Documento de Propuesta de Diseño de Software I, II y II

Desarrollo de un OVA sobre Métodos de Recolección de Información en la Investigación

Any Vanesa Monterroza Mariota
Bryan Escobar Martínez
Jonathan Pérez Blanquicet
Donaldo Doria Barrios
Tutor: Alexander Enrique Toscano Ricardo

Github: **https://github.com/area-de-informatica/ds1_pa_observatech.git**

Correos:

amonterrozamariota55@correo.unicordoba.edu.co

bfescobarmartinez@correo.unicordoba.edu.co

jperezblanquicet@correo.unicordoba.edu.co

ddoriabarrios3600@correo.unicordoba.edu.co

Descripción del software

El presente proyecto tiene como objetivo diseñar y desarrollar un Objetos Virtuales de Aprendizaje (OVA) que aborden los métodos fundamentales de recolección de información en la investigación: la observación, la entrevista y la encuesta. Estos OVAs estarán diseñados bajo la metodología **MODESEC**, garantizando que sean didácticos, interactivos y alineados con los lineamientos pedagógicos establecidos.

Etapa 1 Diseño de la Aplicación y Análisis de Requisitos

1. Introducción

Propósito del Documento

El presente documento tiene como finalidad documentar el proceso de diseño, análisis e implementación de software de tipo educativo, comercial, OVA, componente o módulo de aplicaciones. Se divide en tres etapas para facilitar el entendimiento y aplicación a gran escala en la asignatura de diseño de software.

- **Etapa 1 Diseño de la Aplicación y Análisis de Requisitos**

Esta etapa cumple la tarea de recoger todas las competencias desarrolladas en todas las áreas de formación del currículo de la licenciatura en Informática y Medios Audiovisuales y ponerlas a prueba en el diseño y análisis de un producto educativo que se base en las teorías de aprendizaje estudiadas, articule las estrategias de enseñanza con uso de TIC y genere innovaciones en educación con productos interactivos que revelen una verdadera naturaleza educativa. Estos productos deben aprovechar las fortalezas adquiridas en las áreas de tecnología e informática, técnicas y herramientas, medios audiovisuales y programación y sistemas, para generar productos software interactivos que permitan a los usuarios disfrutar de lo que aprenden, a su propio ritmo. Todo esto en el marco de un proceso metodológico(metodologías de desarrollo de software como MODESEC, SEMLI, etc.) que aproveche lo aprendido en la línea de gestión y lo enriquezca con elementos de la Ingeniería de Software.

- **Etapa 2: Persistencia de Datos con Backend – Servidor**

En la etapa 2 se continua con los lineamientos de la etapa 1, para seguir adicionando elementos de diseño e implementación de software, enfocados en el desarrollo de APIs, servidores o microservicios que permitan soportar aplicaciones cliente del software educativo; en este sentido, el curso presenta los conceptos de los sistemas de bases de datos, su diseño lógico, la organización de los sistemas manejadores de bases de datos, los lenguaje de definición de datos

y el lenguaje de manipulación de datos SQL y NoSQL; de tal manera que los estudiantes adquieran las competencias para analizar, diseñar y desarrollar aplicaciones para gestionar y almacenar grandes cantidades de datos, mediante el uso de técnicas adecuadas como el diseño y modelo lógico y físico de base datos, manejo de los sistemas de gestión de bases de datos, algebra relacional, dominio del lenguaje SQL como herramienta de consulta, tecnología cliente / servidor; igualmente, se definirán los elementos necesarios para el acceso a dichas bases de datos, como la creación del servidor API, utilizando tecnologías de vanguardia como node.js, express, Nest.js, Spring entre otros; para, finalmente converger en el despliegue de la API utilizando servicios de hospedaje en la nube, preferiblemente gratuitos. También podrá implementar servidores o API's con inteligencia artificial o en su defecto crear una nueva capa que consuma y transforme los datos obtenidos de la IA.

El desarrollo del curso se trabajará por proyectos de trabajo colaborativo que serán evaluados de múltiples maneras, teniendo en cuenta más el proceso que el resultado.

- **Etapa 3: Consumo de Datos y Desarrollo Frontend – Cliente**

La etapa 3 el estudiante está en capacidad de establecer la mejor elección de herramientas de consumo de datos y técnicas en aras de lograr el mejor producto a nivel de software o hardware acorde a los requerimientos funcionales y no funcionales del problema a solucionar. En este punto el estudiante puede consumir los datos a través de un cliente que puede ser una aplicación de celular, una aplicación de escritorio, una página web, IoT(internet de las cosas) o incluso, artefactos tecnológicos.

El diseño gráfico es de los requisitos esenciales en la capa de presentación, por lo tanto, se requieren los cursos de diseño gráfico vistos previamente. Los elementos anteriores nos permiten elegir el paradigma y tecnología para desarrollar nuestras aplicaciones, teniendo en cuenta que podríamos desarrollar aplicaciones de tipo cliente.

Alcance del Proyecto OVA sobre Métodos de Recolección de Información en la Investigación

Este módulo educativo tiene como propósito central el desarrollo de una estrategia digital interactiva que fortalezca el aprendizaje autónomo y significativo de los métodos básicos de recolección de datos (observación, entrevista, encuesta) usados en procesos investigativos escolares.

Se enfoca especialmente en estudiantes que se inician en la investigación, generalmente de básica secundaria o media, y busca resolver problemáticas frecuentes como:

- La falta de materiales didácticos atractivos y actualizados.
- La escasa interacción en clases tradicionales para adquirir habilidades prácticas.
- La inexistencia de entornos digitales adaptados al contexto académico que permitan aprender haciendo.

A través del diseño y uso de Objetos Virtuales de Aprendizaje (OVAs), se integran recursos interactivos que simulan escenarios reales y fomentan la exploración, la toma de decisiones y la reflexión activa en torno a la recolección de datos. El módulo es escalable, adaptable y pensado para uso híbrido o completamente virtual.

Funcionalidades Actuales y Futuros.

1. Registro de usuarios
2. Crear narrativa
3. Editar narrativa
4. Buscar narrativa
5. Eliminar narrativa
6. Recomendar método de recolección (Asistente)
7. Sugerir de tipos de observación
8. Generar análisis de la observación para evaluar
9. Formular entrevistas
10. Publicar en redes sociales
11. Generar análisis de muestra de entrevistas
12. Generar formularios
13. Generar análisis de muestra de encuestas
14. Generar actividades prácticas tipo caso
15. Generar retroalimentación
16. Generar evaluación diagnóstica
17. Implementar retroalimentación

18. Generar nivelación sugerida
19. Generar evaluación final
20. Visualizar módulos completados mediante line de tiempo interactiva
21. Asignar insignias gamificadas
22. Mostrar definiciones a partir de glosario interactivo
23. Generar de autoevaluación
24. Mostrar ejercicios
25. Mostrar ejemplos
26. Mostrar videos
27. Generar evaluaciones
28. Generar reporte
29. Mostrar listado de "Sabías que..."
30. Cambiar modo claro/oscuro
31. Integrar con Google Classroom
32. Integrar de plugin para Moodle.
33. Enviar notificaciones por correo
34. Activar soporte para accesibilidad

Definiciones y Acrónimos

API: Interfaz de Programación de Aplicaciones (Application Programming Interface).

DBMS: Sistema de Gestión de Bases de Datos (Database Management System).

SQL: Lenguaje de Consulta Estructurada (Structured Query Language).

HTTP: Protocolo de Transferencia de Hipertexto (Hypertext Transfer Protocol).

REST: Transferencia de Estado Representacional (Representational State Transfer).

JSON: Notación de Objetos de JavaScript (JavaScript Object Notation).

JWT: Token de Web JSON (JSON Web Token).

CRUD: Crear, Leer, Actualizar y Borrar (Create, Read, Update, Delete).

ORM: Mapeo Objeto-Relacional (Object-Relational Mapping).

MVC: Modelo-Vista-Controlador (Model-View-Controller).

API RESTful: API que sigue los principios de REST.

CI/CD: Integración Continua / Entrega Continua (Continuous Integration / Continuous Delivery).

SaaS: Software como Servicio (Software as a Service).

SSL/TLS: Capa de sockets seguros/Seguridad de la Capa de Transporte (Secure Sockets Layer/Transport Layer Security).

HTML: Lenguaje de Marcado de Hipertexto (Hypertext Markup Language).

CSS: Hojas de Estilo en Cascada (Cascading Style Sheets).

JS: JavaScript.

DOM: Modelo de Objeto del Documento (Document Object Model).

UI: Interfaz de Usuario (User Interface).

UX: Experiencia del Usuario (User Experience).

SPA: Aplicación de Página Única (Single Page Application).

AJAX: Asincrónico JavaScript y XML (Asynchronous JavaScript and XML).

CMS: Sistema de Gestión de Contenido (Content Management System).

CDN: Red de Distribución de Contenido (Content Delivery Network).

SEO: Optimización de Motores de Búsqueda (Search Engine Optimization).

IDE: Entorno de Desarrollo Integrado (Integrated Development Environment).

CLI: Interfaz de Línea de Comandos (Command Line Interface).

PWA: Aplicación Web Progresiva (Progressive Web App).

2. Descripción General

Objetivos del Sistema

Funcionalidad General

Restricciones

3. Requisitos Funcionales

Casos de Uso

Diagrama de caso de uso

Diagramas de Flujo de Casos de Uso

Descripción detallada de cada caso de uso

CASO No. 1 Crear Pizarra

ID:		
Nombre		
Actores		
Objetivo		
Urgencia		
Esfuerzo		
Pre-condiciones	-	
Flujo Normal	Docente	Sistema

Flujo alternativo 1		
Flujo alternativo 2		
Post-condiciones		
Exepciones		

Comentado [k1]: Son los estados finales con los que termina el caso de uso, estos se deben cumplir y cualquier situación que no permita cumplirlos debe quedar consignada como flujo alternativo.

Comentado [k2]: Son excepciones que no dependen del sistema o actores y que se deben resolver en otro flujo.

Prioridad de Requerimientos

4. Requisitos No Funcionales

Requisitos de Seguridad

Requisitos de Usabilidad

Requisitos de Escalabilidad

5. Modelado E/R

Diagrama de Entidad-Relación

Diagrama Relacional

Script de modelo relacional

Descripción de Entidades y Relaciones

Entidades:

Relaciones:

Reglas de Integridad Referencial

Colecciones (NoSLQ)

6. Anexos

Diagramas Adicionales

Referencias

Etapa 2: Persistencia de Datos con Backend

7. Introducción

Propósito de la Etapa

Alcance de la Etapa

Definiciones y Acrónimos

8. Diseño de la Arquitectura de Backend

Descripción de la Arquitectura Propuesta

Componentes del Backend

Diagramas de Arquitectura

9. Elección de la Base de Datos

Evaluación de Opciones (SQL o NoSQL)

Justificación de la Elección

Diseño de Esquema de Base de Datos

10. Implementación del Backend

Elección del Lenguaje de Programación

Creación de la Lógica de Negocio

Desarrollo de Endpoints y APIs

Autenticación y Autorización

11. Conexión a la Base de Datos

Configuración de la Conexión

Desarrollo de Operaciones CRUD

Manejo de Transacciones

12. Pruebas del Backend

Diseño de Casos de Prueba

Ejecución de Pruebas Unitarias y de Integración

Manejo de Errores y Excepciones

Etapa 3: Consumo de Datos y Desarrollo Frontend

13. Introducción

Propósito de la Etapa

Alcance de la Etapa

Definiciones y Acrónimos

14. Creación de la Interfaz de Usuario (UI)

Diseño de la Interfaz de Usuario (UI) con HTML y CSS

Consideraciones de Usabilidad

Maquetación Responsiva

15. Programación Frontend con JavaScript (JS)

Desarrollo de la Lógica del Frontend

Manejo de Eventos y Comportamientos Dinámicos

Uso de Bibliotecas y Frameworks (si aplicable)

16. Consumo de Datos desde el Backend

Configuración de Conexiones al Backend

Obtención y Presentación de Datos

Actualización en Tiempo Real (si aplicable)

17. Interacción Usuario-Interfaz

Manejo de Formularios y Validación de Datos

Implementación de Funcionalidades Interactivas

Mejoras en la Experiencia del Usuario

18. Pruebas y Depuración del Frontend

Diseño de Casos de Prueba de Frontend

Pruebas de Usabilidad

Depuración de Errores y Optimización del Código

19. Implementación de la Lógica de Negocio en el Frontend

Migración de la Lógica de Negocio desde el Backend (si necesario)

Validación de Datos y Reglas de Negocio en el Frontend

20. Integración con el Backend

Verificación de la Comunicación Efectiva con el Backend

Pruebas de Integración Frontend-Backend

ANEXOS

Diagramas UML

- **Diagrama de Casos de Uso (Use Case Diagram):** Este diagrama muestra las interacciones entre los actores (usuarios) y el sistema. Puede ayudar a identificar las funcionalidades clave y los actores involucrados.
- **Diagrama de Secuencia (Sequence Diagram):** Estos diagramas muestran la interacción entre objetos y actores a lo largo del tiempo. Puedes utilizarlos para representar cómo los usuarios interactúan con la pizarra en un flujo de trabajo específico.
- **Diagrama de Clases (Class Diagram):** Puedes utilizar este diagrama para modelar las clases y estructuras de datos subyacentes en el sistema, como usuarios, pizarras, comentarios, revisiones, etc.

- **Diagrama de Estados (State Diagram):** Este diagrama puede ser útil para modelar el comportamiento de la pizarra en diferentes estados, como "edición", "visualización", "comentario", etc.
- **Diagrama de Despliegue (Deployment Diagram):** Puedes utilizar este diagrama para representar cómo se despliega la aplicación en servidores y cómo interactúa con otros componentes del sistema, como el CMS.
- **Diagrama de Componentes (Component Diagram):** Este diagrama puede ayudar a representar la estructura de componentes del software, como la interfaz de usuario, la lógica de negocio, las bibliotecas y los servicios utilizados.
- **Diagrama de Actividad (Activity Diagram):** Puedes usar este diagrama para modelar flujos de trabajo o procesos específicos, como el flujo de trabajo de creación y edición de contenido en la pizarra.
- **Diagrama de Comunicación (Communication Diagram):** Similar a los diagramas de secuencia, estos diagramas muestran interacciones entre objetos y actores, pero pueden ser más simples y enfocados en la comunicación.
- **Diagrama de Paquetes (Package Diagram):** Este diagrama puede ayudar a organizar y visualizar los paquetes y módulos del software, lo que es útil para el diseño modular.
- **Diagrama de Objetos (Object Diagram):** Puedes utilizar este diagrama para representar instancias de clases y cómo interactúan en un escenario específico.