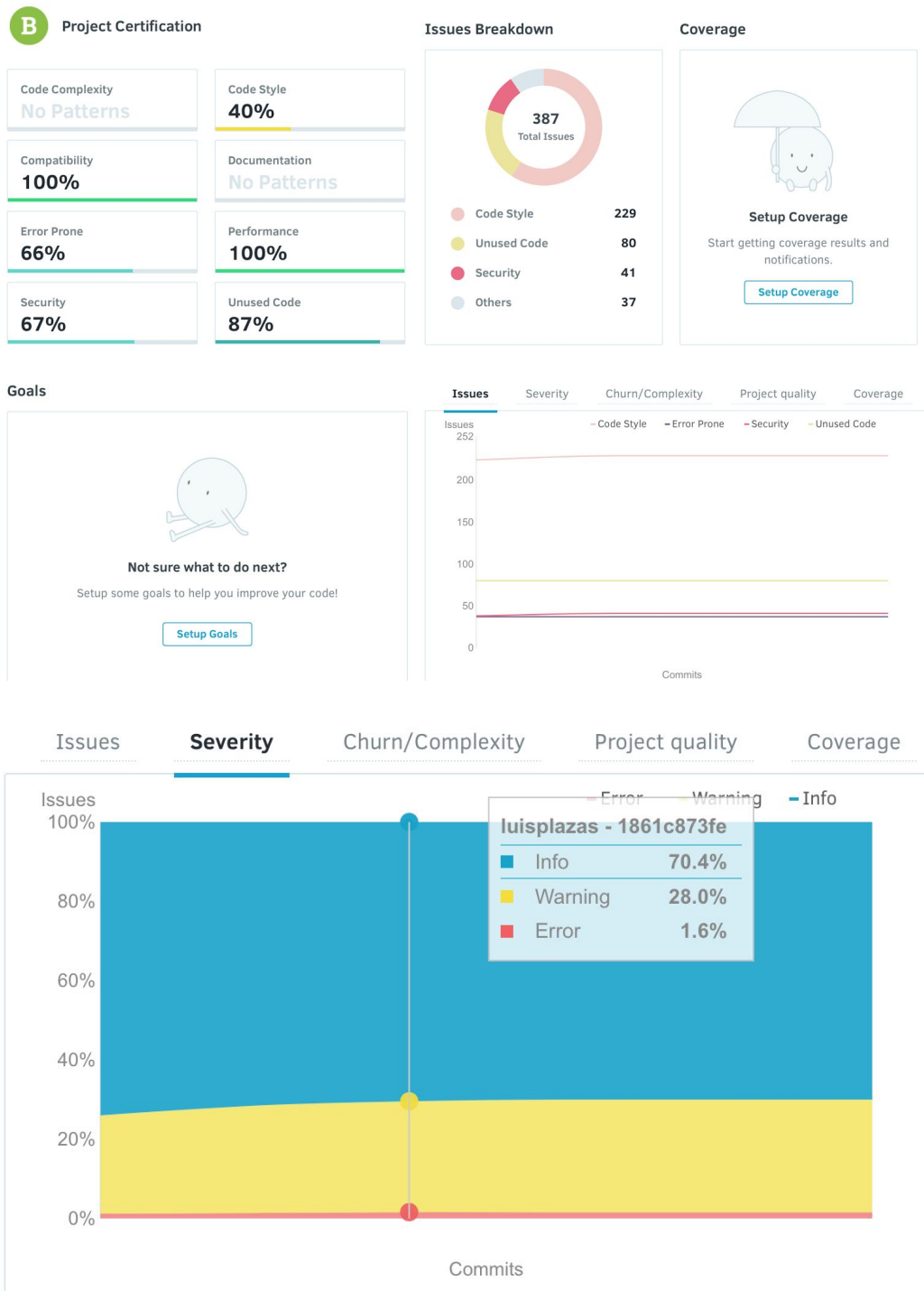


# Experimento 3

Grupo **St. ValenTeam**

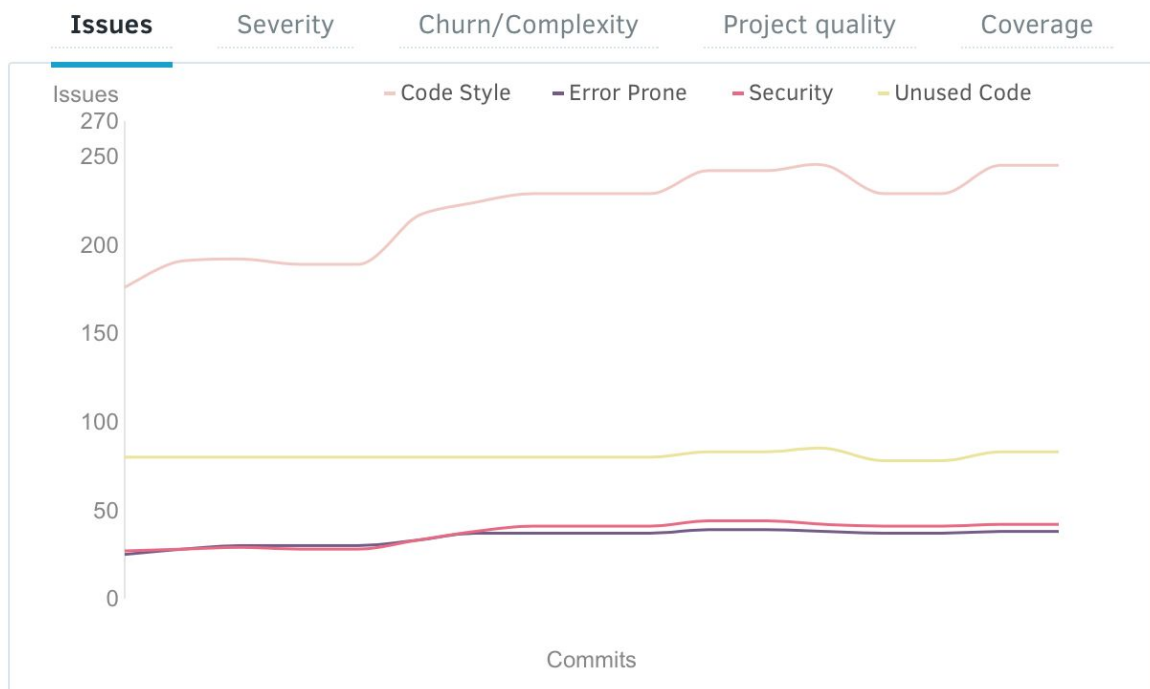
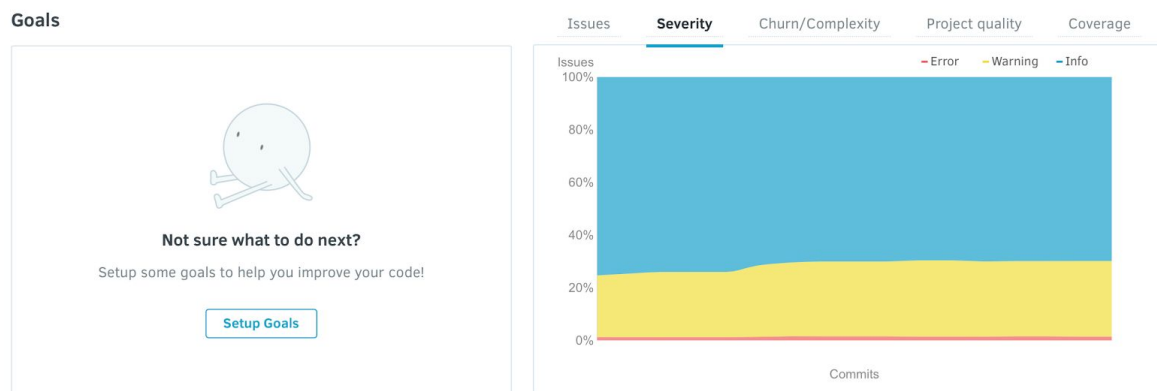
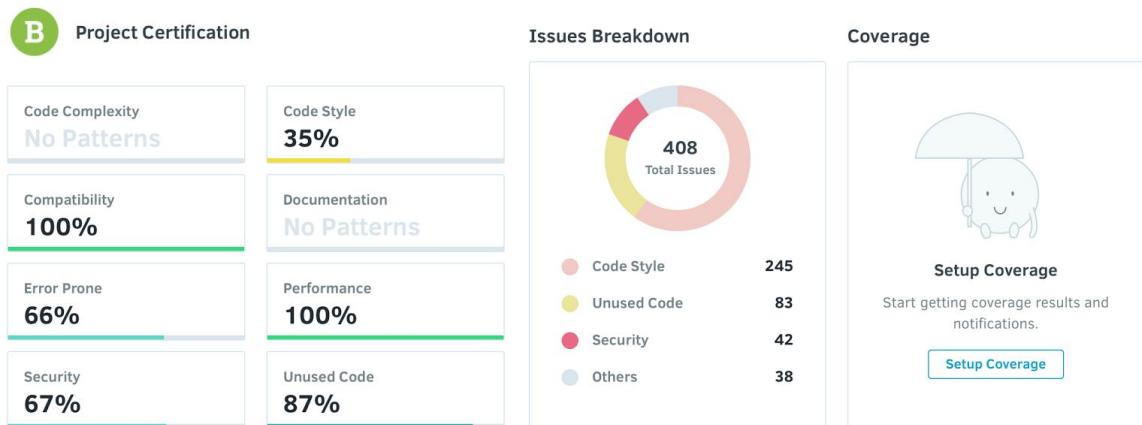
## Métricas antes de aplicar patrones

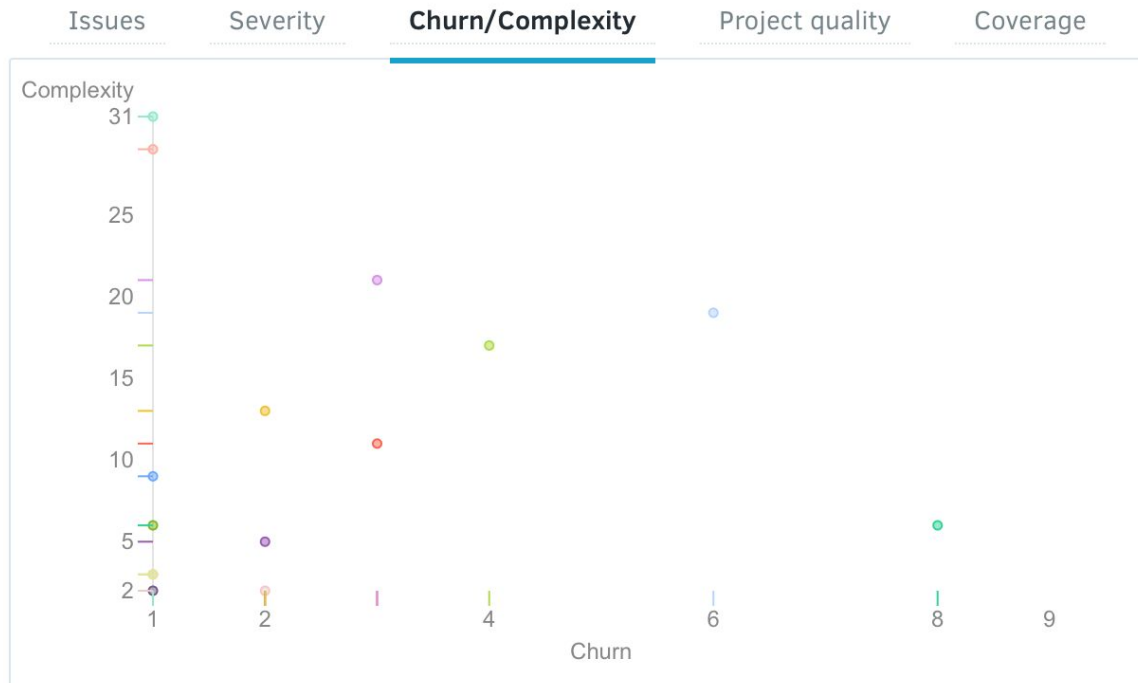




Se evidencia que la mayoría de issues corresponden al estilo del código, al observar los issues se observa que la mayoría de estos son generados por las librerías de javascript que se utilizaron en la interfaz. También se observa que hay un 13% de código sin usar e issues de seguridad los cuales son generados por el uso de cookies en la interfaz. Sin embargo, cerca del 70% de los issues son informativos y solo el 1% pueden causar errores; es decir, la severidad de los issues es baja. Por último, se evidencia que la mayoría de archivos presentan pocos cambios y baja complejidad lo que implica una mayor modificabilidad.

## Métricas después de aplicar patrones





Por otra parte, se observó que tras aplicar los patrones de modificabilidad aumentaron ligeramente los issues relacionados con el estilo del código y código sin usar. Sin embargo, se redujeron los issues de seguridad. Se evidencia también que la proporción de issues informativos y de error no se vieron modificados por los cambios.

Por otra parte, después de aplicar los patrones de modificabilidad se incrementaron los cambios en los archivos (churn) aunque la mayoría se sigue ubicando en baja complejidad por lo cual se mantiene el atributo de calidad de modificabilidad.

## Análisis crítica colaborativa

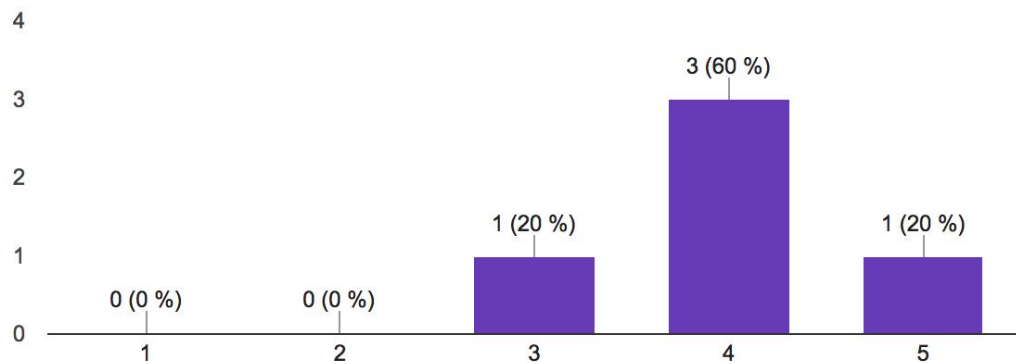
Se realiza una encuesta a 5 personas a partir de la cual se encontraron los siguientes resultados. Además de los resultados se presentan los cambios que se realizaron a la interfaz con el objetivo de aplicar la crítica colaborativa.

### Módulo inicio de sesión

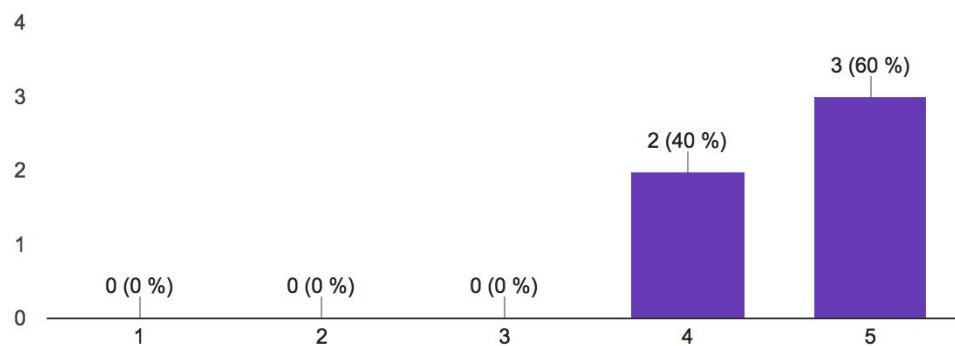
Se observó que según la crítica colaborativa, para los usuarios es fácil encontrar la forma de iniciar sesión y recordarán con facilidad los pasos que hicieron para iniciar sesión. Las recomendaciones que encontramos fue ampliar el tamaño del botón de inicio de sesión y cambiar el color ya que no resaltaba.

Resultados:

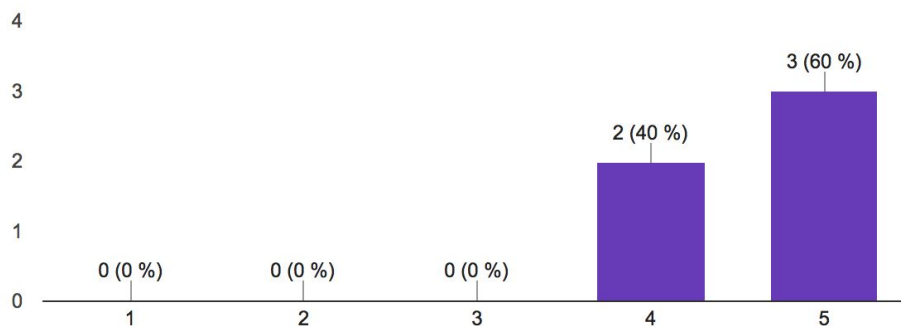
¿Qué tan fácil es para el usuario encontrar la opción de iniciar sesión?  
(5 respuestas)



¿Qué tanto deben explorar los usuarios para encontrar las opciones que le permitirán iniciar sesión?  
(5 respuestas)



¿Después de la actual acción, los usuarios recordarán qué pasos realizaron con el fin de completar la tarea de iniciar sesión en una próxima ocasión?  
(5 respuestas)



## ¿Qué problemas encontró al iniciar sesión o en la página principal?

(5 respuestas)

No se ve claramente el botón de inicio de sesión

Ninguno

Está bien

El botón de inicio de sesión es un poco pequeño y se pierde dentro del fondo de la pagina

El botón para iniciar sesión es muy pequeño.

**Solución:** Para aplicar las recomendaciones se cambió el color y tamaño del botón como se muestra en la imagen siguiente:

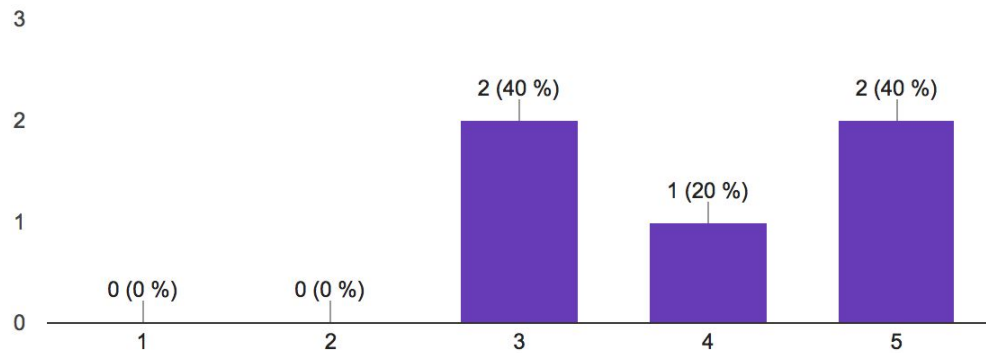


## Módulo Médico

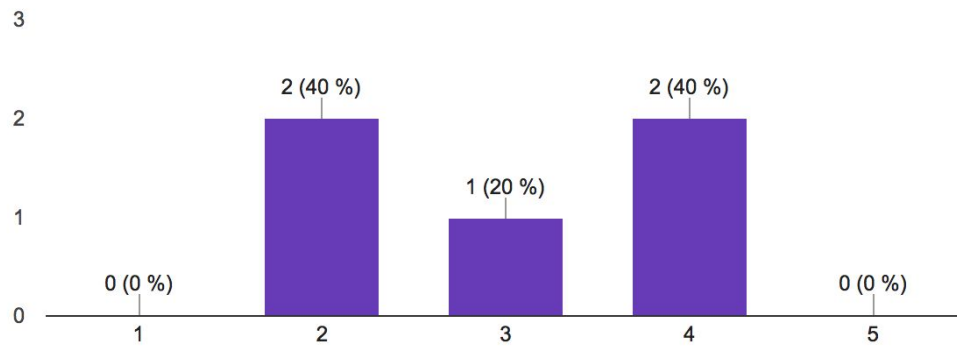
Como resultados del módulo del médico encontramos que se requería colocar la fila del paciente del color del estado de manera que para el médico fuera más fácil seleccionar al paciente que requiere su atención. Dado que antes solo se colocaba el estado del paciente en una fila como atributo. También, recibimos la crítica de que al colocar solo las tablas de mediciones no era intuitivo el historial de un paciente.

### Resultados:

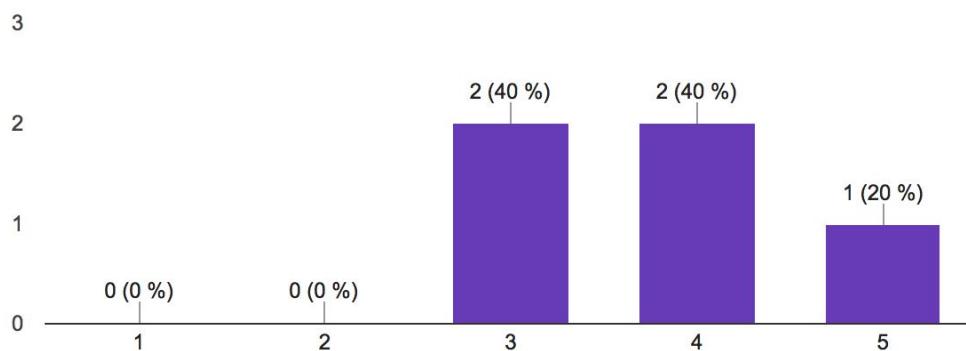
¿Qué tan fácil es para un médico encontrar a un paciente? (5 respuestas)



¿Qué tan fácil es identificar el estado de un paciente? (5 respuestas)

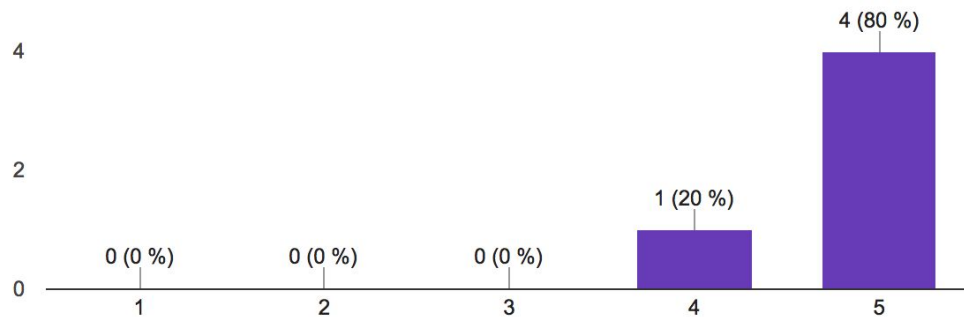


¿Qué tanto debe explorar un médico para encontrar la información de un paciente?  
(5 respuestas)

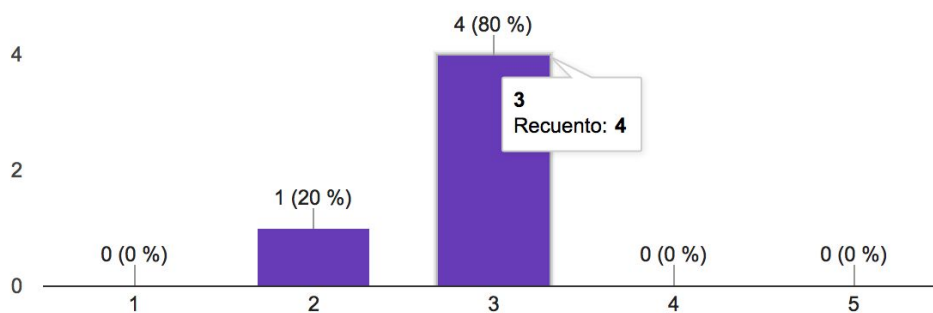


## Después de la actual sesión ¿los médicos recordaran los pasos para ver todos sus pacientes?

(5 respuestas)



## ¿Qué tan fácil es inspeccionar la historia clínica de un paciente? (5 respuestas)



## ¿Qué problemas encontró en el módulo médico? (5 respuestas)

Debería verse el color del estado de un paciente

debería haber un buscador de pacientes. Una tabla de mediciones no dice nada.

La historia clínica se ve mal

No es tan fácil saber cuando un paciente de ese médico tiene una alerta roja o amarilla

Identificar a los pacientes con estado de salud rojo es crítico. La historia clínica del paciente no es muy fácil de leer.

### Solución:

Por ello decidimos colocar gráficas para cada uno de los tipos de mediciones como se muestra a continuación:



## Pacientes

Estado de los pacientes

Show 10 entries

Search:

Cédula	Nombre	Marcapasos
	Pedro null	0
101354256	Carlos Perez	123
245367154	Pedro Pablo Sanchez	144
102485429432	Pedro Ardila	0

Showing 1 to 4 of 4 entries

Previous

1

Next

## Historia clínica

Volver a todos mis pacientes

Información general

Perfil

Antecedentes

Información personal

Nombre:

Carlos Perez

Dirección:

coco

Cédula:

101354256

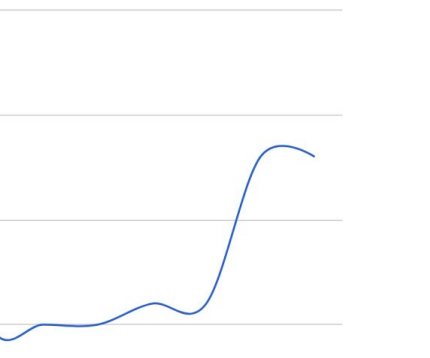
Email:

ca.p@gmail.com

Enviar consejo

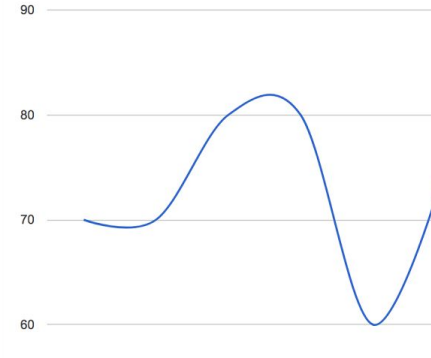
Enviar consejo

Frecuencia cardiaca



Fecha	Frecuencia
2017-1-30 21:00:00	100
2017-1-30 21:00:00	90
2017-2-1 00:46:40	90
2017-2-1 00:46:40	91
2017-2-1 04:33:20	90
2017-2-2 04:33:20	98

Presión sanguínea



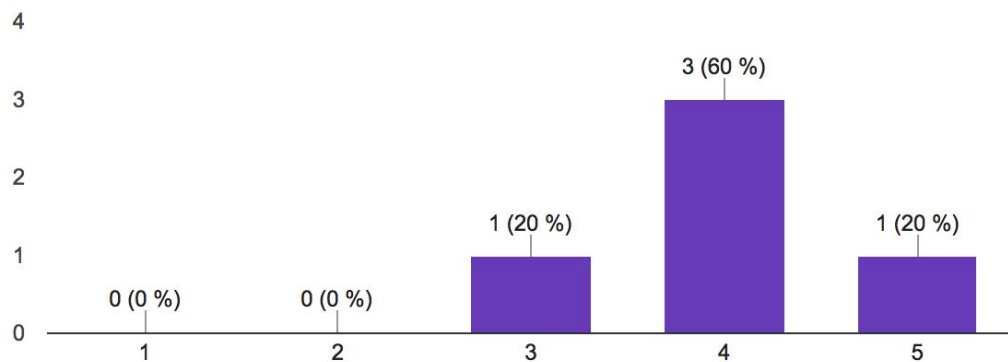
Fecha	Presión
2017-1-30 21:00:00	70
2017-2-1 00:46:40	82
2017-2-1 04:33:20	60
2017-2-2 04:33:20	75

## Módulo Paciente

Como resultado de la crítica colaborativa obtuvimos que era necesario mostrar de manera más intuitiva el estado del paciente y no solo tablas. Además nos indicaron que la interfaz se veía desordenada a pesar de que eran claros los pasos para ver el estado de un paciente.

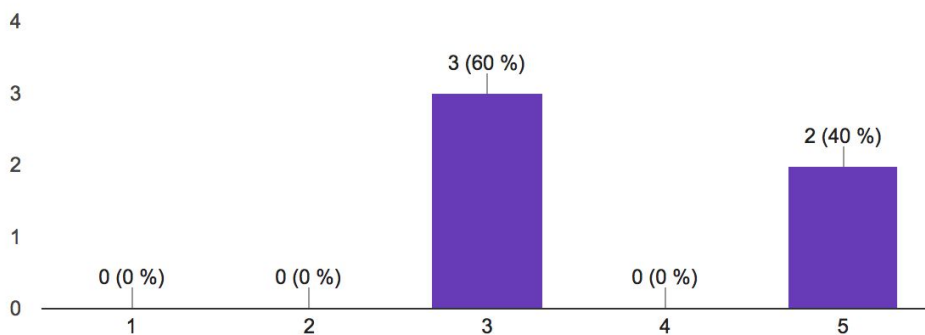
## Resultados:

¿Qué tan fácil es ver la información de un paciente? (5 respuestas)



Después de la actual sesión ¿los pacientes recordaran los pasos para ver su información?

(5 respuestas)



¿Qué problemas encontró en el módulo paciente? (4 respuestas)

Se ve desordenado

Una tabla de mediciones no dice nada

Igual que en médico

Al ver las tablas del paciente es difícil saber que dato es de cada histórico o medida, es confuso ver renglones iguales

## Solución:

Por ello modificamos la interfaz de la siguiente manera:



## Patrones de Diseño

En aras de mejorar el atributo de calidad de modificabilidad, se han implementado los siguientes tres patrones de diseño. Se presenta el estado anterior al uso del patrón, seguido del actual y por último como mejora la modificabilidad del proyecto.

- **Observer:** Anteriormente, cada que llegaba la medición de tipo Emergencia, era necesario notificar de manera directa a todos los médicos y cada quien revisaba si la medición era de uno de sus pacientes. Ahora, al implementar el patrón Observer, cada que llega una medición de tipo emergencia se notifica a los médicos y cada uno de estos, cuando tengan espacio para hacerlo, revisan la notificación y si esta es de un paciente de su interés se genera una alerta en la servicio web y móvil de los médicos interesados en dicho paciente. Este cambio permite una mayor modificabilidad a futuro, pues en caso de algún nuevo requerimiento, los nuevos interesados se pueden agregar directamente a la lista de notificados y así atender cuando se ha generado una nueva emergencia y no tener que estar verificando cada cierto tiempo en busca de que existan o no emergencias.

- **Mediator:** Antes del experimento 3, las clases y métodos usados para el manejo o representación de objetos persistentes (Médico,Paciente,Medición,etc) tenían -cada uno por su cuenta- libre acceso a la base de datos. Ahora, se ha implementado un Mediator a cargo del acceso a esta, este patrón de diseño está representado en el proyecto a través de la clase APPController el cual tiene una instancia de tipo EPCrudService para cada objeto que se maneja en la base de datos. Actualmente es la clase APPController la encargada de gestionar las operaciones en la base de datos, por lo que ahora cada clase y método que necesiten acceder a esta, deben hacerlo a través de un llamado a la instancia del objeto EPCrudService que requiera para funcionar (ej: la instancia medicosCrud para acceder a la tabla médicos de la base de datos). Con esta implementación ha mejorado la modificabilidad, pues ahora de requerirse algún cambio en un método o incluso de proveedor del servicio base de datos, las modificaciones se haran unicamente en la clase EPCrudService y EPCrudService sin necesidad de buscar todos los métodos dentro del proyecto que se vieran afectados por el cambio. Por último, al limitar los accesos a una sola clase se ha mejorado el atributo de seguridad.
- **Facade:** Anteriormente cada que un método necesitaba acceder a información de un paciente, era necesario crear una instancia de este objeto y hacer múltiples llamados hasta obtener la información de interés. Ahora, se ha creado una interface Facade con los métodos relacionados con el objeto paciente, de esta forma, cada que un método necesita información de este objeto contacta únicamente a través de la fachada, reduciendo así el número de llamados y el acoplamiento entre clases. Esta estrategia mejora la modificabilidad del proyecto debido a que ahora cada nuevo método o cambio en alguno existente se refleja únicamente a través de la Facade. Por ejemplo, si la manera de obtener los consejos de un paciente cambia, no es necesario cambiar todos los métodos distribuidos en el código que lo hacían, sino que únicamente realizar un único cambio en la clase que implementa la interface y mantener el llamado del método (el de la fachada) intacto para que no se altere los métodos interesados en dicha información. de igual manera de alterarse el llamado los cambios representan un menor número de líneas a modificar.