

# Arquitectura de Sistemas Clínicos de Alta Velocidad: Definición Técnica y Operativa de MedLogger CLI

## 1. Introducción: La Paradoja de la Documentación Clínica y el Paradigma "Keyboard-First"

La práctica de la enfermería en entornos críticos —como las Unidades de Paciente Crítico (UPC), Urgencias y Oncología— se desarrolla en un escenario de alta presión cognitiva y temporal. La documentación clínica, fundamental para la continuidad del cuidado y la seguridad del paciente, se ha convertido paradójicamente en una carga administrativa que compite con el tiempo de atención directa. Los sistemas de Registro Clínico Electrónico (EHR) actuales, diseñados bajo paradigmas de interfaz gráfica de usuario (GUI) tradicionales, dependen excesivamente de la interacción mediante puntero (mouse), menús anidados y formularios rígidos. Esta fricción interactiva no solo ralentiza el registro, sino que induce errores por fatiga de decisión y fomenta prácticas peligrosas como el "copiar y pegar" acrítico de notas previas sin la debida verificación de datos cambiantes.

El proyecto "MedLogger CLI" surge como una respuesta arquitectónica y ergonómica a esta problemática. Se propone un sistema de documentación "Keyboard-First" (prioridad al teclado) que adapta la eficiencia de los editores de código modales (inspirados en Vim/Neovim) y la fluidez de herramientas de productividad ejecutiva (como Superhuman o Linear) al dominio clínico. El objetivo no es convertir a los enfermeros en programadores, sino dotarles de una interfaz de "Alta Velocidad Cognitiva" que permita transcribir el pensamiento clínico estructurado a la velocidad del habla, garantizando simultáneamente la estructuración de datos en segundo plano para fines de auditoría, investigación y soporte a la decisión.

Este informe técnico analiza exhaustivamente la arquitectura de software, la experiencia de usuario (UX), el modelo de datos y las estrategias de seguridad necesarias para implementar MedLogger CLI. Se integra evidencia de proyectos clínicos avanzados, como sistemas de gestión de heridas oncológicas y talleres de calidad asistencial, para fundamentar las decisiones de diseño en requisitos reales de trazabilidad, seguridad y eficiencia operativa.

## 2. Análisis del Dominio Clínico y Requisitos Cognitivos

Para diseñar una herramienta efectiva, es imperativo comprender la naturaleza de la información que gestiona. El análisis de registros de enfermería reales en entornos de alta complejidad revela que la documentación es inherentemente narrativa pero posee una estructura semántica rígida subyacente.

### 2.1. Deconstrucción de la Narrativa Clínica

Al examinar notas de evolución estándar y especializada, se identifican patrones recurrentes que trascienden el texto libre. Una nota de "Evolución Standard" en UPC, por ejemplo, sigue una secuencia lógica: Neurología (Glasgow, pupilas), Hemodinamia (PAM, ritmo, uso de drogas vasoactivas), Ventilatorio (modo, parámetros, saturación), Infeccioso (antibióticos, curvas febriles) y Metabólico/Eliminación.

La complejidad de estos registros es variable y dependiente del contexto. Una nota de "Ventilación Mecánica Invasiva" requiere documentar parámetros precisos: "Modo ACVC, Vtp xxx, PEEP xxx, Fresp PVA de xxxx". Un error en la transcripción de la PEEP (Presión Positiva al Final de la Espiración) o de la dosis de noradrenalina no es un error tipográfico menor; es un evento adverso potencial. Los sistemas actuales, al obligar al profesional a navegar entre pestañas para ingresar estos datos, rompen el flujo de pensamiento. MedLogger CLI debe permitir la inserción lineal de estos bloques complejos ("chunks") sin abandonar el editor, utilizando "Snippets Clínicos Inteligentes" que guíen el llenado de las variables críticas (placeholders).

## 2.2. Variabilidad y Necesidad de Estandarización

La investigación cualitativa en entornos oncológicos destaca que la falta de estandarización en los protocolos y la variabilidad en el juicio clínico generan diferencias en la calidad del cuidado y duplicación de recursos. MedLogger CLI actúa como un "riel" cognitivo: al invocar un snippet para "Curación de Herida Oncológica", el sistema no solo provee texto, sino que estructura el proceso mental del enfermero según estándares validados (como el acrónimo TIMERS para evaluación de tejidos). Esto transforma el editor de texto en un motor de soporte a la decisión clínica pasivo, donde la arquitectura de la información induce el cumplimiento de protocolos de seguridad.

## 3. Modelo Mental y UX: Adaptación de la Edición Modal a la Enfermería

La propuesta de valor central de MedLogger es la adaptación de los "Modos" de Vim para segregar las intenciones del usuario, reduciendo el riesgo de errores accidentales y optimizando la velocidad de navegación.

### 3.1. Arquitectura de Modos Clínicos

En lugar de una traducción literal de los comandos de programación, se redefine la semántica de los modos para el flujo de trabajo hospitalario.

| Modo Vim Tradicional | Modo MedLogger CLI            | Propósito Clínico y Comportamiento   |
|----------------------|-------------------------------|--|
| <b>NORMAL</b>        | <b>Modo Lectura / Guardia</b> | Estado por defecto. Las teclas no escriben texto, sino que ejecutan acciones de navegación y revisión. Permite al enfermero leer la historia clínica sin riesgo de insertar caracteres basura ("dsfdsf") por |

|                      |                               |   |
|----------------------|-------------------------------|---|
| Modo Vim Tradicional | Modo MedLogger CLI            | Propósito Clínico y Comportamiento  |
|                      |                               | error. Las teclas j y k (o flechas) desplazan la vista por secciones semánticas (ej. saltar de "Antecedentes" a "Plan"), no por líneas visuales.  |
| <b>INSERT</b>        | <b>Modo Narrativa</b>         | Se activa explícitamente (ej. tecla i). Es el único modo donde se ingresa texto libre. Diseñado para capturar matices cualitativos que no caben en formularios ("paciente refiere ansiedad por visita familiar"). La salida de este modo devuelve inmediatamente al estado seguro de Lectura. |
| <b>VISUAL</b>        | <b>Modo Estructuración</b>    | Permite seleccionar texto narrativo para etiquetarlo semánticamente. Ejemplo: Seleccionar "EVA 7/10" y aplicar la etiqueta Dato:Dolor. Esto conecta el texto no estructurado con el modelo de datos subyacente sin alterar la legibilidad humana.   |
| <b>COMMAND</b>       | <b>Modo Ejecutivo (Cmd+K)</b> | El centro de control. Invocado mediante Cmd+K o :, despliega la Paleta de Comandos. Desde aquí se gestionan acciones del sistema ("Guardar", "Interconsulta", "Alta") y la inserción de conocimiento clínico (Snippets, Protocolos).  |

### 3.2. La Paleta de Comandos como Cerebro Operativo

Inspirada en la eficiencia de Superhuman y Linear, la paleta de comandos reemplaza la navegación por menús profundos. En un entorno de enfermería, la memoria muscular es más fiable que la memoria visual.

- **Búsqueda Difusa (Fuzzy Search):** El enfermero no necesita recordar si el protocolo se llama "Manejo de Catéter Venoso Central" o "CVC Cuidados". Al escribir cvc, el sistema debe ofrecer inmediatamente los snippets relevantes: "Instalación CVC", "Curación CVC", "Retiro CVC".
- **Contextualidad:** La paleta es sensible al contexto del paciente. Si el paciente está ingresado en "UPC Coronaria", la paleta prioriza comandos y snippets cardiológicos (:ecg, :troponinas) sobre los oncológicos o traumáticos, reduciendo el ruido cognitivo.

### 3.3. Ergonomía y Reducción de Fatiga

El diseño debe considerar la fatiga física y mental de turnos de 12 o 24 horas. La interfaz oscura (Dark Mode) por defecto reduce la fatiga visual en estaciones de enfermería nocturnas. El uso exclusivo del teclado elimina los micro-movimientos repetitivos de alcanzar el mouse, reduciendo el riesgo de lesiones por esfuerzo repetitivo (RSI) y manteniendo las manos en la "zona de acción" principal.

## 4. Arquitectura Tecnológica: Selección del Stack y Justificación

La elección de tecnologías no es arbitraria; responde a la necesidad de equilibrar rendimiento, mantenibilidad y seguridad en un entorno donde el hardware puede ser obsoleto y la red inestable.

### 4.1. El Dilema del Motor de Edición: CodeMirror 6 vs. Alternativas

Para lograr la experiencia de "Vim" en un navegador, se requiere un control granular sobre el manejo de entradas y el renderizado de texto.

- **CodeMirror 6 (Recomendado):** Representa el estado del arte en editores web. Su arquitectura modular permite cargar solo lo necesario.
  - *Ventaja Crítica:* Posee un modelo de estado funcional e inmutable que se separa de la vista (DOM). Esto facilita la implementación de un "Shadow DOM" o modelo de datos paralelo (JSON) que se actualiza con cada transacción del editor, crucial para mantener sincronizados el texto visible y los datos estructurados.
  - *Soporte Vim:* Existen adaptadores robustos que emulan la máquina de estados de Vim, permitiendo personalizar los keymaps para el dominio clínico.
- **Monaco Editor (VS Code):** Aunque potente, su peso (varios megabytes) y consumo de memoria son excesivos para una aplicación que debe cargar instantáneamente en PCs hospitalarios antiguos. Su arquitectura está demasiado acoplada a la edición de código de software, lo que dificulta la personalización para "código clínico" (texto enriquecido con datos).
- **Textarea "Vitaminado":** Descartado. No permite resaltado de sintaxis (necesario para diferenciar visualmente los valores variables de los textos fijos en los snippets) ni gestión avanzada de rangos de selección para la navegación por tabuladores.

### 4.2. Estrategia de Plataforma: Web-First con Capacidades Híbridas

Se propone una arquitectura **Single Page Application (SPA)** construida con **React** y **TypeScript**.

- **Justificación:** React ofrece el ecosistema más maduro para interfaces complejas basadas en estado. TypeScript añade una capa de seguridad de tipos indispensable para manejar estructuras de datos clínicos complejos (FHIR, SBAR) sin errores en tiempo de ejecución.
- **Despliegue Híbrido:** La aplicación se distribuye principalmente como web (accesible vía intranet hospitalaria). Para estaciones críticas o uso offline, se encapsula en **Tauri**. Tauri

utiliza el WebView nativo del sistema operativo (WebView2 en Windows, WebKit en macOS/Linux) y un backend en Rust, resultando en binarios minúsculos (<10MB) y un consumo de RAM drásticamente menor que Electron. Esto es vital para hardware legado en hospitales públicos.

### 4.3. Stack Recomendado

| Capa                | Tecnología          | Justificación Técnica y Clínica   |
|---------------------|---------------------|---|
| Frontend Framework  | React + Vite        | Velocidad de desarrollo y optimización de assets. Vite permite Hot Module Replacement (HMR) instantáneo, acelerando la iteración de UI.                                       |
| Lenguaje            | TypeScript          | Tipado estático estricto para prevenir errores en el manejo de dosis y datos sensibles.   |
| Editor Core         | CodeMirror 6        | Motor ligero, modular, accesible y con excelente soporte para modos Vim personalizados.   |
| State Management    | Zustand             | Gestión de estado minimalista y eficiente, ideal para manejar la sesión del usuario y el estado transitorio de la nota sin la complejidad de Redux.                           |
| Command Palette     | cmdk                | Librería "headless" (sin estilos) de alto rendimiento, accesible y componible, perfecta para integrar con el sistema de diseño propio.  |
| UI System           | Radix UI + Tailwind | Primitivas accesibles (cumplimiento WCAG para accesibilidad hospitalaria) y estilizado atómico de alto rendimiento.   |
| Validación de Datos | Zod                 | Definición de esquemas de datos en tiempo de ejecución. Permite validar que un snippet insertado cumple con la estructura requerida (ej. rangos de presión arterial válidos). |

## 5. Diseño del Motor de Snippets Clínicos: De Texto a Datos

El motor de snippets es el corazón de la eficiencia de MedLogger. No se trata de simples

macros de sustitución de texto, sino de micro-aplicaciones que se ejecutan dentro del editor.

## 5.1. Anatomía de un Snippet Clínico Inteligente

Un snippet debe encapsular tanto el contenido narrativo como la lógica de datos. Se propone un esquema JSON que soporte "Placeholders" (huecos rellenables), "Selectores" (opciones predefinidas) y lógica condicional básica.

Considérese el siguiente ejemplo basado en las notas de "Hemodinamia UPC" identificadas en la investigación :

```
{
  "id": "hemodinamia-upc-v2",
  "trigger": "/hemo",
  "label": "Evaluación Hemodinámica Completa",
  "category": "Cardiovascular",
  "context": ["UPC", "Coronaria"],
  "body":,
  "logic": {
    "use_dva": {
      "if": "${1} == 'inestable'",
      "then_insert": "Requiere soporte vasoactivo: ${dva_name} a  
${dva_dose} mcg/kg/min.",
      "else_insert": "Sin requerimientos de drogas vasoactivas."
    }
  },
  "data_mapping": {
    "pam": "${2}",
    "fc": "${4}",
    "perfusión": "${6}"
  }
}
```

## 5.2. Dinámica de Interacción y Placeholders

Al invocar /hemo, el editor renderiza el texto. El cursor se sitúa automáticamente en el primer placeholder (\${1}).

- **Navegación Tabular:** El usuario selecciona "inestable" y presiona Tab. El sistema salta inmediatamente al siguiente campo (PAM). No es necesario usar las flechas ni el mouse.
- **Lógica Condicional:** Si el usuario selecciona "inestable", el motor de snippets inyecta dinámicamente la línea de "Requiere soporte vasoactivo". Esta capacidad de *templating* dinámico previene la inconsistencia frecuente en notas copiadas donde se dice "paciente inestable" pero se olvida documentar las drogas, o viceversa.
- **Validación Inline:** Los campos numéricos (como PAM o FC) pueden tener validadores asociados en el esquema JSON (ej. min: 30, max: 250). Si el usuario introduce un valor fuera de rango, el editor muestra una advertencia visual discreta (subrayado amarillo) sugiriendo verificación, sin bloquear el flujo de escritura pero alertando sobre posibles errores de digitación.

### 5.3. Gobernanza y Versionado de Snippets

Los snippets no deben ser estáticos ni locales. Se gestionan como código ("Clinical Content as Code").

- **Repositorio Central:** Los archivos JSON de snippets se almacenan en un repositorio Git controlado.
- **Versionado:** Cada cambio en un protocolo clínico (ej. cambio en la meta de PAM de 65 a 70 mmHg) genera una nueva versión del snippet.
- **Distribución:** Al iniciar sesión, MedLogger CLI descarga la última versión de los snippets aprobados para la unidad del usuario. Esto garantiza que todos los enfermeros utilicen el estándar de documentación más reciente, resolviendo el problema de la variabilidad clínica.

## 6. Estructura de Datos, Persistencia y Sincronización

El desafío técnico más complejo es conciliar la naturaleza no estructurada del texto con la necesidad de datos estructurados para bases de datos relacionales y análisis.

### 6.1. El Patrón "Shadow Object" (Objeto Sombra)

MedLogger CLI implementa un patrón de diseño donde el documento tiene dos representaciones simultáneas:

1. **Representación Visual:** El texto que ve y edita el enfermero.
2. **Representación de Datos (Shadow JSON):** Un objeto oculto que se actualiza en tiempo real.

Cuando el usuario completa el placeholder de "PAM" con "70", el motor de snippets actualiza el Shadow JSON:

```
"vital_signs": {  
  "mean_arterial_pressure": { "value": 70, "unit": "mmHg",  
  "timestamp": "2023-10-27T08:00:00Z" }  
}
```

Si el usuario borra manualmente la frase "PAM sobre 70 mmHg", el motor detecta la eliminación del rango asociado y remueve la entrada del JSON. Esto asegura que la base de datos estructurada sea siempre un reflejo fiel de la nota narrativa, eliminando la discrepancia común donde el texto dice una cosa y los campos estructurados dicen otra.

### 6.2. Estrategia "Local-First" y Tolerancia a Fallos

Dada la infraestructura crítica y a veces precaria de las redes hospitalarias, la aplicación debe funcionar plenamente sin conexión a internet.

- **Persistencia Local (IndexedDB/RxDB):** Cada pulsación de tecla (keystroke) se guarda en una base de datos local en el navegador (IndexedDB). Esto previene la pérdida de datos ante cierres accidentales o cortes de energía. RxDB es una excelente opción tecnológica aquí, ya que permite una sincronización reactiva con el backend cuando la red se restablece.

- **Cifrado en Reposo:** Dado que el navegador almacena datos sensibles de salud (PHI), la base de datos local IndexedDB debe estar cifrada. Se recomienda el uso de **AES-256** para cifrar el contenido de los registros antes de guardarlos localmente, utilizando una clave derivada de las credenciales del usuario que se mantiene solo en memoria volátil durante la sesión.

### 6.3. Backend y Modelo de Datos Relacional

Para la persistencia a largo plazo y la colaboración, se utiliza una arquitectura de backend robusta.

- **Base de Datos: PostgreSQL** es el estándar por su fiabilidad y soporte para JSONB. Esto permite almacenar la estructura flexible del Shadow JSON junto con columnas relacionales estrictas para indexación (ID paciente, Fecha, Profesional).
- **Arquitectura de Microservicios vs Monolito:** Siguiendo las definiciones de arquitectura de software, para la fase inicial se recomienda un **Monolito Modular**. Separar prematuramente en microservicios añade complejidad operativa (orquestación, latencia de red) innecesaria para el volumen inicial. Un monolito bien estructurado, donde el módulo de "Usuarios", "Notas" y "Snippets" estén desacoplados lógicamente pero desplegados juntos, facilita el desarrollo y el despliegue.

## 7. Calidad, Seguridad y Cumplimiento Normativo

La seguridad en salud no es una característica opcional, es un requisito funcional. MedLogger incorpora principios de seguridad y calidad desde el diseño.

### 7.1. Trazabilidad y Auditoría Granular

Los sistemas tradicionales registran *quién* guardó la nota y *cuándo*. MedLogger debe ir más allá para cumplir con estándares de calidad avanzados.

- **Audit Trail de Composición:** El sistema registra *cómo* se creó la nota.
  - [10:05:01] User X triggered snippet 'examen-fisico-normal'
  - [10:05:05] User X modified field 'lung\_sounds' from 'normal' to 'crepitos bases'
  - [10:05:30] User X signed note
- **Utilidad Forense:** Este nivel de detalle permite distinguir entre una nota creada automáticamente sin revisión (tiempo de edición < 2 segundos) y una nota cuidadosamente editada. Esto es vital para la defensa legal del profesional y para la auditoría de calidad de los procesos de enfermería.

### 7.2. Seguridad en la Medicación y "Guarda-Raíles" (Guardrails)

Integrando los hallazgos del taller de calidad sobre administración segura de medicamentos, MedLogger puede actuar como una barrera de seguridad activa.

- **Alertas de Alto Riesgo:** Si un usuario intenta usar un snippet para administrar un medicamento de alto riesgo (ej. Heparina, Insulina), el sistema puede activar un "Modo de Confirmación" que exige una segunda validación (simulando el doble chequeo).
- **Indicadores de Proceso:** El sistema puede calcular automáticamente indicadores de calidad, como el tiempo transcurrido entre la orden médica y el registro de administración



(TAT), alimentando los tableros de gestión de la unidad sin requerir recolección manual de datos.

### 7.3. Privacidad y Minimización de Datos

Siguiendo los principios expuestos en el proyecto WoundCare , se aplica el principio de **Mínima Necesidad**. El sistema local solo descarga y cachea los datos de los pacientes asignados al turno del enfermero, minimizando la superficie de exposición de datos en caso de robo del dispositivo físico. La autenticación se maneja mediante **OIDC (OpenID Connect)** con rotación frecuente de tokens JWT, asegurando que el acceso se revoque inmediatamente al finalizar el turno.

## 8. Interoperabilidad: Integración con el Ecosistema Clínico

MedLogger CLI no puede ser una isla; debe integrarse con el HIS (Hospital Information System) existente.

### 8.1. Estrategia FHIR (Fast Healthcare Interoperability Resources)

HL7 FHIR es el estándar de facto para el intercambio de datos clínicos. MedLogger debe implementar un adaptador que mapee su modelo de datos interno a recursos FHIR.

| Datos MedLogger                 | Recurso FHIR             | Uso   |
|---------------------------------|--------------------------|---|
| Nota de Enfermería completa     | Composition              | Documento clínico legal que agrupa toda la información del evento.                                |
| Texto narrativo renderizado     | DocumentReference        | Versión PDF/HTML inmutable de la nota para visualización en otros sistemas.                       |
| Signos Vitales (PAM, FC, SatO2) | Observation              | Datos discretos extraídos del Shadow JSON para alimentar las gráficas de tendencias del paciente. |
| Procedimientos (CVC, Sonda)     | Procedure                | Registro de intervenciones realizadas.  |
| Administración de Fármacos      | MedicationAdministration | Registro del cumplimiento de la terapia farmacológica.  |

### 8.2. Integración Legacy (HL7 v2)

Dado que muchos hospitales aún operan con estándares antiguos, se debe contemplar una capa de integración ("Anti-Corruption Layer") que traduzca los objetos JSON modernos de MedLogger a mensajes HL7 v2 (ej. mensajes ORU^R01 para envío de resultados/notas) si la integración nativa FHIR no está disponible.

## 9. Plan de Implementación y Hoja de Ruta (Roadmap)

La implementación de un cambio de paradigma tan radical requiere una estrategia gradual para gestionar la resistencia al cambio y mitigar riesgos técnicos. Se propone un plan en cuatro fases, alineado con las estrategias de pilotaje clínico.

## **Fase 1: Prototipo "Single-Player" y Validación Ergonómica (Meses 1-2)**

- **Objetivo:** Validar que el modelo de interacción Vim-like es usable por enfermeros reales.
- **Alcance:** SPA local (sin backend). Persistencia en navegador. Carga manual de JSONs de snippets.
- **Actividades Clave:**
  - Diseño de la librería inicial de snippets basada en las notas de (Neuro, UPC, Ventilación). \* Pruebas de usabilidad con enfermeros "Power Users" para ajustar atajos de teclado y sensibilidad de la búsqueda difusa.
  - Implementación del motor de "Shadow JSON".
- **KPI de Éxito:** Velocidad de registro en escenarios simulados superior al método tradicional.

## **Fase 2: Piloto Clínico Controlado (Meses 3-4)**

- **Objetivo:** Pruebas en entorno real con pacientes seleccionados (bajo riesgo).
- **Alcance:** Backend básico para autenticación y guardado centralizado. Cifrado local activado. Generación de PDFs para imprimir y pegar en la ficha física (puente transitorio).
- **Seguridad:** Implementación de logs de auditoría y controles de acceso basados en roles (RBAC).
- **KPI de Éxito:** Adopción voluntaria > 70% en el grupo piloto , cero pérdidas de datos.

## **Fase 3: Despliegue de Unidad y Colaboración (Meses 5-8)**

- **Objetivo:** Operación estándar en una unidad completa (ej. UPC).
- **Alcance:** Sincronización multi-usuario en tiempo real (evitar conflictos de edición). Gestión de perfiles de unidad (snippets específicos para UPC vs Urgencia). Integración de indicadores de calidad automatizados.
- *Actividades Clave:* Capacitación masiva. Migración de datos históricos relevantes.
- **KPI de Éxito:** Reducción del tiempo de documentación por enfermero/turno. Mejora en la completitud de los registros (menos campos vacíos).


## **Fase 4: Integración Sistémica y Escalado (Meses 9+)**

- **Objetivo:** MedLogger como interfaz oficial del EHR.
- **Alcance:** Integración bidireccional FHIR. El EHR alimenta a MedLogger con datos del paciente (demográficos, diagnósticos) y MedLogger devuelve notas estructuradas.
- **Evolución Futura:** Incorporación de modelos de IA/LLM para sugerir snippets basados en el historial del paciente, pero manteniendo siempre el control determinístico y la verificación humana (Human-in-the-loop), alineado con la visión avanzada de WoundCare.

## 10. Conclusiones

MedLogger CLI representa una evolución necesaria en la informática clínica. Al abandonar la metáfora del "formulario de papel digitalizado" y abrazar paradigmas de interacción de alta eficiencia, se ataca la raíz del problema de la carga administrativa en enfermería. La arquitectura propuesta —basada en tecnologías web modernas, desacoplada, segura por diseño y centrada en la estructuración invisible de datos— ofrece una solución robusta que no solo mejora la calidad de vida del profesional, sino que eleva la seguridad del paciente y la calidad de los datos clínicos. La viabilidad técnica está respaldada por el stack seleccionado (React, CodeMirror, Tauri, PostgreSQL), y la viabilidad clínica se fundamenta en la alineación con los flujos de trabajo reales de cuidados críticos y los estándares de calidad vigentes.

### Fuentes citadas

1. next level shit nursing evolution 3000,  
[https://drive.google.com/open?id=1l-w4\\_5D4nmrVz\\_p\\_iPye7ri0EGNJCckuZjRKEifd7dE](https://drive.google.com/open?id=1l-w4_5D4nmrVz_p_iPye7ri0EGNJCckuZjRKEifd7dE) 2.  Formulario Cualitativo ,  
[https://drive.google.com/open?id=1CRlkKgHyuZuAkodvzRZJJe11ZuNQs\\_x98WKVjWyTo-I](https://drive.google.com/open?id=1CRlkKgHyuZuAkodvzRZJJe11ZuNQs_x98WKVjWyTo-I) 3.  
resumen del proyecto app web,  
[https://drive.google.com/open?id=1yEr7IJUyi-6En1N\\_MWZDwempZyQiwdDPpcj6yp\\_rUn4](https://drive.google.com/open?id=1yEr7IJUyi-6En1N_MWZDwempZyQiwdDPpcj6yp_rUn4) 4.  
CUANTITATIVO,  
[https://drive.google.com/open?id=1x9iteWgj5\\_UlLnFV9qf-dfhh\\_i-madvaSn6WVASZwk](https://drive.google.com/open?id=1x9iteWgj5_UlLnFV9qf-dfhh_i-madvaSn6WVASZwk) 5.  
TALLER 2 CALIDAD.docx,  
<https://drive.google.com/open?id=18T4qKnFHpMvPjRKteG9XgPA30HPynzF8>