****

**GESTIÓN DE LA INFORMACIÓN DE SALUD PÚBLICA EN REDES SOCIALES**

**Juan Manuel Loaiza Perez**

**Diego Alejandro Garcia Perez**

**Vanesa Alexandra Acevedo Motato**

**Paula Valentina Torres Jimenez**

**Facultad de ingeniería, Universidad Central**

**Ingeniería de software II**

**Profesor:**

**Miguel Hernandez Bejarano**

**Fecha:**

**22 Mayo 2025**

**Gestión de la Información de Salud Pública en Redes Sociales**

El objetivo de este proyecto es diseñar y desarrollar una aplicación de software integral para la gestión y monitoreo de la información relacionada con temas de salud pública en redes sociales. Esta herramienta permitirá a las autoridades sanitarias y organizaciones encargadas optimizar la recopilación y análisis de contenido relevante, obtener información valiosa sobre la percepción pública de diversos temas de salud y realizar un seguimiento exhaustivo de las interacciones en línea relacionadas con campañas de salud.

A partir del análisis detallado del problema, se identificaron los requisitos esenciales para el desarrollo de la aplicación, que deben ser cumplidos para asegurar su efectividad en el ámbito de la salud pública. Los principales requisitos son los siguientes:

1. Manejo de Datos No Estructurados: La aplicación deberá ser capaz de procesar y gestionar datos no estructurados provenientes de diversas redes sociales. Estos datos incluyen textos de publicaciones y comentarios, hashtags, entre otros relacionados con temas de salud.

2. Monitoreo de la Percepción Pública de la Salud: Se requiere que la aplicación realice un monitoreo continuo de la percepción pública sobre temas de salud, campañas sanitarias o productos específicos de salud, mediante el análisis de contenido proveniente de plataformas sociales como Facebook, Twitter e Instagram.

3. Operaciones CRUD de Información: La aplicación deberá permitir realizar operaciones de gestión sobre los datos obtenidos de las redes sociales, tales como crear, leer, actualizar y eliminar la información recolectada. Esto posibilitará un control completo sobre la gestión de la información relacionada con la salud pública.

4. Consultas Personalizadas: Los usuarios podrán definir criterios de búsqueda específicos para realizar consultas más precisas sobre la información de interés. Los parámetros de búsqueda incluirán, entre otros, palabras clave, hashtags, menciones, temas de salud y rangos de fechas.

5. Visualización de Publicaciones Relevantes: La aplicación mostrará las publicaciones recuperadas que coincidan con los criterios de búsqueda establecidos por el usuario, permitiendo la visualización clara y ordenada de la información relacionada con la salud pública y facilitando la toma de decisiones basada en datos actuales.

Requisitos Funcionales y Técnicos:

• Debe permitir la configuración de filtros personalizados que faciliten la búsqueda de información relevante según los objetivos de las campañas de salud pública.

• La interfaz de usuario debe ser intuitiva y accesible, garantizando una experiencia fácil para los usuarios encargados de monitorear y gestionar la información.

Este sistema se presenta como una solución clave para el fortalecimiento de las estrategias de comunicación en salud pública y el mejoramiento de la gestión de la información sanitaria a través de las redes sociales.

|  |  |
| --- | --- |
| **No.** | **Listado de Requerimientos** |
| 1 | Procesamiento y gestión de datos no estructurados de redes sociales. |
| 2 | Monitoreo continuo de la percepción pública sobre temas de salud. |
| 3 | Implementación de operaciones CRUD para la información recolectada. |
| 4 | Consultas personalizadas con criterios de búsqueda definidos por el usuario. |
| 5 | Visualización clara y ordenada de publicaciones relevantes. |
| 6 | Configuración de filtros personalizados para optimizar la búsqueda. |
| 7 | Interfaz intuitiva y accesible para una experiencia de usuario eficiente. |

**Referencia:** [**Atlassian Cloud**](https://ucentral-team-he7won41.atlassian.net/jira/software/projects/UCENTRAL/boards/3/timeline)

|  |  |  |
| --- | --- | --- |
| **ID** | **Historia de Usuario** | **Descripción** |
| HU-01 | Recopilación de Datos | Como ingeniero/a de sistemas, quiero que la aplicación recopile publicaciones, comentarios y hashtags de redes sociales para analizar la percepción pública sobre temas de salud. |
| HU-02 | Monitoreo de Tendencias | Como ingeniero/a de sistemas, quiero que la aplicación identifique tendencias y patrones en la discusión de temas de salud para evaluar el impacto de las campañas de concienciación. |
| HU-03 | CRUD de Información | Como ingeniero/a de sistemas, quiero poder gestionar (crear, leer, actualizar y eliminar) la información obtenida de redes sociales para mantener una base de datos relevante y actualizada. |
| HU-04 | Consultas Personalizadas | Como ingeniero/a de sistemas, quiero poder realizar búsquedas filtrando por palabras clave, hashtags y fechas para obtener información específica sobre salud pública. |
| HU-05 | Visualización de Datos | Como ingeniero/a de sistemas, quiero ver las publicaciones filtradas en una interfaz ordenada para facilitar la toma de decisiones basada en datos. |

**Referencia:** [**Miro**](https://miro.com/app/board/uXjVIPTKgNY=/?share_link_id=583628988222&shareablePresentation=1)

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario** | **Complejidad (1-5)** | **Prioridad** |
| HU-01 | 4 | Alta |
| HU-02 | 5 | Alta |
| HU-03 | 3 | Media |
| HU-04 | 3 | Media |
| HU-05 | 2 | Alta |

|  |  |
| --- | --- |
| **Historia de Usuario** | **Tareas** |
| HU-01 | Diseñar el módulo de extracción de datos. |
| HU-01 | Implementar la conexión con APIs de redes sociales. |
| HU-01 | Configurar autenticación y permisos de acceso a las APIs. |
| HU-01 | Procesar datos no estructurados (texto, imágenes y videos). |
| HU-01 | Normalizar y limpiar los datos extraídos. |
| HU-01 | Almacenar la información estructurada en la base de datos. |
| HU-01 | Implementar estrategias para la actualización periódica de datos. |
|  | |
| HU-02 | Diseñar algoritmo de análisis de tendencias en redes sociales. |
| HU-02 | Implementar detección de palabras clave y hashtags relevantes. |
| HU-02 | Integrar técnicas de procesamiento de lenguaje natural (NLP). |
| HU-02 | Desarrollar métricas de impacto de tendencias en salud pública. |
| HU-02 | Generar informes automatizados de patrones detectados. |
| HU-02 | Configurar alertas para cambios significativos en tendencias. |
|  | |
| HU-03 | Diseñar la estructura de base de datos con normalización adecuada. |
| HU-03 | Implementar operaciones CRUD para la gestión de la información. |
| HU-03 | Validar la integridad y consistencia de los datos en la BD. |
| HU-03 | Crear interfaz de usuario para la gestión manual de registros. |
| HU-03 | Integrar control de versiones y registros de cambios. |
|  | |
| HU-04 | Diseñar módulo de búsqueda avanzada en la aplicación. |
| HU-04 | Implementar filtros por palabra clave, hashtag y fecha. |
| HU-04 | Optimizar el rendimiento de las consultas para grandes volúmenes de datos. |
| HU-04 | Desarrollar opciones de autocompletado en búsquedas. |
| HU-04 | Implementar paginación y carga dinámica de resultados. |
|  | |
| HU-05 | Diseñar interfaz de visualización de publicaciones relevantes. |
| HU-05 | Implementar categorías y ordenamiento de resultados. |
| HU-05 | Crear gráficos e indicadores de relevancia de publicaciones. |
| HU-05 | Integrar opción de exportación de datos en múltiples formatos (CSV, PDF, JSON). |
| HU-05 | Mejorar la accesibilidad y usabilidad de la interfaz gráfica. |

**Referencia Repositorio:** [**https://github.com/acevedovanesa26/Gestion-de-Salud/tree/main**](https://github.com/acevedovanesa26/Gestion-de-Salud/tree/main)

**Plan de prueba y de calidad del software**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **VITAL DATA** | | | | | | | |
| **CÓDIGO** | | HU01 | | **FECHA** | | Mayo /2025 | |
| Iniciar Sesión | | | | | | | |
| **PRUEBA DE INTEGRACIÓN** | | | | | | | |
| **MÓDULO** | autenticación | **CLASE** | Usuario | **VERSIÓN** | 1.0 | **INGENIERO** |  |
|  |  | | | | | | |
| **OBJETIVOS** | Verificar que el sistema autentique correctamente a los usuarios con credenciales válidas | | | | | | |
| **REQUERIMIENTOS** | -JDK 11+  -Claves API para Twitter, Facebook, Instagram  -Conexión a base de datos a “HealthAnalyticsDB” | | | | | | |
| **PRE CONDICIONES** | El usuario debe estar registrado en la base de datos con un usuario “ —--- Nombre usuario creado—-- ” y contraseña “-----la contraseña que se asignó —---- ” | | | | | | |
| **DETALLES DE LA PRUEBA** | | | | | | | |
| **EJECUCIÓN N°** |  | **FECHA DE EJECUCIÓN** |  | **ING PRUEBA** |  | | |
| **OBSERVACIONES** |  | | | | | | |
| **N°** | **MÉTODO** | **PARÁMETROS** | **RESULTADO ESPERADO** | **RESULTADO OBTENIDO** | **CALIFICACIÓN** | **OBSERVACIONES** | |
| 1 | +iniciar Sesión | Usuario: “juan loaiza”, contraseña: “loaiza123 ” | Inicio de sesión exitoso | Ingresar a la página principal | * **A** * **R** |  | |
| 2 | Usuario: “juan loaiza”, contraseña: “loaiza123456 ” | Error: ”Usuario o Contraseña incorrectos” | Error: ”Usuario o Contraseña incorrectos” | * A * R |  | |
| 3 | Usuario: “juan Manuel”, contraseña: “loaiza123 ” | Error: “Usuario o Contraseña incorrectos” | Error: “Usuario o Contraseña incorrectos” | * A * R |  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **VITAL DATA** | | | | | | | |
| **CÓDIGO** | | HU01 | | **FECHA** | | Mayo /2025 | |
| Extracción y Almacenamiento de Datos | | | | | | | |
| **PRUEBA DE INTEGRACIÓN** | | | | | | | |
| **MÓDULO** | Extracción de Datos | **CLASE** | DataExtractionService | **VERSIÓN** | 1.0 | **INGENIERO** |  |
|  |  | | | | | | |
| **OBJETIVOS** | Verificar que el módulo extrae datos de redes sociales y los almacena correctamente. | | | | | | |
| **REQUERIMIENTOS** | JDK 11+  Librerías HTTP/REST (e.g. OkHttp, Retrofit)  Claves API para Twitter, Facebook, Instagram  Conector JDBC a “HealthAnalyticsDB” | | | | | | |
| **PRE CONDICIONES** | Claves API válidas y activas  Base de datos accesible y esquema de pruebas cargado | | | | | | |
| **DETALLES DE LA PRUEBA** | | | | | | | |
| **EJECUCIÓN** |  | **FECHA DE EJECUCIÓN** |  | **ING PRUEBA** |  | | |
| **OBSERVACIONES** |  | | | | | | |
| **NO** | **MÉTODO** | **PARÁMETROS** | **RESULTADO ESPERADO** | **RESULTADO OBTENIDO** | **CALIFICACIÓN** | **OBSERVACIONES** | |
| 1 | extractData(feeds, fechaI, fechaF) | feeds=["#salud"], fechaI="2025-05-01", fechaF="2025-05-05" | Lista de objetos crudos (texto, imagen, video) | 120 objetos devueltos (85 texto, 25 imágenes, 10 videos) | * A * R |  | |
| 2 | connectToAPIs() |  | Conexión exitosa a todas las APIs | Todas las conexiones OK (Twitter 200, FB 200, Instagram 200) | * A * R |  | |
| 3 | authenticateAPIs() |  | Endpoints responden 200 OK | Autenticación: Twitter OK, FB OK, Instagram OK | * A * R |  | |
| 4 | processRawData(rawList) | 120 objetos | Estructuras intermedias JSON | 120 JSON normalizados con campos: idApi, texto, fecha, tipoMedia | * A * R |  | |
| 5 | normalizeData(intermediate) | 120 JSON | Sin duplicados, encoding corregido | 115 registros únicos tras eliminar 5 duplicados; todo en UTF-8 | * A * R |  | |
| 6 | saveStructuredData(records) | 115 registros | Inserta en tabla y retorna filas afectadas | 115 filas insertadas en posts\_sociales | * A * R |  | |
| 7 | scheduleUpdates(interval) | "24h" | Job programado cada 24 h | Job “DataExtractionJob” programado; próxima ejecución en 2025-05-28T02:00 | * A * R |  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **VITAL DATA** | | | | | | | |
| **CÓDIGO** | | HU02 | | **FECHA** | | Mayo /2025 | |
| Análisis de Tendencias | | | | | | | |
| **PRUEBA DE INTEGRACIÓN** | | | | | | | |
| **MÓDULO** | Análisis de Tendencias | **CLASE** | Servicio de análisis de tendencias | **VERSIÓN** | 1.0 | **INGENIERO** |  |
|  |  | | | | | | |
| **OBJETIVOS** | Detectar patrones y generar métricas de impacto de tendencias en salud. | | | | | | |
| **REQUERIMIENTOS** | Biblioteca NLP (e.g. OpenNLP o spaCy Java)  Acceso a datos ya extraídos  Servicio de notificaciones (e.g. email, webhook) | | | | | | |
| **PRE CONDICIONES** | Al menos 50 registros en tabla posts\_sociales  Pipeline de extracción (HU-01) funcionando | | | | | | |
| **DETALLES DE LA PRUEBA** | | | | | | | |
| **EJECUCIÓN** |  | **FECHA DE EJECUCIÓN** |  | **ING PRUEBA** |  | | |
| **OBSERVACIONES** |  | | | | | | |
| **NO** | **MÉTODO** | **PARÁMETROS** | **RESULTADO ESPERADO** | **RESULTADO OBTENIDO** | **CALIFICACIÓN** | **OBSERVACIONES** | |
| 1. | analyzeTrends(records) | records = 115 posts de la última semana | Top 10 términos/hashtags con su frecuencia diaria | salud (45/día) | * A * R |  | |
| 2. | detectKeywords(records) | records = 115 posts | Lista de keywords con más de 10 apariciones | ["salud", "vacuna", "covid", "hospital", "prevención"] | * A * R |  | |
| 3. | applyNLP(records) | records = 115 posts | Asignar análisis de sentimiento y extracción de entidades | 70 % positivo, 20 % neutro, 10 % negativo; entidades: “OMS”, “Ministerio de Salud”, “Unicef” | * A * R |  | |
| 4. | generateReport(impactData) | impactData = salida de calculateImpact() | Informe PDF/HTML con gráficos y tablas resumen | Generado: TrendsReport\_2025-05.pdf | * A * R |  | |
| 5. | configureAlerts(threshold) | threshold = 20 % | Notificación si una tendencia sube o baja más de 20 % en 24 h | Alerta configurada correctamente; no se disparó en las últimas 24 h | * A * R |  | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **VITAL DATA** | | | | | | |
| **CÓDIGO** | | HU03 | | **FECHA** | | Mayo /2025 |
| Gestión CRUD de la Información | | | | | | |
| **PRUEBA DE INTEGRACIÓN** | | | | | | |
| **MÓDULO** | Gestión de Dato | **CLASE** | DataManagementService | **VERSIÓN** | 1.0 | |
|  |  | | | | | |
| **OBJETIVOS** | Comprobar operaciones CRUD, validación de integridad y logging de cambios. | | | | | |
| **REQUERIMIENTOS** | Conector a “HealthAnalyticsDB”  Framework ORM (e.g. Hibernate)  UI web (e.g. Angular/React) para pruebas de interfaz  Sistema de archivos para logs | | | | | |
| **PRE CONDICIONES** | Esquema de BD creado con tablas posts\_sociales y change\_log  Usuario de pruebas con permisos CRUD | | | | | |
| **DETALLES DE LA PRUEBA** | | | | | | |
| **EJECUCIÓN** |  | **FECHA DE EJECUCIÓN** |  | **ING PRUEBA** |  | |
| **OBSERVACIONES** |  | | | | | |
| **NO** | **MÉTODO** | **PARÁMETROS** | **RESULTADO ESPERADO** | **RESULTADO OBTENIDO** | **CALIFICACIÓN** | **OBSERVACIONES** |
| 1. | createRecord(post) | {idApi:"123", texto:"Prueba",fecha:"2025-05-03"} | Inserta fila y retorna postId | postId=501 (fila creada en posts\_sociales) | * A * R |  |
| 2. | readRecord(postId) | 501 | Devuelve objeto igual al insertado | {postId:501,idApi:"123",texto:"Prueba",fecha:"2025-05-03"} | * A * R |  |
| 3. | updateRecord(postId, campo, valor) | 501, "texto", "Actualizado" | Confirma “Registro actualizado” | “Registro 501 actualizado con éxito” | * A * R |  |
| 4. | deleteRecord(postId) | 501 | Confirma “Registro eliminado” | “Registro 501 eliminado con éxito” | * A * R |  |
| 5. | validateDataIntegrity() |  | Sin nulos ni FK rotas | Chequeo OK: 0 errores de integridad | * A * R |  |
| 6. | openManagementUI() |  | Muestra listado, botones CRUD | UI desplegada con 10 entradas por página, CRUD operativos | * A * R |  |
| 7. | logChange(postId, acción, usuario) | 502, "UPDATE", "tester1" | Inserta en change\_log | Log insertado: {logId:120, postId:502, acción:"UPDATE", user:"tester1", ts:2025-05-28} | * A * R |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **VITAL DATA** | | | | | | | |
| **CÓDIGO** | | HU04 | | **FECHA** | | Mayo /2025 | |
| Búsqueda y Filtrado Avanzado | | | | | | | |
| **PRUEBA DE INTEGRACIÓN** | | | | | | | |
| **MÓDULO** | Búsqueda Avanzada | **CLASE** | SearchService | **VERSIÓN** | 1.0 | **INGENIERO** |  |
|  |  | | | | | | |
| **OBJETIVOS** | Verificar filtros, rendimiento, autocompletado y paginación.  Requerimientos:  Índice en BD (e.g. Elasticsearch o índices SQL)  Librería de autocomplete (e.g. Typeahead) | | | | | | |
| **REQUERIMIENTOS** | Índice en BD (e.g. Elasticsearch o índices SQL)  Librería de autocomplete (e.g. Typeahead) | | | | | | |
| **PRE CONDICIONES** | Mínimo 1.000 registros en BD para medir performance  Índices actualizados | | | | | | |
| **DETALLES DE LA PRUEBA** | | | | | | | |
| **EJECUCIÓN** |  | **FECHA DE EJECUCIÓN** |  | **ING PRUEBA** |  | | |
| **OBSERVACIONES** |  | | | | | | |
| **NO** | **MÉTODO** | **PARÁMETROS** | **RESULTADO ESPERADO** | **RESULTADO OBTENIDO** | **CALIFICACIÓN** | **OBSERVACIONES** | |
| 1. | advancedSearch(filtros) | {keywords:["vacuna"], hashtags:["#salud"], fecha:"2025-05-01"} | Registros que cumplan todos criterios | 23 registros devueltos | * A * R |  | |
| 2. | filterByKeyword(texto) | "covid" | |  | | --- | |  |  |  | | --- | | Posts con “covid” | | 15 registros filtrados | * A * R |  | |
| 3. | filterByHashtag(hashtag) | "#salud" | |  | | --- | |  |  |  | | --- | | Posts con “#salud” | | 30 registros filtrados | * A * R |  | |
| 4. | filterByDate(rango) | {inicio:"2025-05-01", fin:"2025-05-05"} | Posts en rango de fechas | 50 registros filtrados | * A * R |  | |
| 5. | optimizeQuery(query) | query="vacuna covid salud" | Tiempo <200 ms para 1 k registros | Plan OK, ejecución en 145 ms | * A * R |  | |
| 6. | autocomplete(prefix) | "vac" | Sugerencias que empiecen con “vac” | ["vacuna", "vacunación", "vacaciones"] | * A * R |  | |
| 7. | paginateResults(page, size) | 2, 50 | Página 2 con 50 resultados | Devuelve registros **51–100** | * A * R |  | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **VITAL DATA** | | | | | | | |
| **CÓDIGO** | | HU05 | | **FECHA** | | Mayo /2025 | |
| Visualización e Interfaz de Usuario | | | | | | | |
| **PRUEBA DE INTEGRACIÓN** | | | | | | | |
| **MÓDULO** | Visualización de Datos | **CLASE** | VisualizationService | **VERSIÓN** | 1.0 | **INGENIERO** |  |
|  |  | | | | | | |
| **OBJETIVOS** | Asegurar que la UI muestre datos ordenados, exportables y accesibles. | | | | | | |
| **REQUERIMIENTOS** | Framework UI (e.g. React/Angular)  Librería de gráficos (e.g. Chart.js)  Dependencias de accesibilidad (WCAG) | | | | | | |
| **PRE CONDICIONES** | Datos filtrados disponibles (salida de HU-04)  Entorno gráfico habilitado | | | | | | |
| **DETALLES DE LA PRUEBA** | | | | | | | |
| **EJECUCIÓN** |  | **FECHA DE EJECUCIÓN** |  | **ING PRUEBA** |  | | |
| **OBSERVACIONES** |  | | | | | | |
| **NO** | **MÉTODO** | **PARÁMETROS** | **RESULTADO ESPERADO** | **RESULTADO OBTENIDO** | **CALIFICACIÓN** | **OBSERVACIONES** | |
| 1 | displayCategories(data) | 23 registros (HU-04) | Agrupa y muestra por categoría | Categorías: “Vacunas” (12), “Hospitales” (6), “Prevención” (5); UI refleja esas tres secciones | * A * R |  | |
| 2 | sortResults(data, orden) | asc | Orden cronológico ascendente | Listado ordenado de más antiguo a más nuevo | * A * R |  | |
| 3 | generateCharts(data) | métricas de relevancia | Gráficos de barras/líneas renderizados | Se muestran: | * A * R |  | |
| 4 | exportData(data, formato) | "CSV" | Descarga CSV válido | Archivo “export\_2025-05.csv” generado (1.2 MB), 23 filas | * A * R |  | |
| 5 | improveAccessibility() |  | Cumple ARIA y contraste WCAG | Validación automática: 0 errores de accesibilidad; lector de pantalla recorre en orden lógico | * A * R |  | |