documento de ingenieria de software para el proyecto de gestion de datos

**MY HOSPITAL**

**ANTHONY BARAHONA**

**NATALIA ISAZA SERRANO**

**COTECNOVA**

**VERSION 2.0**

TABLA DE CONTENIDO

[HOJA DE CONTROL DEL DOCUMENTO 2](#_Toc22160967)

[HISTORIAL DE REVISIONES 3](#_Toc22160968)

[1 INTRODUCCION 4](#_Toc22160969)

[1.1 Propósito 4](#_Toc22160970)

[1.2 Usuarios 4](#_Toc22160971)

[1.3 Alcance 4](#_Toc22160972)

[1.4 Cronograma 5](#_Toc22160973)

[1.5 Glosario, acrónimos y definiciones 5](#_Toc22160974)

[2 VISTA GENERAL DEL PROYECTO 6](#_Toc22160975)

[2.1 Propósito, alcance y objetivos 7](#_Toc22160976)

[2.2 Aclaraciones 7](#_Toc22160977)

[¿Qué es SCRUM? 8](#_Toc22160978)

[2.2.1 Roles de usuario. 8](#_Toc22160979)

[2.2.2 Proceso de desarrollo. 8](#_Toc22160980)

[3 DISEÑO Y ARQUITECTURA DEL SISTEMA 9](#_Toc22160981)

[3.1 Descripción general 9](#_Toc22160982)

[3.2 Ingeniería de requisitos 9](#_Toc22160983)

[3.2.1 Requisitos funcionales 9](#_Toc22160984)

[- Consulta 9](#_Toc22160985)

[- Manejar/administrar modulo 10](#_Toc22160986)

[- Manejar/Administrar Citas 10](#_Toc22160987)

[- Manejar/Administrar Pacientes 11](#_Toc22160988)

[- Manejar/Administrar Cita medica 11](#_Toc22160989)

[3.2.2 Otros requisitos 12](#_Toc22160990)

[3.2.3 Requisitos no funcionales 13](#_Toc22160991)

[- RNF1 13](#_Toc22160992)

[- RNF2 13](#_Toc22160993)

[- RNF3 13](#_Toc22160994)

[- RNF4 13](#_Toc22160995)

[- RNF5 13](#_Toc22160996)

[- RNF6 13](#_Toc22160997)

[- RNF7 14](#_Toc22160998)

[3.3 Diseño de base de datos 14](#_Toc22160999)

[4 PROCESOS DEL SISTEMA 19](#_Toc22161004)

[4.1 Actores y casos de uso 19](#_Toc22161005)

[4.2.1 Actores 19](#_Toc22161006)

[4.2.2 Casos de uso 20](#_Toc22161007)

[ Caso de uso – 01 CONSULTA 20](#_Toc22161008)

[ Caso de uso – 02 Manejar modulo 21](#_Toc22161009)

[ Caso de uso – 03 Manejar CITA 22](#_Toc22161010)

[ Caso de uso – 04 Manejar PACIENTE 23](#_Toc22161011)

[ Caso de uso – 05 Manejar CITA médica 24](#_Toc22161012)

[5 PROTOTIPOS DEL SISTEMA 25](#_Toc22161013)

[6 MODELO DE DESARROLLO 33](#_Toc22161014)

[7 DIAGRAMAS DE ACTIVIDADES 34](#_Toc22161015)

[6.1 Log in 34](#_Toc22161016)

[6.2 Usuario 34](#_Toc22161017)

[6.3 Pacientes 36](#_Toc22161018)

[Citas 38](#_Toc22161019)

[Ajustes 41](#_Toc22161020)

[8 DIAGRAMA DE CLASES 44](#_Toc22161021)

[9 DIAGRAMA DE COMPONENTES 45](#_Toc22161022)

[10 ARQUITECTURA ADAPTADA 46](#_Toc22161023)

[11 DESARROLLO DEL PROYECTO 47](#_Toc22161024)

[12 PRUEBAS 51](#_Toc22161025)

[12.1 Pruebas Inicio de sesión 51](#_Toc22161026)

[12.2 Pruebas superadmin 58](#_Toc22161027)

[12.3 Pruebas usuario admin 65](#_Toc22161028)

[12.4 Pruebas de usuario médico 67](#_Toc22161029)

[12.5 Datos incorrectos al editar un usuario 69](#_Toc22161030)

# TABLA DE ILUSTRACIONES

[Ilustración 1 Cronograma sprint 9](#_Toc22161556)

[Ilustración 2 Diagrama DB Modelo Relacional 21](#_Toc22161557)

[Ilustración 3 Diagrama DB Modelo entidad relación 22](#_Toc22161558)

[Ilustración 4 Diagrama CU general 23](#_Toc22161559)

[Ilustración 5 Caso de uso – 01 CONSULTA 24](#_Toc22161560)

[Ilustración 6 Caso de uso – 02 Manejar modulo 25](#_Toc22161561)

[Ilustración 7 Caso de uso – 03 Manejar CITA 26](#_Toc22161562)

[Ilustración 8 Caso de uso – 04 Manejar PACIENTE 27](#_Toc22161563)

[Ilustración 9 Caso de uso – 05 Manejar CITA medica 28](#_Toc22161564)

[Ilustración 10 PS-Usuarios 29](#_Toc22161565)

[Ilustración 11 PS-Crear usuarios 30](#_Toc22161566)

[Ilustración 12 PS-Editar usuarios 30](#_Toc22161567)

[Ilustración 13 PS-Editar usuarios (eliminar) 31](#_Toc22161568)

[Ilustración 14 PS-Paciente 31](#_Toc22161569)

[Ilustración 15 PS-Crear pacientes 32](#_Toc22161570)

[Ilustración 16 PS-Editar pacientes 32](#_Toc22161571)

[Ilustración 17 PS-Editar pacientes (eliminar) 33](#_Toc22161572)

[Ilustración 18 PS-Citas 33](#_Toc22161573)

[Ilustración 19 PS-Crear citas 34](#_Toc22161574)

[Ilustración 20 PS-Editar citas 34](#_Toc22161575)

[Ilustración 21 PS-Editar citas (eliminar) 35](#_Toc22161576)

[Ilustración 22 PS-Ordenes 35](#_Toc22161577)

[Ilustración 23 PS-Crear ordenes 36](#_Toc22161578)

[Ilustración 24 PS-Editar ordenes 36](#_Toc22161579)

[Ilustración 25 PS-Editar ordenes (eliminar) 37](#_Toc22161580)

[Ilustración 26 PS-Historias (Pacientes) 37](#_Toc22161581)

[Ilustración 27 log in usuario 38](#_Toc22161582)

[Ilustración 28 modificaciones usuario 39](#_Toc22161583)

[Ilustración 29 crear usuario 40](#_Toc22161584)

[Ilustración 30 buscar usuario 40](#_Toc22161585)

[Ilustración 31 modificaciones paciente 41](#_Toc22161586)

[Ilustración 32 crear paciente 42](#_Toc22161587)

[Ilustración 33 buscar paciente 42](#_Toc22161588)

[Ilustración 34 modificaciones citas 43](#_Toc22161589)

[Ilustración 35 crear cita 44](#_Toc22161590)

[Ilustración 36 buscar cita 44](#_Toc22161591)

[Ilustración 37 modificar ordenes 45](#_Toc22161592)

[Ilustración 38 modificar ajustes arl 46](#_Toc22161593)

[Ilustración 39 modificar ajustes eps 47](#_Toc22161594)

[Ilustración 40 nuevos ajustes 48](#_Toc22161595)

[Ilustración 41 Diagrama de clases 49](#_Toc22161596)

[Ilustración 42 diagrama de componentes 50](#_Toc22161597)

[Ilustración 43 cliente-servidor 51](#_Toc22161598)

# HOJA DE CONTROL DEL DOCUMENTO

|  |  |  |  |
| --- | --- | --- | --- |
| **Título** | Documento de ingeniería de software. Sistema de gestión de datos:  My Hospital | | |
| **Descripción** | My Hospital es una aplicación desarrollada para facilitar la gestión de información y recursos un de hospital.  (Citas, usuarios, pacientes y ordenes medicas) | | |
| **Autor** | Anthony Barahona Villamil | | |
| **Responsable** | Cristhian Urrego Salazar (SCRUM Master) | | |
| **Organización** | Corporación de Estudios Tecnológicos del Norte del Valle | | |
| **Categoría** | Documentación técnica | | |
| **Palabras clave** | Hospital, Software, SCRUM, Gestión | | |
| **Versión** | 1.2 | **Fecha** | 11 septiembre de 2019 |

# HISTORIAL DE REVISIONES

Primer sprint

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 30 agosto de 2019 | 1.0 | VERSIÓN INICIAL | Anthony Barahona Villamil |
| 1 septiembre de 2019 | 1.1 | VERSIÓN AJUSTADA | Anthony Barahona Villamil |
| 11 septiembre de 2019 | 1.2 | FINAL | Anthony Barahona Villamil |

Segundo sprint

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 19 Septiembre de 2019 | 1.3 | VERSIÓN INICIAL | Natalia Isaza Serrano |
| 11 Octubre de 2019 | 1.4 | VERSIÓN AJUSTADA | Natalia Isaza Serrano |
| 17 Octubre de 2019 | 1.5 | FINAL | Natalia Isaza Serrano |

# INTRODUCCION

Este documento está destinado a describir el plan de desarrollo de la aplicación My Hospital y sus especificaciones técnicas.

El proyecto se desarrolla y describe con base en el modelo de desarrollo de software SCRUM, que junto con el Lenguaje Unificado de Modelado (UML), constituye una de las metodologías más utilizadas para el análisis, diseño, implementación y documentación de sistemas. Por ello, este documento dispone la terminología propia de dichas metodologías incluyendo así el alcance My Hospital, e identificando los actores y casos de uso.

## Propósito

El propósito principal de este documento, es brindar toda información necesaria para controlar y desarrollar correctamente el proyecto My Hospital, describiendo así sus funcionalidades.

## Usuarios

Este documento es concebido como herramienta de referencia, y no como texto con finalidad divulgativa o para usuarios finales. Los actores previstos para la consulta de este documento son:

* Responsables My Hospital: como herramienta de seguimiento y para control del proceso de desarrollo.
* Desarrolladores: como guía de actividades, calendario de fases y ejecución.
* Auditores: para verificar el ajuste de lo proyectado a la realidad de la implementación.

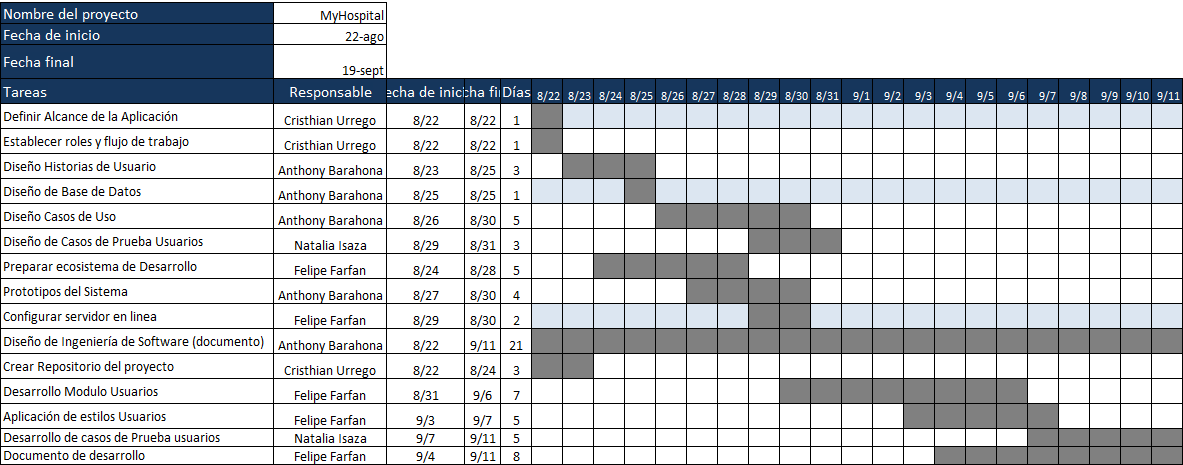
## Alcance

A solicitud del Producto Owner, el software debe contar con las siguientes características; Pacientes, médicos, historia clínica, administrativos, tratamientos, autorizaciones, incapacidades.

Se estableció que el software contaría con los siguientes módulos, Usuarios, Pacientes, Citas, Órdenes e Historia de Usuario.

## Cronograma

La siguiente ilustración representa el cronograma de actividades que se deben llevar a cabo durante el primer sprint del proyecto



El segundo sprint del proyecto

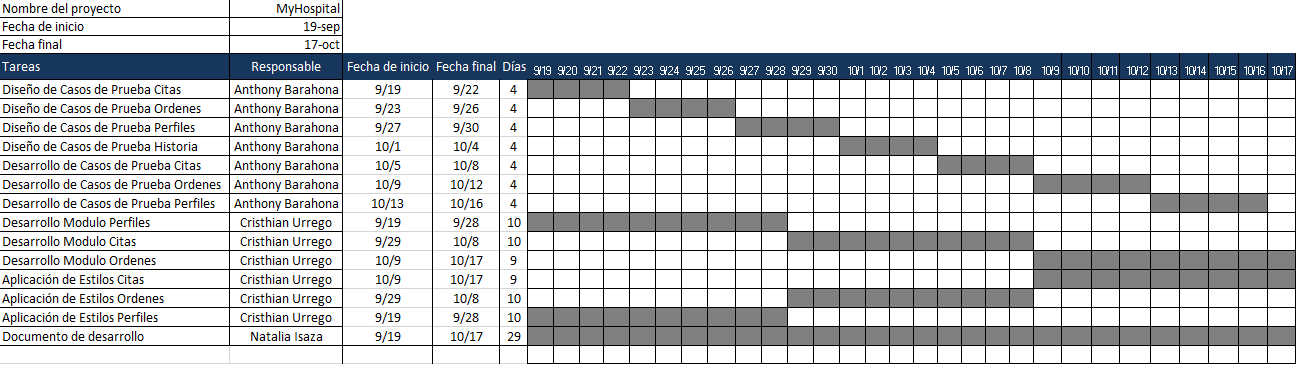


Ilustración 1 Cronograma sprint

## Glosario, acrónimos y definiciones

Con el fin de clarificar la terminología utilizada en el presente documento, se adjunta un breve prontuario de acrónimos, siglas y términos de uso específico.

|  |  |
| --- | --- |
| Bug | En el contexto de desarrollo SW, error descubierto en un programa |
| DB | Siglas en ingles de Data Base. Lo que en español, Base de datos |
| COTECNOVA | Corporación de Estudios Tecnológicos del Norte del Valle |
| CU | Caso de uso |
| FK | Foreign Key |
| FW | Framework. Terminología de arquitectura de desarrollo |
| GC | Gestión de calidad |
| HW | Hardware |
| IEEE | Institute of Electrical and Electronics Engineers |
| ISO | International Organization for Standarization |
| OS | Operating System |
| PK | Primary Key |
| RF | Requisito Funcional |
| RNF | Requisito no Funcional |
| SCRUM | Metodología organizativa relacionada con los sistemas ágiles de desarrollo |
| SGBD | Sistema Gestor de Bases de Datos. También se conoce como DBMS |
| SI | Sistema de Información |
| SW | Software |
| Stakeholder | Parte interesada en el proceso de desarrollo, como interno o como CL, en terminología SCRUM |
| UML | Unified Modeling Language |

# VISTA GENERAL DEL PROYECTO

## Propósito, alcance y objetivos

El propósito del proyecto My Hospital, una iniciativa de un grupo de estudiantes de la carrera de Ingeniería de Sistemas por ciclos propedéuticos de COTECNOVA. Quienes, en cumplimiento a las actividades propuestas por el docente encargado de la asignatura de Ingeniería de Software II, Establecen desarrollar un software de gestión para el manejo de información y recursos que posee un hospital.

Para un correcto funcionamiento del sistema, las características requeridas por el éste deberán ser las siguientes:

* Captura de información
  + Usuarios del sistema deberán suministrar la información perteneciente a los pacientes
  + Usuarios del sistema con un rol especifico(superadmin, admin, médico) en el deberán suministrar la información perteneciente a cada una de las citas a las que asista o no un paciente
* Explotación de la información
  + Usuarios a fin podrán servirse de la información que remite un médico (rol de usuario) después de cada cita (consulta médica asistida o no por un paciente)
  + Usuarios podrán servirse de una búsqueda recursiva que contiene el historial clínico de cada paciente

## Aclaraciones

Sabiendo que la metodología utilizada para este proyecto es SCRUM, es indispensable hacer mención de cómo funciona, sus roles de participación y el proceso de desarrollo que contempla.

Es una metodología ágil de desarrollo que permite abordar proyectos complejos desarrollados en

Entornos dinámicos y cambiantes de un modo flexible. Se caracteriza por su flexibilidad y adaptación a los contextos de la organización y del cliente, permite presentar resultados anticipados al producto final, etc.

## ¿Qué es SCRUM?

## Roles de usuario.

Existen cinco roles dentro de la metodología SCRUM, los cuales son:

* Product owner: Es el cliente o el representante de él.
* Scrum master: Es el representante y director del proyecto, su función es la de gestionar y simplificar la ejecución de tareas dentro del proceso de desarrollo.
* Scrum team: Son los individuos encargados de desarrollar el producto según indicadores. Estos a su vez se dividen en tres partes que son.
  + Desarrollador: aquel o aquellos que deben crear el código fuente de la iteración, generar los manuales técnicos y de usuario.
  + Analista: aquel o aquellos que deben especificar cada proceso y crear los soportes o documentos para futuras consultas del equipo (Team).
  + Tester: aquel o aquellos encargados de generar documentos de planeación y soporte de las pruebas que se ejecutaran para probar las funcionalidades del sistema, además, ejecutar también las mismas.

## Proceso de desarrollo.

El proceso de desarrollo de SCRUM se realiza por ciclos cortos y de duración fija (iteraciones de 2 semanas o más). Cada iteración debe proporcionar un resultado que incremente el producto final, siendo una posible entrega para alguna solicitud del cliente.

# DISEÑO Y ARQUITECTURA DEL SISTEMA

## Descripción general

El sistema de gestión My Hospital está preparado para la actual demanda digital y cuenta con unas funcionalidades descritas a continuación:

Módulo de usuario, que permite la gestión de usuarios encargados de administrar el sistema; módulo de citas, enfocado al desarrollo de las citas solicitadas y asistidas por cada paciente del hospital; módulo de órdenes, diseñado para la gestión de las prescripciones realizadas durante las citas médicas y que a su vez corresponden a una única cita; un apartado de consulta llamado “Historia”, esta consulta es recursiva y acude directamente a la historia clínica de cada paciente vinculado al hospital.

## Ingeniería de requisitos

## Requisitos funcionales

## Consulta

|  |  |  |
| --- | --- | --- |
| CODIGO | RF1 | |
| DESCRIPCION | Consulta realizada para conocer la historia, citas u órdenes medicas asociadas a un paciente | |
| PRIORIDAD | Media | |
| ACCION INDICADORA | Paso 1 | Solicita búsqueda relacionada de un PACIENTE |
| Paso 2 | Mostrar la solicitud realizada |
| Paso3 | Finalizar consulta |
| RF ASOCIADOS |  | |

## Manejar/administrar modulo

|  |  |  |
| --- | --- | --- |
| CODIGO | RF2 | |
| DESCRIPCION | Manejar (crear, editar o eliminar) un USUARIO, PACIENTE, CITA o una ORDEN | |
| PRIORIDAD | Alta | |
| ACCION INDICADORA | Paso 1 | Solicita realizar alguna acción sobre un modulo |
| Paso 2 | Mostrar las posibles opciones para manejar un modulo |
| Paso3 | Manejar un modulo |
| Paso 4 | Guardar cambios del manejo del modulo |
| Paso 5 | Finalizar |
| RF ASOCIADOS |  | |

## Manejar/Administrar Citas

|  |  |  |
| --- | --- | --- |
| CODIGO | RF3 | |
| DESCRIPCION | Manejar (crear, modificar o cancelar) una CITA para un PACIENTE | |
| PRIORIDAD | Alta | |
| ACCION INDICADORA | Paso 1 | Solicita realizar alguna acción sobre una CITA para un PACIENTE |
| Paso 2 | Mostrar las posibles opciones para manejar una CITA para un PACIENTE |
| Paso3 | Manejar una CITA para un PACIENTE |
| Paso 4 | Guardar cambios del manejo de la CITA para un PACIENTE |
| Paso 5 | Finalizar |
| RF ASOCIADOS |  | |

## Manejar/Administrar Pacientes

|  |  |  |
| --- | --- | --- |
| CODIGO | RF4 | |
| DESCRIPCION | Manejar (crear, editar) un PACIENTE en la BD | |
| PRIORIDAD | Media | |
| ACCION INDICADORA | Paso 1 | Solicita realizar alguna acción sobre un PACIENTE |
| Paso 2 | Mostrar las posibles opciones para manejar un PACIENTE |
| Paso3 | Manejar un PACIENTE |
| Paso 4 | Guardar cambios del manejo del PACIENTE |
| Paso 5 | Finalizar |
| RF ASOCIADOS |  | |

## Manejar/Administrar Cita medica

|  |  |  |
| --- | --- | --- |
| CODIGO | RF5 | |
| DESCRIPCION | Manejar la CITA de un PACIENTE en la BD | |
| PRIORIDAD | Alta | |
| ACCION INDICADORA | Paso 1 | Solicita realizar alguna acción sobre una CITA del PACIENTE |
| Paso 2 | Mostrar las posibles opciones para manejar una CITA del PACIENTE |
| Paso3 | Manejar una CITA del PACIENTE |
| Paso 4 | Guardar cambios del manejo de la CITA del PACIENTE |
| Paso 5 | Finalizar |
| RF ASOCIADOS |  | |

## Otros requisitos

* **Ruby on Rail**

Ruby on Rails es un entorno de desarrollo web de código abierto que está optimizado para la satisfacción de los programadores y para la productividad sostenible. Te permite escribir un buen código evitando que te repitas y favoreciendo la convención antes que la configuración.

“Rails”, es un armazón para construir aplicaciones web que acceden a bases de datos, Obteniendo un conjunto de librerías, automatismos y convenciones destinados a resolver los problemas más comunes a la hora de desarrollar una aplicación web, para que el programador pueda concentrarse en los aspectos únicos y diferenciales de su proyecto en lugar de los problemas recurrentes; es utilizado para ayudar a construir aplicaciones modernas de internet como: Twitter, Scribd, Hulu, Xing, Soundcloud, Basecamp, Github, etc.

* **PostgreSQL**

PostgreSQL es un gestor de bases de datos relacional y orientado a objetos. Su licencia y desarrollo es de código abierto, siendo mantenida por una comunidad de desarrolladores, colaboradores y organizaciones comerciales de forma libre y desinteresadamente. Esta comunidad es denominada PDGD (PostgreSQL Global Development Group, por sus siglas en inglés).

Es reconocido actualmente como uno de los sistemas gestores de bases de datos relacionales más potentes del mercado. Presenta fácil accesibilidad, es multiplataforma y está disponible para su utilización en casi todos los sistemas operativos utilizados en la actualidad sin disminuir su rendimiento.

* **Heroku**

Heroku es una plataforma como servicio de computación en la Nube que soporta distintos lenguajes de programación.

Heroku es propiedad de Salesforce.com, es una de las primeras plataformas de computación en la nube, que fue desarrollada desde junio de 2007, con el objetivo de soportar solamente el lenguaje de programación Ruby, pero posteriormente se ha extendido el soporte a Java, Node.js, Scala, Clojure y Python y PHP.

* **MaterializeCSS**

Es un framework que hace más fácil el proceso de creación de diseño web. Es un framework moderno y responsivo que se basa en Material Design. Tiene muchos componentes y clases predefinidas, para hacer uso de ellas mucho fácilmente. Aumenta el ritmo de desarrollo

Materialize cuenta con componentes personalizados con unos estilos por defecto, que se pueden manejar fácilmente para generar animaciones y ciertas transiciones, de tal manera que para el desarrollador web es mucho más fácil hacer su tarea de diseño web.

## Requisitos no funcionales

## RNF1

Los datos modificados en la DB deben ser actualizados permanentemente para todos los usuarios

## RNF2

Toda transacción y funcionalidad del sistema debe responder en menos de 5 segundos

## RNF3

Los permisos de acceso podrán ser cambiados únicamente por el usuario que tenga el rol de Superadmin

## RNF4

El sistema entrará en modo hibernación después de 30 minutos de inactividad

## RNF5

El sistema debe contar con mensajes de error informativos enfocados a usuario final

## RNF6

La aplicación debe tener un diseño “responsive”, que garantice una adecuada visualización desde múltiples dispositivos electrónicos

## RNF7

La interfaz gráfica del sistema debe ser interactiva para su fácil manipulación

## Diseño de base de datos

Una DB correctamente diseñada podrá proporcionar a los usuarios acceso a información actualizada y precisa. Para un funcionamiento idóneo de la base de datos es necesario:

## Propósito de la DB

La base de datos de My Hospital tiene como finalidad brindar a cada Usuario una lista detallada de opciones que depende del rol con el cual se inicie sesión, así:

|  |  |  |  |
| --- | --- | --- | --- |
| **Tipos de usuario** | SUPERADMIN | ADMIN | MEDICO |
| **Descripción** | Root | Administrativo | Doctor, especialista |

* SUPERADMIN: Tiene las características de un usuario tipo ROOT de cualquier sistema, de esta manera tiene derecho de manejar a su totalidad:
  + Usuarios del sistema.
  + Tablas del sistema y funcionalidades (PACIENTES, CITAS, ÓRDENES).
* ADMIN: Tiene control únicamente sobre los PACIENTES y todo en lo que a ellos concierne, es decir, las CITAS (solicitadas, asistidas o in-asistidas), ORDENES generadas por los doctores, y gestionar el MEDICO que atenderá la cita del paciente.
* MEDICO: Tiene control únicamente sobre las CITAS y ORDENES que genera en el sistema para cada PACIENTE
* PACIENTE: es el cliente del sistema, éste puede hacer solicitudes únicamente al usuario ADMIN.

Es de resaltar que como cualquier hospital, el sistema lleva una HISTORIA clínica perteneciente a cada paciente, es una búsqueda recursiva en el frontend de la aplicación la cual puede servirse a cualquier USUARIO previamente identificado.

## Estructura de la DB

Para lograr un óptimo entendimiento del funcionamiento de la DB, es preciso identificar el cómo están compuestas las tablas

.

* USUARIO

|  |  |
| --- | --- |
| **Descripción** | **Variable de tipo** |
| Id\_usuario | PK |
| Email | String |
| Contraseña | String |
| Nombre | String |
| Tipo\_usuario (Rol del usuario) | Integer |

* PERFILES

|  |  |
| --- | --- |
| **Descripción** | **Variable de tipo** |
| Id\_perfil | PK |
| Id\_usuario | FK |
| cedula | String |
| Email | String |
| Tipo\_sangre | String |
| Fecha\_nacimiento | Date |
| Nombre | String |
| Dirección | String |
| Teléfono | Integer |
| Alergias | String |
| Discapacidad | String |
| Notas | String |
| Id\_ep | FK |
| Id\_ciudad | FK |
| Id\_arl | FK |

* ORDENES

|  |  |
| --- | --- |
| **Descripción** | **Variable de tipo** |
| Id\_orden | PK |
| Id\_cita | FK |
| Tipo\_orden | Integer |
| Descripción | Text |

* CITAS

|  |  |
| --- | --- |
| **Descripción** | **Variable de tipo** |
| Id\_cita | PK |
| Id\_perfil | FK |
| Id\_usuario | FK |
| Fecha | Date |
| Diagnostico | Text |
| Asistencia | Boolean |

* CIUDAD

|  |  |
| --- | --- |
| **Descripción** | **Variable de tipo** |
| Id\_ciudad | PK |
| nombre | String |

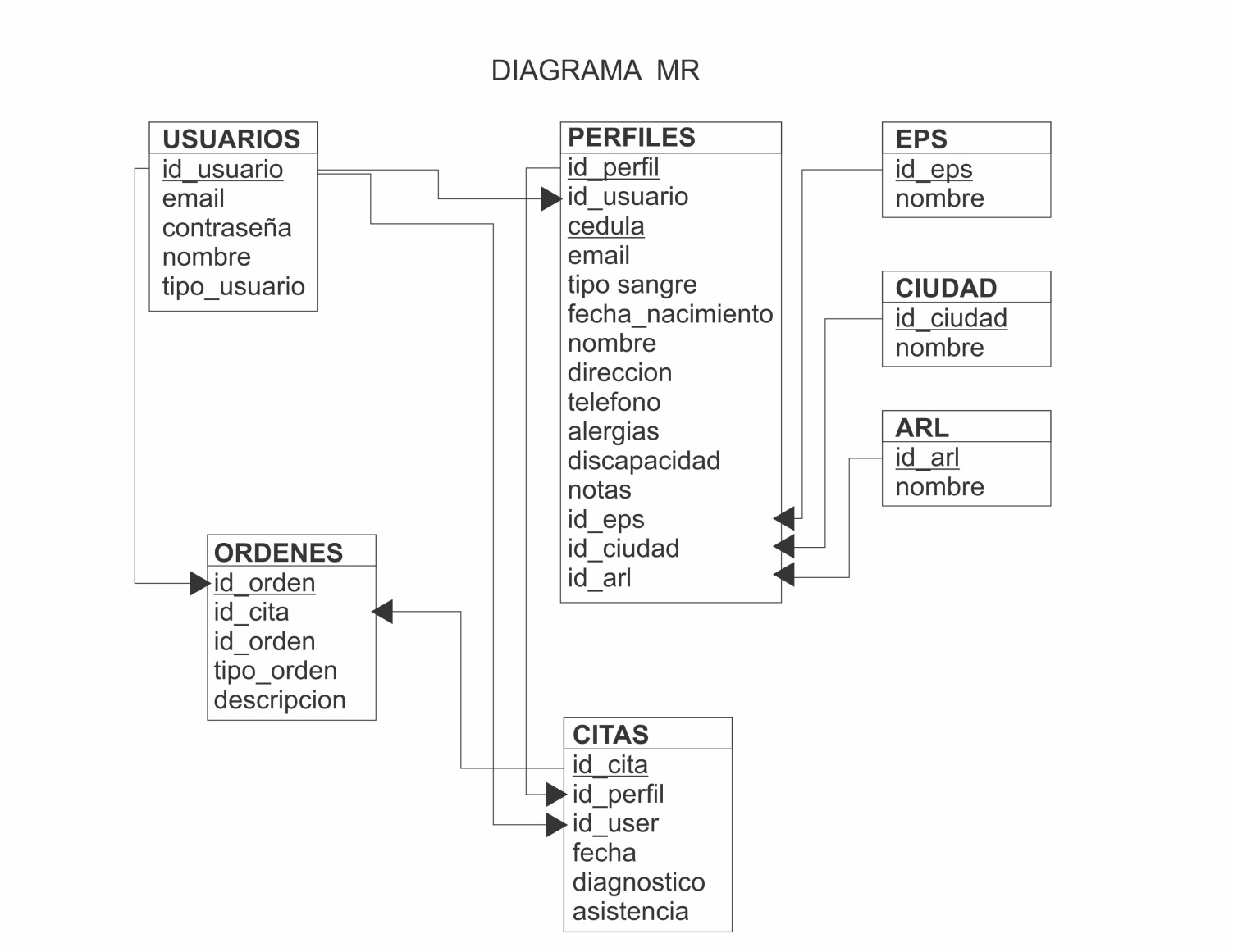
* EPS

|  |  |
| --- | --- |
| **Descripción** | **Variable de tipo** |
| Id\_eps | PK |
| nombre | String |

* ARL

|  |  |
| --- | --- |
| **Descripción** | **Variable de tipo** |
| Id\_arl | PK |
| nombre | String |

## Diseño de la DB

Ilustración 2 Diagrama DB Modelo Relacional

## Modelo conceptual

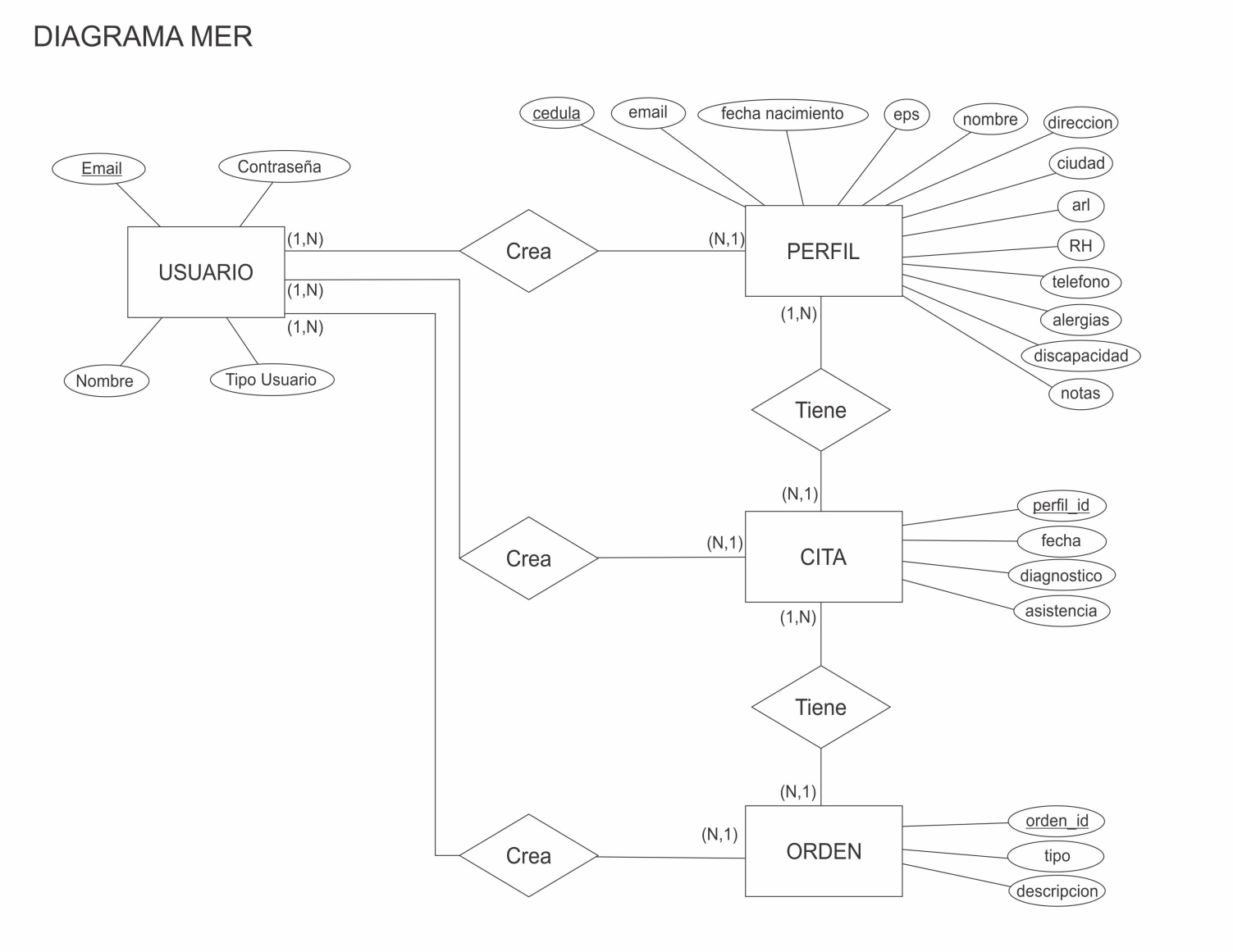


Ilustración 3 Diagrama DB Modelo entidad relación

# PROCESOS DEL SISTEMA

## Actores y casos de uso

A continuación se presenta el diagrama de caso de uso desde una perspectiva general, que abarca todo el funcionamiento interno del sistema.

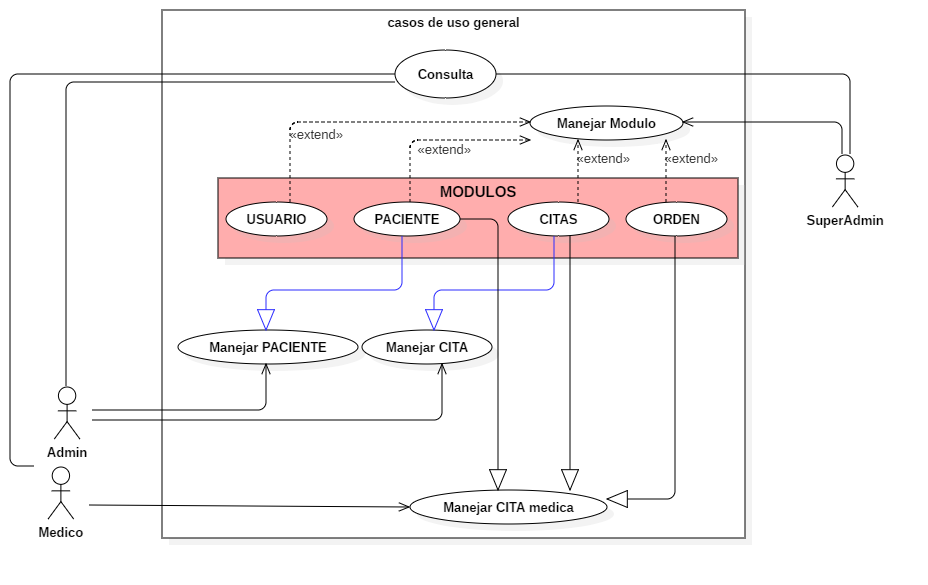


Ilustración 4 Diagrama CU general

## Actores

Se denomina actor a toda entidad externa del sistema que se relaciona con éste y que demanda una funcionalidad. Los actores definidos para el sistema My Hospital son:

* USUARIO: Es aquel individuo que opera el sistema. Para el sistema existen tres tipos de usuarios que son: SUPERADMIN, ADMIN y MEDICO
* PACIENTE: Es aquel individuo que se sirve del sistema y no demanda funcionalidades sobre el mismo.

## Casos de uso

Las siguientes ilustraciones representan casos de uso precisos del sistema.

## Caso de uso – 01 CONSULTA

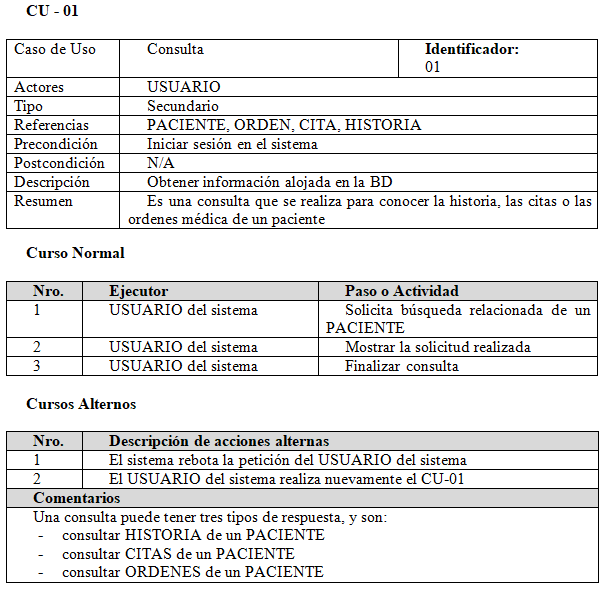


Ilustración 5 Caso de uso – 01 CONSULTA

## Caso de uso – 02 Manejar modulo

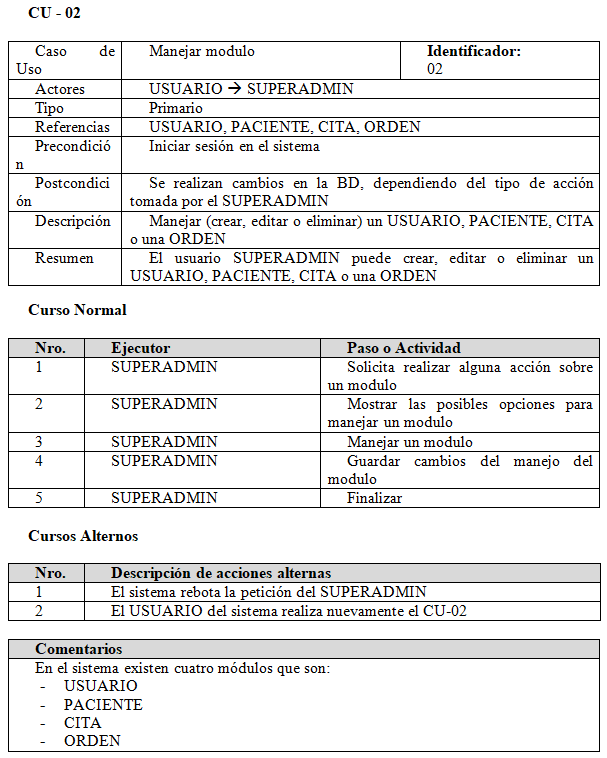


Ilustración 6 Caso de uso – 02 Manejar modulo

## Caso de uso – 03 Manejar CITA

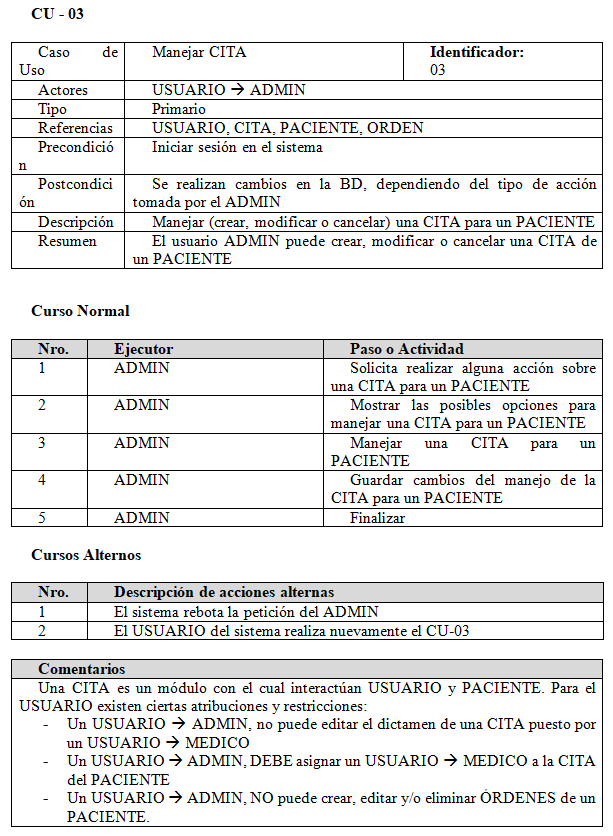


Ilustración 7 Caso de uso – 03 Manejar CITA

## Caso de uso – 04 Manejar PACIENTE

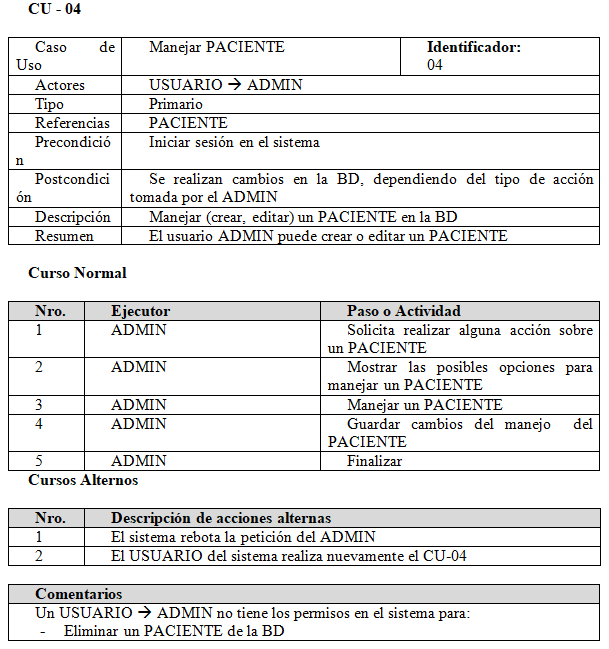


Ilustración 8 Caso de uso – 04 Manejar PACIENTE

## Caso de uso – 05 Manejar CITA médica

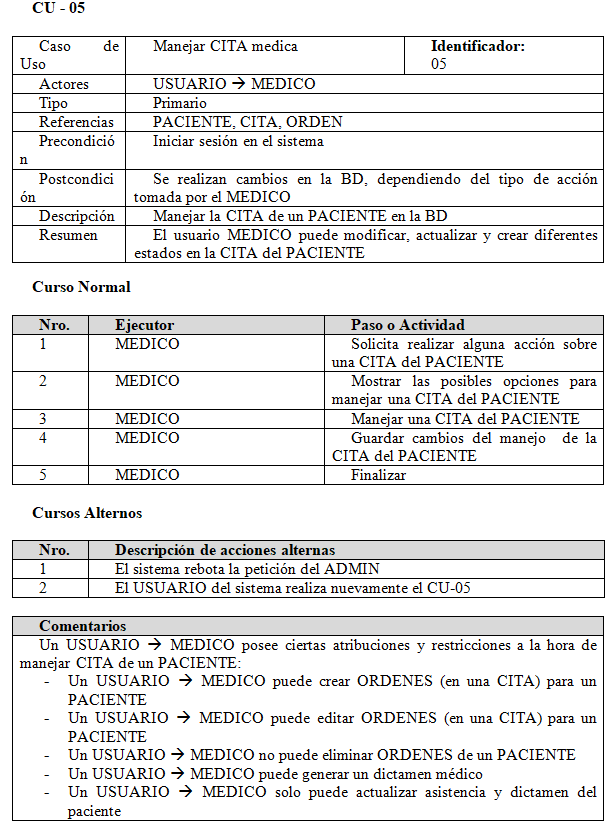


Ilustración 9 Caso de uso – 05 Manejar CITA medica

# PROTOTIPOS DEL SISTEMA

Las siguientes ilustraciones representan el prototipo de interfaz que podría apreciarse en el sistema.

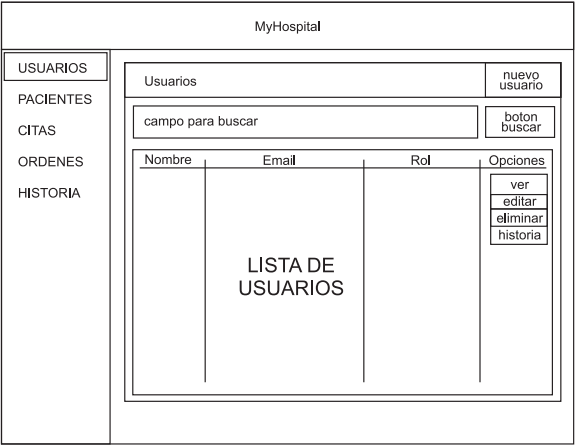


Ilustración 10 PS-Usuarios

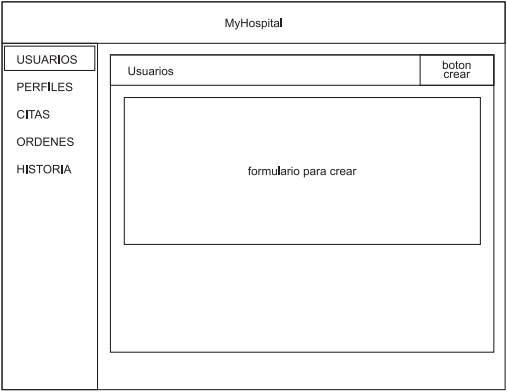


Ilustración 11 PS-Crear usuarios

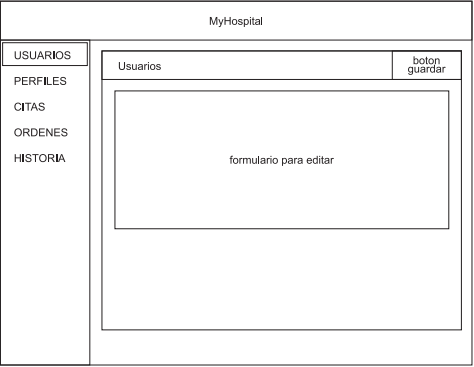


Ilustración 12 PS-Editar usuarios

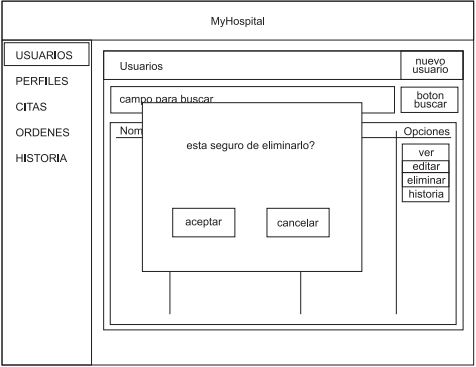


Ilustración 13 PS-Editar usuarios (eliminar)

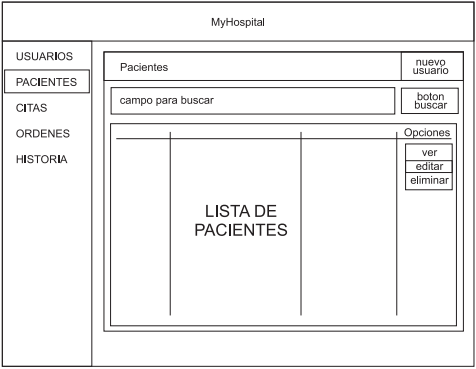


Ilustración 14 PS-Paciente

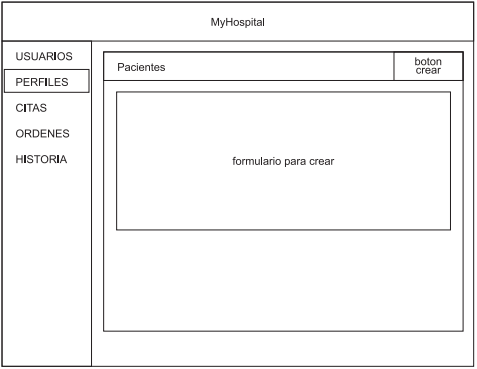


Ilustración 15 PS-Crear pacientes

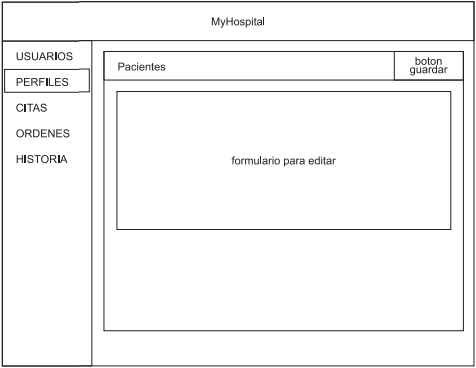


Ilustración 16 PS-Editar pacientes

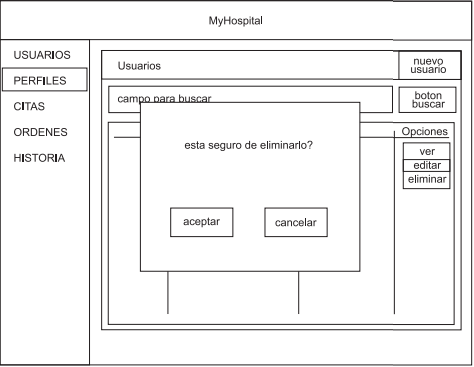


Ilustración 17 PS-Editar pacientes (eliminar)

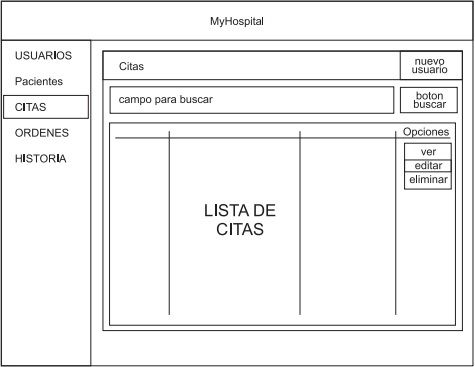


Ilustración 18 PS-Citas

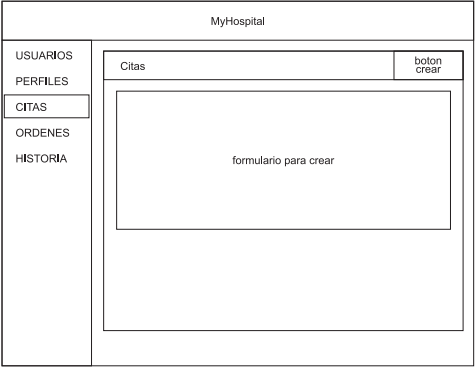


Ilustración 19 PS-Crear citas

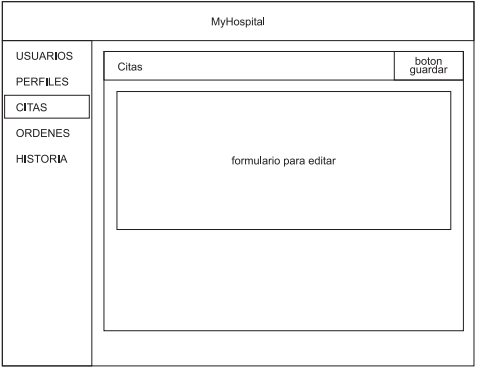


Ilustración 20 PS-Editar citas

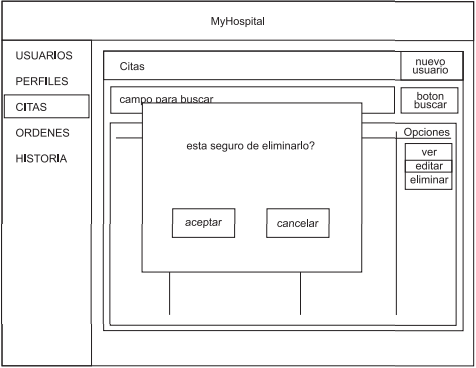


Ilustración 21 PS-Editar citas (eliminar)

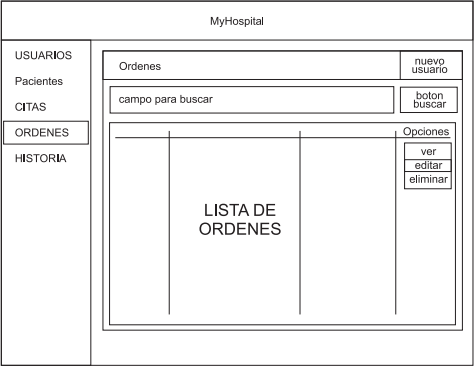


Ilustración 22 PS-Ordenes

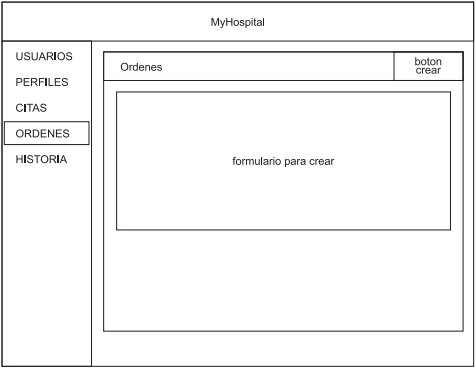


Ilustración 23 PS-Crear ordenes

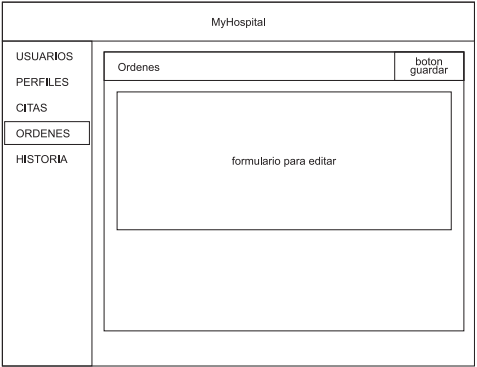


Ilustración 24 PS-Editar ordenes

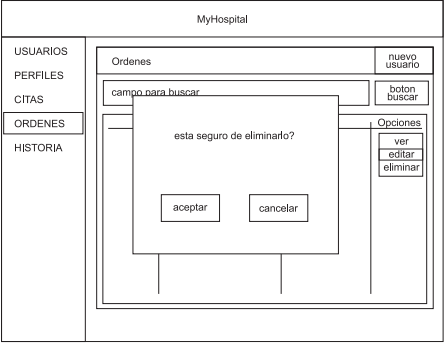


Ilustración 25 PS-Editar ordenes (eliminar)

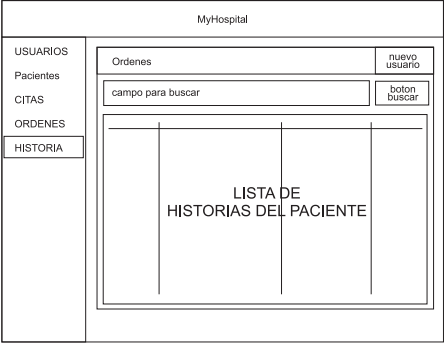


Ilustración 26 PS-Historias (Pacientes)

# MODELO DE DESARROLLO

MVC es una propuesta de diseño de software utilizada para implementar sistemas donde se requiere el uso de interfaces de usuario. Surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado, donde se potencie la facilidad de mantenimiento, reutilización del código y la separación de conceptos.

Su fundamento es la separación del código en tres capas diferentes, acotadas por su responsabilidad, en lo que se llaman Modelos, Vistas y Controladores, o lo que es lo mismo, Model, Views & Controllers, si lo prefieres en inglés. En este artículo estudiaremos con detalle estos conceptos, así como las ventajas de ponerlos en marcha cuando desarrollamos.

MVC es un "invento" que ya tiene varias décadas y fue presentado incluso antes de la aparición de la Web. No obstante, en los últimos años ha ganado mucha fuerza y seguidores gracias a la aparición de numerosos frameworks de desarrollo web que utilizan el patrón MVC como modelo para la arquitectura de las aplicaciones web.

Por qué MVC

La rama de la ingeniería del software se preocupa por crear procesos que aseguren calidad en los programas que se realizan y esa calidad atiende a diversos parámetros que son deseables para todo desarrollo, como la estructuración de los programas o reutilización del código, lo que debe influir positivamente en la facilidad de desarrollo y el mantenimiento.

Los ingenieros del software se dedican a estudiar de qué manera se pueden mejorar los procesos de creación de software y una de las soluciones a las que han llegado es la arquitectura basada en capas que separan el código en función de sus responsabilidades o conceptos. Por tanto, cuando estudiamos MVC lo primero que tenemos que saber es que está ahí para ayudarnos a crear aplicaciones con mayor calidad.

Modelos

Es la capa donde se trabaja con los datos, por tanto contendrá mecanismos para acceder a la información y también para actualizar su estado. Los datos los tendremos habitualmente en una base de datos, por lo que en los modelos tendremos todas las funciones que accederán a las tablas y harán los correspondientes selects, updates, inserts, etc.

No obstante, cabe mencionar que cuando se trabaja con MCV lo habitual también es utilizar otras librerías como PDO o algún ORM como Doctrine, que nos permiten trabajar con abstracción de bases de datos y persistencia en objetos. Por ello, en vez de usar directamente sentencias SQL, que suelen depender del motor de base de datos con el que se esté trabajando, se utiliza un dialecto de acceso a datos basado en clases y objetos.

Vistas

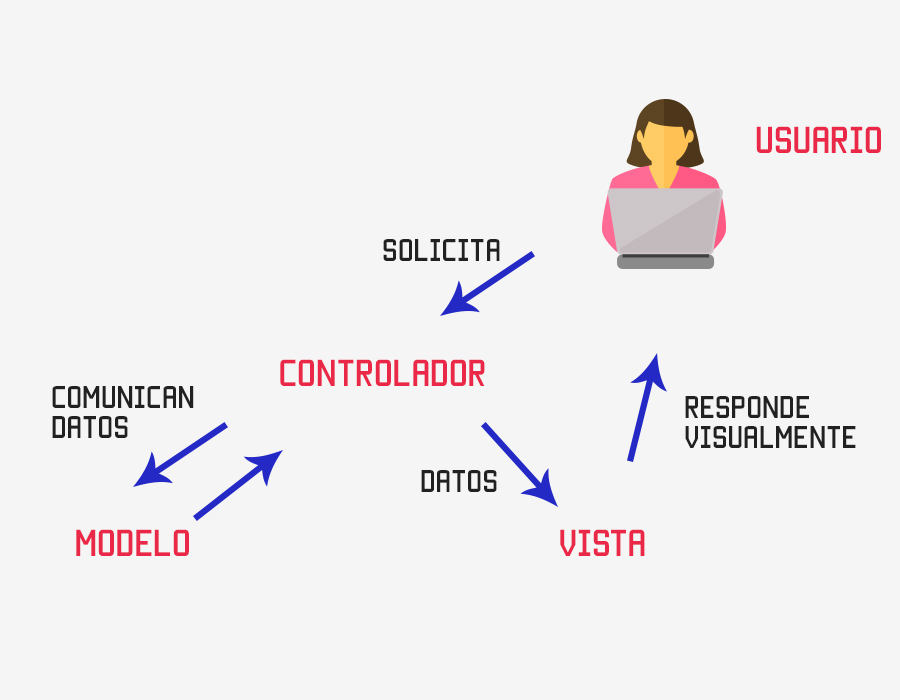
Las vistas, como su nombre nos hace entender, contienen el código de nuestra aplicación que va a producir la visualización de las interfaces de usuario, o sea, el código que nos permitirá renderizar los estados de nuestra aplicación en HTML. En las vistas nada más tenemos los códigos HTML y PHP que nos permite mostrar la salida.

En la vista generalmente trabajamos con los datos, sin embargo, no se realiza un acceso directo a éstos. Las vistas requerirán los datos a los modelos y ellas se generará la salida, tal como nuestra aplicación requiera.

Controladores

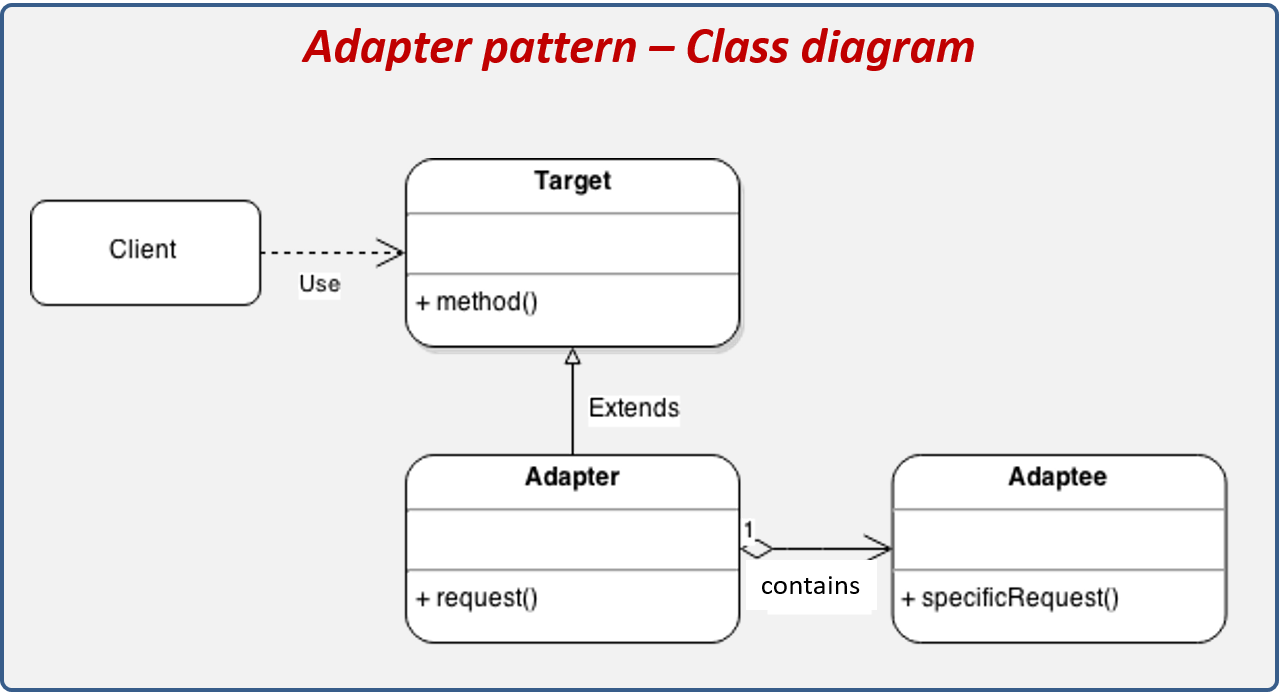
Contiene el código necesario para responder a las acciones que se solicitan en la aplicación, como visualizar un elemento, realizar una compra, una búsqueda de información, etc.

En realidad es una capa que sirve de enlace entre las vistas y los modelos, respondiendo a los mecanismos que puedan requerirse para implementar las necesidades de nuestra aplicación. Sin embargo, su responsabilidad no es manipular directamente datos, ni mostrar ningún tipo de salida, sino servir de enlace entre los modelos y las vistas para implementar las diversas necesidades del desarrollo.

****

# PATRÓNES DE DISEÑO

**ADAPTER:** El patrón de diseño Adapter es utilizado cuando tenemos interfaces de software incompatibles, las cuales a pesar de su incompatibilidad tiene una funcionalidad similar. Este patrón es implementado cuando se desea homogeneizar la forma de trabajar con estas interfaces incompatibles, para lo cual se crea una clase intermedia que funciona como un adaptador. Esta clase adaptador proporcionará los métodos para interactuar con la interface incompatible.



Estructura del patrón Adapter.

Los componentes que conforman el patrón son los siguientes:

Client: Actor que interactua con el Adapter.

Target: Interface que nos permitirá homogenizar la forma de trabajar con las interfaces incompatibles, esta interface es utilizada para crear los Adapter.

Adapter: Representa la implementación del Target, el cual tiene la responsabilidad de mediar entre el Client y el Adaptee. Oculta la forma de comunicarse con el Adaptee.

Adaptee: Representa la clase con interface incompatible.

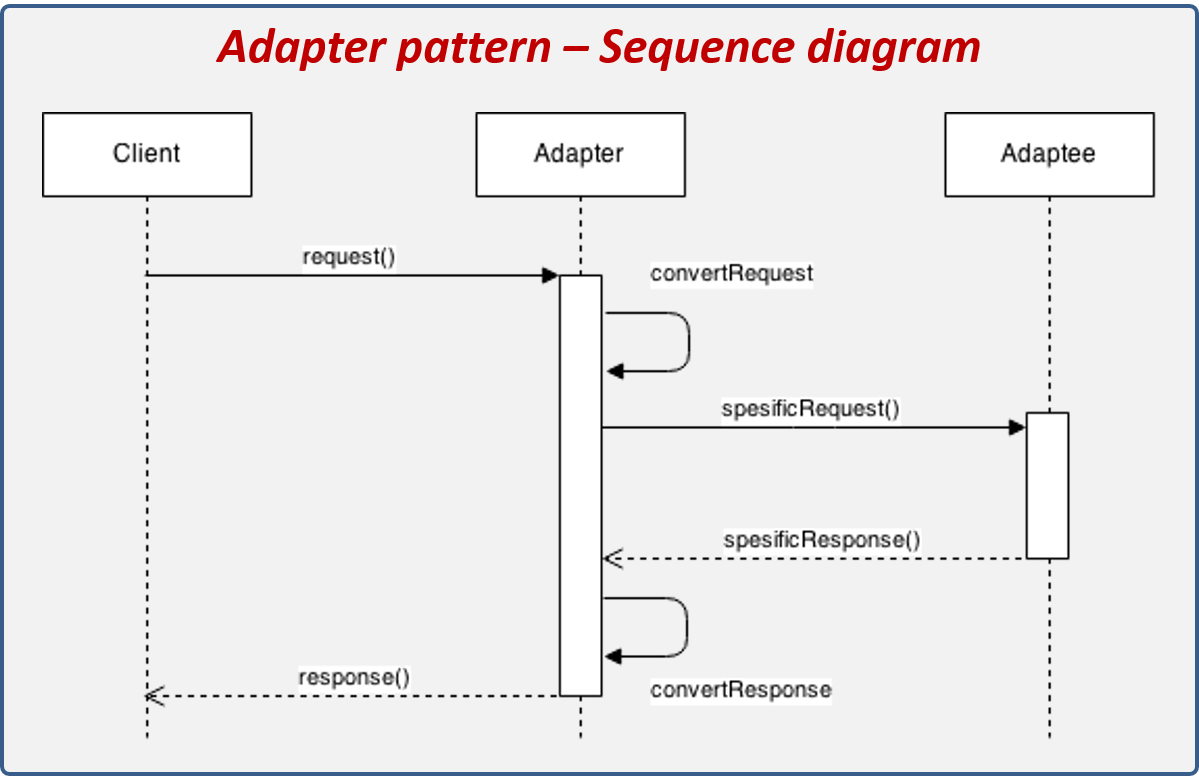


Diagrama de secuencia del patrón Adapter.

El Client invoca al Adapter con parámetros genéricos.

El Adapter convierte los parámetros genéricos en parámetros específicos del Adaptee.

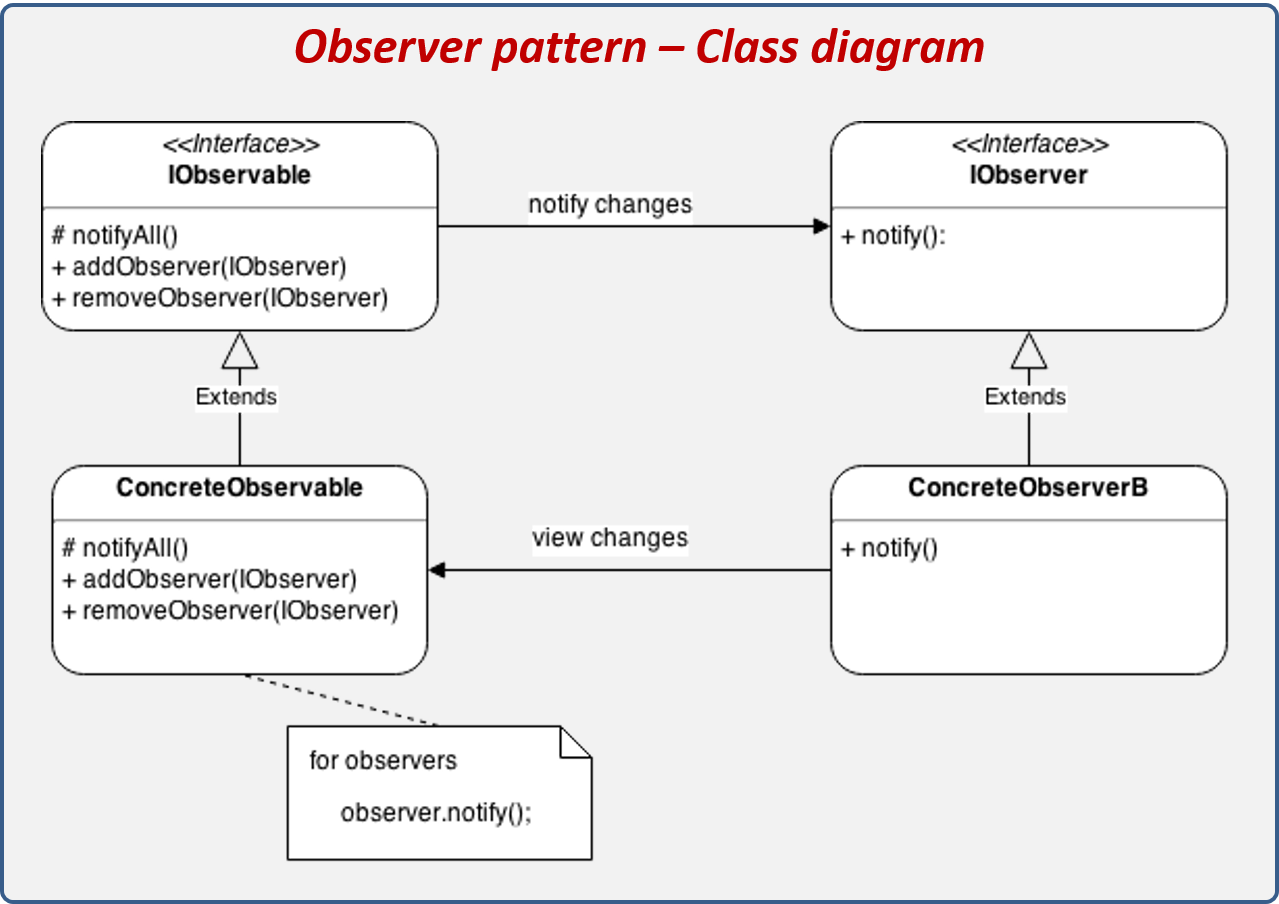
El Adapter invoca al Adaptee.

El Adaptee responde.

El Adapter convierte la respuesta del Adaptee a una respuesta genérica para el Client.

El Adapter responde al Client con una respuesta genérica.

**OBSERVER** El patrón de diseño Observer permite observar los cambios producidos por un objeto, de esta forma, cada cambio que afecte el estado del objeto observado lanzará una notificación a los observadores; a esto se le conoce como Publicador-Suscriptor. Observer es uno de los principales patrones de diseño utilizados en interfaces gráficas de usuario (GUI), ya que permite desacoplar al componente gráfico de la acción a realizar.



Estructura del patrón de diseño Observer.

Los componentes del patrón se explican a continuación:

IObservable: Interface que deben de implementar todos los objetos que quieren ser observados, en ella se definen los métodos mínimos que se deben implementar.

ConcreteObservable: Clase que desea ser observada, ésta implementa IObservable y debe implementar sus métodos.

IObserver: Interfaces que deben implementar todos los objetos que desean observar los cambios de IObservable.

ConcreteObserver: Clase concreta que está atenta de los cambios de IObserver, esta clase hereda de IObserver y debe de implementar sus métodos.

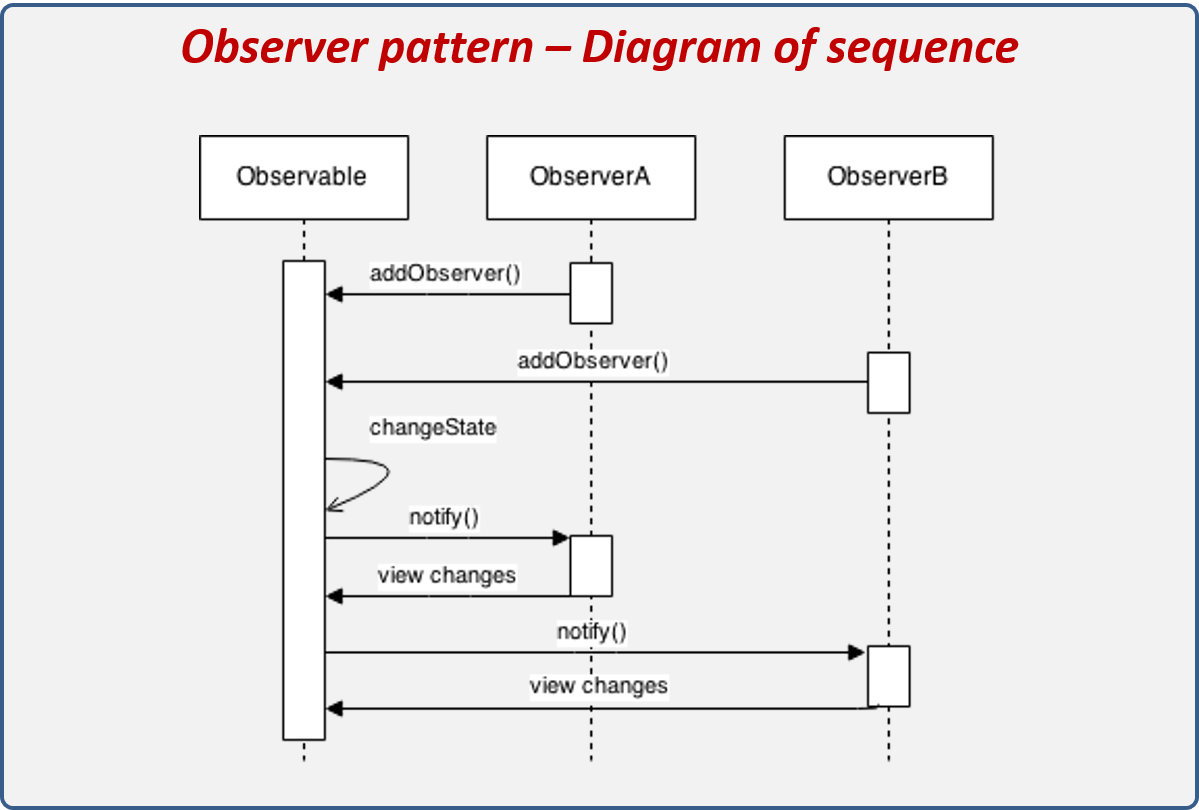


Diagrama de secuencia del patrón Observer.

El ObserverA se registra con el objeto Observable para ser notificado de algún cambio.

El ObserverB se registra con el objeto Observable para ser notificado de algún cambio.

Ocurre algún cambio en el estado del Observable.

Todos los Observers son notificados con el cambio ocurrido.

# DIAGRAMAS DE ACTIVIDADES

## 6.1 Log in

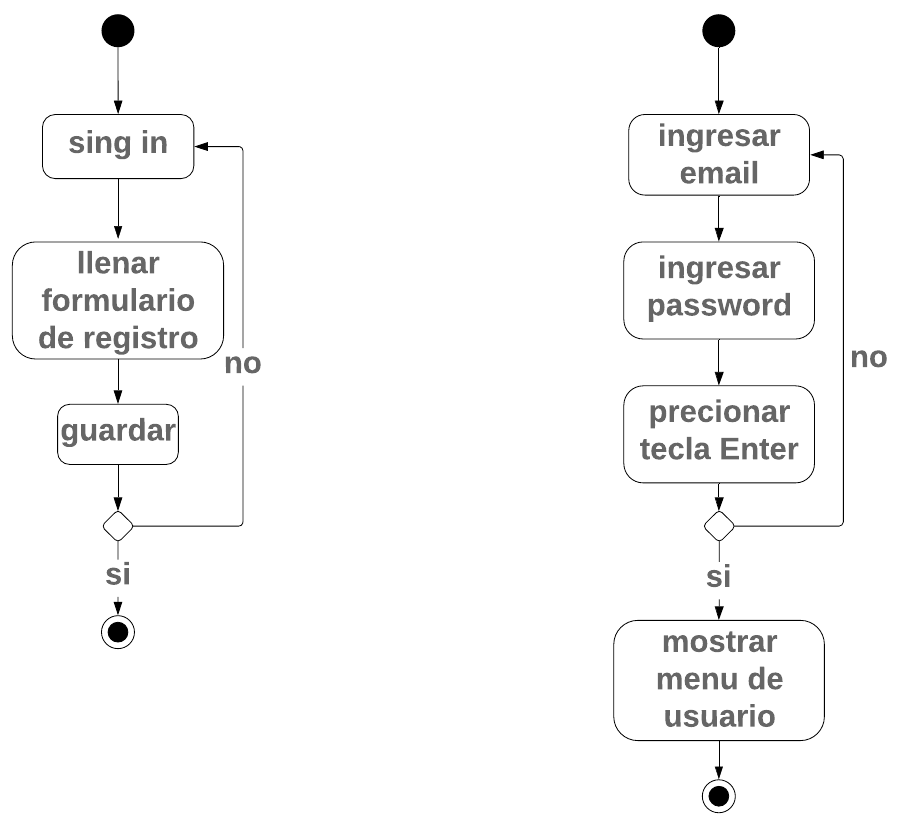


Ilustración 27 log in usuario

## 6.2 Usuario

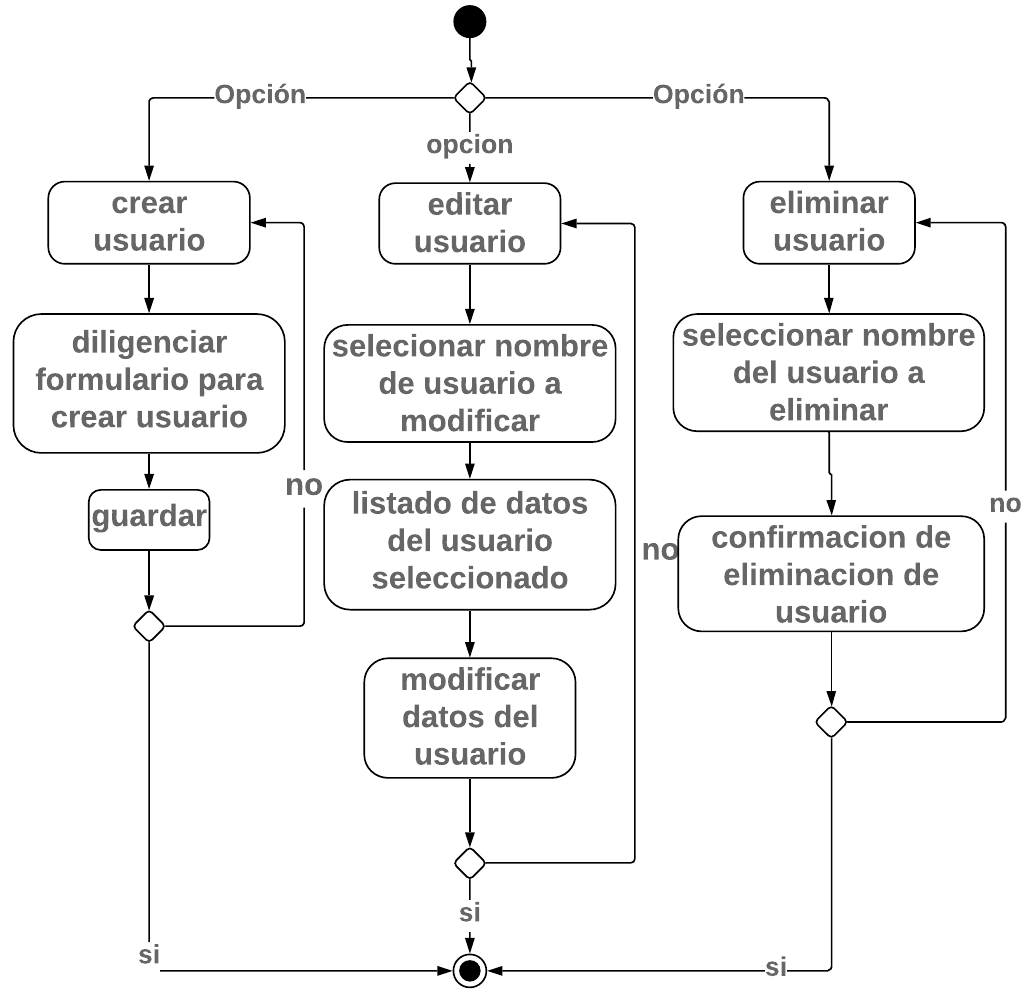


Ilustración 28 modificaciones usuario

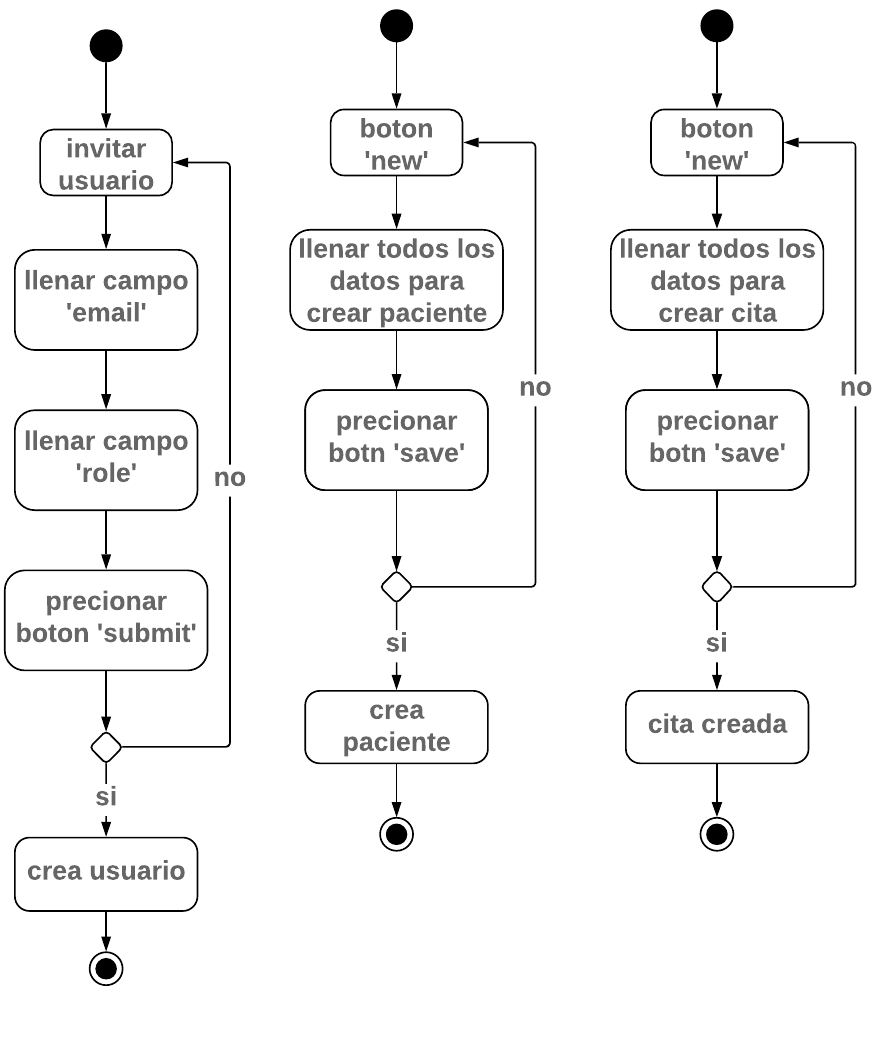


Ilustración 29 crear usuario

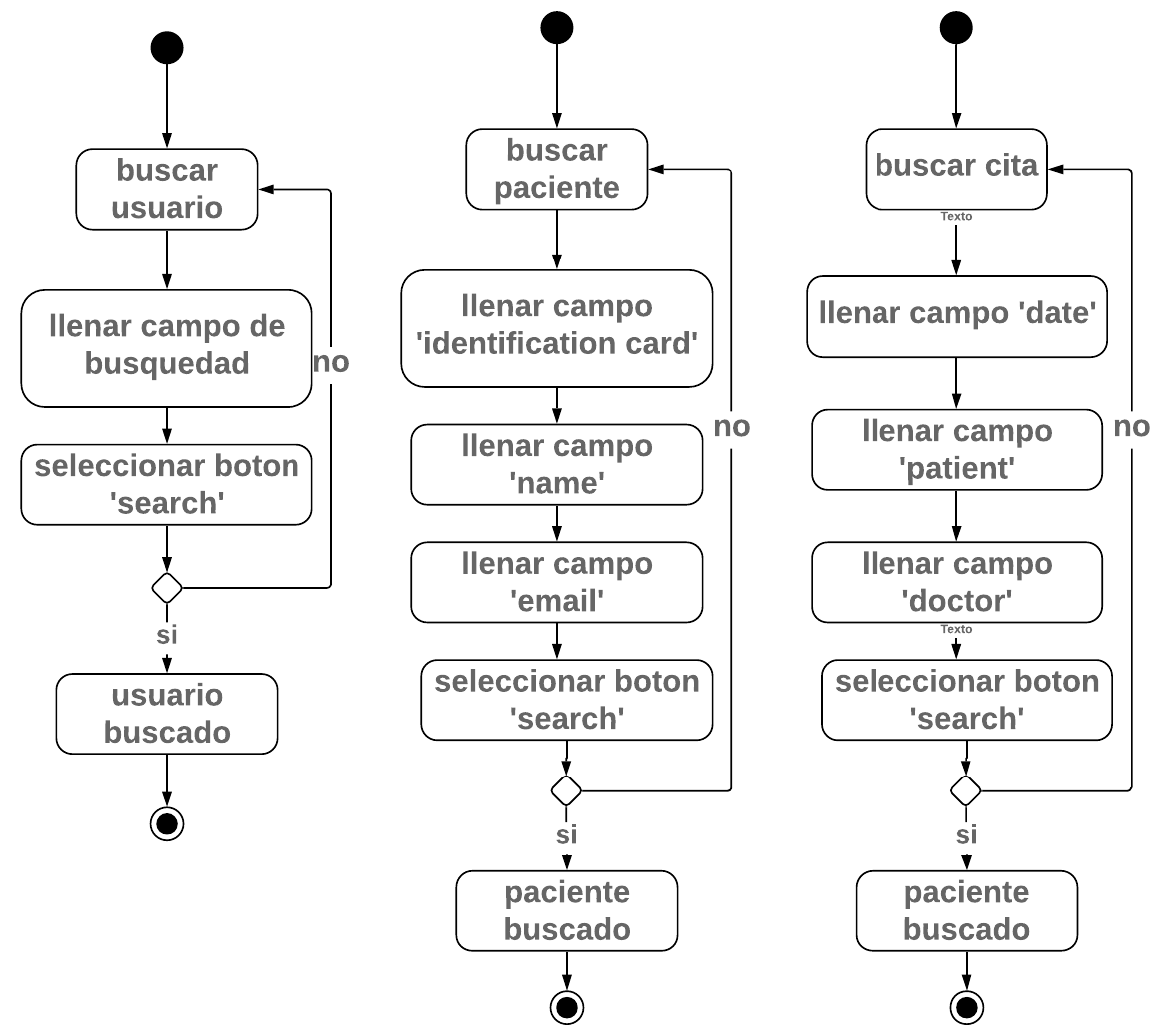


Ilustración 30 buscar usuario

## 6.3 Pacientes

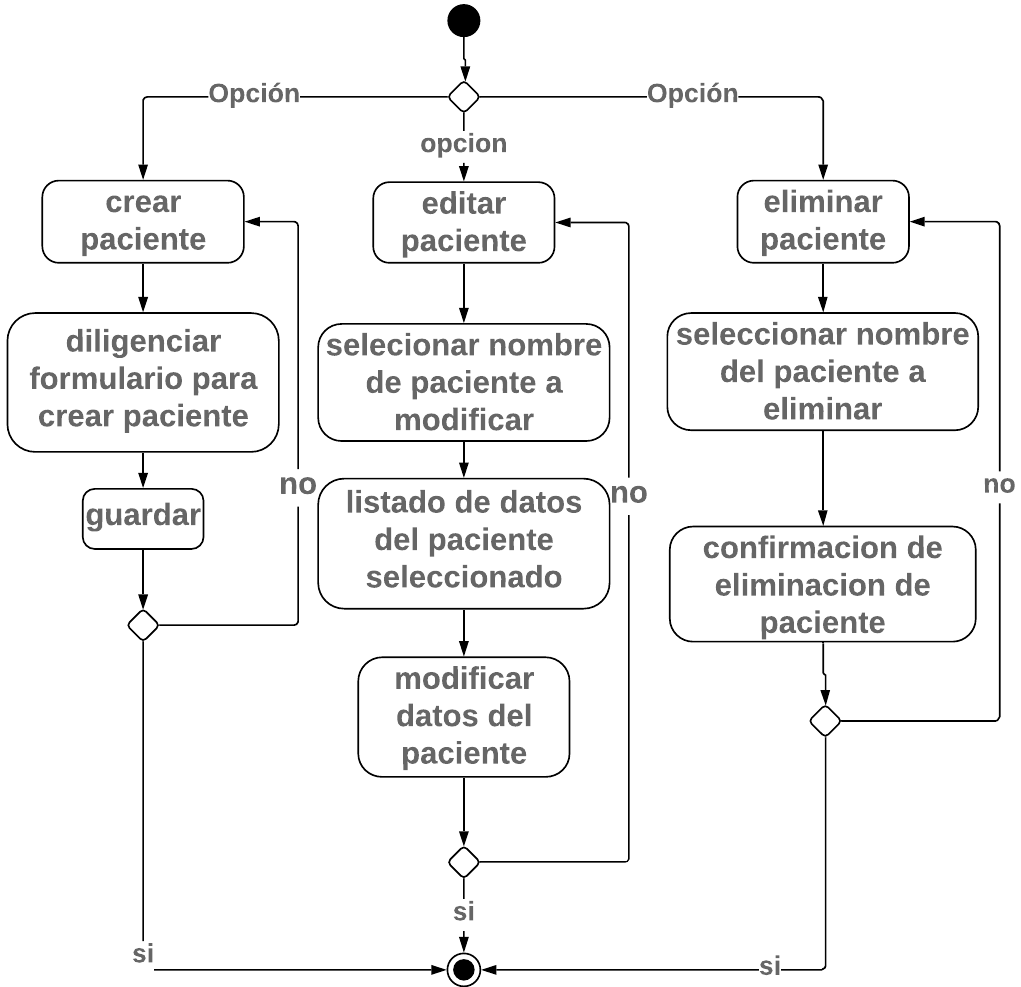


Ilustración 31 modificaciones paciente

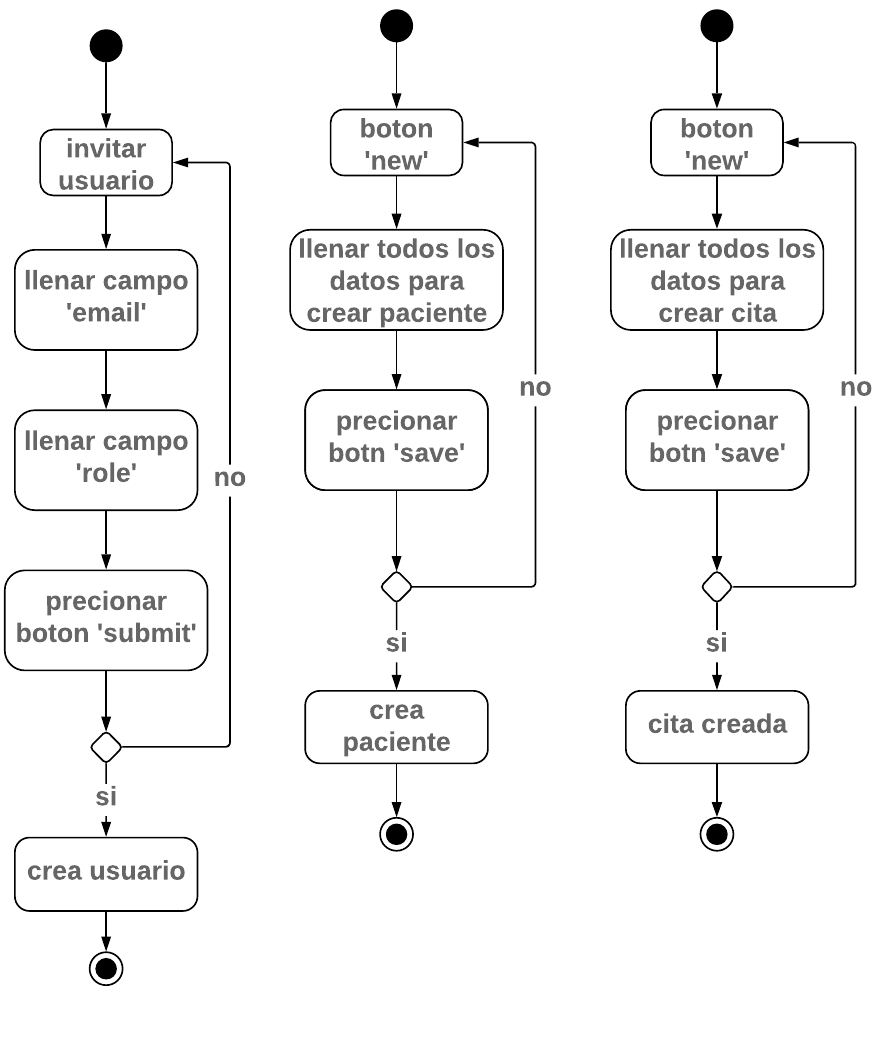


Ilustración 32 crear paciente

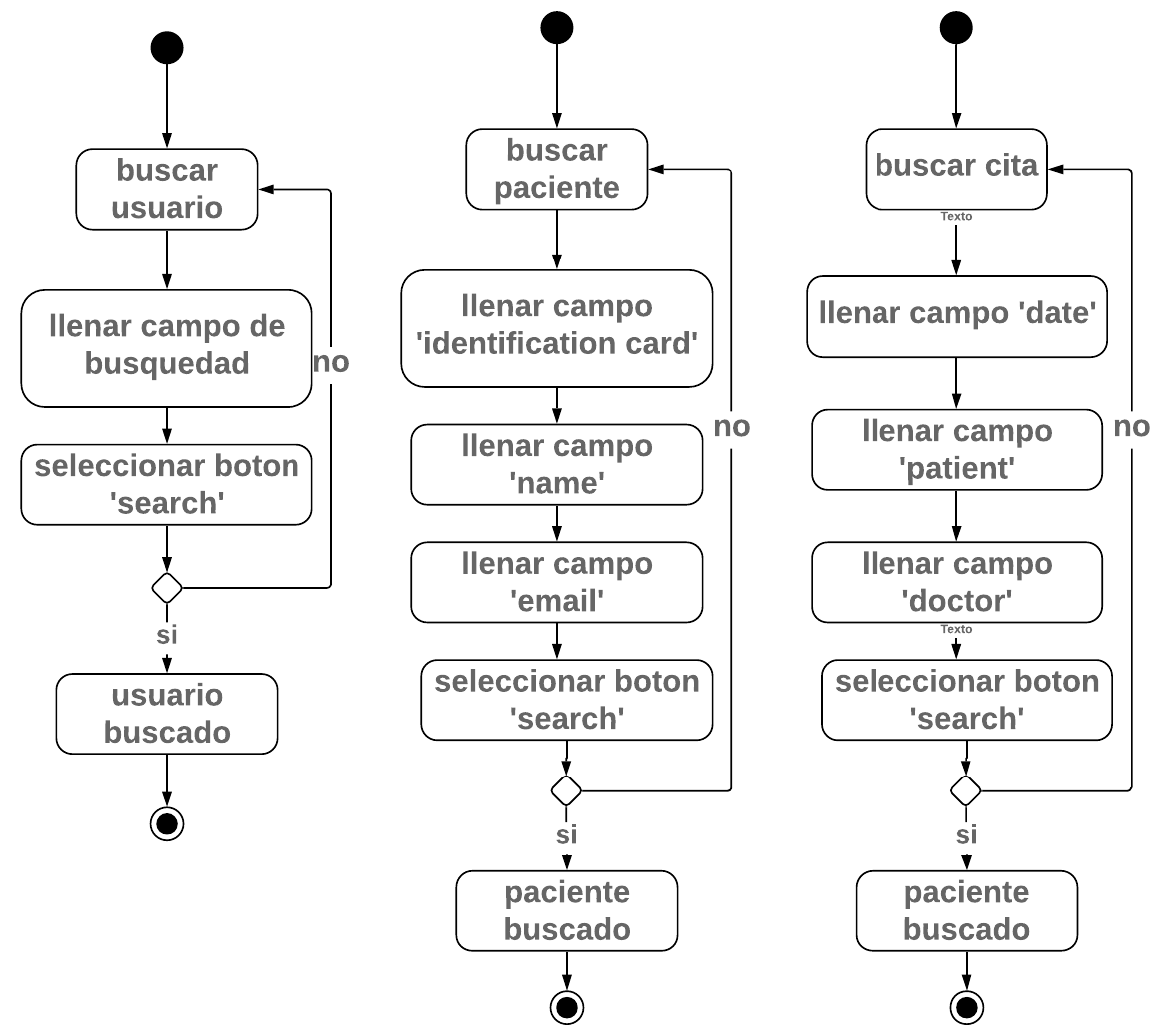


Ilustración 33 buscar paciente

## Citas

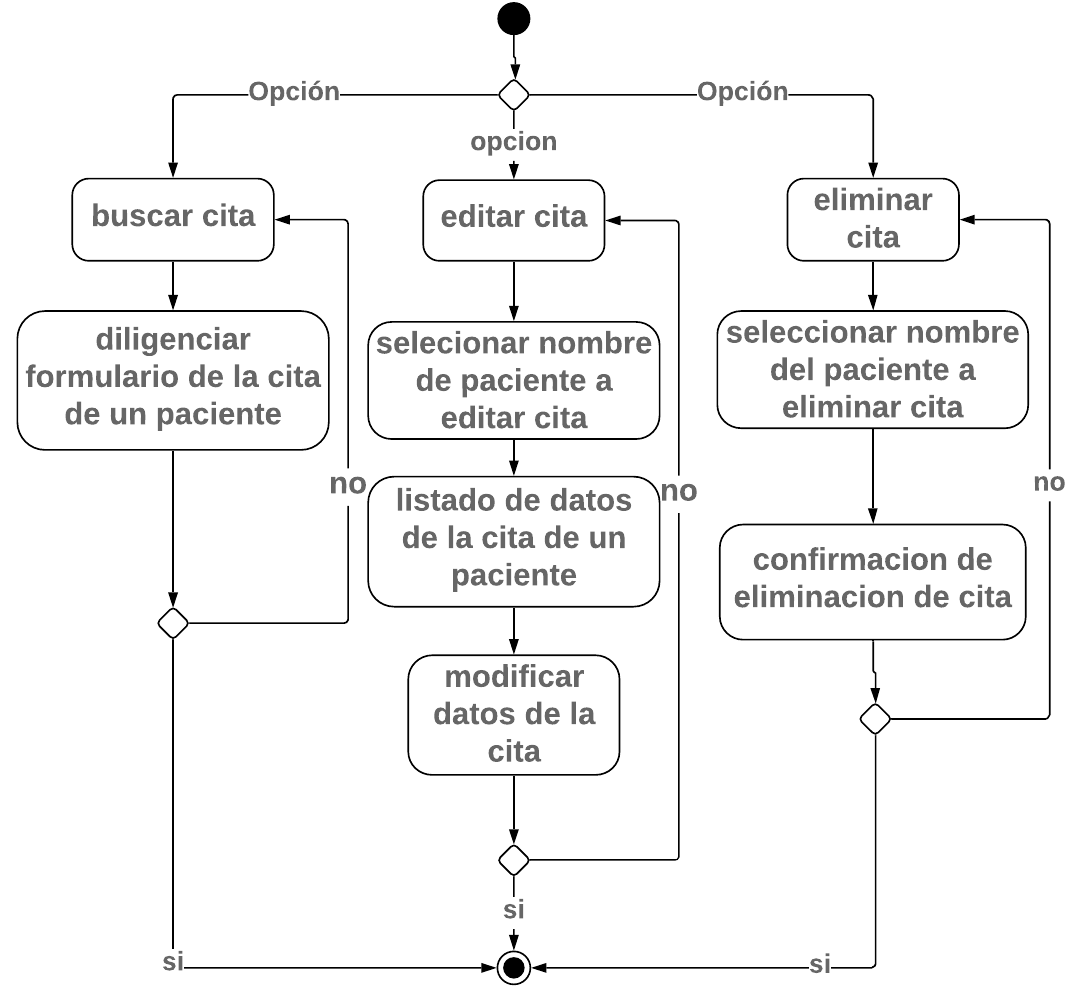


Ilustración 34 modificaciones citas

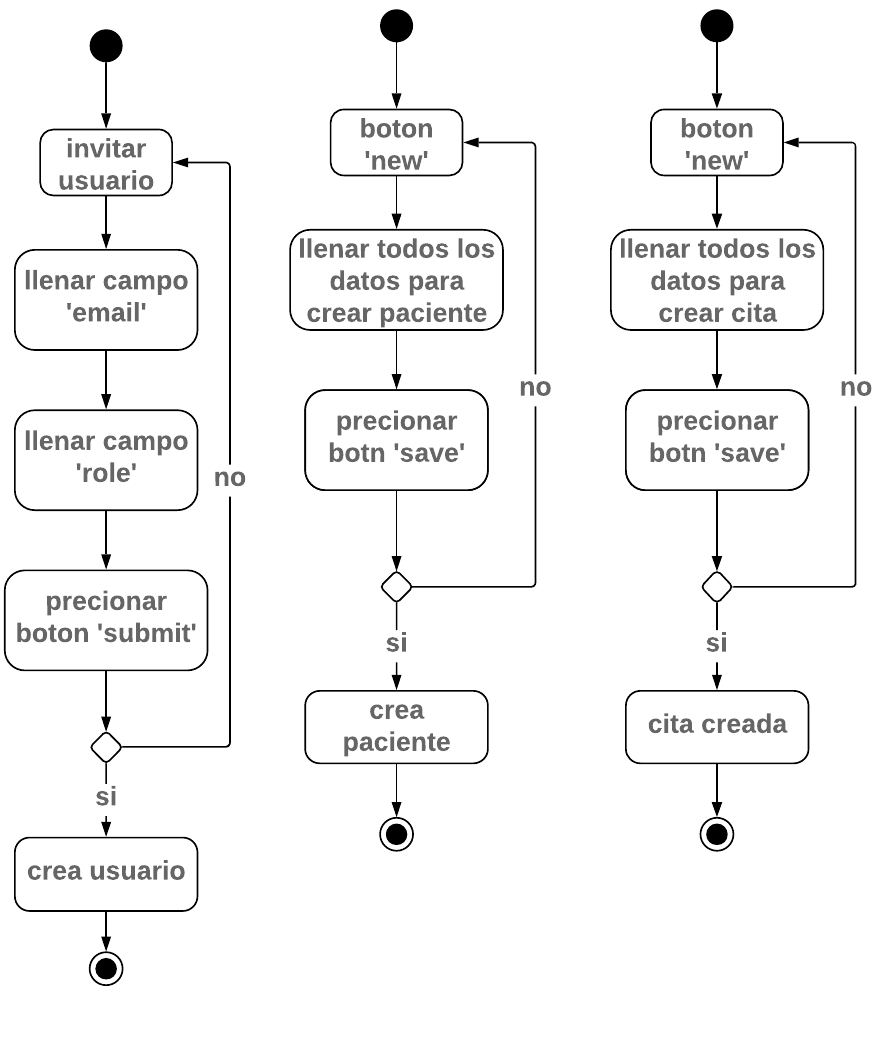


Ilustración 35 crear cita

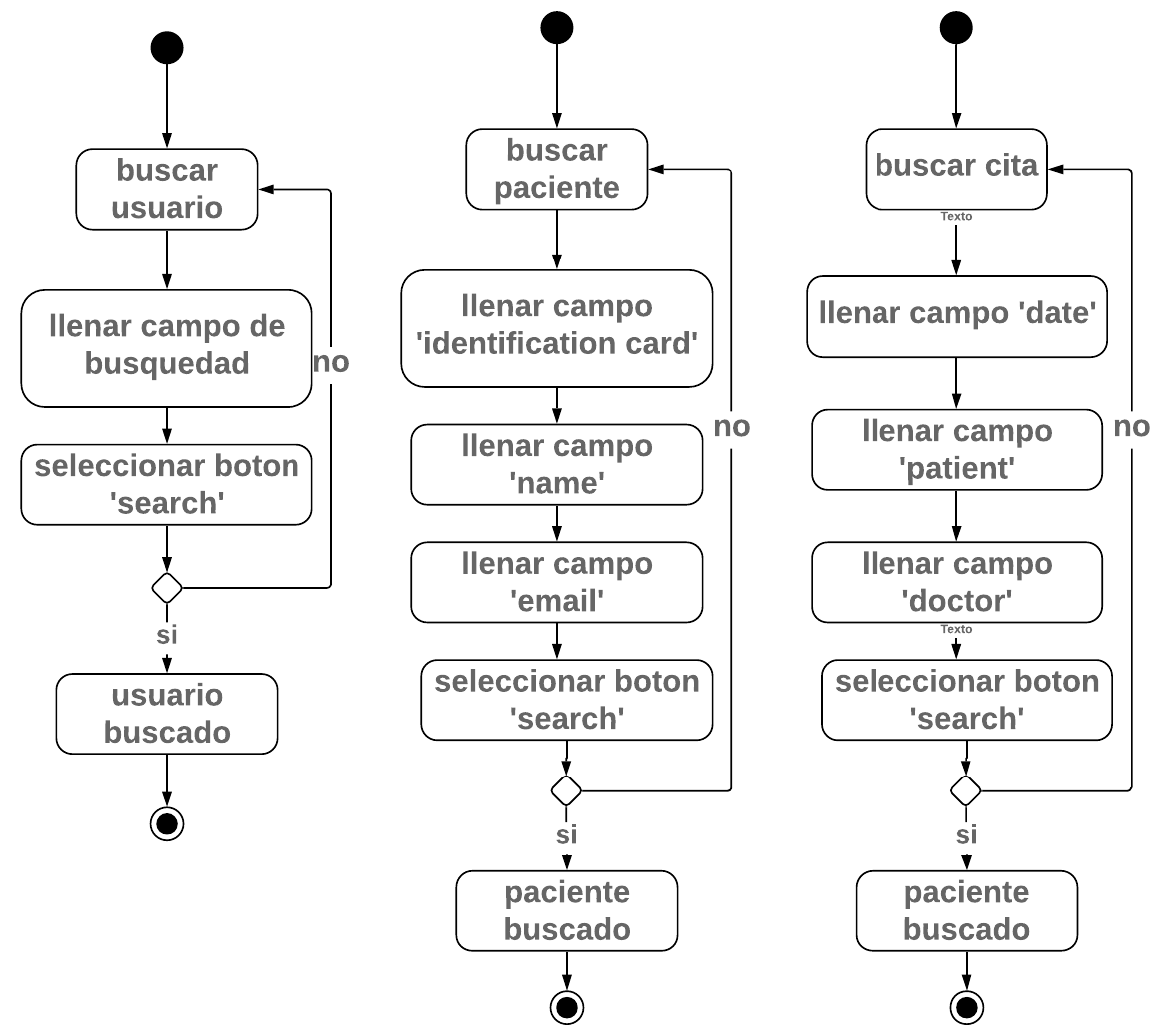


Ilustración 36 buscar cita

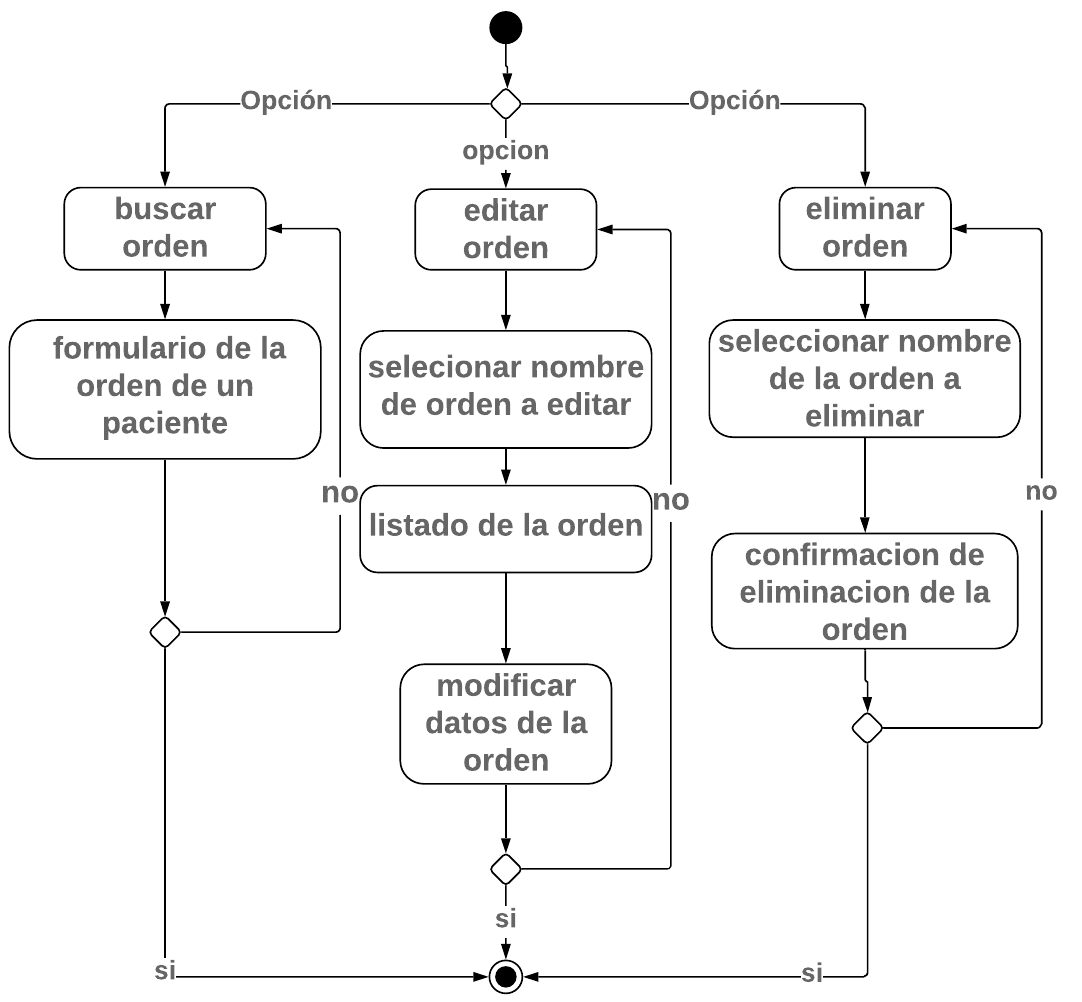


Ilustración 37 modificar ordenes

## Ajustes

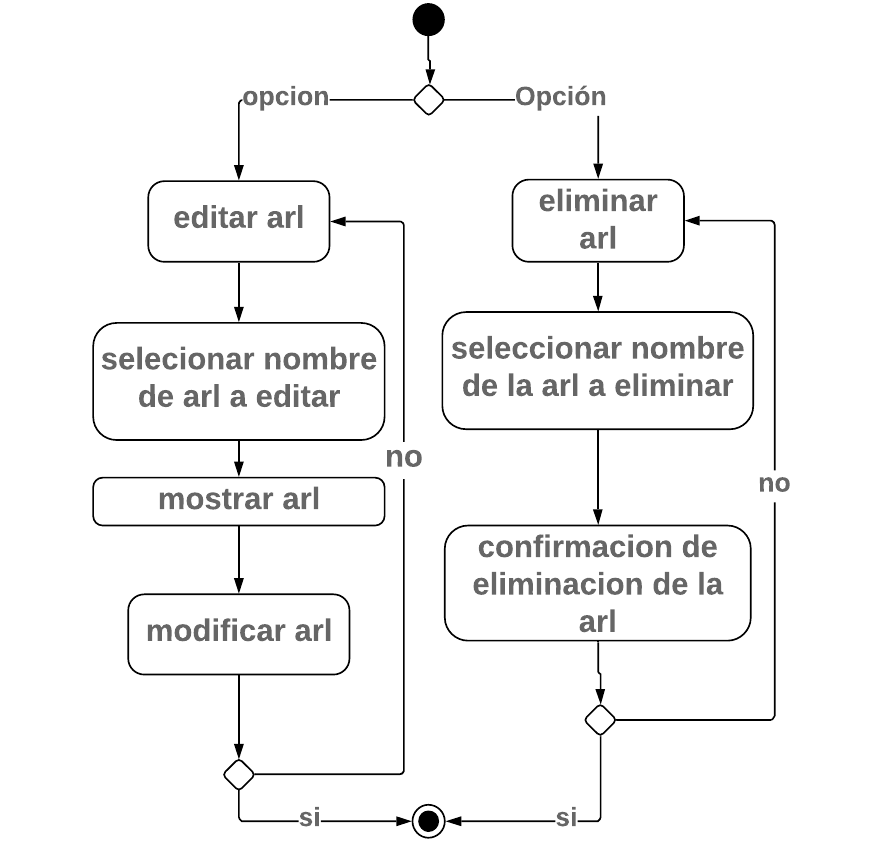


Ilustración 38 modificar ajustes arl

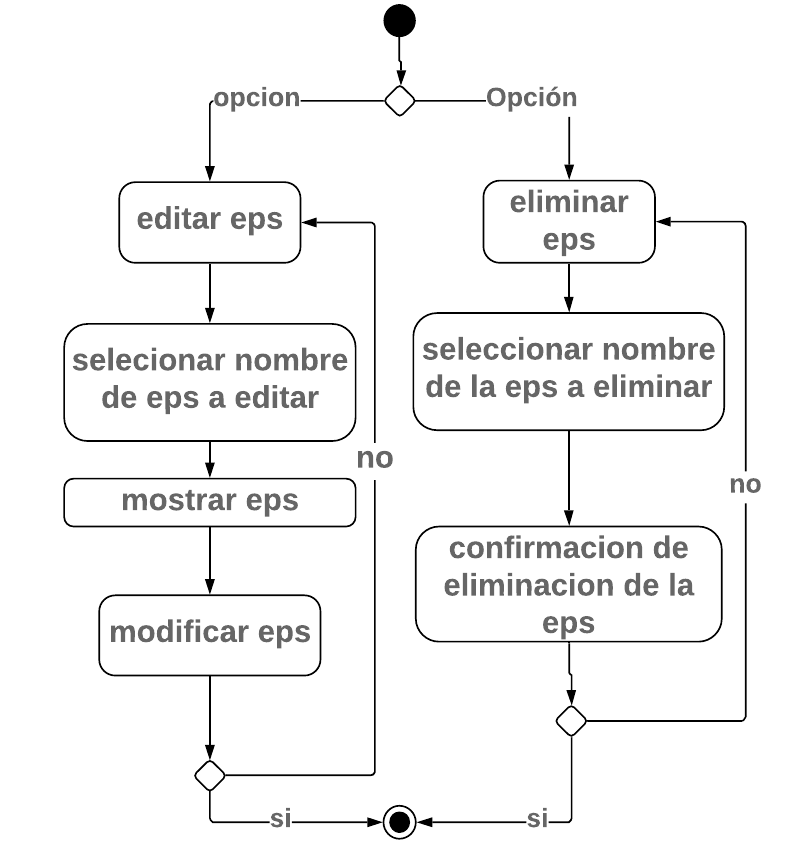


Ilustración 39 modificar ajustes eps

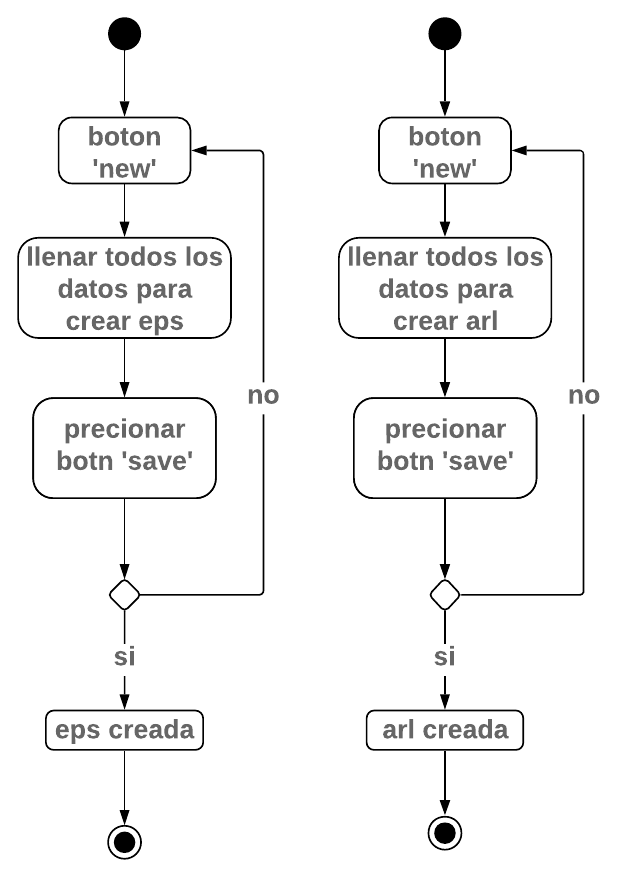


Ilustración 40 nuevos ajustes

# DIAGRAMA DE CLASES

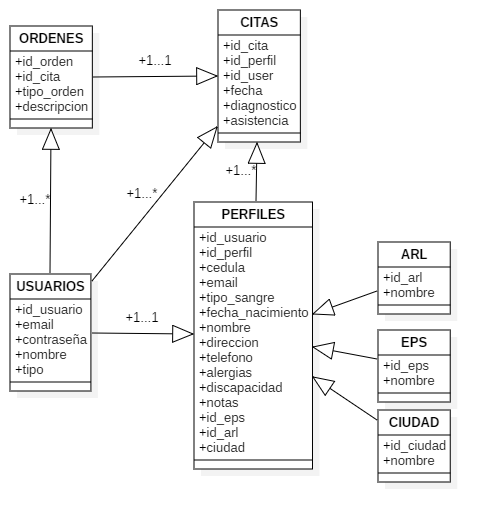


Ilustración 41 Diagrama de clases

# DIAGRAMA DE COMPONENTES

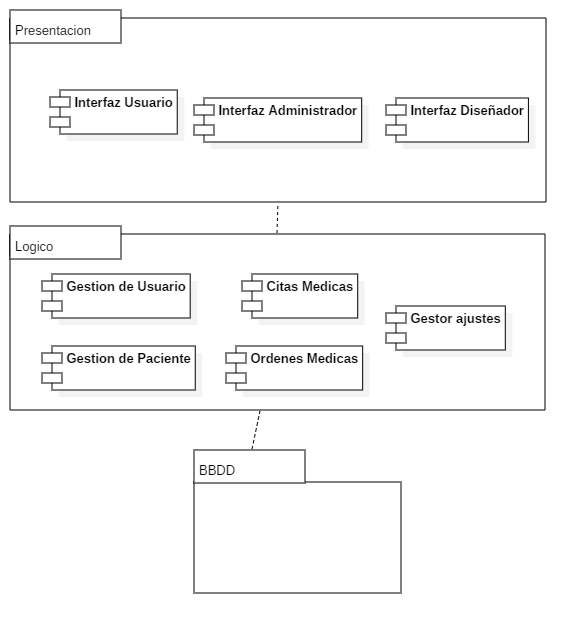


Ilustración 42 diagrama de componentes

# ARQUITECTURA ADAPTADA

**Cliente – Servidor**

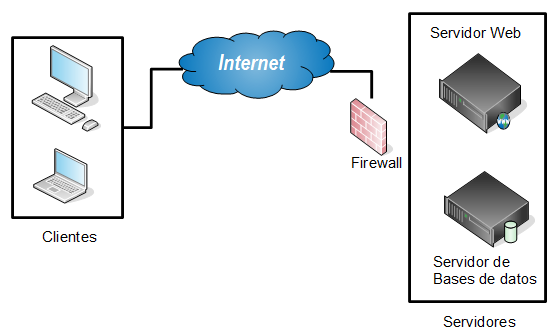


Ilustración 43 cliente-servidor

# DESARROLLO DEL PROYECTO

**Ruby on Rails**

Ruby on Rails es un entorno de desarrollo web de código abierto que está optimizado para la satisfacción de los programadores y para la productividad sostenible. Te permite escribir un buen código evitando que te repitas y favoreciendo la convención antes que la configuración.

**¿Qué es Rails?** Rails es un armazón para construir aplicaciones web que acceden a bases de datos, Obteniendo un conjunto de librerías, automatismos y convenciones destinados a resolver los problemas más comunes a la hora de desarrollar una aplicación web, para que el programador pueda concentrarse en los aspectos únicos y diferenciales de su proyecto en lugar de los problemas recurrentes.

**¿Para qué se utiliza?** Para ayudar a construir aplicaciones modernas de internet como: Twitter, Scribd, Hulu, Xing, Soundcloud, Basecamp, Github, etc.

**¿De dónde sale?** Rails fue creado en 2003 por David Heinemeier Hansson y desde entonces ha sido extendido por el Rails core team, más de 2.100 colaboradores y soportado por una extensa y activa comunidad.

**PostgreSQL**

PostgreSQL es un gestor de bases de datos relacional y orientado a objetos. Su licencia y desarrollo es de código abierto, siendo mantenida por una comunidad de desarrolladores, colaboradores y organizaciones comerciales de forma libre y desinteresadamente. Esta comunidad es denominada PDGD (PostgreSQL Global Development Group, por sus siglas en inglés).

Es reconocido actualmente como uno de los sistemas gestores de bases de datos relacionales más potentes del mercado. Presenta fácil accesibilidad, es multiplataforma y está disponible para su utilización en casi todos los sistemas operativos utilizados en la actualidad sin disminuir su rendimiento.

**Heroku**

Heroku es una plataforma como servicio de computación en la Nube que soporta distintos lenguajes de programación.

Heroku es propiedad de Salesforce.com, es una de las primeras plataformas de computación en la nube, que fue desarrollada desde junio de 2007, con el objetivo de soportar solamente el lenguaje de programación Ruby, pero posteriormente se ha extendido el soporte a Java, Node.js, Scala, Clojure y Python y PHP.

**MaterializeCSS**

Es un framework que hace más fácil el proceso de creación de diseño web. Es un framework moderno y responsivo que se basa en Material Design. Tiene muchos componentes y clases predefinidas, para hacer uso de ellas mucho fácilmente. Aumenta el ritmo de desarrollo

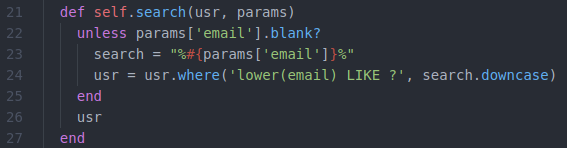
Materialize cuenta con componentes personalizados con unos estilos por defecto, que se pueden manejar fácilmente para generar animaciones y ciertas transiciones, de tal manera que para el desarrollador web es mucho más fácil hacer su tarea de diseño web.

Durante el primer sprint, el equipo de desarrolló el módulo de usuarios, sus funcionalidades y detalles de seguridad

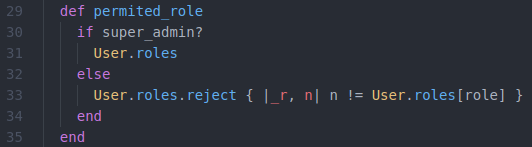
**Módulo de Usuario**

|  |  |
| --- | --- |
| Elaborado por: | Andrés Felipe Farfán Hernández |
| Fecha: | Agosto 30 del 2019 |

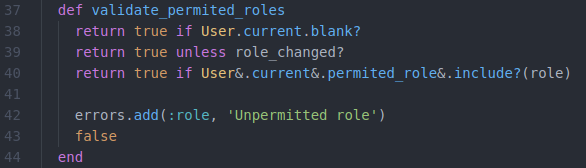
|  |  |
| --- | --- |
| Elaborado por: | Andrés Felipe Farfán Hernández |
| Fecha: | Septiembre 1 del 2019 |



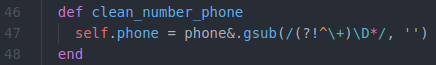
Método search se utiliza para la búsqueda de usuarios por Email, el método recibe un objeto con una lista de usuarios y unos parámetros de búsqueda.



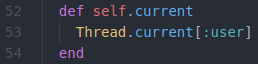
Método permited\_role se utiliza para mostrar el listado de roles permitidos por cada usuario.



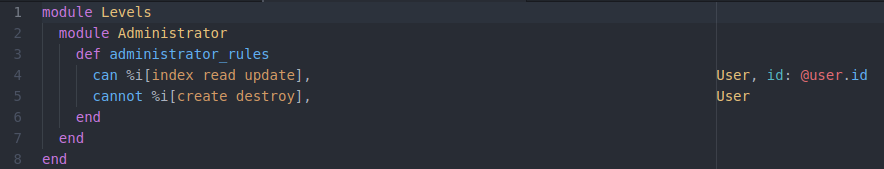
Método validate\_permited\_roles se utiliza para validar si el role asignado a un usuario, es un role permitido para un usuario.

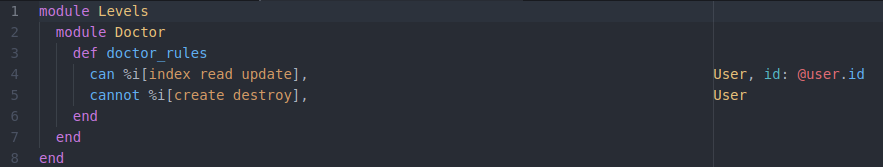


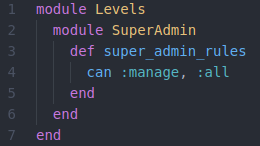
Método clean\_number\_phone se utiliza para limpiar los caracteres especiales y textos, del campo de teléfono.



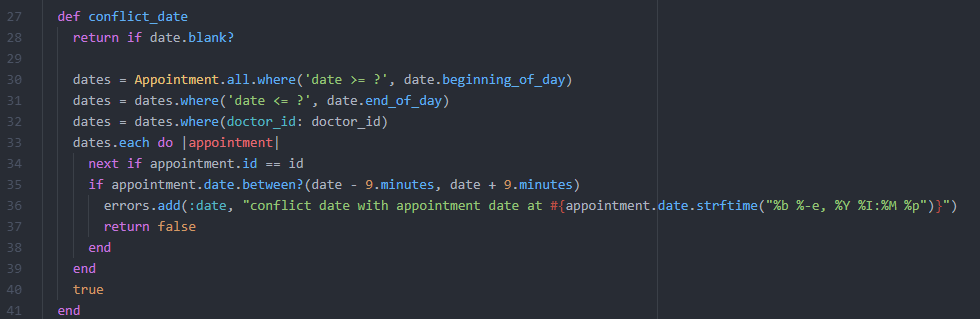
Método current se utiliza para almacenar la instancia del usuario que esta logueado en el sistema.







Reglas de los usuarios para el módulo de usuarios.



Método para para identificar si hay conflictos de fechas cuando se crea o edita una cita

# PRUEBAS

## Pruebas Inicio de sesión

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 31/08/2019 |
| Hora: | 5:00 p.m |
| Código: | CT1 |
| Nombre: | Log in USUARIO |
| Descripción: | El USUARIO debe estar registrado para hacer uso de sistema y para ello debe ingresar los campos de ‘Email’ y su ‘Password”. |
| Caso de uso: | Inicio sesión |
| Precondiciones: | El USUARIO debe estar registrado en el sistema. |
| Pos condiciones: | El USUARIO accede al sistema y se registra su acceso en la tabla de registro de actividad. |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El usuario accede al sistema en la url principal | http://radiant-forest-88745.herokuapp.com/ | El sistema abrirá correctamente sin mostrar ningún error |
| 2 | EL sistema solicita credenciales | ‘email’  ‘Password’ | El sistema dejará llenar estos campos con su respectiva información |
| 3 | El USUARIO ingresa proporcionando su email y password | Email:natalia@myhospital  Contraseña:123456 | los campos fueron llenados con éxito |
| 4 | El sistema valida las credenciales del usuario |  | el sistema validó los datos y entra al menú principal de usuarios |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: | Los usuario ya deben estar creados con respectivo id\_usuario, nombre, email, password, tipo de usuario |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 09/09/2019 |
| Hora: | 10:00 p.m |
| Código: | CT2 |
| Nombre: | sign in USUARIO |
| Descripción: | El USUARIO debe registrarse para hacer uso de sistema y para ello debe ingresar los campos de ‘Email’ y su ‘Password. |
| Caso de uso: | Inicio sesión |
| Precondiciones: | n/n |
| Pos condiciones: | El USUARIO inicia sesión en el sistema |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El usuario accede al sistema en la url principal | http://radiant-forest-88745.herokuapp.com/ | El sistema abrirá correctamente sin mostrar ningún error |
| 2 | El usuario accede a ‘sign in’ para registrarse | <http://radiant-forest-88745.herokuapp.com/users/sign_in> | Botón ‘sing in’ funcionando correctamente y lo lleva a la ventana de registro |
| 2 | EL sistema solicita credenciales | ‘email’  ‘Password’ | El sistema deja llenar estos campos con su respectiva información |
| 3 | El USUARIO ingresa proporcionando su email y password | Email:natalia@myhospital  Contraseña:123456 | Lo campos fueron llenados con éxito |
| 4 | El sistema crea nuevo usuario del sistema | muestra mensaje ‘invalid email or password’ | el sistema no deja que un nuevo usuario del sistema se registre |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK |  | Falló | X | No ejecutado |  |

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 31/08/2019 |
| Hora: | 5:00 p.m |
| Código: | CT3 |
| Nombre: | Olvidaste tu contraseña |
| Descripción: | El USUARIO quiere iniciar sesión pero el sistema no lo deja entrar ya que olvido su contraseña y puede cambiarla ingresando a ‘forgot your password |
| Caso de uso: | Inicio sesión |
| Precondiciones: | El usuario debe olvidar su contraseña |
| Pos condiciones: | La contraseña del usuario debe ser cambiada |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El usuario accede al sistema en la url principal | http://radiant-forest-88745.herokuapp.com/ | El sistema abrirá correctamente sin mostrar ningún error |
| 2 | El usuario accede a ‘forgot your password’ para cambiar contraseña | <http://radiant-forest-88745.herokuapp.com/users/password/new> | Accede correctamente al campo |
| 2 | EL sistema solicita credenciales | ‘email’ | El sistema dejará llenar estos campos con su respectiva información |
| 3 | El USUARIO ingresa su email | Email:natalia@myhospital | Lo campos fueron llenados con éxito y da ‘enter’ |
| 4 | El sistema valida datos | Muestra mensaje ‘You will receive an email with instructions on how to reset your password in a few minutes.’ | la contraseña no fue cambiada con éxitos |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK |  | Falló | X | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: |  |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 09/09/2019 |
| Hora: | 11:00 p.m |
| Código: | CT4 |
| Nombre: | Log in USUARIO no registrado |
| Descripción: | El USUARIO no registrado intentara iniciar sesión ingresar los campos de ‘Email’ y su ‘Password. NO debe iniciar sesión si el correo electrónico y contraseña son incorrectos |
| Caso de uso: | Inicio sesión |
| Precondiciones: | El USUARIO no debe estar registrado en el sistema. |
| Pos condiciones: | El sistema deberá mostrar algún error. |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El usuario accede al sistema en la url principal | http://radiant-forest-88745.herokuapp.com/ | El sistema abrirá correctamente sin mostrar ningún error |
| 2 | EL sistema solicita credenciales | ‘email’  ‘Password’ | El sistema deja llenar estos campos con su respectiva información |
| 3 | El USUARIO no existente ingresa proporcionando su email y password | Email:nataliaisazaa@gmail.com  Contraseña:123456 | los campos fueron llenados con éxito |
| 4 | El sistema valida las credenciales del usuario | Muestra mensaje ‘Invalid Email or password. ’ | No deja ingresar al menú principal del sistema |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: |  |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## Pruebas superadmin

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 010/09/2019 |
| Hora: | 5:00 p.m. |
| Código: | CT5 |
| Nombre: | Crear Usuario |
| Descripción: | El usuario SUPERADMIN puede crear un nuevo usuario en el sistema |
| Caso de uso: | Manejar módulo |
| Precondiciones: | El usuario SUPERADMIN debe iniciar sesión en el sistema y tener los privilegios de crear un nuevo usuario |
| Pos condiciones: | Después de crear el nuevo usuario, se realizan cambios en la BD |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El SUPERADMIN accede al menú principal | <http://radiant-forest-88745.herokuapp.com/> | El sistema funciona correctamente, no lance error |
| 2 | EL SUPERADMIN baja a la parte de ‘invite user’ e ingresa los campos requeridos | Email: nataliaisaza@myhospital  Role: Admin  Botón ‘submit’ | Los campos fueron llenados con éxito |
| 3 | El sistema valida las credenciales y crear nuevo usuario | Mensaje ‘An invitation email has been sent to nataliaisaza@myhospital.com.’ | El nuevo usuario del sistema fue creado con éxito |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: | Cuando el SUPERADMIN crea un usuario este queda grabado en la lista de usuario del sistema, con un pequeño error que el rol del nuevo usuario lo muestra en la parte de PHONE del usuario, posteriormente esto se puede editar |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 010/09/2019 |
| Hora: | 5:00 p.m. |
| Código: | CT6 |
| Nombre: | editar Usuario |
| Descripción: | El usuario SUPERADMIN puede editar un nuevo usuario en el sistema |
| Caso de uso: | Manejar módulo |
| Precondiciones: | El usuario SUPERADMIN debe iniciar sesión en el sistema y tener los privilegios de editar un usuario |
| Pos condiciones: | Se realizan cambios en el usuario y estos se guardan en la BD |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El SUPERADMIN accede al menú principal | <http://raiant-forest-88745.herokuapp.com/> | El sistema funciona correctamente, no lance error |
| 2 | EL SUPERADMIN selecciona un usuario al que le quiere realizar cambios y hace clic en el lápiz de editar | <http://radiant-forest-88745.herokuapp.com/users/7/edit> | la ventana editar abre correctamente |
| 3 | El SUPERADMIN realiza cambios en el usuario | Name: Natalia Isaza  Email:nataliaisaza@myhospital.com  Phone:12345678912345  Role: Administrator  Botón ‘save’ | Los campos fueron llenados con éxito |
| 4 | Sistema valida los cambios y guarda | Mensaje: ‘User was successfully updated.’  Botón ‘back’ : salir de editar usuario | Los cambios fueron modificados correctamente |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: | el teléfono del usuario tiene que tener mínimo 10 caracteres |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 10/09/2019 |
| Hora: | 10:00 a.m. |
| Código: | CT7 |
| Nombre: | eliminar Usuario |
| Descripción: | El usuario SUPERADMIN puede eliminar un usuario en el sistema |
| Caso de uso: | Manejar módulo |
| Precondiciones: | El usuario SUPERADMIN debe iniciar sesión en el sistema y tener los privilegios de eliminar un usuario |
| Pos condiciones: | El usuario es eliminado del sistema |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El SUPERADMIN accede al menú principal | <http://raiant-forest-88745.herokuapp.com/> | El sistema funciona correctamente, no lance error |
| 2 | EL SUPERADMIN selecciona un usuario al que le quiere eliminar y hace clic en el basura de eliminar | Mensaje: ‘está seguro de eliminarlo’  Botón ‘aceptar’ | la ventana eliminar abre |
| 3 | el sistema valida los cambios | Mensaje ‘User was successfully destroyed.’ | El usuario fue eliminado con éxito |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: | el usuario también se puede eliminar cuando lo estamos editando que esta la opción ‘delete user’ |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 10/09/2019 |
| Hora: | 10:00 a.m. |
| Código: | CT8 |
| Nombre: | buscar Usuario |
| Descripción: | El usuario SUPERADMIN puede buscar un usuario en el sistema |
| Caso de uso: | Manejar módulo |
| Precondiciones: | El usuario SUPERADMIN debe iniciar sesión en el sistema y tener los privilegios de buscar un usuario |
| Pos condiciones: | El usuario es encontrado en el sistema |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El SUPERADMIN accede al menú principal | <http://raiant-forest-88745.herokuapp.com/> | El sistema funciona correctamente, no lance error |
| 2 | EL SUPERADMIN va a la línea de búsqueda y coloca el correo del usuario a buscar | Email: ’natalia@myhospital’  ‘enter’ | El campo es llenado correctamente |
| 3 | el sistema valida búsqueda | Datos del usuario buscado | el usuario de encontró exitosamente |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: | Si un usuario no está en la lista de búsqueda simplemente la lista aparece vacía sin datos exitosos |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

## Pruebas usuario admin

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tester: | | | | Natalia Isaza Serrano | | | | | | |
| Fecha: | | | | 04/09/2019 | | | | | | |
| Hora: | | | | 1:00 a.m. | | | | | | |
| Código: | | | | CT9 | | | | | | |
| Nombre: | | | | Manejo USUARIOADMIN | | | | | | |
| Descripción: | | | | El USUARIO ADMIN tiene los privilegios editarse su propia información | | | | | | |
| Caso de uso: | | | |  | | | | | | |
| Precondiciones: | | | | El USUARIOADMIN debe tener los privilegios solo para editar su propia información | | | | | | |
| Pos condiciones: | | | | Se realiza cambios a de su propia información | | | | | | |
| Paso | Acción | | | | | Valores de los Campos | | Resultado esperado | | | |
| 1 | El ADMIN accede al sistema en la url principal | | | | | <http://radiant-forest-88745.herokuapp.com/users> | | El sistema funcione correctamente, no lance error al abrir | | | |
| 4 | El ADMIN puede editar su propia información | | | | Phone:12345678912346  Editar  Phone:12345678912347  save | | | Modificación realizada con éxito | | | |
| 5 | El sistema valida cambios | | | | ‘translation missing: en.share.success\_update’ | | | Los cambios fueron modificados con éxito | | | |
| OK | | X | Falló | | | |  | | No ejecutado |  | |

|  |  |
| --- | --- |
| Comentarios: | Un USUARIOADMIN no tiene los permisos en el sistema para:  -eliminar un USUARIOS de la BD  -encargar roles a otros USUARIOS  -editar o ver otros usuarios del sistema |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tester: | | | | Natalia Isaza Serrano | | | | | | |
| Fecha: | | | | 04/09/2019 | | | | | | |
| Hora: | | | | 1:00 a.m. | | | | | | |
| Código: | | | | CT10 | | | | | | |
| Nombre: | | | | Manejo USUARIOADMIN | | | | | | |
| Descripción: | | | | El USUARIOADMIN tiene los privilegios de buscarse a sí mismo | | | | | | |
| Caso de uso: | | | |  | | | | | | |
| Precondiciones: | | | | El USUARIOADMIN debe tener solo el privilegio de buscarse a sí mismo y no a otros usuarios del sistema | | | | | | |
| Pos condiciones: | | | | Se realizara búsqueda | | | | | | |
| Paso | Acción | | | | | Valores de los Campos | | Resultado esperado | | | |
| 1 | El ADMIN accede al sistema en la url principal | | | | | <http://radiant-forest-88745.herokuapp.com/users> | | El sistema funcione correctamente, no lance error al abrir | | | |
| 2 | EL ADMIN realiza búsqueda de sí mismo | | | | Email: natalia@myhospital | | | Búsqueda exitosa | | | |
| 3 | El sistema muestra la información del usuario buscado | | | | Name: Natalia  Email:natalia@myhospital.com  Phone:12345678912345  Role: MEDICO  Botón(buscar, editar) | | | El sistema muestra correctamente esta información y sus botones funcionan | | | |
| OK | | X | Falló | | | |  | | No ejecutado |  | |

|  |  |
| --- | --- |
| Comentarios: | el usuario admin solo se puede buscar a sí mismo y no a otros usuarios del sistema |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_**

## Pruebas de usuario médico

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tester: | | | | Natalia Isaza Serrano | | | | | | |
| Fecha: | | | | 04/09/2019 | | | | | | |
| Hora: | | | | 1:00 a.m. | | | | | | |
| Código: | | | | CT11 | | | | | | |
| Nombre: | | | | Manejo USUARIOMEDICO | | | | | | |
| Descripción: | | | | El USUARIO ADMIN tiene los privilegios editarse su propia información | | | | | | |
| Caso de uso: | | | |  | | | | | | |
| Precondiciones: | | | | El USUARIOMEDICO debe tener los privilegios solo para editar su propia información | | | | | | |
| Pos condiciones: | | | | Se realiza cambios a de su propia información | | | | | | |
| Paso | Acción | | | | | Valores de los Campos | | Resultado esperado | | | |
| 1 | El MEDICO accede al sistema en la url principal | | | | | <http://radiant-forest-88745.herokuapp.com/users> | | El sistema funcione correctamente, no lance error al abrir | | | |
| 4 | El MEDICO puede editar su propia información | | | | Phone:12345678912348  Editar  Phone:12345678912347  save | | | Modificación realizada con éxito | | | |
| 5 | El sistema valida cambios | | | | ‘translation missing: en.share.success\_update’ | | | Los cambios fueron modificados con éxito | | | |
| OK | | X | Falló | | | |  | | No ejecutado |  | |

|  |  |
| --- | --- |
| Comentarios: | Un USUARIOMEDICO no tiene los permisos en el sistema para:  -eliminar un USUARIOS de la BD  -encargar roles a otros USUARIOS  -editar o ver otros usuarios del sistema |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tester: | | | | Natalia Isaza Serrano | | | | | | |
| Fecha: | | | | 04/09/2019 | | | | | | |
| Hora: | | | | 1:00 a.m. | | | | | | |
| Código: | | | | CT12 | | | | | | |
| Nombre: | | | | Manejo USUARIOMEDICO | | | | | | |
| Descripción: | | | | El USUARIOMEDICO tiene los privilegios de buscarse a sí mismo | | | | | | |
| Caso de uso: | | | |  | | | | | | |
| Precondiciones: | | | | El USUARIOMEDICO debe tener solo el privilegio de buscarse a sí mismo y no a otros usuarios del sistema | | | | | | |
| Pos condiciones: | | | | Se realizará búsqueda | | | | | | |
| Paso | Acción | | | | | Valores de los Campos | | Resultado esperado | | | |
| 1 | El MÉDICO accede al sistema en la url principal | | | | | <http://radiant-forest-88745.herokuapp.com/users> | | El sistema funcione correctamente, no lance error al abrir | | | |
| 2 | EL MÉDICO realiza búsqueda de sí mismo | | | | Email: natalia@myhospital | | | Búsqueda exitosa | | | |
| 3 | El sistema muestra la información del usuario buscado | | | | Name: Natalia  Email:natalia@myhospital.com  Phone:12345678912345  Role: MÉDICO  Botón(buscar, editar) | | | El sistema muestra correctamente esta información y sus botones funcionan | | | |
| OK | | X | Falló | | | |  | | No ejecutado |  | |

|  |  |
| --- | --- |
| Comentarios: | el usuario admin solo se puede buscar a sí mismo y no a otros usuarios del sistema |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_**

## Datos incorrectos al editar un usuario

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 010/09/2019 |
| Hora: | 5:00 p.m. |
| Código: | CT13 |
| Nombre: | editar Usuario con datos incorrectos |
| Descripción: | se realizará prueba con datos incorrectos en editar Usuario |
| Caso de uso: | Manejar módulo |
| Precondiciones: | El usuario SUPERADMIN debe iniciar sesión en el sistema y tener los privilegios de editar un usuario |
| Pos condiciones: | debe mostrar algún tipo de error y no guardar cambios |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El SUPERADMIN accede al menú principal | <http://raiant-forest-88745.herokuapp.com/> | El sistema funciona correctamente, no lance error |
| 2 | EL SUPERADMIN selecciona un usuario al que le quiere realizar cambios y hace clic en el lápiz de editar | <http://radiant-forest-88745.herokuapp.com/users/7/edit> | la ventana editar abre correctamente |
| 3 | El SUPERADMIN realiza cambios inválidos en el usuario | Name: Natza  Email:nataa@myhospital.com  Phone:125  Role: Administrator  Botón ‘save’ | Los campos fueron llenados con éxito |
| 4 | Sistema valida datos | Mensaje : ‘3 Errors prohibited this User from being saved:  Email is invalid  Email is invalid  Phone is too short (minimum is 10 characters)’ | Los cambios no fueron modificados correctamente ya que hay errores |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: | el teléfono del usuario tiene que tener mínimo 10 caracteres  email es incorrecto |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 010/09/2019 |
| Hora: | 5:00 p.m. |
| Código: | CT14 |
| Nombre: | No Crear Usuario si ya existe |
| Descripción: | No deberá crear un nuevo usuario si el USUARIO ya existe en el sistema |
| Caso de uso: | Manejar módulo |
| Precondiciones: | El usuario ya debe estar creado en el sistema |
| Pos condiciones: | El sistema debe mostrar error y no dejar crear un nuevo usuario |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El SUPERADMIN accede al menú principal | <http://radiant-forest-88745.herokuapp.com/> | El sistema funciona correctamente, no lance error |
| 2 | EL SUPERADMIN baja a la parte de ‘invite user’ e ingresa los campos requeridos con un correo que ya existe en el sistema | Email: nataliaisaza@myhospital  Role: Admin  Botón ‘submit’ | Los campos fueron llenados con éxito |
| 3 | El sistema valida y muestra error | Mensaje ‘1 error prohibited this user from being saved: email has already been taken’ | El nuevo usuario no fue creado. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: | Cada usuario tiene correo electrónico único |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 12/09/2019 |
| Hora: | 5:00 p.m |
| Código: | CT515 |
| Nombre: | Log in USUARIO sin password |
| Descripción: | El sistema no debería iniciar sesión sin haber llenado el capo ‘Password. |
| Caso de uso: | Inicio sesión |
| Precondiciones: | N/A |
| Pos condiciones: | El sistema no dejara iniciar sesión si no hay password |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El usuario accede al sistema en la url principal | http://radiant-forest-88745.herokuapp.com/ | El sistema abrirá correctamente sin mostrar ningún error |
| 2 | EL sistema solicita credenciales | ‘email’  ‘Password’ | El sistema dejará llenar estos campos con su respectiva información |
| 3 | El USUARIO ingresa proporcionando su email y no el password | Email:natalia@myhospital  Contraseña: "vacío" | los campos fueron llenados con éxito |
| 4 | El sistema valida las credenciales del usuario | Mensaje: ‘invalid email or password.’ | el sistema no deja iniciar sesión |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: |  |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |
| --- | --- |
| Versión del documento: | 1.0 |

|  |  |
| --- | --- |
| Tester: | Natalia Isaza Serrano |
| Fecha: | 12/09/2019 |
| Hora: | 5:00 p.m |
| Código: | CT516 |
| Nombre: | Log in USUARIO sin email |
| Descripción: | El sistema no debería iniciar sesión sin haber llenado el capo ‘email’. |
| Caso de uso: | Inicio sesión |
| Precondiciones: | N/A |
| Pos condiciones: | El sistema no dejara iniciar sesión si no hay email |

|  |  |  |  |
| --- | --- | --- | --- |
| Paso | Acción | Valores de los Campos | Resultado esperado |
| 1 | El usuario accede al sistema en la url principal | http://radiant-forest-88745.herokuapp.com/ | El sistema abrirá correctamente sin mostrar ningún error |
| 2 | EL sistema solicita credenciales | ‘email’  ‘Password’ | El sistema dejará llenar estos campos con su respectiva información |
| 3 | El USUARIO ingresa proporcionando su password y no el Email | Email: ´vacío´  Contraseña:123456 | los campos fueron llenados con éxito |
| 4 | El sistema valida las credenciales del usuario | Mensaje: ‘invalid email or password.’ | el sistema no deja iniciar sesión |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OK | X | Falló |  | No ejecutado |  |

|  |  |
| --- | --- |
| Comentarios: |  |

**Vo Bo Director: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**