

REGULARIZACIÓN

El alumno, quien cursa la materia para obtener la regularidad debe aprobar un informe que será desarrollado durante el cursado.

Informe durante el cursado.

Durante el cursado se solicitará por parte de la cátedra un solo informe que deberá contener los siguientes puntos.

1. Presentación del grupo (Datos de contacto, datos personales, docentes de la cátedra, fecha de presentación)
2. Análisis de requerimientos e investigación de los procesos y el caso de negocio si aplicase.
3. Buscar formas de mejorar la situación actual inclusive cuando requiera cambios en los procesos que no afecten el correcto funcionamiento.
4. Justificación de Ingeniería de Software respecto a lo aprendido (Diagramas que considere necesario, que justifiquen el análisis y conocimiento del problema)
5. Detallar los módulos que se van a desarrollar. Que contengan un texto breve de su funcionalidad y agregar Diagrama de casos de uso (Fig. 1 Ejemplo de la tabla)
6. Descripción de cómo se manejara la información según le corresponda, ejemplo: SQL, noSQL y otros. Análisis coincidentes con la elección del manejo.
7. Esquema de colaboración con la institución si aplicase; como van a realizar el refinamiento de los requisitos.
8. Cómo planean su trabajo en grupo, y propuesta de distribución de responsabilidades.
9. Diagrama de gantt tentativo para el desarrollo del proyecto, apuntando a proyectos de no más de 1 año de desarrollo ni menos de 4, salvo autorización expresa de la cátedra.

Puntos del informe a redactar:

1. **Presentación del grupo:** En este punto lo que se pretende es una carátula y los datos personales de los integrantes del grupo.
2. **Contexto del problema a sistematizar o idea a desarrollar:** Se busca conocer el contexto del problema o innovación que realiza. En el caso que se busque sistematizar un proceso de una empresa X conocer brevemente cómo se trabaja hoy en día con ese circuito / proceso. Y cómo lo van a resolver. En el caso de hacer un desarrollo de una tecnología / Sistema / Software, el contenido de este apartado debe reflejar información de una investigación mínima del campo a innovar y conocimientos previos de desarrollos similares.
3. **Justificación de Ingeniería de Software respecto a lo aprendido** (Diagramas que considere necesario, que justifiquen el análisis y conocimiento del problema) : En este apartado se busca tener todo los diagramas de análisis de requerimientos necesarios para el desarrollo planteado, pero deben justificar la elección de forma inteligente luego de investigar las opciones.
4. **Detallar los módulos que se van a desarrollar.** Que contengan un texto breve de su funcionalidad y agregar Diagrama de casos de uso (Fig. 1 Ejemplo de la tabla: En este apartado se busca tener un listado con una descripción breve de cada módulo a desarrollar, casos de uso y tabla de flujos (Fig. 2 Ejemplo de la tabla).
5. **Diagrama de modelado de datos (Relacional o NoSql):** Se espera un análisis de la forma en que se van a guardar los datos de la aplicación. Si es relacional un diagrama de la base de datos que van a crear que solo impacte lo que corresponde a lo datos de la aplicación. Dejar afuera manejo de usuarios y tablas creadas por frameworks. Si es NoSql un diagrama que explique el tipo de base de datos NoSql que se va a usar, en otro caso, documentar según corresponda en las buenas prácticas.

1. Presentación del Grupo

Esta sección sirve para introducir a los miembros del equipo y los detalles básicos del proyecto. Es la primera impresión, así que debe ser clara y completa.

- **Integrantes:** Liste el nombre completo de cada miembro del equipo, seguido de su información de contacto. Esto es crucial para la comunicación.
 - Ejemplo:
 - Juan Pérez - juan.perez@email.com
 - Ana García - ana.garcia@email.com

- Docentes de la Cátedra: Incluya los nombres de los profesores a cargo para demostrar que el equipo conoce el contexto académico.
- Fecha de Presentación: Especifique la fecha en que se expone el trabajo.

2. Análisis de Requerimientos e Investigación del Caso de Negocio

Aquí deben demostrar que han entendido a fondo el problema que van a resolver. Es el fundamento de todo el proyecto.


- Caso de Negocio: Describir el problema central o la oportunidad que su proyecto busca abordar. Expliquen por qué es importante para la organización o los usuarios.
 - Ejemplo: *La gestión de citas en la clínica X se realiza de forma manual, lo que provoca una alta tasa de errores en los horarios y un registro ineficiente de los pacientes. Esto causa insatisfacción en los clientes y una pérdida de tiempo para el personal administrativo.*
- Procesos Actuales: Detallen paso a paso cómo se realiza la tarea actualmente. Identifiquen las deficiencias, los cuellos de botella y los puntos débiles.
- Requerimientos: Con base en la investigación, definan los requerimientos funcionales (lo que el sistema debe hacer) y no funcionales (cómo debe funcionar, por ejemplo, ser rápido, seguro, fácil de usar).

3. Búsqueda de Formas de Mejorar la Situación Actual

En esta sección, se espera que presenten su visión de la solución. No se limiten a la tecnología; propongan cambios en los procesos de trabajo para maximizar el impacto de su proyecto.

- Propuesta de Mejora: Describen cómo su solución tecnológica (el software) va a resolver los problemas identificados. Expliquen los cambios en los procesos que el software facilitará.
 - Ejemplo: *Proponemos un sistema de agendamiento de citas online que permite a los pacientes reservar su turno. El personal de recepción ya no gestionará las citas por teléfono, sino que supervisará el sistema, validará los turnos y enviará recordatorios automáticos por correo electrónico.*
- Justificación de las mejoras: Expliquen los beneficios concretos de estos cambios.

4. Justificación de Ingeniería de Software



Esta parte es clave para demostrar su conocimiento en la materia. Utilicen diagramas UML para modelar y justificar su análisis y diseño.

- Propósito: Expliquen que los diagramas sirven para visualizar la estructura y el comportamiento del sistema antes de escribir código, asegurando un diseño sólido.
- Diagramas a Incluir: Elijan los diagramas más relevantes para su proyecto y expliquen brevemente qué muestra cada uno.
 - Diagrama de Casos de Uso: Muestra la interacción entre los usuarios (actores) y el sistema.
 - Diagrama de Clases: Define la estructura de las entidades del sistema (clases, atributos, relaciones).
 - Diagrama de Secuencia: Detalla el orden de las interacciones entre los objetos del sistema para una funcionalidad específica.
 - [Aquí pueden insertar sus propios diagramas]

5. Detalle de Módulos.

Dividan su software en módulos lógicos. Esto hace que el proyecto sea más manejable y fácil de entender.

- Definición de Módulos: Describan cada módulo con un nombre y un breve texto sobre su funcionalidad principal.

6. Manejo de la Información.

Justifiquen la elección de la tecnología de base de datos. Esto demuestra un análisis de las necesidades de datos del proyecto.

- SQL (Bases de Datos Relacionales): Elijan esta opción si sus datos tienen una estructura bien definida y las relaciones entre ellos son importantes.
 - Ejemplo: *Optamos por una base de datos SQL como PostgreSQL, ya que la información de pacientes, citas y médicos tiene una estructura rígida y es crucial mantener la integridad referencial.*
- NoSQL (Bases de Datos No Relacionales): Elijan esta opción si los datos son dinámicos, semi-estructurados, o si necesitan una alta escalabilidad y flexibilidad.
 - Ejemplo: *Utilizaremos una base de datos NoSQL como MongoDB, ya que manejaremos un gran volumen de datos de sensores IoT que no tienen un*

esquema fijo y necesitamos flexibilidad para agregar nuevos tipos de datos en el futuro.

7. Esquema de Colaboración con la Institución.

Describan cómo mantendrán una comunicación constante con el cliente o la institución para asegurar que el proyecto se alinee con sus expectativas.

- Refinamiento de Requisitos: Expliquen los métodos que usarán para validar y refinar los requerimientos.
 - Ejemplo: *Realizaremos reuniones semanales con el personal de la clínica para presentar avances y recopilar retroalimentación. Usaremos prototipos de baja fidelidad para validar la interfaz y asegurarnos de que el flujo de trabajo es intuitivo para los usuarios finales.*

8. Plan de Trabajo en Grupo y Responsabilidades.

Muestren cómo se organizarán para trabajar de manera eficiente.

- Metodología de Trabajo: Mencionen si usarán una metodología de colaboración, puede ser metodología vista en clase o un esquema que permita el desarrollo de una forma razonable.
- Distribución de Responsabilidades: Asignen roles y responsabilidades claras a cada miembro del equipo.
 - Ejemplo:
 - Project Manager: Juan Pérez (Encargado de la planificación y comunicación)
 - Desarrollador Backend: Ana García (Encargada de la lógica del servidor y la base de datos)
 - Desarrollador Frontend: Carlos López (Encargado de la interfaz de usuario)

Diagrama de Gantt Tentativo.

Este diagrama es fundamental para visualizar la planificación del proyecto en el tiempo. Muestra las fases, tareas y sus plazos.

- Propósito: Expliquen que el Gantt es una herramienta para planificar, coordinar y dar seguimiento al progreso.

- Plantilla de Gantt: Utilicen esta tabla para crear un cronograma de 4 a 12 meses.

Entregas

Durante el cursado se realizará la entrega del informe, que tendrá a lo sumo, dos revisiones por parte de la cátedra. Si las correcciones que se solicitan no se cumplen, el alumno puede desaprobar el informe y necesitará recurrir la materia.

En caso de tener este informe aprobado antes de finalizar el cuatrimestre, se procederá a que el equipo inicie con el desarrollo de la aplicación, para luego ser evaluado por el equipo en instancias finales.

El informe aprobado será firmado por el docente evaluador, siendo esto suficiente sin intervención de otra evaluación externa.

En caso de disolverse el grupo, la situación quedará sujeta a lo que los docentes resuelvan al momento, pudiendo realizar sugerencias para continuar el cursado. Esta situación excepcional resulta crítica cuando ocurre cerca del cierre, pudiendo generar una condición de libre en caso de desacuerdos de originalidad en la idea; la decisión final de la condición del alumno en esta situación será exclusivamente dictaminada por el equipo docente.

Evaluación

Una vez aprobado el informe requerido por la cátedra el alumno deberá presentarse con al menos uno de los dos docentes de la cátedra para presentar el software desarrollado. El mismo debe estar finalizado, salvo excepciones se aceptan trabajos no finalizados, cito: 95% o similar. Esto quiere decir que ya debe tener todos los módulos desarrollados y listo para mostrar al docente evaluador el cual realizará un dictamen en el cual aclara los puntos evaluados y en el caso de que no se cumpla uno dejará constancia del porque no cumplió el alumno con dicho requisito, además de tener un diálogo más detallado de lo que se espera.

La evaluación se realizará presencial en el edificio de la sede central Oro Verde, en caso que el docente o el/los alumnos no puedan asistir presencialmente por una situación extraordinaria los docentes podrían invitar virtualmente, en caso excepcionales. En las revisiones, todos los miembros del grupo deben estar, caso contrario no serán



evaluados.

Las evaluaciones que no sean aprobadas, serán sujetas a una nueva revisión, destacando los puntos faltantes para su aprobación mediante dictamen fundado. La decisión de alguno de los miembros de la cátedra, es suficiente para aprobar/desaprobar el examen ; si bien el alumno puede dejar sugerencias en el dictamen al momento evaluativo, la decisión del docente será inapelable.

Docente Evaluadores

1. Ing. IA Diaz Santiago R. (diaz.santiago@uader.edu.ar)
2. An. Rios Nuñez Marcos Elias (rios.marcos@uader.edu.ar)

