UNIVERSIDAD DE CONCEPCIÓN
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA ELECTRICA

SISTEMA REMOTO DE CAPTURA DE VARIABLES FISIOLÓGICAS PARA EL APOYO EN CUIDADOS PALIATIVOS EN EL HOGAR

POR

Gonzalo Andrés Rojas Bernard

Memoria de Título presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título profesional de Ingeniero(a) Civil Biomédico

Profesor(es) Guía

Prof. Esteban Pino Quiroga

Profesional Supervisor

Jaime Jiménez Ruiz

Agosto 2022
Concepción
(Chile)

Agradecimientos

A mis padres y hermana, principalmente por brindarme apoyo y cariño incondicional todos los días para poder trabajar, estudiar y enfocarme en lo que me gusta hacer, los amo con todo mi corazón.

A mis amigos, quienes se preocupaban de sacarme de la rutina para poder disfrutar y compartir con ellos, los quiero mucho.

A Valeria Ortega, por ser un gran pilar en todo lo que he logrado hasta el momento, no sabes cuánto valoro el tiempo contigo.

A mis primos por ser como hermanos para mí, gracias por acompañarme siempre.

Y, por último, a mis perritos, únicamente por existir.

Resumen

En el presente documento se informa sobre la implementación de un sistema de monitorización continua mediante un dispositivo de sabanilla con sensores integrados no invasivos. El sistema contará con la habilitación del dispositivo en sí, la implementación de una plataforma web para la gestión y visualización en tiempo real de un paciente/usuario y de la realización de una prueba piloto en pacientes reales para evaluar la sabanilla en cuestión.

El objetivo general de este estudio es mejorar los cuidados paliativos de pacientes mediante un monitoreo con sensores no invasivos y remotos, esto mediante la incorporación de una sabanilla en la cama del paciente. Este estudio se hizo en colaboración con la empresa **Healthtracker Analytics**.

La sabanilla funciona mediante sensores FSR que detectan cambios de presión y los transforman a voltaje (V). Estos cambios de presión son enviados a un microcontrolador ESP32 el cual cuenta con un módulo WiFi que estará conectado a la red correspondiente al hogar del paciente. Desde el microcontrolador se envían datos a un servidor otorgado por la Universidad de Concepción en donde se podrá observar la actividad de cada sabanilla activa.

Esto fue logrado mediante la tramitación de la documentación del Comité de Ética Científico (CEC) proporcionado por **RedSalud** con objetivo de obtener su aprobación y posteriormente el consentimiento informado del paciente firmado para luego poder proceder a la implementación del dispositivo en el hogar del individuo en la prueba piloto.

Se utilizaron prototipos del dispositivo pertenecientes al Laboratorio de Ing. Civil Biomédica de la Universidad de Concepción, primero verificando el funcionamiento de todos sus componentes.

A modo de conclusión, se monitoreó continuamente el estado de un paciente postrado en cuidados paliativos. Este sistema es capaz de saber el estado actual del paciente, obtener su frecuencia respiratoria en tiempo real, predecir la posición actual del paciente y detectar periodos fuera de cama, datos que serán útiles tanto como para al cuidador como para el equipo de salud encargado del individuo.

Abstract

The present paper reports on the implementation of a continuous monitoring system using a non-invasive, integrated sensor-embedded bedpan device. The system will have the enablement of the device itself, the implementation of a web platform for the real-time management and visualization of a patient/user and of the realization of a pilot test on real patients to evaluate the bed sheet in question.

The overall objective of this study is to improve palliative care of patients through non-invasive and remote sensor monitoring by incorporating a bed sheet in the patient's bed. This study was done in collaboration with the company **Healthtracker Analytics.**

This prototype sheet works by FSR sensors that detect pressure changes and transform them into voltage (V). These pressure changes are sent to an ESP32 microcontroller which has a WiFi module that will be connected to the network corresponding to the patient's home. From the microcontroller, data is sent to a server provided by the University of Concepción where the activity of each active sheet can be observed.

This was achieved by processing the documentation of the Scientific Ethics Committee (CEC) provided by **RedSalud** with the aim of obtaining its approval and subsequently the patient's signed informed consent to proceed with the implementation of the device in the individual's home in the pilot test.

Prototypes of the device belonging to the Biomedical Civil Engineering Laboratory of the Universidad de Concepción were used, first verifying the functioning of all its components.

In conclusion, the condition of a patient who is bedridden due to an illness was continuously monitored. This system can know the patient's current state, knowing his or her respiratory rate in real time, predicting the patient's current position, among other capabilities that will be useful for both the caregiver and the health team in charge of the individual.

Tabla de Contenidos

Lista de Tablas

Lista de Figuras

Capítulo 1. Introducción

1.1. Introducción General

En el mundo ocurren 15 millones de casos nuevos de cáncer al año, de los cuales un 60% termina en muerte, lo que equivale a 7,4 millones de personas. De ellos, un 75% a un 80% sufre dolores en todas las etapas de la enfermedad [1].

El dolor por cáncer ocurre en un tercio de quienes están en tratamiento activo y en más de dos tercios de quienes están en una etapa avanzada o terminal. El 80% de los dolores son debido a causa tumoral, el 20% a causa del tratamiento de la enfermedad. Sin embargo, existe un 10% en el cual radica otras causas tales como síndromes inducidos por el cáncer y causas no relacionadas con la enfermedad (infarto al miocardio, enfermedad isquémica, entre otras) [2].

El problema actual radica en que no existe una comunicación rápida y eficiente del estado actual del paciente con el personal médico correspondiente (cuidador o equipo de salud), considerando que un seguimiento continuo mediante un monitoreo no invasivo puede ser indispensable para mejorar la calidad de vida del paciente. Es por esto que monitorear de forma remota el estado de la persona en etapa avanzada o terminal de su enfermedad, permitiría al equipo de salud tener un diagnóstico concreto y basado en el estado actual del paciente.

La presente memoria, busca evaluar con un dispositivo de sabanilla con sensores de presión integrados de manera no invasiva, un monitoreo continuo con gestión de pacientes y visualización en tiempo real mediante la implementación de una plataforma web. Esto a través de una prueba piloto para obtener retroalimentación de la funcionalidad de la sabanilla en casos de pacientes reales. Para la realización de la prueba piloto, se habilitaron los dispositivos otorgados por la Universidad de Concepción para reparar sensores averiados, habilitar comunicación con un servidor que recibirá los datos, soldar nuevas placas de circuito correspondientes a la sabanilla, para luego verificar un correcto funcionamiento entre todos los componentes.

Se considerará el estado del paciente (refiriéndose a su capacidad para moverse) a quien se proporcionará dicho dispositivo un porcentaje de movilidad de 40% o menor según la escala de funcionalidad paliativa (PPS) [2]. Esta escala indica que el paciente se encuentra incapaz de realizar la mayoría de las actividades cotidianas y necesita asistencia general. El dispositivo es implementado como se muestra en la Fig. 1.

Para el envío de datos obtenidos de la sabanilla al servidor correspondiente, se utilizó un sistema de adquisición y almacenamiento de datos el cual consiste en un circuito con multiplexores, microprocesador (ESP 32) con módulo WiFi para el manejo de datos del dispositivo en cuestión

1.2. Objetivos

1.2.1 Objetivo General

Implementar sistema de monitorización continua mediante la habilitación de dispositivos, diseño de plataforma web y realización de prueba piloto en pacientes.

1.2.2 Objetivos Específicos

- Obtener autorización del comité de ética científico (CEC).
- Comprobar la funcionalidad de sensores integrados a los dispositivos.
- Configurar servidor de la Universidad de Concepción para correcta comunicación con dispositivos.
- Implementar plataforma web para control y visualización en tiempo real de pacientes/usuarios que ocupen el dispositivo.
- Realizar prueba piloto.
- Analizar resultados obtenidos de prueba piloto.

1.3. Alcances y Limitaciones

El presente trabajo es un sistema de monitorización continua para comprobar y evaluar la funcionalidad del prototipo de sabanilla. Se contará con una plataforma web en donde el cuidador y el equipo correspondiente podrá acceder a los datos extraídos de la sabanilla para realizar un análisis concreto del estado actual del paciente y su visualización en tiempo real durante la prueba piloto.

1.4. Metodología

1. Selección y preparación de dispositivos de monitorización continua.

Se analizarán dispositivos de monitorización continua (sabanillas) disponibles en el Laboratorio de Ing. Civil Biomédica de la universidad. Las sabanillas serán revisadas para

identificar sensores averiados o algún error en el circuito para posteriormente corregirlos y contar con prototipos funcionales para su futura implementación.

2. Utilización de dispositivos para el monitoreo remoto y de plataforma para manejo de datos.

Se utilizarán dispositivos, basado en experiencias previas, consistente en una sábana que se instala en la cama del paciente, sin contacto directo con la piel. Se propone una plataforma web para la visualización desde el equipo correspondiente en donde se podrá analizar el estado de los pacientes que estén monitoreados con el dispositivo en cuestión. Al igual un modelo de visualización en tiempo real por cada paciente.

3. Implementación de plataforma web para visualizar y gestionar pacientes con sabanilla.

Mediante programación en Python, se escribirá un código el cual habilita una página web, en la cual se permitirá la visualización en tiempo real del usuario acostado con la sabanilla implementada en su respectiva cama. Los datos serán procesados con el tal de poder predecir la posición actual del usuario acostado.

Además, permitirá una gestión sencilla de los usuarios que estén dentro del proyecto, mediante un ingreso fácil y una posterior asignación a una sabanilla que se encuentre disponible, esto es, que no esté asignada a ningún paciente y que esté ingresada a la base de datos.

4. Prueba piloto de prototipo en pacientes.

Con la aprobación del CEC, se buscarán pacientes voluntarios en atención domiciliaria asociados a ***AtencionDomiciliaria*** y ***RedSalud*** que deseen participar en la prueba del dispositivo. Se incluyen ciertos criterios de exclusión e inclusión:

Criterios de inclusión:

- Paciente postrado, con PPS inferior o igual a 40%.
- Paciente lúcido.
- Diagnóstico de Cáncer.
- Ubicado en la Región Metropolitana.

- Atendidos por un mismo equipo clínico.
- Deben firmar el consentimiento informado.
- Debe ser mayor de 18 años.
- Cuenta con conexión a Internet.

Criterios de exclusión:

- Paciente con peso superior a 120 kg.
- Paciente no cuenta con red de apoyo en el hogar.

5. Análisis de resultados.

Se busca comprobar si el sistema implementado para monitoreo continuo logró su objetivo, mediante un análisis del procesamiento utilizado en la prueba piloto en términos de estabilidad de los componentes tales como el servidor, la plataforma web y los algoritmos utilizados en esta, tales como un algoritmo de predicción de posiciones, frecuencia respiratoria, entre otros.

1.5. Temario

Capítulo 1:

Introducción.

Cuenta con la introducción al tema de monitoreo no invasivo en conjunto de la metodología y objetivos del proyecto.

Capítulo 2:

Marco teórico.

En este capítulo se relatan los trabajos previos que se realizaron con la sabanilla en orden cronológico.

Capítulo 3:

Habilitación de dispositivos.

Se indican los avances realizados en el laboratorio de Ing. Civil Biomédica en conjunto con las pruebas realizadas para verificar el funcionamiento de los dispositivos.

Capítulo 4:

Plataforma web para visualización de datos.

Capítulo con información detallada de la implementación y funcionalidad de la plataforma web diseñada para la gestión de pacientes.

Capítulo 5:

Prueba piloto.

Contará con la información recopilada de la prueba piloto realizada con la sabanilla.

Capítulo 6:

Conclusiones y trabajo futuro.

Contará con la conclusión, discusión y trabajo futuro del rendimiento de la sabanilla.

Capítulo 2. Marco Teórico

2.1. Introducción

Se han realizado experiencias anteriores en las cuales se ha hecho uso de la sabanilla para la captación de señales fisiológicas y algunos parámetros asociados con el objetivo de mejorar la calidad de vida o trabajo de las personas o pacientes. En específico, el uso de la sabanilla fue enfocado al estudio del sueño (polisomnografía), en donde se añadieron las variables fisiológicas de los voluntarios que participaron anteriormente, al igual que en pruebas de laboratorio [3].

2.2. Investigación

Dentro de la revisión de trabajos previos se encuentran memorias de título anteriores en conjunto con publicaciones realizadas para la prueba del dispositivo en cuestión en distintos escenarios en donde se consideraron necesario su uso.

Primero, se considera necesario saber algunos conceptos para comprender la necesidad que refleja la utilización de un dispositivo como este.

2.2.1 Cuidados paliativos

Los cuidados paliativos y el alivio de dolor son los cuidados que tienen como finalidad mejorar la calidad de vida de una persona que se encuentra con una enfermedad que pone su vida en peligro, como lo es el cáncer [4]. El objetivo de estos cuidados es evitar y tratar lo más pronto posible los síntomas y los efectos secundarios de la enfermedad sufrida por el paciente y de su tratamiento correspondiente, esto abarca problemas psicológicos, sociales y espirituales.

Estos cuidados pueden proporcionarse a lo largo del tratamiento de cáncer, pero es más común en pacientes con etapas más avanzadas en donde la movilidad se le ve severamente reducida. Puede proveerse desde el diagnóstico hasta el final de la vida.

Los pacientes que estén bajo cuidados paliativos, generalmente se encuentran en su hogar recibiendo prestaciones de profesionales cada semana o cada mes, aquí es donde entra la importancia de un dispositivo no invasivo de monitorización continua del paciente, lo cual evitaría visitas innecesarias, priorizando el bienestar y privacidad del individuo.

2.2.2 Monitoreo en el hogar

En el hogar del paciente es donde se realizará la monitorización continua con el dispositivo de sabanilla. Como es un dispositivo no invasivo, el paciente estará en la comodidad de su casa sin saber que está siendo monitoreado continuamente para saber su estado actual cuando se encuentre acostado en su cama, lo cual la mayoría de pacientes que se encuentren con tratamiento de cáncer pasan la mayor parte del tiempo postrados en la cama debido a la misma enfermedad o al tratamiento recibido, lo cual implique que quizá necesite más prestaciones por semana o diarias.

2.2.3 Efectos del tratamiento contra el cáncer

Este tratamiento puede tener muchos efectos secundarios, debido a que los tratamientos más comunes para esta enfermedad son la quimioterapia y radioterapia, los cuales normalmente dañan células sanas. Uno de los efectos secundarios más comunes son náuseas y vómitos, debido a que estos tratamientos pueden hacer que la persona se enferme del estómago hasta de solo pensar en el tratamiento de la enfermedad. Existen medicamentes que pueden ayudar a controlar las náuseas para que el paciente se sienta mejor, esto también puede estar incluido en el caso de que una persona solicite cuidados paliativos [5].

El cáncer por si mismo y sus tratamientos en la mayoría de los casos causan dolor, y esto puede dificultar las actividades cotidianas de la persona y su calidad de vida. En etapas avanzadas, el paciente ya se encuentra postrado en la cama la mayor parte de sus días, requiriendo asistencia para casi todas sus actividades.

Es ahí en donde un dispositivo que pueda medir variables fisiológicas de manera no invasiva y remota puede llegar a tener un gran uso para el cuidador o el equipo de salud asignado al paciente que se encuentra postrado para que su tratamiento pueda ser manejado de una manera más eficiente y cómoda para la persona mediante un monitoreo continuo de su estado actual para otorgar una ayuda en la toma de decisiones clínicas correspondientes a su situación de salud.

2.2.4 Estudio del sueño

El estudio del sueño se ha mostrado ser de gran importancia para analizar la severidad de algún trastorno del sueño que puede estar padeciendo una persona. Un examen común es la polisomnografía, la cual es útil para determinar la seriedad de dichos trastornos, esto también dando ayuda al personal médico para tomar una mejor decisión sobre el tratamiento a seguir [6].

El examen mencionado anteriormente muestra resultados positivos en la detección de algún

trastorno del sueño, pero tiene la desventaja de que tiene que ser realizado en un centro de salud o laboratorio, en cambio, se han realizado pruebas de sueño en casa (HST), con la ayuda de un monitor portátil pero que se usa únicamente para buscar apnea obstructiva del sueño (AOS), mediante el uso de sensores de flujo de aire que se deben ubicar debajo de la nariz, cinturones para mantener los componentes en la posición correcta, entre otras instrucciones adicionales para un monitoreo correcto [7].

Tomando en cuenta los factores invasivos de las pruebas mencionadas anteriormente, un dispositivo que sea capaz de monitorear continua y remotamente desde la comodidad del hogar y que el paciente no sienta incomodidades al usarlo, como lo es la sabanilla expuesta en la presente memoria, tendría beneficios tanto para el paciente como para el personal de salud para un correcto diagnóstico.

2.2.5 Trabajos previos en sabanilla de monitoreo

El primer trabajo que se inició como tesis de magíster, publicado por Dörner A. (2013), con el uso de la sabanilla fue una propuesta de evaluación objetiva para análisis de calidad de sueño, en donde se implementó la sabanilla utilizando 24 sensores de presión la cual sería ubicada bajo las sábanas de la cama del paciente. Este dispositivo adquirió la señal respiratoria y detectó los movimientos del cuerpo (BM) durante la noche, también tomando en cuenta otros parámetros tales como el tiempo en cama (TB), el porcentaje de tiempo en intervalos de sueño menores a 20 minutos, entre otros. Con estos parámetros se propuso un índice de calidad de sueño para un análisis objetivo en pacientes que padecen de algún trastorno del sueño. Este estudio fue validado y probado en 12 voluntarios, entre ellos 4 diagnosticados previamente con Síndrome de Apnea Obstructiva del Sueño (SAHOS) [8].

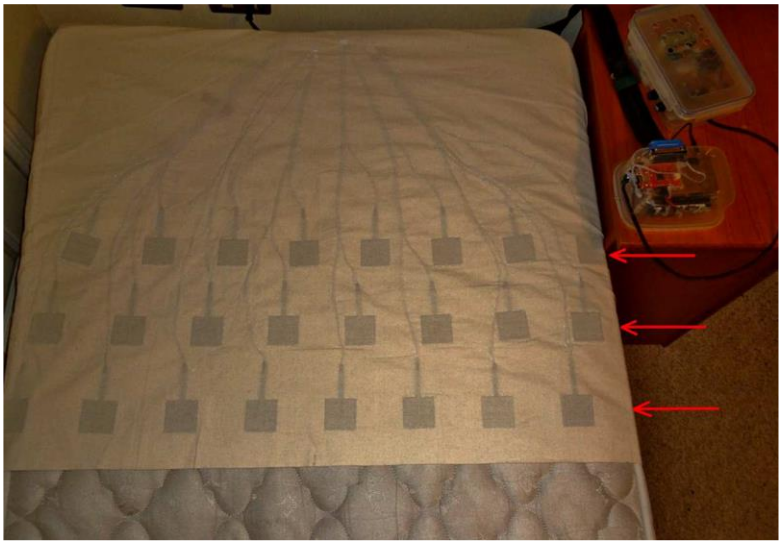


Figura 1: Sabanilla implementada en cama, cada línea roja es una fila de ocho sensores FSR.

Fuente: Astrid Ivonne Dörner De La Paz, "*Propuesta de Evaluación Objetiva para análisis de Calidad de Sueño*"[8].

El seguimiento del trabajo mencionado anteriormente fue una validación de la malla de sensores de presión para estudios del sueño, memoria de título publicada por Moran A. (2014). En este trabajo se mejoró el diseño de la sabanilla implementando 2 nuevos equipos para realizar mediciones en pacientes derivados hacia un laboratorio del sueño y en voluntarios monitoreados en sus hogares. Los resultados de este trabajo validaron el uso de la sabanilla como prediagnóstico para descartar a personas que no sufren apneas del sueño utilizando el índice de calidad de sueño [9].

Luego se realizó un trabajo con este dispositivo sobre un estudio de la calidad de sueño en instalaciones mineras para evaluar si el número de accidentes está relacionado con dormir de manera inadecuada. En este trabajo se realizan mediciones de distintos parámetros para medir el índice de calidad de sueño que se propuso en los trabajos anteriormente mencionados. Los resultados de este trabajo mostraron que se podrían hacer unas mejoras en el sitio de trabajo como el control de humedad, tecnología en camas y aislamiento de ruido y calor para obtener una mejor calidad de sueño [10].

Con el trabajo anterior se realizó la validación de este dispositivo para estudios sobre la calidad del sueño, específicamente para el diagnóstico de apnea del sueño. Este trabajo valida algoritmos que se utilizaron para conseguir los parámetros necesarios para medir el índice de calidad de sueño en conjunto con la detección de apneas durante el sueño [11].

Un trabajo con cierta similitud al presente se trata de una prueba con la sabanilla en

pacientes postrados con cuidados paliativos implementando un sistema no invasivo, capaz de dar soporte a los cuidadores o equipo de salud respecto a las necesidades que presenta el paciente postrado, memoria de título publicada por Oliva C. (2019). Se diseñaron hardware y algoritmos que fueron capaces de medir y procesar las señales obtenidas del dispositivo, también diseñando un sistema de comunicación para el envío de datos y un interfaz en donde se pueden visualizar de manera amigable el estado actual del paciente y sus distintos reportes necesarios [12].

Finalmente, un trabajo se enfocó en crear una plataforma en donde se pueden observar de mejor manera los datos recibidos desde la sábana mediante la implementación de una aplicación web, memoria de título publicada por Tardones J. (2021). En este trabajo, se utilizó un sistema Cliente-Servidor en el cual los clientes serían los usuarios que cuenten con un dispositivo de la sabanilla que envían datos por WiFi. El servidor corresponde a un Docker dedicado y remoto el cual pertenece al Laboratorio de Ing. Civil Biomédica. En este servidor se crea la base de datos mediante la creación de tablas utilizando PostgreSQL y algoritmos en Python [13].

Cada sabanilla era identificada mediante el módulo de ESP32 correspondiente, los cuales eran la fuente que enviaba los datos hacia el servidor para luego ser analizados en la aplicación web anteriormente mencionada [13].

2.3. Discusión

Los trabajos mencionados anteriormente fueron de gran ayuda en la evolución de la sabanilla como dispositivo capaz de captar señales fisiológicas de manera no invasiva. A medida que fue avanzando en el tiempo, se le fueron añadiendo nuevas modificaciones ya sea a su diseño de circuito, la implementación de nuevos algoritmos y la creación de nuevas interfaces para poder facilitar el uso de terceros y poder convertir este dispositivo en uno que se le pueda dar utilidad cuando se trate de pacientes que pasan la mayor parte del tiempo postrado en la cama debido a su enfermedad, mayormente por tratamientos oncológicos.

Los cuidados paliativos se podrían apoyar del envío de datos desde la sabanilla para poder tomar decisiones clínicas concretas y correctas basándose en el estado actual del paciente y sus necesidades en el momento.

Capítulo 3. Habilitación de dispositivos

3.1. Introducción

Se han realizado avances prácticos en la investigación mediante asistencias al laboratorio de la universidad y la tramitación de la documentación del comité de ética científico para la aprobación del uso del dispositivo en pacientes de la Región Metropolitana. Los avances en el laboratorio se han enfocado en la verificación de inventario, contando la cantidad de dispositivos que actualmente se encuentran en la universidad, en la habilitación del funcionamiento del circuito que se conecta a la sabanilla que posteriormente envía datos al servidor, ocupando una Raspberry Pi para su simulación de servidor y un ESP32 para el envío de datos recibidos desde la sabanilla y en la soldadura de componentes a las nuevas placas impresas habilitadas por el Laboratorio de Ing. Civil Biomédica. Los componentes utilizados para el envío de datos y funcionamiento de la sabanilla en las primeras pruebas se muestran en la Fig. 2 y Fig. 3.

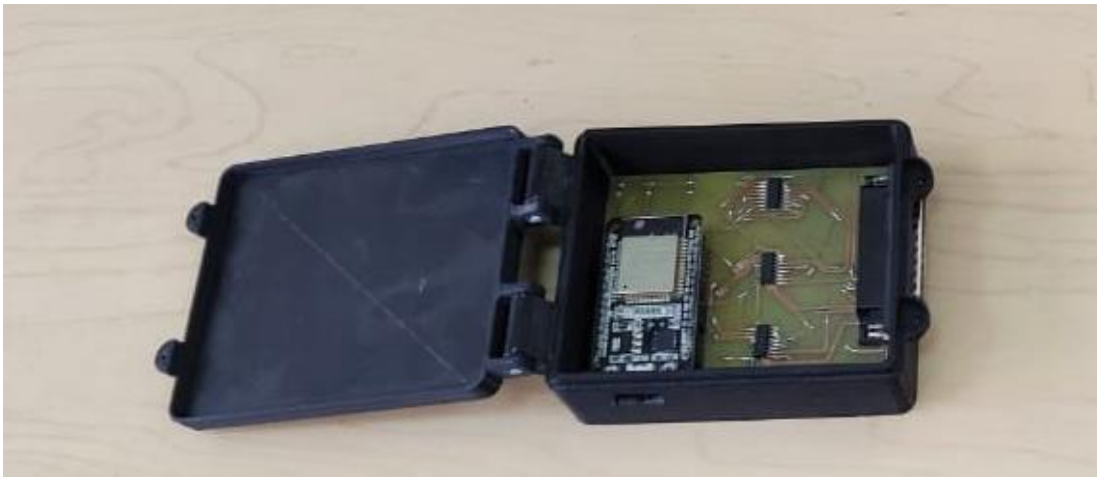


Figura 2: Componentes de circuito en caja para funcionamiento de sabanilla (Placa, ESP32, multiplexores).

Figura 3: Componentes de circuito para verificación de funcionamiento de sabanilla (Router y Raspberry Pi).

3.2. Diagrama de sistema propuesto

Para la incorporación del circuito que recibe y envía los datos hacia el servidor, se implementaron los respectivos algoritmos a los microcontroladores correspondientes para su funcionamiento. El funcionamiento del circuito se muestra la Fig. 4.

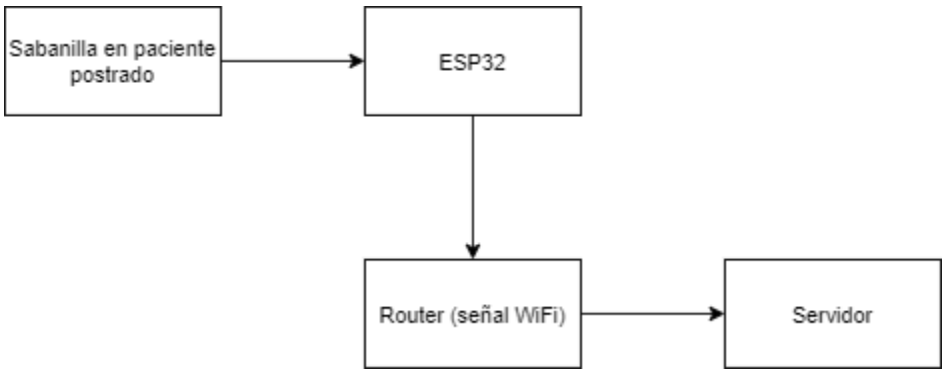


Figura 4: Diagrama de flujo funcionamiento circuito sabanilla.

Como se puede observar, la sabanilla está implementada en la cama del paciente postrado, en donde la tarjeta ESP32 recibe los datos de los sensores integrados. Luego, el ESP32 que está conectado a la red WiFi, envía los datos al servidor en donde pueden ser visualizados y analizados de forma remota.

3.3. Configuración Raspberry Pi y prueba de sabanillas

En el laboratorio, se utilizó una Raspberry Pi 3B como servidor en donde se instalaron las librerías correspondientes para su correcto funcionamiento, para luego, una vez habilitado, implementar las mismas dependencias en el servidor otorgado por la Universidad de Concepción.

Para el uso de la Raspberry Pi se tuvo que instalar un sistema operativo (OS) mediante un adaptador de tarjeta SD a un computador personal para poder visualizar dicho OS en un monitor externo y posteriormente instalar las librerías necesarias y fijar una dirección IP para la conexión con el Router y el ESP32. También se realizó una capacitación de Ubuntu para el conocimiento de los comandos para la instalación de archivos necesarios, para luego implementarlos en el servidor de la UdeC.

Se instalaron todas las dependencias necesarias para el funcionamiento del envío de datos desde la tarjeta ESP32 hacia la Raspberry Pi, incluyendo las librerías de Python que requiere el código actual que envía información del dispositivo a una base de datos habilitada en PostgreSQL.

Se conectó a un computador personal el circuito el cual contiene el microcontrolador ESP32 para la detección de las señales provenientes de la sabanilla para la verificación de la ejecución que debe ir implementado en este. Se comprobó su correcto funcionamiento mediante el uso de la interfaz que posee el Router WiFi para simular la conexión WiFi en la casa de un paciente. También el ESP32 enciende un LED azul cuando la conexión al Router y a la Raspberry Pi es exitosa.

Con las conexiones necesarias para que el dispositivo sea capaz del envío de datos, se pudo verificar el funcionamiento de todos los dispositivos habilitados por el Laboratorio de Ingeniería Civil Biomédica, verificando que sensores FSR estaban averiados o saturados. Esto se logró mediante la visualización de los datos enviados en tiempo real, actualizados en la terminal de Ubuntu mientras se está ejecutando el código en Python que lee y envía datos.

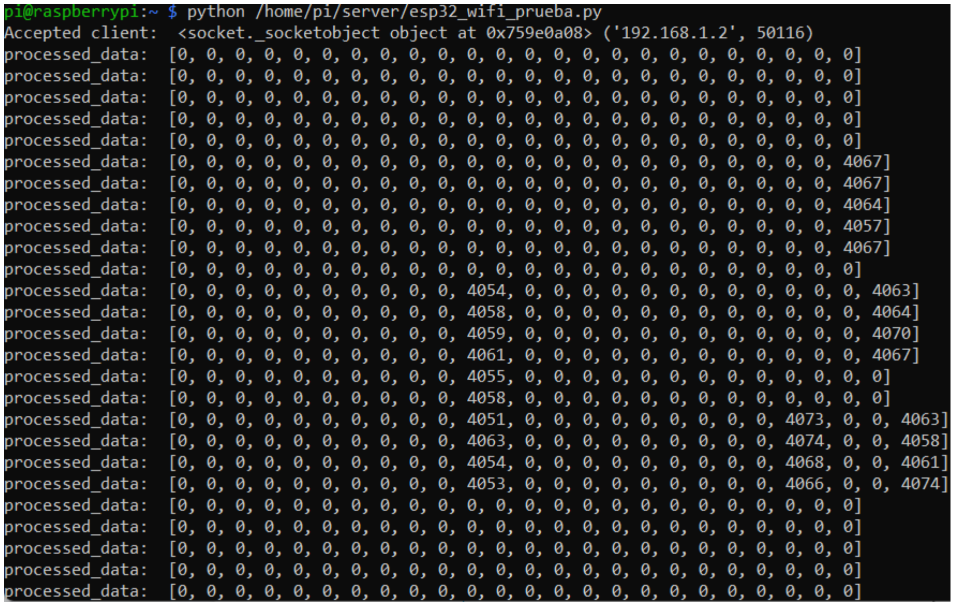


Figura 5: Visualización de sensores presionados en terminal Ubuntu.

En la figura 5 se puede observar una lista de datos procesados con 24 elementos, cada uno corresponde a un sensor FSR implementado en la sabanilla. Cuando un sensor está en 0, significa que no está siendo presionado y cuando muestra un número mayor a 0 (dependiendo de cuanta presión se aplique en el sensor), se está presionando un sensor.

En la Fig. 6 se puede observar la ubicación de cada sensor FSR en la sabanilla.

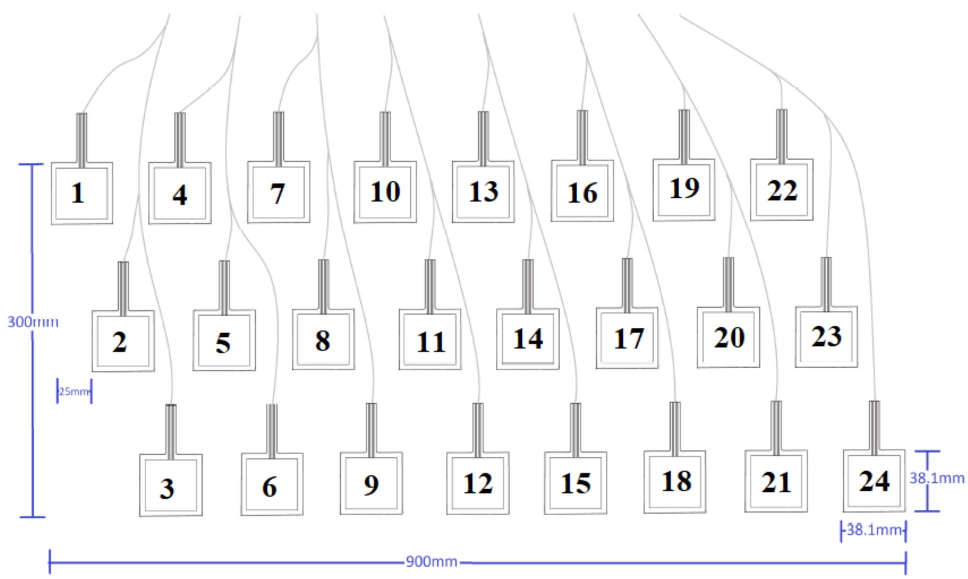


Figura 6: Sensores enumerados en el orden que están ubicados en el dispositivo [9].

A partir de las pruebas realizadas en el laboratorio con las sábanas disponibles, se obtienen los siguientes resultados que se pueden observar en la Tabla 1.

TABLA I

OBSERVACIONES DE DISPOSITIVOS PROBADOS EN EL LABORATORIO

Dispositivo	Observaciones
Sabanilla N°0	Primera sábana, no se utilizará en prueba piloto.
Sabanilla N°1	2 sensores averiados (1, 11) y 1 sensor débil (23).
Sabanilla N°2	Sensores 13 y 15 levantan la misma señal.
Sabanilla N°3	1 sensor saturado (22).
Sabanilla N°4	Sábana con todos los sensores funcionando correctamente.
Sabanilla N°5	8 sensores saturados (20, 17, 4, 8, 5, 22, 14, 10).
Sabanilla N°6	Sábana con todos los sensores funcionando correctamente.
Sabanilla N°7	1 sensor averiado (12).
Sabanilla N°8	1 sensor averiado (4) y 1 sensor saturado
Total: 9 sabanillas.	

A partir de estos resultados se observa que los dispositivos están relativamente en buen estado a excepción de la sabanilla N° 5, la cual cuenta con un grán número de sensores saturados, esto quizá debido a que no contaba con una funda y el su conector DB25 para conectarse con la placa estaba destapado.

3.4. Implementación de nuevas placas

Se facilitó acceso al laboratorio de Ing. Civil Biomédica en donde se encuentran las máquinas específicas para la impresión de 6 placas PCB con el circuito implementado para después realizar pruebas de funcionamiento con las sabanillas. Se recibió el archivo EAGLE correspondiente al esquemático de la tarjeta con algunos cambios específicos para reducir el ruido eléctrico,

interferencia, entre otros. El cambio más significativo fue añadir un plano de tierra, así asegurando la misma tierra común para todos los componentes de la tarjeta.

Se realizó la compra de los componentes necesarios (multiplexores, resistencias (10kOhm), conectores DB25 y pin header) para la posterior soldadura de estos en su lugar correspondiente como se muestra en la Fig. 5. Para soldar se utilizó un cautín, flux, estaño y ocasionalmente un PCB holder para sostener de mejor manera las placas y realizar de mejor manera la soldadura, dado a que se tratan de componentes relativamente pequeños.

Para realizar la soldadura, se dejó la placa en un lugar fijo y plano, a una altura conveniente para una observación óptima, con la ayuda de una lupa se hizo más sencilla esta última acotación. Se utilizó una pasta fluida llamada Flux, la cual hace que se adhiera el estaño exclusivamente al cobre (lugares de soldadura). Luego, se sostuvo el cautín con la mano dominante y el estaño en la otra, acercando lentamente el cautín y estaño al lugar de la soldadura con el componente en posición.

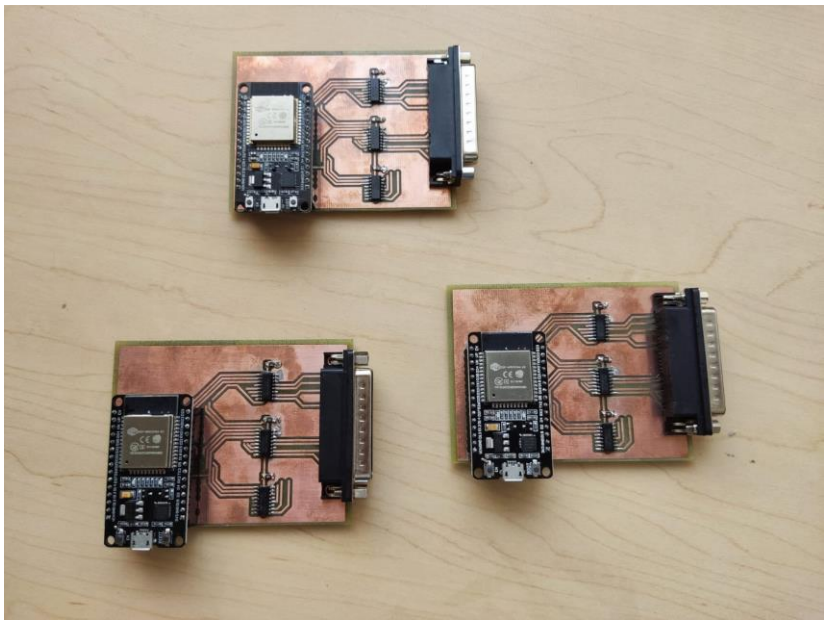


Figura 7: Nuevas placas soldadas junto a tarjeta ESP32.

Las nuevas placas fueron posteriormente probadas, de la misma forma en la que fueron probados los sensores FSR (consola Ubuntu), para asegurar que no hubieran puentes entre las líneas de cobre impresas.

El diseño de las placas fue proporcionado previamente mediante la plataforma Eagle, la cual permitió posteriormente la impresión de esta.

3.5. Arreglo de sensores averiados o saturados

Luego de identificar los sensores que estaban averiados o saturados, como se muestra en la Tabla 1, se retiraron los que efectivamente estaban malfuncionando, corroborando esta infomación con un multímetro. Luego de retirarlos se realizo una costura mediante hilo y aguja en el lugar de donde se retiró el sensor para volver a habilitarlo nuevamente.

Luego de la costura, se utilizó un pegamento conductor de electricidad (de plata, especificamente), para unir los terminales del sensor FSR al hilo conductor que va hacia el cable que se conecta a la placa en donde se encuentra el microcontrolador ESP32.

3.6. Habilitación servidor de la Universidad de Concepción

Como se ha mencionado anteriormente, para las pruebas de sabanillas se estaba ocupando una Raspberry Pi para la simulación del servidor en donde se almacenará la base de datos y ejecutará el código correspondiente al envío de información de los sensores. Luego de que las pruebas con las sabanillas terminaron, se contactó con la Universidad de Concepción para habilitar un servidor en Linux e introducir todas las dependencias necesarias para que pueda funcionar sin ningún problema. Se habilitó un sub-servidor con unas credenciales que fueron otorgadas. Se verificó que la conexión del ESP32 al servidor fuera correcta mediante la consola de Arduino IDE [14] y que la información esté llegando a la base de datos en PostgreSQL mediante el software PgAdmin4 [15].

Con el servidor habilitado, se escribió un código en un archivo *.bash* el cual verifica que el código que envía información de los sensores se esté ejecutando continuamente en el caso de una posible falla. Esto se realizó mediante el uso de *crontab* [16], el cual es un programador de tareas, cuya función en el servidor fue ejecutar cada 1 minuto el archivo *.bash* creado para que verifique si el código se está ejecutando correctamente. Si el código no se está ejecutando de manera correcta, el proceso se termina y lo comienza de nuevo, asegurando así, que no hayan pérdidas de información mayores a 1 minuto.

```
#!/bin/sh
SERVICE='esp32_code.py'

if [ $(ps ax | grep -v grep | grep $SERVICE | wc -l) -eq "4" ] > /dev/null
then
    echo "$SERVICE service is running, everything is fine" > /dev/null
else
    echo "$SERVICE is not running. Starting."
    sudo kill -9 $(pidof python2 /root/server/$SERVICE)
    python2 /root/server/$SERVICE &
fi
```

Figura 8: Archivo *.bash* que se ejecuta cada 1 minuto para verificar que el código de envío de información de sensores se esté ejecutando correctamente.

3.7. Configurar conexión WiFi en tarjeta ESP32 desde dispositivo móvil

En pruebas anteriores, la conexión a WiFi del ESP32 se realizaba ingresando las credenciales directamente al código en Arduino implementado en la tarjeta, haciendo que la única manera de cambiar esas credenciales fuera ingresando directamente a la tarjeta ESP32 y editar su código, lo cual no es óptimo para los cuidadores/equipo clínico en la prueba piloto. Para facilitar esto, se implementó un código en donde el ESP32 genera un punto de acceso (PA) WiFi en donde, una vez conectado, genera un servidor web (http://192.168.4.1) si es que no encuentra credenciales guardadas en la memoria. El servidor web se implementó con HTML en donde se integraron una caja de selecciones, la cual entrega las señales de WiFi más cercanas, y en la otra caja se debe ingresar la contraseña correspondiente.

Una vez conectado, empieza el envío de datos al servidor PostgreSQL de la Universidad de Concepción.

(a) (b)

Figura 9: Servidor web de ingreso de credenciales WiFi.

(a): Ingreso de crenciales WiFi, (b): Mensaje de credenciales ingresadas correctamente.

3.8. Discusión

La conexión a la Raspberry Pi resultó ser exitosa en las primeras pruebas, también para replicar los mismos pasos (instalación de dependencias, uso de comandos en Linux, etc.) cuando ya se habilitó el servidor de la Universidad de Concepción, acelerando el proceso de conexión considerablemente. Se modificaron los algoritmos integrados al microcontrolador ESP32 para que los datos sean directamente enviados al servidor y se creó una interfaz para que los datos sean fácilmente visualizados y gestionados, con una vista amigable para el cuidador del paciente y/o el equipo de salud correspondiente al paciente en observación.

Las pruebas que se realizaron con la sabanilla hicieron uso de una camilla para habilitar una correcta posición para simular un paciente acostado en su cama, se recibieron datos correctos y se verificó si existían sensores de presión FSR averiados para su reemplazo. El diagrama del sistema de monitoreo implementado en un paciente, con todos los componentes en funcionamiento se observa en la Fig. 10.

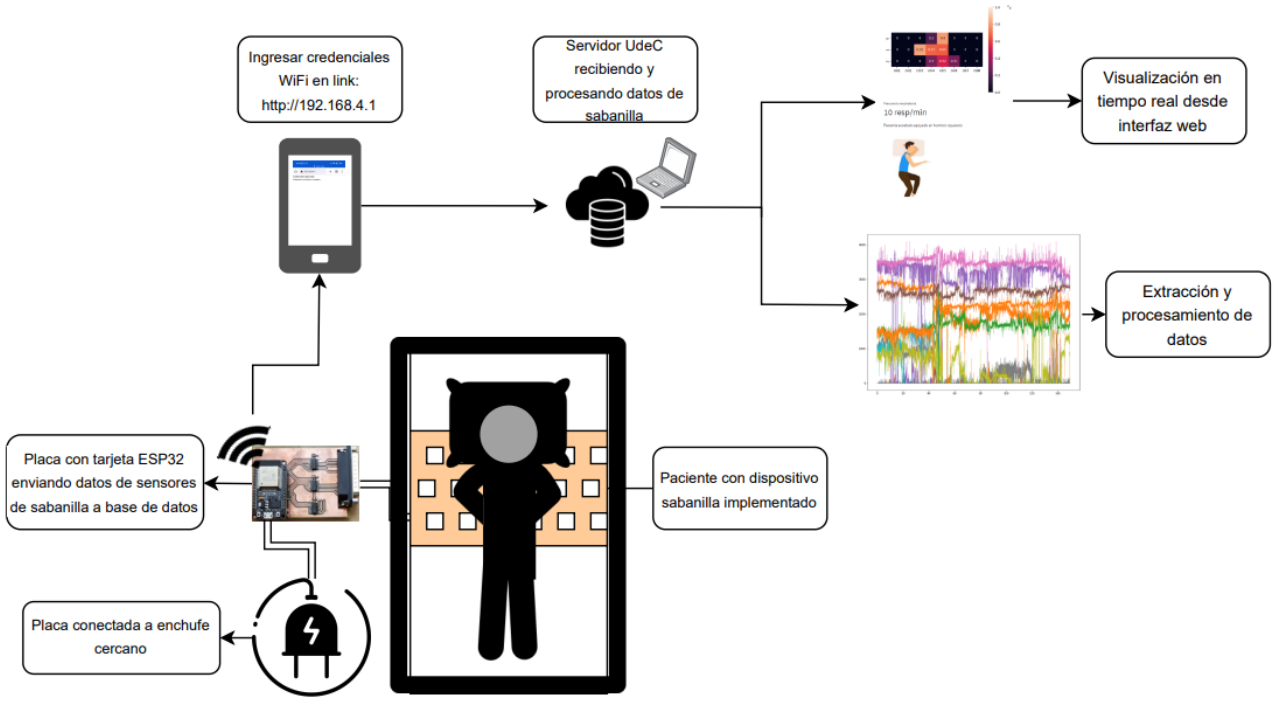


Figura 10: Diagrama sistema de monitoreo continuo.

Capítulo 4. Plataforma Web para visualización de datos

4.1. Introducción

La utilización de una interfaz web es de gran utilidad para realizar un seguimiento y una gestión eficaz a la hora de una instalación o cambio de una sabanilla, al igual para la visualización en tiempo real del estado del paciente.

4.2. Implementación

Utilizando el lenguaje de programación Python mediante Spyder de Anaconda ® [17], se creó una interfaz destinada al seguimiento y la visualización en tiempo real de pacientes que estén asignados a una sabanilla para su monitoreo contínuo. La librería que se utilizó es Streamlit [18], la cual con las dependencias correspondientes, puede ser utilizada para la visualización de pacientes, la gestión de sabanillas y comunicación rápida con la base de datos en PostgreSQL. El diseño de la página web se modificó considerando los widgets disponibles de Streamlit [19] y la forma en la que se ejecuta la interfaz.

Se definieron funciones específicas para cada servicio que se desea usar, por ejemplo, cuando se ingresa por primera vez a la interfaz (o se recarga la página), se llama automáticamente a una función que cumple el servicio de inicio de sesión (Fig. 11). Una vez iniciada la sesión, dependiendo de que interfaz se desea visualizar, se llama a la función que cumplirá ese servicio y se mantendrá en esa a menos que otra interfaz sea seleccionada o se cierre la sesión.

Todas las funciones encontradas en la plataforma son controladas mediante una función maestra llamada *main()*, la cual contiene los widgets correspondientes a una caja de selecciones (*select box*), en donde proyecta los interfaces disponibles en la página web.

Al comienzo del código correspondiente a la interfaz se realiza la conexión a la base de datos en PostgreSQL, alojada en el servidor de la UdeC, debido a que la mayoría de las funciones correspondientes a los sub-interfaces requieren está conexión para realizar los servicios requeridos. Esto ingresando la dirección IP especificada correspondiente al servidor. Si la conexión a la base de datos es exitosa, el código no arrojará ningún error, de lo contrario no será posible el ingreso a la interfaz.

La manera en que Streamlit funciona es que cada vez que se realiza una acción (presionar un botón, ingresar a un sub-interfaz o ingresar algún input), el código se vuelve a ejecutar desde el

comienzo, por lo que fue necesario que la conexión de la base de datos esté fuera de las funciones, es decir, en el código base. También fue necesario la utilización de variables de estado de la sesión (*session_state()*) [20], las cuales son útiles para que la página en donde se visualiza la interfaz tenga algún tipo de memoria, en donde las variables ingresadas en la sesión se mantienen en un estado persistente, que cada vez que se realice una acción o se recargue la página dentro de la misma sesión, mantendrán su valor. Gracias a esto, cada vez que se recarga la página mientras el usuario navega por los sub-interfaces, el código recordará la variable específica de la sesión, haciendo que la sub-interfaz seleccionado no se reinicie. Lo mismo es aplicado cuando se inicia la sesión, debido a que el código tiene programado para que cada vez que se ingrese a la interfaz, o se recargue la página manualmente, se tengan que ingresar las credenciales nuevamente (debido a las variables de estado de la sesión), añadiendo cierto nivel de seguridad.

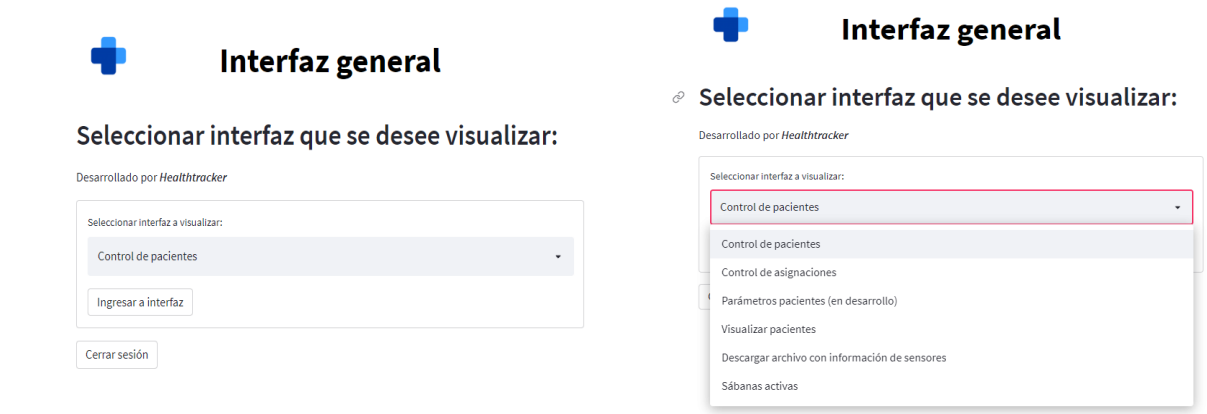
Por defecto, el diseño de Streamlit viene en un tono oscuro, por lo que se debe crear una carpeta llamada *.streamlit* en el servidor en una ruta específica con los parámetros de diseño de la página web. Por estética de las imágenes utilizadas, se escogió un color blanco con letras negras como se observa en las siguientes figuras.

La interfaz puede ser ingresado y utilizado a través de las plataformas de celular (sea Android o iPhone) o de una computadora, debido a que se trata de una página web.

Figura 11: Interfaz de inicio de sesión.

4.3. Plataforma web

Una vez ingresadas las credenciales correctas en la interfaz de inicio de sesión, la función *login()* (inicio de sesión) llama a la función *main()*, la cual es responsable de mostrar los sub-interfaces disponibles en la página web (Fig. 12). Para que se llame por defecto la función de inicio de sesión, se utiliza un condicional en Python, el cual ocupa la variable *__name__* que por defecto tiene como valor "*__main__*" cuando el código es ejecutado, por lo tanto, si *__name__* es igual a "*__main__*" (siempre), que ejecute la función *login()*.



(a) (b)

Figura 12: Interfaz principal.

(a): Vista al ingresar a interfaz web, (b): Opciones de sub-interfaces en la interfaz web.

Las funciones de los sub-interfaces son las siguientes:

1. Control de pacientes:

Para la implementación de esta sub-interfaz, se conectó con la base de datos habilitada en PostgreSQL debido a que actúa directamente sobre ella, actualizando tablas mediante el uso de comandos para añadir, eliminar o visualizar filas.

La función de esta sub-interfaz es añadir a la base de datos en PostgreSQL los pacientes que estén con un dispositivo de sabanilla implementado en su cama, para un posterior seguimiento en sus asignaciones correspondientes. Los campos que se piden son; ID del paciente, nombre del paciente, domicilio, fecha de nacimiento y fecha de derivación. También es posible eliminar un paciente debido a un mal ingreso o si se terminó el ciclo de investigación con el dispositivo del

paciente. Todo lo anterior se realiza mediante comandos de ingreso, actualización y eliminación de filas o columnas en PostgreSQL.

La ídea es que apenas se tramite el uso del dispositivo con algún paciente, este sea ingresado al sistema para actualizar la base de datos y empezar el monitoreo continuo con el uso de los demás sub-interfaces.

Figura 13: Plataforma de ingreso de pacientes (ingresar paciente).

2. Control de asignaciones:

Esta interfaz permite un seguimiento de las asignaciones que se han hecho de cada sabanilla a cada paciente ingresado a la base de datos, por ejemplo si se realiza un cambio de sabanilla debido a un mal funcionamiento de un dispositivo o componente, se cataloga como no vigente y se finaliza la sesión del paciente con esa sabanilla, asignando una fecha de término al monitoreo. Los campos que se ingresan son únicamente el ID del paciente y el ID de la sábana a asignar y la tabla creada posee dichos campos y automáticamente se añade la fecha de inicio de monitoreo, la fecha de término en el caso de una desasignación y la dirección MAC asignada a la determinada sabanilla.

La función de esta sub-interfaz es tener un historial de uso de la sabanilla de modo que si se desea extraer información desde la base de datos de un dispositivo, se sepa, de acuerdo a la fecha, con que paciente estaba asignada una determinada sabanilla y si está actualmente cursando un monitoreo continuo o si bien, se le asignó una fecha de término, lo cual permitiría consultar información hasta dicha fecha.

Figura 14: Plataforma de asignación de pacientes (con datos de prueba).

3. Parámetros pacientes (en desarrollo):

Esta sub-interfaz está actualmente en desarrollo debido a que no se han considerado parámetros de alerta hasta el momento, pero para un trabajo futuro, esta sub-interfaz tendrá la función de ingresar a la base de datos unos parámetros correspondientes a un paciente, debido a que no todos van a tener las mismas alertas en el caso de que se midan variables fisiológicas tales como la frecuencia respiratoria o el tiempo en la misma posición.

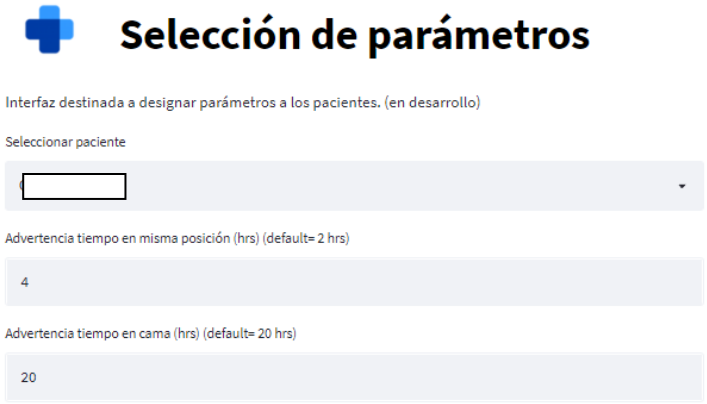


Figura 15: Plataforma de selección de parámetros (con datos de prueba, en desarrollo).

4. Visualizar pacientes:

La función de esta interfaz es la monitorización en tiempo real de los pacientes que estén acutalmente asignados a un dispositivo. Los pacientes asignados son mostrados en una lista en conjunto con la información necesaria para identificar a cada individuo. Hay un botón de selección con nombre “Visualizar paciente en tiempo real”, con el cual es posible ver en tiempo real la dstribución de sensores activos (que esté presionando actualmente el paciente), y los que están inactivos. En conjunto con esta información se muestra la estimación de la posición actual del paciente, ya sea prono, supino, o de lado.

La visualización en tiempo real se implementó utilizando la librería de Python *seaborn* [21], la cual contiene diversos gráficos con los cuales es posible observar datos enviados a través de la librería. En específico se utilizó el mapa de calor o *heatmap*, con el cual se definieron las dimensiones para que coincidieran con los sensores FSR de la sabanilla. Una vez programado el *heatmap*, se observa la activación de los sensores como se muestra en la Fig. 16.

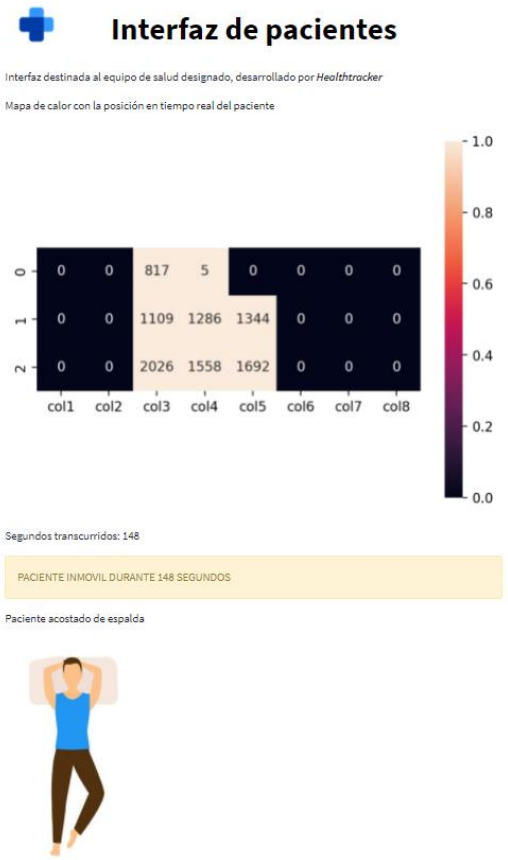
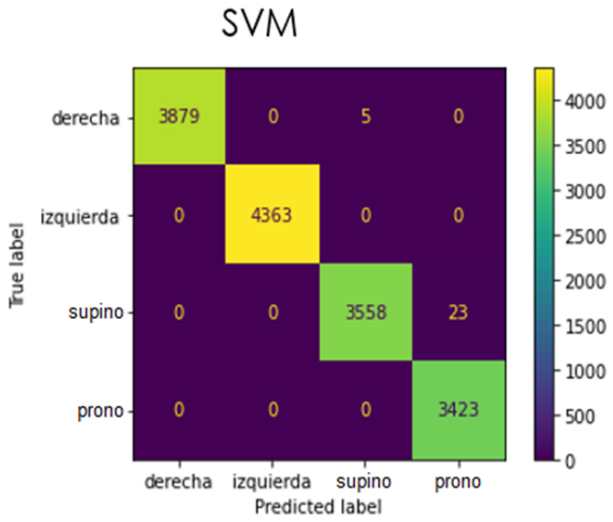


Figura 16: Información de sensores desplegada en tiempo real en conjunto con la predicción de posición en la cama (simulación).

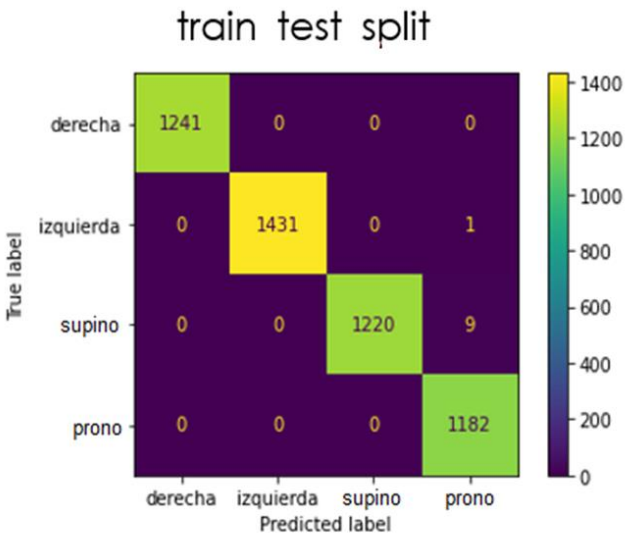
Además de visualizar la activación de los sensores, se entrenó un clasificador para estimar la posición del paciente utilizando una librería de *machine learning* de Python llamada *sklearn* o *scikit-learn* [22]

Para la estimación de la posición actual del paciente en tiempo real, se realizó un entrenamiento de un clasificador llamado *Máquina de Vectores de Soporte* o SVM por sus siglas en inglés, que en resumen es un algoritmo de aprendizaje automático de *Machine Learning* que busca la mejor recta, o hiperplano, que separe las clases de un conjunto de muestras [23].

Para el uso de este clasificador, se realizaron grabaciones en el dispositivo sabanilla de distintas posiciones (prono, supino, de lado), para luego, mediante el uso de la interfaz, específicamente la sub-interfaz de descarga de archivo con información de sensores, se etiquetaron los archivos correspondientes a la posición grabada para luego aplicarle, mediante un código en lenguaje Python (Spyder), el clasificador SVM.



(a)



(b)

Figura 17: Matrices de confusión.

(a): Clasificador SVM, (b): Entrenamiento con 1/3 de los datos.

Como se puede observar en la Fig. 8, se calcularon las matrices de confusión del clasificador en conjunto con un entrenamiento del clasificador utilizando 1/3 de los datos totales, se puede observar un resultado satisfactorio dado a que un mínimo de los datos fue predicho erróneamente. La manera de observar los datos predichos acertados, son los que están en la diagonal de la matriz, que, en este caso, son los números más grandes. Los datos erróneos están fuera de la diagonal de la matriz, tomando por ejemplo la matriz de entrenamiento de datos (train_test_split) en la Fig. 18, los

datos que están encerrados en rojo son los datos que fueron erróneamente predichos (minoría), debido a que el verdadero valor (True label), en el caso de supino, 9 de esos datos fueron predichos como la posición prona (Predicted label), pero es un valor insignificante comparado a los 1220 datos de supino correctamente predichos.

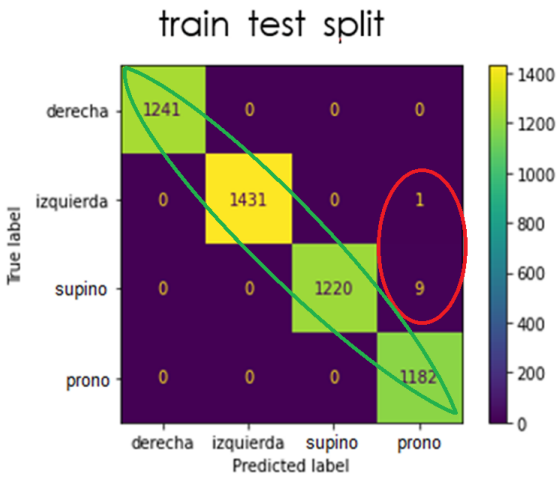


Figura 18: Matriz de confusión de entrenamiento.

1/3 de datos; verde: datos predichos correctamente, rojo: datos predichos erróneamente.

Se realizaron pruebas de laboratorio con el clasificador integrado en la interfaz web y su funcionamente fue satisfactorio en relación a la predicción de posiciones en tiempo real, como se puede ver en la Fig. 16.

Además de la predicción de posiciones, se modificó un algoritmo en Python de cálculo de la frecuencia respiratoria utilizando las señales recibidas de los dispositivos para que sea implementado en tiempo real. Esto mediante guardando el dato correspondiente a la frecuencia respiratoria en la base de datos de PostgreSQL, generando así un histórico de información con el cual se realiza la consulta del último dato para mostrarlo en la sub-interfaz de visualización de pacientes (Fig 18).

El cálculo de la frecuencia respiratoria se realiza utilizando la función de la librería de Python *scipy* llamada *signal*, la cual tiene un comando llamado *find_peaks()*, el cual como dice el nombre, encuentra los peaks (puntos más altos), dentro de una ventana de tiempo, que en este caso es de 1 minuto. La cantidad de peaks dentro de esa ventana nos permite poder calcular la frecuencia respiratoria e ir actualizando esa información cada 1 minuto en la base de datos en PostgreSQL para poder obtener un histórico de información.

Figura 19: Frecuencia respiratoria visualizada en sub-interfaz de visualización de pacientes

5. Descargar archivo con información de sensores:

Esta sub-interfaz permite seleccionar un paciente para posteriormente obtener la información en una fecha determinada de los sensores de la sabanilla, con el objetivo de realizar, si es necesario u óptimo, un procesamiento a la señal formada para fines de la investigación. La forma en la que funciona es mediante consultas al servidor PostgreSQL con una fecha específica, la cual se va

actualizando a través de la interfaz dependiendo de la selección del usuario, como se muestra en la Fig. 20.

Figura 20: Sub-interfaz de descarga de archivo con datos de sensores (de acuerdo a rango de fechas).

Esta interfaz tiene una función de procesamiento de información ingresada a la base de datos, es posible descargar información basado en una fecha y hora específica o en las últimas horas específicas. El archivo descargado es un archivo *.csv* conteniendo información de los sensores en el rango de tiempo seleccionado anteriormente.

Figura 21: Sub-interfaz de descarga de archivo con datos de sensores (de acuerdo las últimas horas).

La función de la descarga en las últimas horas es para obtener la información más rápida en comparación a seleccionar la fecha y hora de la consulta.

El archivo resultante en formato *.csv* al momento de la descarga se visualiza como se observa en la Fig. 22.

FECHA	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16	s17	s18	s19	s20	s21	s22	s23	s24
2022-07-22 1	0	0	405	0	0	1465	1106	0	0	332	2910	0	822	602	0	0	1117	0	3679	0	0	0	2038	0
2022-07-22 1	0	0	411	0	0	1467	1104	0	0	337	2922	0	818	609	0	0	1113	0	3472	0	0	0	2041	0
2022-07-22 1	0	0	407	0	0	1463	1092	0	0	335	2933	0	831	603	0	0	1095	0	3274	0	0	0	2041	0
2022-07-22 1	0	0	405	0	0	1467	1085	0	0	381	2950	0	829	595	0	0	1093	0	3277	0	0	0	2047	0
2022-07-22 1	0	0	402	0	0	1462	1081	0	0	382	2960	0	830	601	0	0	1083	0	3426	0	0	0	2038	0
2022-07-22 1	0	0	384	0	0	1461	1065	0	0	389	2960	0	833	592	0	0	1088	0	3408	0	0	0	2042	0
2022-07-22 1	0	0	389	0	0	1463	1062	0	0	384	2966	0	837	587	0	0	1073	0	3282	0	0	0	2055	0
2022-07-22 1	0	0	397	0	0	1470	1072	0	0	389	2989	0	832	592	0	0	1077	0	3653	0	0	0	2047	0
2022-07-22 1	0	0	379	0	0	1469	1067	0	0	401	2970	0	839	585	0	0	1072	0	3573	0	0	0	2047	0
2022-07-22 1	0	0	372	0	0	1463	1067	0	0	395	2990	0	855	587	0	0	1070	0	3440	0	0	0	2044	0
2022-07-22 1	0	0	369	0	0	1463	1066	0	0	391	2974	0	855	589	0	0	1073	0	3050	0	0	0	2053	0
2022-07-22 1	0	0	363	0	0	1469	1067	0	0	400	2963	0	867	587	0	0	1067	0	3062	0	0	0	2043	0
2022-07-22 1	0	0	368	0	0	1472	1072	0	0	402	2891	0	863	589	0	0	1054	0	3094	0	0	0	2062	0
2022-07-22 1	0	0	367	0	0	1471	1074	0	0	400	2736	0	867	582	0	0	1051	0	3202	0	0	0	2042	0
2022-07-22 1	0	0	364	0	0	1463	1070	0	0	390	2800	0	869	587	0	0	1047	0	2957	0	0	0	2054	0
2022-07-22 1	0	0	363	0	0	1471	1066	0	0	394	2810	0	859	577	0	0	1047	0	3717	0	0	0	2043	0
2022-07-22 1	0	0	358	0	0	1471	1059	0	0	390	2784	0	874	589	0	0	1052	0	3602	0	0	0	2056	0
2022-07-22 1	0	0	363	0	0	1471	1055	0	0	400	2683	0	867	595	0	0	1048	0	3451	0	0	0	2055	0
2022-07-22 1	0	0	368	0	0	1472	1054	0	0	427	2608	0	878	589	0	0	1051	0	3447	0	0	0	2057	0
2022-07-22 1	0	0	362	0	0	1484	1040	0	0	426	2480	0	867	592	0	0	1043	0	3440	0	0	0	2061	0
2022-07-22 1	0	0	367	0	0	1488	1043	0	0	431	2530	0	875	579	0	0	1048	0	3435	0	0	0	2058	0
2022-07-22 1	0	0	358	0	0	1488	1028	0	0	444	2527	0	863	587	0	0	1043	0	3435	0	0	0	2064	0
2022-07-22 1	0	0	352	0	0	1486	1022	0	0	422	2576	0	854	583	0	0	1041	0	3394	0	0	0	2058	0
2022-07-22 1	0	0	364	0	0	1481	1019	0	0	431	2559	0	859	583	0	0	1040	0	3589	0	0	0	2061	0

Figura 22: Archivo *.csv* con datos de sensores.

En el archivo *.csv*, la primera fila indica el número de columnas, siendo la columna 0 la correspondiente a la fecha y las columas 1-24 los respectivos sensores. Los sensores que no están presionados son denotados por un 0, luego, dependiendo de la intensidad, el número varía entre 0 y 4095 (12 bytes).

6. Sábanas activas:

Esta interfaz es importante para ver en el momento de implementación de la sabanilla, verificar si la base de datos en PostgreSQL está recibiendo datos, corroborando así que todos los componentes están funcionando correctamente y la informacion está siendo recibida en el servidor de la Universidad de Conepción.

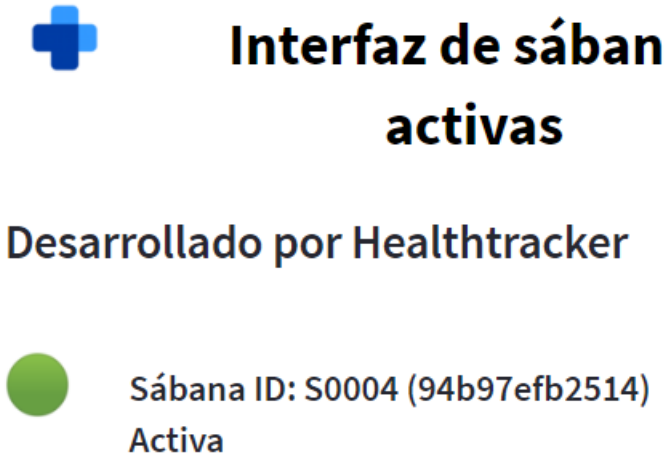


Figura 23: Sub-interfaz de verificación de sabanillas activas.

Si una sabanilla está enviando información correctamente al servidor, se mostrará el ID de la sabanilla, en conjunto con su dirección MAC, corroborando que dicha sabanilla esta activa y enviando datos.

En el caso de que ninguna sabanilla está enviando información el servidor, se mostrará el mensaje mostrado en la Fig. 24.

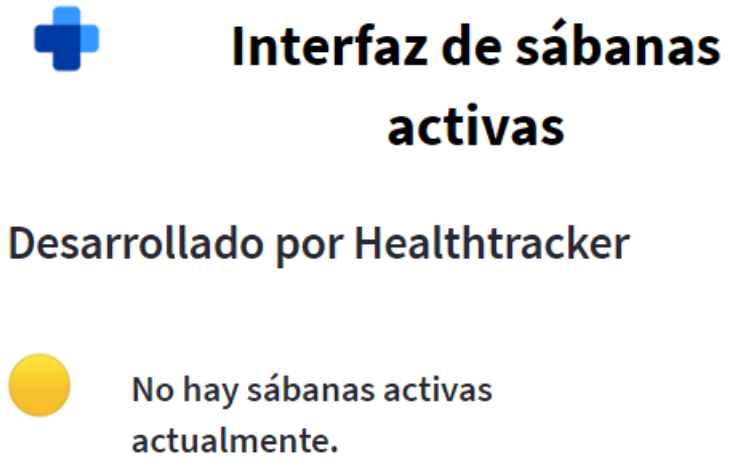


Figura 24: Sub-interfaz de verificación de sabanillas activas (ninguna sabanila enviando datos).

La manera en que detecta que una sabanilla está enviando información, es mediante consultas a la base de datos si es que se ha actualizado la fecha del último dato enviado, esto es debido a que cada dataset de información de sensores tiene una columna llamada *time*, la cual muestra la fecha en que los datos fueron enviados y recibidos. Si se actualiza esa fecha, es decir, si se envío un nuevo dato, es que se considera de que la sabanilla está activa.

4.4. Implementación en servidor

Para un óptimo funcionamiento de la interfaz web y que el código correspondiente a este se esté ejecutando 24/7 se implementó en el servidor otorgado por la UdeC.

Anteriormente, para primeras pruebas de que los sub-interfaces estén funcionando, se ejecutaba el código de la plataforma web vía Streamlit desde un computador personal, el cual no tendría acceso remoto debido a que no estaría abierto el puerto correspondiente del router y además no estaría encendido constantemente, haciendo que el acceso a la plataforma dependa del usuario que ejecute el código.

En el servidor se escribió un archivo *.bash* (ejecutable), el cual revisa si la interfaz se está ejecutando correctamente cada un minuto para que se pueda acceder en cualquier momento, sumado la tramitación para abrir el puerto correspondiente del router de la universidad.

```
#!/bin/sh
SERVICE='Healthtracker.py'

if ps ax | grep -v grep | grep $SERVICE > /dev/null
then
    echo "$SERVICE service running, everything is fine" > /dev/null #sin mensaje
else > /dev/null  #sin mensaje
    echo "$SERVICE is not running. Starting."
    . /home/grojas/ambientes/sabana/bin/activate
    streamlit run /home/grojas/server/server/interfaz_sabana/$SERVICE &> /dev/null & #sin mensaje
fi
```

Figura 25: Archivo *.bash* que revisa si la plataforma web se está ejecutando correctamente.

El archivo *.bash* busca en los procesos activos el código correspondiente a la interfaz, el cual está descrito en la Fig. 25 como '*Healthtracker.py*', si el código no se encuentra en los procesos activos, se vuelve a ejecutar para que esté funcionando correctamente.

Además, se implementó un ambiente virtual con una versión de Python más reciente (Python 3.8), esto hace que todas las librerías de Python que sean instaladas cuando este ambiente esté activo, sean de la misma versión y no sean confundidas con las que ya están instaladas con versiones antiguas. Esto ayuda también a no confundir a los códigos que se están ejecutando actualmente en el servidor, dado a que además del código de la interfaz, también se está ejecutando el código de envío de datos de la sabanilla, y este ocupa una versión más antigua (Python 2), por lo que el uso de un ambiente virtual fue crucial para la implementación de la plataforma web.

4.5. Utilización de la interfaz web en gestión de pacientes

Para la verificación de un correcto funcionamiento de todos los sub-interfaces, se realizaron pruebas de laboratorio en donde se ingresó un paciente, se procedió a asignarlo a una sabanilla, se verificó que esta estuviera enviando información al servidor para luego ver la información de sensores en tiempo real.

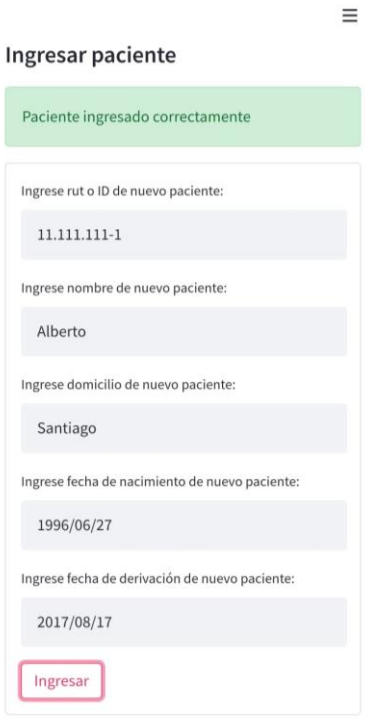


Figura 26: Captura de pantalla de ingreso de paciente ficticio (desde celular Android).

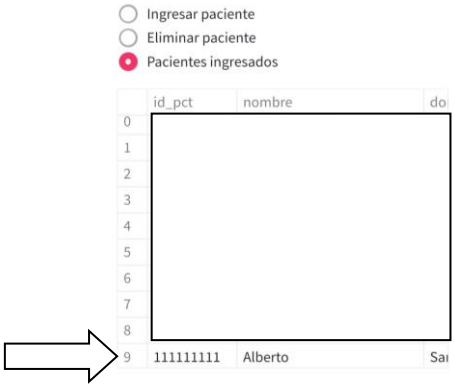


Figura 27: Captura de pantalla verificación de ingreso en base de datos de paciente ficticio (desde celular Android).

Figura 28: Captura de pantalla de asignación de sabanilla a paciente ficticio (desde celular Android).

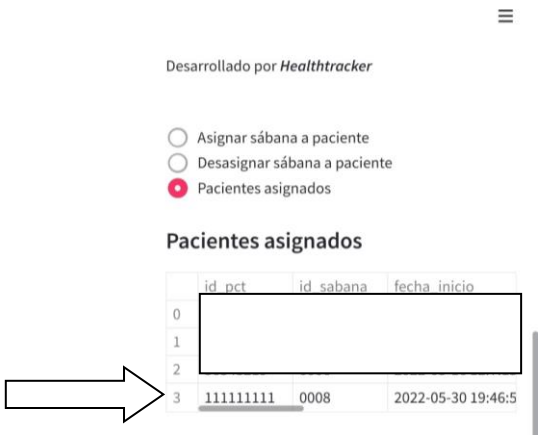


Figura 29: Captura de pantalla verificación de paciente ficticio asignado (desde celular Android).

Luego de verificar que toda sub-interfaz esté funcionando correctamente, se procedió a realizar la prueba piloto en donde se implementó la sabanilla en un paciente con una movilidad muy restringida para su monitoreo continuo con ayuda de la interfaz web diseñada para el proyecto en cuestión.

4.6. Discusión

El uso de la interfaz en pruebas de laboratorio fue exitoso, esto se corroboró mediante la revisión de la base de datos, analizando si se añadían correctamente los pacientes nuevos y si se asignaba correctamente el paciente a una nueva sabanilla. La plataforma web implementada en el servidor no muestra intermitencia de disponibilidad al intentar ingresar, esto es gracias al archivo que revisa constantemente cada un minuto si el código correspondiente a la interfaz se está ejecutando correctamente.

Capítulo 5. Prueba piloto

5.1. Instalación de sabanilla

Se realizó una visita al domicilio del paciente en Santiago, el cual firmó la documentación correspondiente al consentimiento informado del comité de ética científico de **Redsalud**, para la posterior instalación del dispositivo en la cama. Para una instalación correcta, se asignaron ciertos pasos a seguir correspondientes a la preparación para implementar la sabanilla para que inicie su monitoreo continuo con el paciente asignado. Estos pasos a seguir se escribieron en los *checklist* correspondientes a la etapa del proyecto SISCAP - CP.

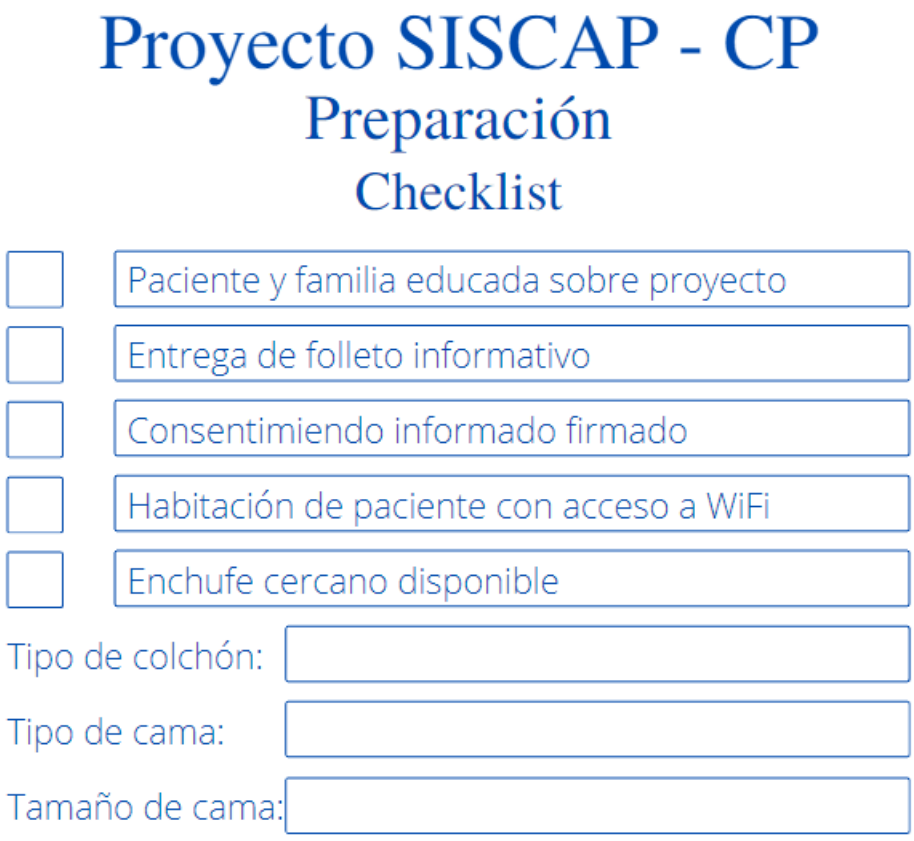


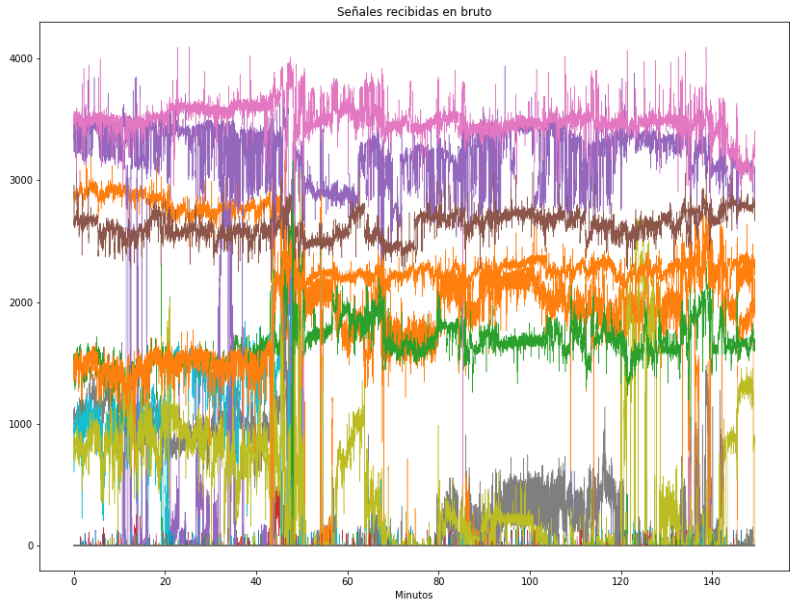
Figura 30: Checklist preparación de proyecto SISCAP – CP.

Figura 31: Checklist instalación sabanilla de proyecto SISCAP – CP.

Luego de verificar que todas las casillas del *checklist* de preparación del proyecto estuvieran marcadas, se procedió a la instalación, en donde se fue marcando cada casilla mientras se completaba la instalación de forma correcta. La verificación de funcionamiento se realizó mediante el uso de la plataforma web anteriormente mencionada.

5.2. Recolección de datos

Luego de verificar el envío de datos al servidor de la Universidad de Concepción, se procedió a esperar un periodo de tiempo para que haya suficientes datos para un posterior procesamiento y análisis de estos. La descarga de datos se hizo a través de la plataforma web en la sub-interfaz de descarga de archivo con datos de sensores, en donde se descarga un archivo *.csv* al computador o celular que se está ocupando para ingresar a la plataforma. Este archivo *.csv* (Fig. 22), contiene la información de sensores en un periodo de tiempo previamente seleccionado (Fig. 20, Fig. 21), la cual, mediante un código en Python se pueden visualizar los sensores.



(a)

(b)

Figura 32: Señales recibidas desde sabanilla instalada.

(a): Señales en bruto recibidas de sensores, (b): Sensores reflejados en la sabanilla (simulado).

En la Fig. 32, se observan las señales correspondientes a los sensores de la sabanilla instalada en la cama del paciente en un tiempo aproximado de 2 horas y 28 minutos. Cada señal de distinto color es un sensor distinto, los cuales se activan dependiendo la posición del paciente en la cama. Por ejemplo, si el paciente se encuentra al medio de la cama, presionaría los sensores que están en la mitad del dispositivo, y los que no están presionados (a los bordes de la sabanilla), sus señales marcan el valor 0, como se puede observar en la Fig. 32. Para un mejor entendimiento del posicionamiento de los sensores de presión en la sabanilla, observar la Fig. 32 (b).

5.3. Análisis de datos

A las señales recibidas de los sensores se les realizó un procesamiento mediante códigos en Python. La primera etapa del procesamiento es la visualización de las señales y poder identificar ciertos patrones en la información recibida, por ejemplo, en la Fig. 33, la cual es la imagen mostrada en la Fig. 32, se puede observar un cambio de posición, debido a que es un salto brusco de saturación de los sensores, los cuales sucede cuando se presionan más fuertes unos y se dejan de presionar otros en un corto periodo de tiempo.

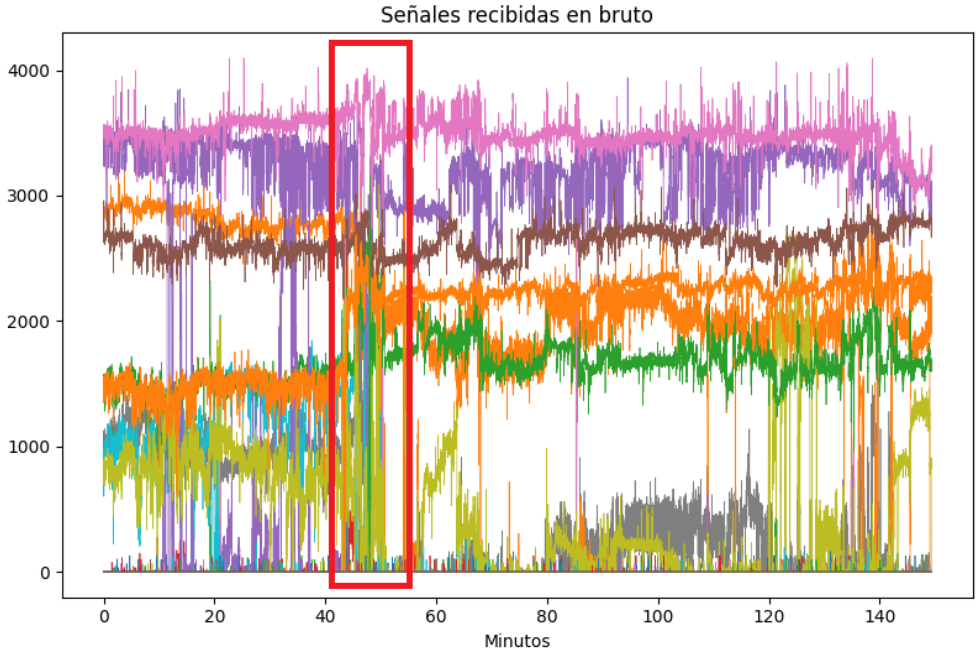


Figura 33: Movimiento corporal detectado en señales de sensores.

También se pueden observar señales correspondientes a la frecuencia respiratoria, debido a que la expansión y contracción del tórax (respiración), provoca un cambio de presión periódica en los sensores que se encuentren en esa zona, por lo que se pueden identificar distintas señales las cuales corresponden a la respiración del paciente. Un algoritmo detecta la mejor calificada señal respiratoria y la utiliza para procesamientos posteriores.

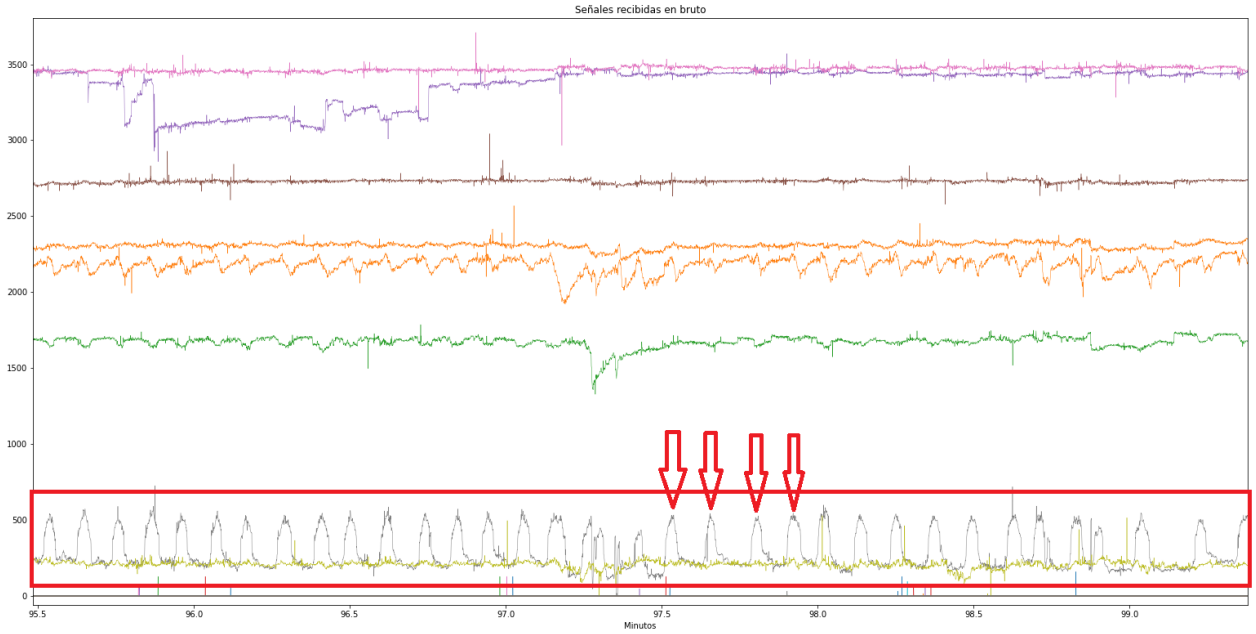


Figura 34: Señal de sensor de presión correspondiente a la respiración.

Luego, mediante consultas a la base de datos se puede obtener un histórico de información de frecuencia respiratoria correspondiente a la obtenida en la prueba piloto con el dispositivo implementado en la cama del paciente.

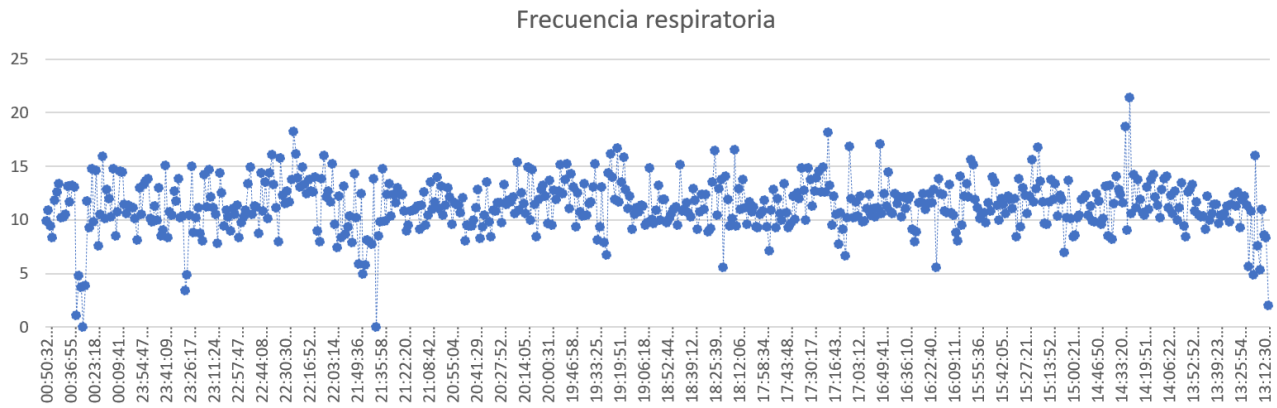


Figura 35: Histórico frecuencia respiratoria (Resp/min).

Como se puede observar en la Fig. 35, el paciente muestra un promedio de frecuencias entre las 10 y 15 respiraciones por minuto (RPM), siendo algunas excepciones posiblemente debido a que el paciente se cambió de posición en la cama o que no se encuentre en ella durante un periodo de tiempo por alguna razón específica. Por otro lado, su promedio se encuentra entre los rangos normales de frecuencia respiratoria. El histórico muestra un tiempo de lectura de 12 horas aproximadamente.

Los tiempos fuera de cama del paciente también se pueden visualizar en las señales recibidas del paciente, se ven como espacio en blanco en donde ningún sensor está presionado o algunos que están activos debido a que se dejó algo encima de la cama, esto se observa en la Fig. 36.

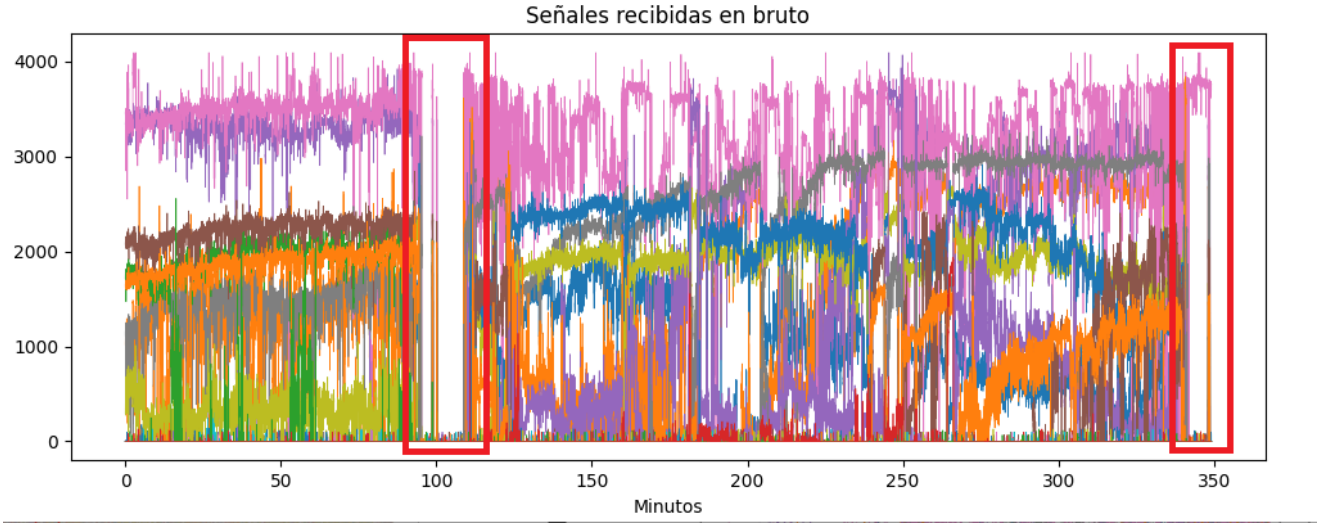


Figura 36: Secciones fuera de cama del paciente.

También hay secciones en donde el paciente no vuelve a la cama en largos periodos de tiempo, esto puede dar una pequeña visión al estado actual del paciente, que no necesariamente está postrado en cama si no que durante el día puede ser capaz (quizá no solo) de realizar actividades y/o moverse alrededor de su hogar, esto se ve reflejado en la Fig. 37.

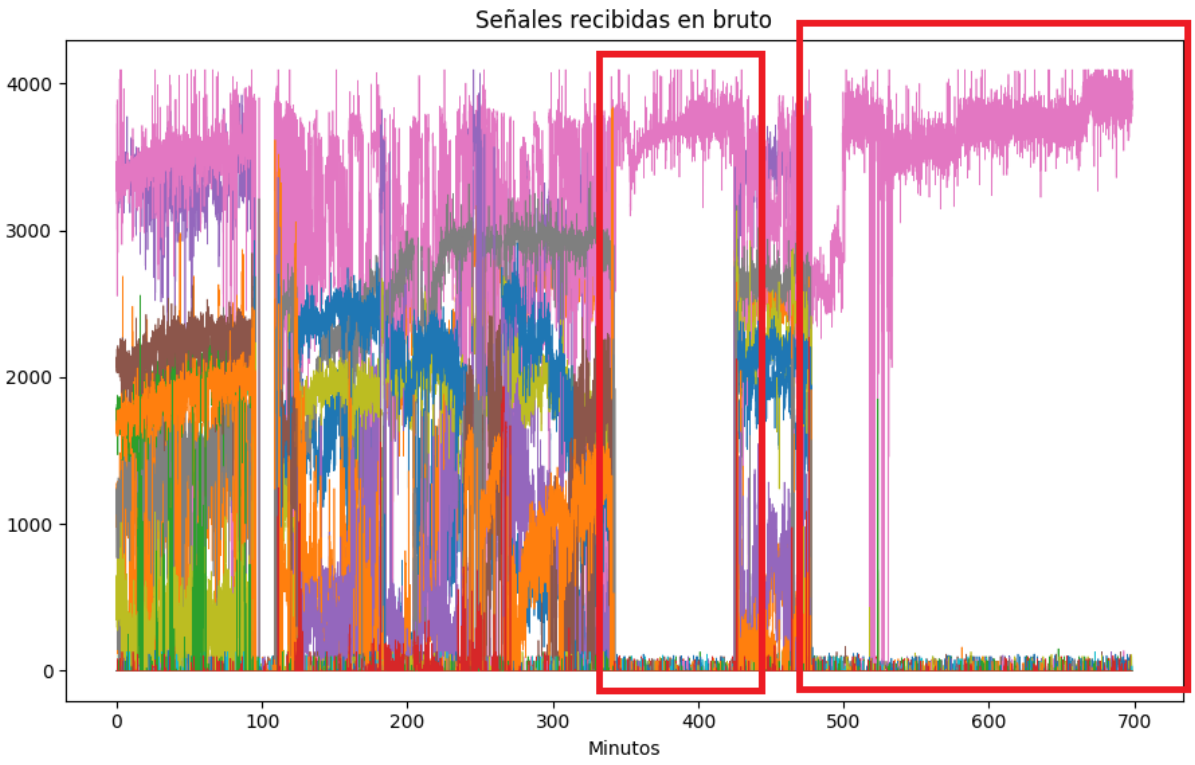
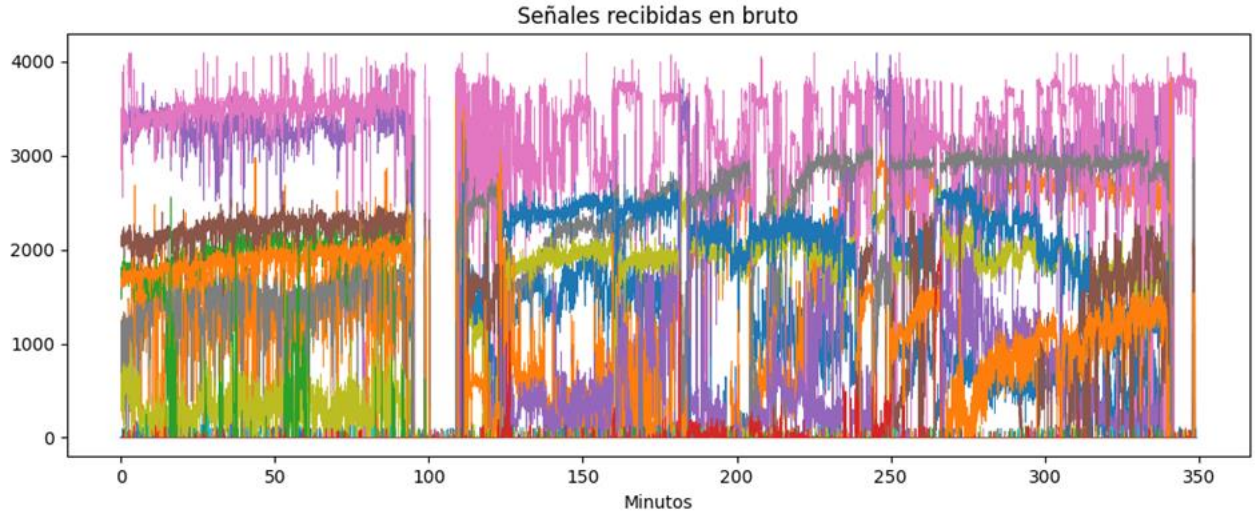


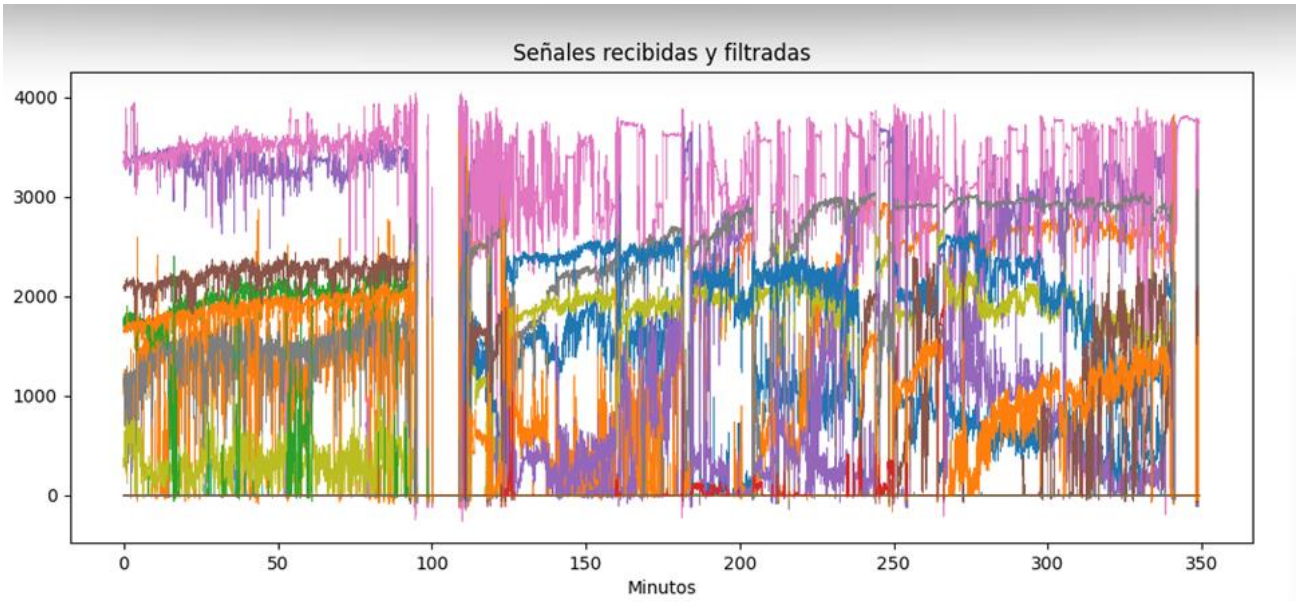
Figura 37: Secciones fuera de cama del paciente (periodos más largos, con un sensor saturado).

Se observa una señal saturada, la cual puede ser un sensor presionado debido a algún objeto encima de la cama del paciente, o debido a algún cruzamiento de cables por un mal posicionamiento de la sábana debido a los movimientos del paciente.

Luego, para una visualización más limpia de las señales, se realizó un filtro pasa-bajo utilizando Python, el cual niega las señales que no entregan información útil y conserva las que están adecuadas para un posterior procesamiento y análisis. Las señales filtradas se observan con menos ruido y más claridad, como se muestra en la Fig. 38.



(a)



(b)

Figura 38: Señales en bruto v/s señales filtradas.

(a): Señales recibidas en bruto de sensores de sabanillas, (b): Señales después de aplicar filtro pasa-bajo (Periodo de muestreo = 5 segundos, Frecuencia de muestreo = 30 Hz, Frecuencia de corte = 0.8 Hz, Frecuencia de Nyquist = 0.5 * Frec. muestreo)

Con la información recibida de las señales, fue posible implementar la plataforma web en donde, en conjunto, se realizan múltiples procesamientos para poder visualizar de manera más sencilla y amigable el estado actual del paciente asignado a la sabanilla.

En la Fig. 19 se observa el estado actual del paciente con todos los procesamientos implementados para entregar una información sencilla y útil para el equipo clínico y de desarrolladores.

5.4. Discusión

La interfaz web mostró ser una buena plataforma de gestión de pacientes a la hora de realizar la prueba piloto, se pudo visualizar al paciente en tiempo real con las señales recibidas de los sensores, al igual que predecir su posición actual y calcular la frecuencia respiratoria en el instante, en intervalos de 1 minuto.

Se demostró que en periodos largos de tiempo es posible realizar un monitoreo continuo mediante el uso de distintos códigos que cumplen funciones específicas para posteriormente visualizar al paciente y su estado actual. La sabanilla no mostró incomodidades en la cama del paciente dado a que los sensores bajo las sábanas comunes de la cama y de la sábana impermeable, son imprescindibles para el paciente asignado.

Idealmente sería ideal seguir añadiendo pacientes a la prueba piloto, contar con 5 como mínimo para examinar a largo plazo un envío de datos más pesado y con más dispositivos conectados, y si la prueba piloto mencionada es un éxito, el paso siguiente sería la fabricación de más sabanillas para realizar un monitoreo continuo a gran escala, con significativamente mayor número de pacientes.

Capítulo 6. Conclusiones generales

6.1. Discusión

En base a los trabajos previos mencionados anteriormente y al avance que se ha realizado en presente memoria, se muestra que la sabanilla ha sido validada y probada en voluntarios, teniendo la capacidad de monitorear de manera continua y no invasiva distintos parámetros que brindan una ayuda al personal de salud para tomar una decisión correcta al momento de realizar un diagnóstico.

El trabajo realizado en el laboratorio demuestra la posibilidad de utilizar los avances de las investigaciones previas e implementarlas para realizar una prueba piloto efectiva en pacientes con cuidados paliativos.

Con la implementación de la interfaz web se abrió la posibilidad de un monitoreo en tiempo real, con una gestión de sabanillas y pacientes sencilla para que el equipo correspondiente al paciente asignado al dispositivo.

6.2. Conclusiones

Con los avances realizados en el laboratorio de Ingeniería Civil Biomédica de la Universidad de Concepción, se podrá habilitar más dispositivos para realizar pruebas futuras en pacientes postrados por tratamiento de cáncer para empezar su monitoreo continuo por largos periodos de tiempo.

Se verificó el envío de información hacia la base de datos para poder ser visualizados de manera remota por el personal de salud asignado al paciente y/o el cuidador encargado, esto será útil para facilitar la comunicación en tiempo real sobre el estado actual del paciente para una toma de decisiones correcta a la hora de realizar un diagnóstico.

Se realizaron pruebas de laboratorio exitosas, incluyendo el uso de la interfaz web para simular pacientes ficticios a los cuales se les asignan sabanillas para luego monitorearlas en tiempo real, con la ayuda de los sub-interfaces integrados en la plataforma. Con la implementación de la interfaz en el servidor de la Universidad de Concepción, se hizo posible el acceso en cualquier momento al enlace entregado por Streamlit, el cual corresponde a la dirección IP del servidor. Se ingresó al enlace mediante un computador y celular, con lo que se corroboró una visualización agradable y sencilla.

Se realizó la prueba piloto con un paciente para verificar un envío de datos y monitorear continuamente a largo plazo la presión detectada por los sensores FSR con procesamientos

posteriores para medir variables fisiológicas, y también y no menos importante, la comodidad del paciente al tener integrada la sabanilla en la cama.

6.3. Trabajo Futuro

La sabanilla cuenta con un diseño enfocado para la medición de parámetros útiles para determinar el estado actual del paciente. La implementación de algún otro sensor tales como de temperatura serían de gran utilidad para la detección de fiebre o de baja temperatura corporal, los cuales son parámetros que ayudarían sin duda la toma de decisión para un diagnóstico correcto. Otro trabajo futuro es buscar opciones más efectivas y/o económicas que los actuales sensores FSR para la detección de presión en la cama, debido a que aún queda espacio para que el dispositivo sea aún más certero para el análisis de datos obtenidos de este.

Con respecto al servidor, la interfaz web y el envío de datos, se podría actualizar a un servidor con mayores recursos para poder manejar la gran cantidad de datos recibidos con mayor rapidez. Al mismo tiempo, también realizar una optimización del código que envía los datos para que únicamente se guarde en la base de datos información que sea relevante y útil para un procesamiento óptimo.

Bibliografía

[1] Subsecretaría de salud pública, "*Alivio del dolor por cáncer avanzado y cuidados paliativos*", Guía clínica AUGE, MINSAL, 2011.

[2] Pedro E. Pérez-Cruz y Francisco Acevedo C. "*Escalas de estado funcional (o performance status) en cáncer*", Clasificaciones en Gastroenterología, Gastroenterol, latinoam 2014; Vol 25, N°3: 219 – 226.

[3] E. J. Pino, A. Dörner De la Paz, P. Aqueveque, J. A. P. Chávez and A. A. Morán, "Contact pressure monitoring device for sleep studies," *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2013, pp. 4160-4163, doi: 10.1109/EMBC.2013.6610461.

[4] Instituto Nacional del Cáncer de EE. UU, "*Cuidados paliativos durante el cáncer*", 20 de Octubre de 2017, [En línea], disponible en: https://www.cancer.gov/espanol/cancer/cancer-avanzado/opciones-de-cuidado/hoja-informativa-cuidados-paliativos

[5] Centros para el Control y la Prevención de Enfermedades (CDC), "*Efectos secundarios del tratamiento contra el cáncer*", Los sobrevivientes de cáncer, 12 de mayo del 2021, [En línea], disponible en: https://www.cdc.gov/spanish/cancer/survivors/patients/side-effects-of-treatment.htm

[6] Clínica Somno, "*Polisomnografía*", 2 de septiembre del 2021, [En línea], disponible en: http://www.somno.cl/polisomnografia/?utm_term=&utm_campaign=Publicidad+polisomnografia+Regiones&utm_source=adwords&utm_medium=ppc&hsa_acc=5906872826&hsa_cam=1533133803&hsa_grp=79082158003&hsa_ad=410367465518&hsa_src=g&hsa_tgt=dsa-858102696658&hsa_kw=&hsa_mt=&hsa_net=adwords&hsa_ver=3&gclid=Cj0KCQiAzfuNBhCGARIsAD1nu-9EajIQlNlfKCav_0x1ImKF4_6gZebyzVqsKjBehUNqy48qKEAXW50aAi_BEALw_wcB

[7] The University of Tennessee Medical Center, "*Estudio del sueño en casa*", 2011 – 2017, [En línea], disponible en: https://www.utmedicalcenter.org/es/medical-care/medical-services/procedures-treatments/home-sleep-study/

[8] Astrid Ivonne Dörner De La Paz, "*Propuesta de Evaluación Objetiva para análisis de Calidad de Sueño*", Tesis de Magister en Ciencias de la Ingeniería con mención en Ingeniería

Eléctrica, Diciembre 2013, Departamento de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de Concepción.

[9] Alejandra Andrea Morán Perrin, "*Validación de Malla de Sensores de Presión para Estudios del Sueño*", Memoria de Título, Ingeniero Civil Biomédico, Diciembre 2014, Departamento de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de Concepción.

[10] E. J. Pino, A. D. De la Paz and P. Aqueveque, "*Noninvasive Monitoring Device to Evaluate Sleep Quality at Mining Facilities*," in *IEEE Transactions on Industry Applications*, vol. 51, no. 1, pp. 101 – 108, Jan. – Feb. 2015, DOI: 10.1109/TIA.2014.2334740.

[11] E. J. Pino, A. A. Morán, A. D. De la Paz and P. Aqueveque, "*Validation of non-invasive monitoring device to evaluate sleep quality*," *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2015, pp. 7974 – 7977, DOI: 10.1109/EMBC.2015.7320242.

[12] Camila Francesca Oliva Cortés, "*Sistema No Invasivo para Apoyo en Cuidados Domiciliarios de Pacientes Postrados*", Memoria de Título, Ingeniero Civil Biomédico, Agosto 2019, Departamento de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de Concepción.

[13] Jorge Joaquín Tardones Ortiz, "*Plataforma de reporte de datos de dispositivo de estudio de sueño*", Memoria de Título, Ingeniero Civil Biomédico, Enero 2021, Departamento de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de Concepción.

[14] Arduino IDE, Arduino IDE 1.8.19, 2022 Arduino, [En línea], disponible en: https://www.arduino.cc/en/software

[15] pgAdmin, "*PostgreSQL Tools*", 2022 PostgreSQL Community Association of Canada, [En línea], disponible en: https://www.pgadmin.org/

[16] Crontab (5), "*Linux manual page*", HTML rendering created 2021-08-27 by Michael Kerrisk, [En línea], disponible en: https://man7.org/linux/man-pages/man5/crontab.5.html

[17] Anaconda, "*Data Science Technology*", 2022 Anaconda Inc., [En línea], disponible en: https://www.anaconda.com/

[18] Streamlit, "*A faster way to build and share data apps*", 2022 Streamlit Inc., [En línea], disponible en: https://streamlit.io/

[19] Streamlit documentation, "*API reference*", 2022 Streamlit Inc., [En línea], disponible en:

https://docs.streamlit.io/library/api-reference

[20] Streamlit documentation "*Session state*", 2022 Streamlit Inc., [En línea], disponible en: https://docs.streamlit.io/library/api-reference/session-state

[21] Seaborn "*statistical data visualization*", © Copyright 2012 – 2021, Michael Waskom, Created using Sphinx 3.3.1, [En línea], disponible en: https://seaborn.pydata.org/

[22] Scikit-learn, "*Machine Learning in Python*", 2007 – 2022, scikit-learn developers (BSD License), [En línea], disponible en: https://scikit-learn.org/stable/

[23] Joaquín Amat Rodrigo, Máquinas de Vector Soporte "*Support Vector Machines, SVMs*", Abril 2017, Attribution 4.0 International (CC BY 4.0), [En línea], disponible en: https://www.cienciadedatos.net/documentos/34_maquinas_de_vector_soporte_support_vector_machines

Anexos

Anexo 1: Presentación para instalación de sabanilla

Para la implementación de la sabanilla en la cama de un paciente, se deben seguir ciertos pasos para corroborar una instalación correcta. Por ello, se elaboró una presentación para el equipo de salud asignado enumerando estos pasos para que sea más sencilla y eficaz la instalación.

Figura 39: Presentación instalación sabanilla.

Anexo 2: Instructivo para implementación interfaz en el servidor

Para una posterior instalación correcta de la plataforma web en algún servidor, se escribió un instructivo con comandos necesarios para descargar todas las dependencias e implementar los ambientes virtuales requeridos para una correcta ejecución del código correspondiente a la interfaz web que se utilizó para la prueba piloto en pacientes.

Instalación interfaz dispositivo sabanilla en servidor Linux

1. Instalación de Python 3.8 o mayor

Primero, se deben instalar los prerrequisitos, los cuales son las librerías de desarrollo para compilar el código de fuente de Python3.8.
Esto se hace ingresando a los comandos a la terminal:

```
$ sudo apt-get install build-essential checkinstall
$ sudo apt-get install libreadline-gplv2-dev libncursesw5-dev libssl-dev libsqlite3-dev
tk-dev libgdbm-dev libc6-dev libbz2-dev libffi-dev zlib1g-dev
```

Luego, se descarga Python 3.8, se recomienda crear un directorio en donde almacenar los archivos, aquí se escoge el nombre *opt*:

```
$ mkdir opt
$ cd /opt
$ sudo wget https://www.python.org/ftp/python/3.8.12/Python-3.8.12.tgz
```

Luego se extrae el archivo:

```
$ sudo tar xzf Python-3.8.12.tgz
```

Finalmente, correr los siguientes comandos:

```
$ cd Python-3.8.12
$ sudo ./configure --enable-optimizations
$ sudo make altinstall
```

Corroborar versión de Python:

```
$ python3.8 --version

Output: Python 3.8.12
```

2. Creación de ambiente virtual

Para la creación de un ambiente virtual es necesario el programa *virtualenv*, el cual se instala mediante el siguiente comando en la terminal:

```
$ pip3.8 install virtualenv
```

Luego, se recomienda crear una carpeta para almacenar los ambientes que se desean crear, en este caso la carpeta se llamará *ambientes*:

```
$ mkdir ambientes
```

```
$ cd /ambientes
```

En la carpeta *ambientes* se debe correr el siguiente comando de *virtualenv*:

```
$ virtualenv <nombre_ambiente>
```

En donde <nombre_ambiente> es el nombre del ambiente que se desea crear, el cual le asigna el usuario, para este caso se usará el nombre *sabana*.
Cuando se haya creado el ambiente, habrá un directorio en la carpeta *ambientes* con el nombre del ambiente que se le asigno, se puede corroborar con el siguiente comando:

```
$ ls
```

```
Output: sabana
```

Luego, para activar el ambiente, se debe ingresar el siguiente comando:

```
$ source /ambientes/sabana/bin/activate
```

Y en la terminal aparecerá el nombre del ambiente de la siguiente forma:

```
(sabana) root@:/home/user/ambientes#
```

Lo cual corrobora que el ambiente ha sido activado y está listo para la instalación de dependencias.
Se puede desactivar el ambiente virtual mediante el siguiente comando:

```
$ deactivate
```

Nota: La creación de un ambiente virtual es para evitar confusiones con librerías anexas a este ambiente, así todo está en la versión correspondiente, incluyendo Python.

3. Instalación cron

Para la verificación de que el código correspondiente a la interfaz de la sabanilla se esté ejecutando, es necesario el administrador de procesos *cron*, el cual ejecuta procesos o guiones a intervalos regulares (cada minuto, hora, día, etc).
Para su instalación, se debe ingresar el siguiente comando:

```
$ sudo apt-get update
```

```
$ sudo apt-get install cron
```

Luego de instalarlo, se empieza el servicio cron mediante el comando:

```
$ sudo /etc/init.d/cron start
```

Y se verifica que está corriendo, ingresando:

```
$ ps -ef | grep cron
```

```
Output: root     <PID>    0  0 00:00 ?        00:00:00 /usr/sbin/cron
```

4. Instalación de dependencias (Python 3.8 o mayor)

Las librerías a utilizar junto a sus comandos de instalación para la implementación del interfaz son las siguientes:

Nota: Las siguientes librerías deben ser instaladas con el ambiente activado (ver punto 2.)

1- psycopg2 2.9.3

```
$ pip3.8 install psycopg2==2.9.3
```

2- asyncio 3.4.3

```
$ pip3.8 install asyncio==3.4.3
```

3- seaborn 0.11.2

```
$ pip3.8 install seaborn==0.11.2
```

4- streamlit 1.9.0

```
$ pip3.8 install streamlit==1.9.0
```

5- streamlit-autorefresh 0.0.1

```
$ pip3.8 install streamlit_autorefresh==0.0.1
```

6- numpy 1.22.4

```
$ pip3.8 install numpy==1.22.4
```

7- openpyxl 3.0.10

```
$ pip3.8 install openpyxl==3.0.10
```

8- matplotlib 3.5.2

```
$ pip3.8 install matplotlib==3.5.2
```

9- pandas 1.4.2

```
$ pip3.8 install pandas==1.4.2
```

10- Pillow 9.1.1

```
$ pip3.8 install pillow==9.1.1
```

11- `scikit-learn 1.1.1`

```
$ pip3.8 install sklearn
```

12- `scipy 1.8.1`

```
$ pip3.8 install scipy==1.8.1
```

5. Creación de carpeta con archivos necesarios para ejecutar código de interfaz

El código de la interfaz (*interfaz_general.py*) debe estar en una misma carpeta que los siguientes archivos para su ejecución:
Imágenes:
- bed_alone.png
- bed_sheet_logo.png
- boca_abajo.png
- boca_arriba.png
- c_amarillo.png
- c_rojo.png
- c_verde.png
- hombro_izquierdo.png
- hombro_derecho.png
- ht.jpg
- ht2.jfif
- logo.png

Archivos extra:
- datos_pos.pickle
- database_pctes.xlsx

Carpetas extra:
- datos_pos

Archivos de carpeta (datos_pos):
- datos_pos.pickle
- derecha.csv
- izquierda.csv
- prono.csv
- supino.csv

Nota: El código *interfaz_general.py* en el trabajo fue ubicado en una carpeta llamada "*interfaz_sabana*", en los siguientes pasos se utilizará la dirección del código con esa carpeta.

6. Ejecutar comando de ejecución y verificar funcionamiento

Desde la carpeta en donde se ubicaron los archivos anteriormente, junto al código *interfaz_general.py*, se debe ingresar en la consola el siguiente comando:

```
$ streamlit run interfaz_general.py
```

En donde luego, se indicará en el output la dirección IP local en donde deben ingresar para acceder a este (se debe estar conectado a la misma red WiFi por el momento).
Para acceder de manera remota, se debe abrir el puerto correspondiente al mostrado en el output (por defecto es el 8501).
Otra manera de acceder es mediante un *reverse proxy*, ocupando los puertos 443 u 80, los cuales también deben estar abiertos desde el router en cuestión. La apertura de puertos (port forwarding) dependen de la compañía o servicio de internet otorgado.

7. Creación de archivo bash para la verificación de ejecución

Para la creación de un archivo *.bash*, se debe ir a la carpeta en donde se desea crear el archivo e ingresar el siguiente comando:

```
$ touch archivo.bash
```

En donde se puede colocar el nombre que uno desee. Una vez creado, se debe editar dicho archivo, para esto, se puede utilizar el editor *nano*. Si no posee *nano*, se puede instalar mediante el siguiente comando:

```
$ sudo apt-get -y install nano
```

Una vez instalado, se ingresa el siguiente comando para editar el archivo:

```
$ nano archivo.bash
```

Dentro del archivo, se debe ingresar el siguiente código:

```
#!/bin/sh
SERVICE='interfaz_general.py'

if ps ax | grep -v grep | grep $SERVICE > /dev/null
then
    echo "$SERVICE service running, everything is fine" > /dev/null #sin mensaje
else > /dev/null  #sin mensaje
    echo "$SERVICE is not running. Starting."
    . /home/user/ambientes/sabana/bin/activate
    streamlit run /home/user/interfaz_sabana/$SERVICE &> /dev/null & #sin mensaje
fi
```

Luego, se le deben dar los permisos correspondientes para que quede como un archivo ejecutable, esto mediante el comando:

```
$ sudo chmod +x archivo.bash
```

Nota: Recordar que el código se encuentra en la carpeta "*interfaz_sabana*" y el ambiente "*sabana*" a activar se encuentra en la carpeta "*ambientes*".

8. Edición del crontab

Ya teniendo instalado y activado *cron*, para añadir tareas programadas se debe ingresar el siguiente comando en la terminal:

```
$ crontab -e
```

En donde, similar a usando el editor *nano*, se deben ingresar unas líneas de código al final del archivo.

Las líneas de código correspondientes a la verificación de que el archivo *.bash* se esté ejecutando cada un minuto son las siguientes:

```
$ * * * * * /home/user/carpeta/archivo.bash
```

Quedando un archivo similar al siguiente:

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
* * * * * /home/user/carpeta/archivo.bash
```

Nota: Recordar dejar un salto de línea (\n) al final del archivo (crontab) para que sea correctamente ejecutado.

Para verificar que esté funcionando, se debe esperar un minuto y luego ejecutar el siguiente comando:

```
$ ps -ef | grep interfaz_general
```

En donde saldrá el proceso activo de output correspondiente al interfaz de la sabanilla. Si no arroja ningún output es que hubo algún error y se recomienda verificar los pasos anteriores.

9. Ingreso al enlace

Luego de verificar que todos los pasos anteriores se realizaron correctamente, el código se debe estar ejecutando constantemente, por lo que para ingresar, se debe escribir el enlace entregado en el punto 6 o al momento de ejecutar el archivo *.bash*, debido a que Streamlit te entrega el enlace al momento de que el código es ejecutado.

Resumen FI

UNIVERSIDAD DE CONCEPCION – FACULTAD DE INGENIERIA
RESUMEN DE MEMORIA DE TITULO

Departamento	**:** Departamento de Ingeniería Eléctrica
Carrera	**:** Ingeniería Civil Biomédica
Nombre del memorista	**:** Gonzalo Andrés Rojas Bernard
Título de la memoria	**:** Sistema remoto de captura de variables fisiológicas para el apoyo en cuidados paliativos en el hogar
Fecha de la presentación oral	**:** 01 de septiembre, 2022, 13:00 hrs.
Profesor(es) Guía	**:** Esteban Pino Quiroga
Profesor(es) Revisor(es)	**:** César García
Concepto	**:**
Calificación	**:**

Resumen (máximo 200 palabras)

En el presente documento se informa sobre la implementación de un sistema de monitorización continua mediante un dispositivo de sabanilla con sensores integrados no invasivos. El sistema contará con la habilitación del dispositivo en sí, la implementación de una plataforma web para la gestión y visualización en tiempo real de un paciente/usuario y de la realización de una prueba piloto.

El objetivo general de este estudio es mejorar los cuidados paliativos de pacientes mediante un monitoreo con sensores no invasivos y remotos, esto mediante la incorporación de una sabanilla en la cama del paciente. Este estudio se hizo en colaboración con la empresa **Healthtracker Analytics**.

La sabanilla funciona mediante sensores FSR que detectan cambios de presión y los transforman a voltaje (V). Estos cambios de presión son enviados a un microcontrolador ESP32 el cual cuenta con un módulo WiFi que estará conectado a la red correspondiente al hogar del paciente.

A modo de conclusión, este sistema es capaz de saber el estado actual del paciente, obtener su frecuencia respiratoria en tiempo real, predecir la posición actual del paciente y detectar periodos fuera de cama.