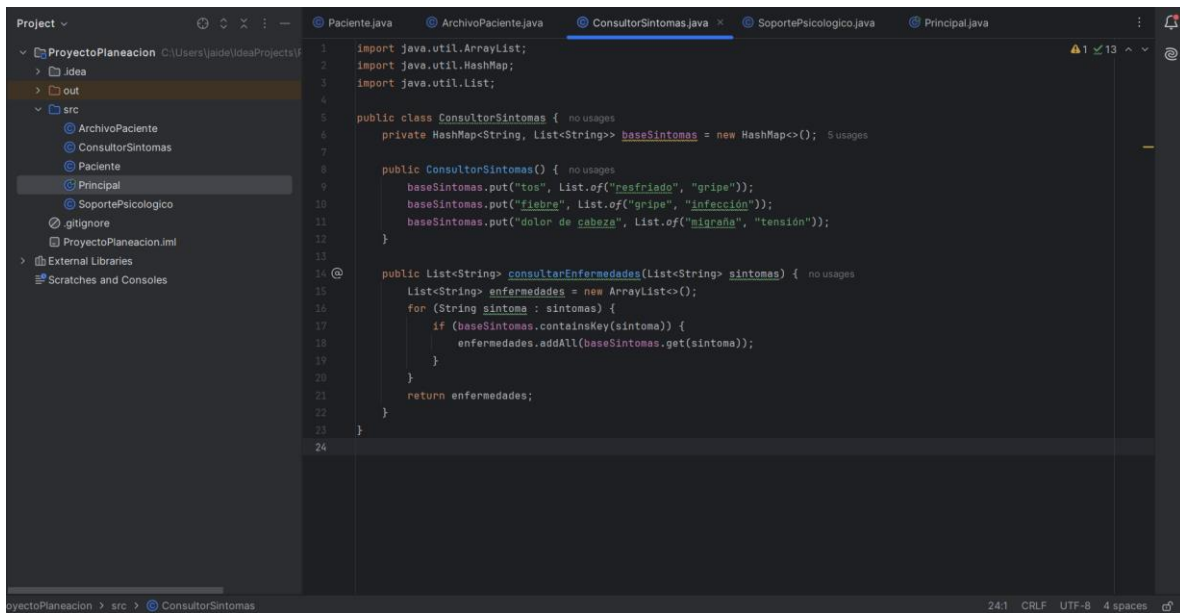
```
1 import java.io.*;
2 import java.util.ArrayList;
3
4 public class ArchivoPaciente {
5     private ArrayList<Paciente> pacientes = new ArrayList<>();
6     private final String filePath = "C:\\\\ProyectoPlaneacion\\\\ArchivoPlaneacion";
7
8     public void cargarPacientes() {
9         try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {
10             String line;
11             while ((line = br.readLine()) != null) {
12                 String[] datos = line.split(",");
13                 pacientes.add(new Paciente(datos[0], Integer.parseInt(datos[1])));
14             }
15         } catch (IOException e) {
16             System.out.println("Error al cargar pacientes: " + e.getMessage());
17         }
18     }
19
20     public void guardarPacientesEnArchivo() {
21         try (BufferedWriter bw = new BufferedWriter(new FileWriter(filePath))) {
22             for (Paciente paciente : pacientes) {
23                 bw.write(paciente.getNombre() + "," + paciente.getEdad());
24                 bw.newLine();
25             }
26         } catch (IOException e) {
27             System.out.println("Error al guardar pacientes: " + e.getMessage());
28         }
29     }
30
31     public void agregarPaciente(Paciente paciente) {
32         pacientes.add(paciente);
33     }
34 }
```



```
4 public class ArchivoPaciente {
5     private ArrayList<Paciente> pacientes = new ArrayList<>();
6     private final String filePath = "C:\\\\ProyectoPlaneacion\\\\ArchivoPlaneacion";
7
8     public void cargarPacientes() {
9         try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {
10             String line;
11             while ((line = br.readLine()) != null) {
12                 String[] datos = line.split(",");
13                 pacientes.add(new Paciente(datos[0], Integer.parseInt(datos[1])));
14             }
15         } catch (IOException e) {
16             System.out.println("Error al cargar pacientes: " + e.getMessage());
17         }
18     }
19
20     public void guardarPacientesEnArchivo() {
21         try (BufferedWriter bw = new BufferedWriter(new FileWriter(filePath))) {
22             for (Paciente paciente : pacientes) {
23                 bw.write(paciente.getNombre() + "," + paciente.getEdad());
24                 bw.newLine();
25             }
26         } catch (IOException e) {
27             System.out.println("Error al guardar pacientes: " + e.getMessage());
28         }
29     }
30
31     public void agregarPaciente(Paciente paciente) {
32         pacientes.add(paciente);
33         guardarPacientesEnArchivo();
34     }
35
36     public ArrayList<Paciente> getPacientes() {
37         return pacientes;
38     }
39 }
40 }
```

Clase CosultorSintomas

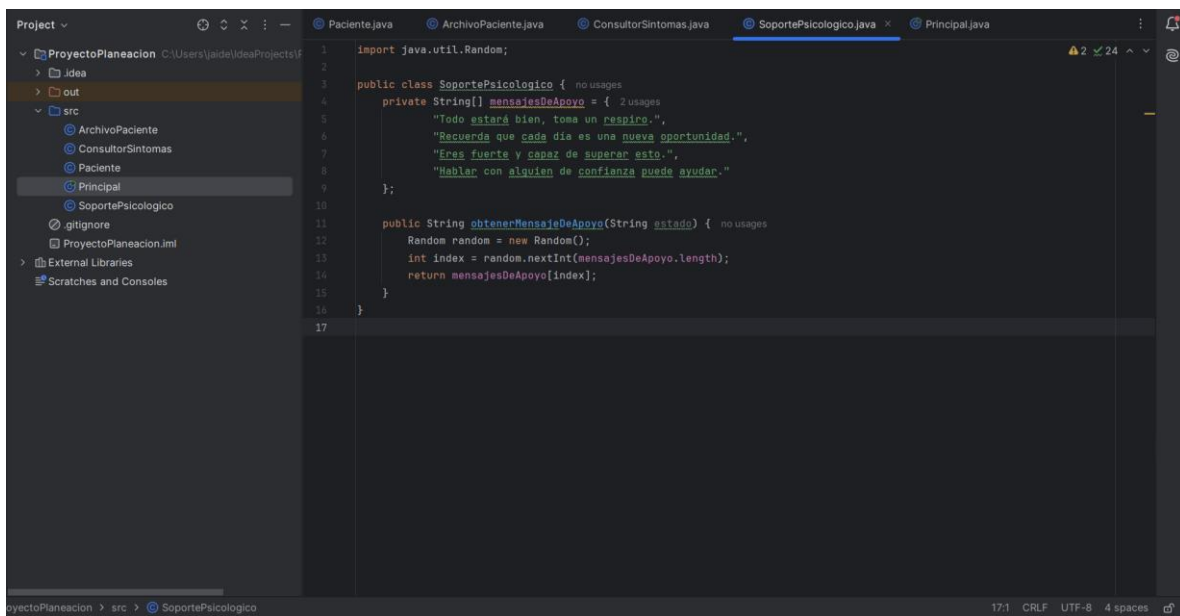
Esta clase permite ingresar síntomas y sugiere posibles enfermedades utilizando una base de datos estática.



```
1 import java.util.ArrayList;
2 import java.util.HashMap;
3 import java.util.List;
4
5 public class ConsultorSintomas {
6     private HashMap<String, List<String>> baseSintomas = new HashMap<>();
7
8     public ConsultorSintomas() {
9         baseSintomas.put("tos", List.of("resfriado", "gripe"));
10        baseSintomas.put("fiebre", List.of("gripe", "infección"));
11        baseSintomas.put("dolor de cabeza", List.of("migraña", "tensión"));
12    }
13
14    @
15    public List<String> consultarEnfermedades(List<String> sintomas) {
16        List<String> enfermedades = new ArrayList<>();
17        for (String sintoma : sintomas) {
18            if (baseSintomas.containsKey(sintoma)) {
19                enfermedades.addAll(baseSintomas.get(sintoma));
20            }
21        }
22        return enfermedades;
23    }
24 }
```

Clase SoportePsicologico

Ofrece mensajes de apoyo psicológico personalizados según el estado de ánimo del usuario.



```
1 import java.util.Random;
2
3 public class SoportePsicologico {
4     private String[] mensajesDeApoyo = {
5         "Todo estará bien, toma un respiro.",
6         "Recuerda que cada día es una nueva oportunidad.",
7         "Eres fuerte y capaz de superar esto.",
8         "Hablar con alguien de confianza puede ayudar."
9     };
10
11    public String obtenerMensajeDeApoyo(String estado) {
12        Random random = new Random();
13        int index = random.nextInt(mensajesDeApoyo.length);
14        return mensajesDeApoyo[index];
15    }
16 }
17
```

Clase Pripical

Clase de ejecución principal del sistema Baymax.

```

1  import java.util.List;
2  import java.util.Scanner;
3
4
5  public class Principal {
6      private static ArchivoPaciente archivoPaciente = new ArchivoPaciente(); 3 usages
7      private static ConsultorSintomas consultorSintomas = new ConsultorSintomas(); 1 usage
8      private static SoportePsicologico soportePsicologico = new SoportePsicologico(); 1 usage
9
10     public static void main(String[] args) {
11         archivoPaciente.cargarPacientes();
12         Scanner scanner = new Scanner(System.in);
13
14         int opcion;
15         do {
16             System.out.println("\n1. Agregar paciente");
17             System.out.println("2. Consultar síntomas");
18             System.out.println("3. Apoyo psicológico");
19             System.out.println("4. Salir");
20             System.out.print("Seleccione una opción: ");
21             opcion = scanner.nextInt();
22             scanner.nextLine();
23
24             switch (opcion) {
25                 case 1:
26                     System.out.print("Nombre del paciente: ");
27                     String nombre = scanner.nextLine();
28                     System.out.print("Edad del paciente: ");
29                     int edad = scanner.nextInt();
30                     archivoPaciente.agregarPaciente(new Paciente(nombre, edad));
31                     System.out.println("Paciente agregado.");
32                     break;

```

```

33         case 2:
34             System.out.print("Ingrese síntomas separados por comas (ej. tos, fiebre): ");
35             String inputSíntomas = scanner.nextLine();
36             List<String> sintomas = List.of(inputSíntomas.split(", "));
37             List<String> enfermedades = consultorSintomas.consultarEnfermedades(sintomas);
38             System.out.println("Posibles enfermedades: " + enfermedades);
39             break;
40         case 3:
41             System.out.print("Estado de ánimo del usuario (ej. triste): ");
42             String estado = scanner.nextLine();
43             System.out.println(soportePsicologico.obtenerMensajeDeApoyo(estado));
44             break;
45         case 4:
46             System.out.println("Saliendo del sistema.");
47             break;
48         default:
49             System.out.println("Opción no válida.");
50     }
51     while (opcion != 4);
52
53     archivoPaciente.guardarPacientesEnArchivo();
54     scanner.close();
55 }
56
57

```

4. Pruebas del Sistema

```
Run Principal x
C:\Users\jaide\.jdk\openjdk-20.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.3\lib\idea_rt.jar=58614:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.3\bin" -Dfile.encoding=UTF-8
1. Agregar paciente
2. Consultar síntomas
3. Apoyo psicológico
4. Salir
Seleccione una opción: 1
Nombre del paciente: Jaider
Edad del paciente: 19
Paciente agregado.

1. Agregar paciente
2. Consultar síntomas
3. Apoyo psicológico
4. Salir
Seleccione una opción: 2
Ingrese síntomas separados por comas (ej. tos, fiebre): tos, fiebre
Posibles enfermedades: [resfriado, gripe, gripe, infección]

1. Agregar paciente
2. Consultar síntomas
3. Apoyo psicológico
4. Salir
Seleccione una opción: 3
Estado de ánimo del usuario (ej. triste): triste
Hablar con alguien de confianza puede ayudar.

1. Agregar paciente
2. Consultar síntomas
```

5. Conclusiones

Observaciones y Rendimiento:

Las pruebas demostraron que el sistema funciona como se espera. Algunas áreas podrían mejorarse, como la gestión de síntomas y la personalización de apoyo psicológico.

Mejoras Futuras:

- Ampliación de la base de datos de síntomas y enfermedades.
- Implementación de un seguimiento emocional avanzado, con gráficos y registros.