**Requerimientos funcionales**

|  |  |
| --- | --- |
| **Nombre** | **Rf. #1. Cargar lista de asistentes y participantes** |
| **Resumen** | Permite cargar lista de asistentes y participantes. |
| **Entradas** | |
| Nombre de archivo de tipo .txt que va a ser leído por java para cargar los nombres de los asistentes con sus datos respectivos | |
| **Resultados** | |
| Ninguna. | |

|  |  |
| --- | --- |
| **Nombre** | **Rf. #2. Buscar asistente en la lista** |
| **Resumen** | Permite buscar asistente en la lista. |
| **Entradas** | |
| Id del asistente. | |
| **Resultados** | |
| Un mensaje que da toda la información del asistente si existe y si no un mensaje que dice “Esta id no pertenece a ningún asistente registrado”. | |

|  |  |
| --- | --- |
| **Nombre** | **Rf. #3. Buscar participante en la lista.** |
| **Resumen** | Permite buscar participante en la lista. |
| **Entradas** | |
| Id del participante. | |
| **Resultados** | |
| Un mensaje que da toda la información del participante si existe y si no un mensaje que dice “Esta id no pertenece a ningún participante registrado”. | |

|  |  |
| --- | --- |
| **Nombre** | **Rf. #4. Pintar un árbol binario con la información de los participantes** |
| **Resumen** | Permite pintar un árbol binario con la información de los participantes |
| **Entradas** | |
| Nombre de la ciudad | |
| **Resultados** | |
| Un gráfico que muestra los participantes en forma de árbol con los datos de cada uno | |

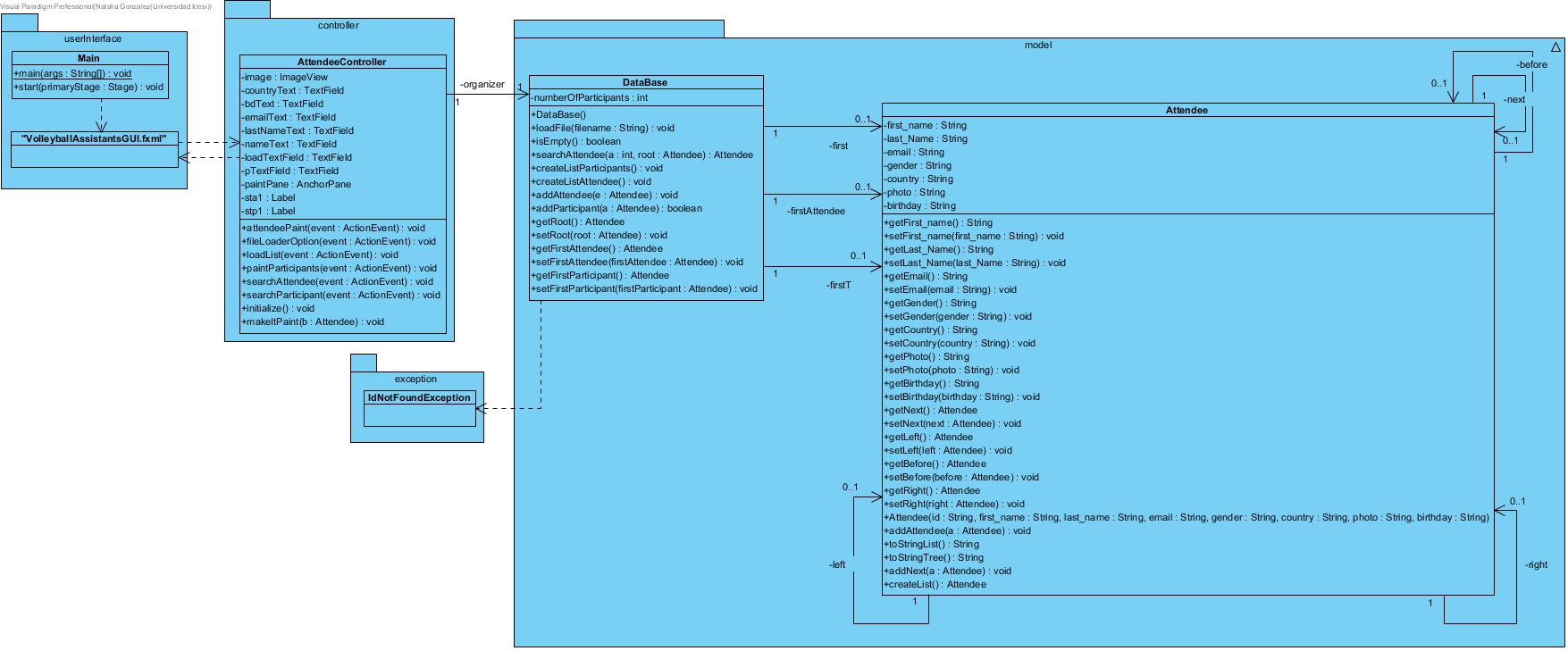
|  |  |
| --- | --- |
| **Nombre** | **Rf. #5. Pintar un árbol binario con la información de los espectadores** |
| **Resumen** | Permite pintar un árbol binario con la información de los participantes |
| **Entradas** | |
| Nombre de la ciudad | |
| **Resultados** | |
| Un gráfico que muestra los espectadores en forma de árbol con los datos de cada uno | |

|  |  |
| --- | --- |
| **Nombre** | **Rf. #6. Escoger aleatoriamente a los participantes y los espectadores** |
| **Resumen** | Permite escoger aleatoriamente a los participantes y los espectadores a partir del grupo de los asistentes |
| **Entradas** | |
| Ninguna | |
| **Resultados** | |
| Ninguna | |

**Trazabilidad al diseño**

|  |  |  |
| --- | --- | --- |
| **Requerimiento funcional** | **Método** | **Clase** |
| **Rf. #1. Cargar lista de asistentes y participantes** | fileLoaderOption(event: ActionEvent): void | AttendeeController |
| loadList(event: ActionEvent): void | AttendeeController |
| loadFile(filename: String): void | DataBase |
| addAttendee(e: Attendee): void | DataBase |
| addAttendee(e: Attendee): void | Attendee |
| **Rf. #2. Buscar asistente en la lista** | searchAttendee(event: ActionEvent): void | AttendeeController |
| searchAssistant(cod: String, act Attendee): Attendee | DataBase |
| makeItPaint(b: Attendee): void | AttendeeController |
| **Rf. #3. Buscar participante en la lista.** | searchParticipant(event: ActionEvent): void | AttendeeController |
| makeItPaint(b: Attendee): void | AttendeeController |
| **Rf. #4. Pintar un árbol binario con la información de los participantes** | painParticipants(event: ActionEvent): void | AttendeeController |
| **Rf. #5. Pintar un árbol binario con la información de los espectadores** |  |  |
| **Rf. #6. Escoger aleatoriamente a los participantes y los espectadores** | createListParticipants(): void | DataBase |
| createListAttendee (): void | DataBase |
| addParticipant(a: Attendee): boolean | DataBase |
| addNext(a: Attendee): void | Attendee |
| createList(): Attendee | Attendee |

**Diagrama de clases**

****