Obligatorio 2 de Diseño de Aplicaciones

Para la segunda entrega se debe tener en cuenta los siguientes cambios o agregados a la letra de la primera entrega:

Mantenimiento de autores: Se desea implementar alta, baja, modificación y listado de autores. De los mismos se conoce los siguientes campos obligatorios:

- Nombre de usuario: alfanumérico de hasta 10 caracteres.
- Nombre y Apellido: cada uno hasta 15 caracteres alfabéticos.
- Fecha de nacimiento: entre 13 y 100 años al momento del registro.

Autores de las frases: A partir de ahora todas las frases registradas en el sistema deberán registrar su autor, seleccionandolo de una lista.

Nueva alarma y forma de procesar las frases: Además de la alarma ya implementada, se desea experimentar con un nuevo tipo de alarma centrada en los autores de las frases. Estas nuevas alarmas se configuran con los siguientes parámetros:

- Tiempo en horas o días: se ingresa un número positivo mayor que cero y se selecciona las unidades.
- Tipo de frases: Se selecciona una de las opciones (positiva/negativa).
- Cantidad de mensajes: número entero positivo mayor que cero y menor que mil.

Ejemplo de procesamiento de las frases y funcionamiento de estas nuevas alarmas: si la alarma es para 5 días, frases negativas y 10 frases, la misma se activa cuando en los últimos 5 días, uno o varios autores han posteado 10 o más frases negativas, debiendo indicar todos los autores que cumplen la condición en el reporte de alarmas.

Reporte de autores: Se desea listar todos los autores ingresados en el sistema, ordenados en forma ascendente o descendente por alguno de los siguientes criterios que el usuario podrá seleccionar:

a. Porcentaje de frases positivas o negativas sobre el total de frases que generó cada uno de los autores. El número se obtiene sumando todas las frases positivas o negativas generadas por el autor, dividido entre la cantidad de frases totales del autor.

b. Cantidad de entidades mencionadas en las frases de cada autor. Este es el número de entidades distintas que el autor ha nombrado en todas sus frases.

c. Promedio diario de frases de cada autor. El número se obtiene dividiendo el total de frases del autor, entre la cantidad de días que pasaron desde la primera frase que aparece en el sistema (la que tiene la fecha más antigua, independientemente de la fecha de creación en el sistema).

El formato del reporte consistirá en una tabla con las siguientes columnas: nombre de usuario, nombre y apellido, el valor del campo elegido por el usuario para ordenar los

autores (porcentaje de frases positivas/negativas; cantidad de entidades mencionadas; o promedio diario de frases).

Persistencia de los datos: En esta nueva versión, la empresa requiere que todos los datos del sistema sean persistidos en una base de datos. De esta manera, la siguiente vez que se ejecute la aplicación se comenzará con dichos datos cargados con el último estado guardado antes de cerrar la aplicación. Los datos se deben ir guardando en la base de datos durante la operativa de la solución, a medida que el usuario realiza las acciones se debe ir almacenando, y no con un botón que salve todo en un bloque o al cerrar la aplicación.

Requerimiento grupos de 3 personas: Además de presentar el reporte de autores en formato tabular como se describe arriba, deberán crear una gráfica de barras para los 10 primeros autores de la lista. Dicha gráfica tendrá en el eje de las abscisas los autores en el mismo orden que aparecen en la lista y en las ordenadas los valores de porcentaje de frases positivas/negativas; cantidad de entidades mencionadas; o promedio diario de frases.

Aplicación a entregar

Se debe entregar una aplicación que contemple toda la funcionalidad descrita en este documento. Se espera que la aplicación se entregue con una base de datos con datos de prueba, de manera de poder comenzar las pruebas sin tener que definir una cantidad de datos iniciales. Por este motivo, la fuente de datos de prueba no debe estar embebida en el código.

Considerar que los datos de prueba entregados deben ser completos y permitir ver reportes no triviales. Se espera que la base de datos incluya como mínimo:

- 20 autores
- 15 entidades
- 10 sentimientos
- 2 alarmas de cada tipo
- 200 frases con fecha desde el 25/06/2029 al 25/06/2020, incluyendo todos los tipos de frases disponibles (positivas/negativas/neutras y grados, si aplica).

Instalación: El costo de instalación de la aplicación debe de ser mínimo y documentado adecuadamente.

NOTA: La totalidad y detalle de los requisitos serán relevados a partir de consultas en el foro correspondiente en aulas, dichas consultas deben tener un título descriptivo de la consulta. Para evitar complejidades innecesarias se realizaron simplificaciones al dominio del problema real.

Persistencia en base de datos

Toda la información contenida en el sistema debe ser persistida en una base de datos Microsoft SQL Server Express 2014. El diseño debe contemplar el modelado de una solución de persistencia adecuada para el problema utilizando Entity Framework (Code First).

Se espera que como parte de la entrega se incluya dos respaldos de la base de datos: uno vacío y otro con datos de prueba. Se recomienda entregar el archivo .bak y también el script .sql para ambas base de datos.

Es condición necesaria para obtener el puntaje mínimo del obligatorio que al menos una entidad del sistema pueda ser persistida.

Mantenibilidad (se mantienen los requerimientos de la primera entrega)

La propia empresa eventualmente hará cambios sobre el sistema, por lo que se requiera un alto grado de mantenibilidad, flexibilidad, calidad, claridad del código y documentación adecuada. Por lo que el desarrollo de todo el obligatorio debe cumplir:

➔ Estar en un repositorio Git.
➔ Haber sido escrito utilizando TDD (desarrollo guiado por pruebas) lo que involucra otras dos prácticas: escribir las pruebas primero (Test First Development) y refactoring. De esta forma se utilizan las pruebas unitarias para dirigir el diseño.
➔ Cumplir los lineamientos de Clean Code (capítulos 1 al 10, y el 12), utilizando las técnicas y metodologías ágiles presentadas para crear código limpio.
➔ Favorecer a los principios y patrones de diseño vistos en el curso (entre ellos SOLID, GRASP, etc).

Pruebas unitarias (se mantienen los requerimientos de la primera entrega)

Se requiere escribir los casos de prueba automatizados con el framework de unit tests visto en el curso (MSTest), documentando y justificando las pruebas realizadas.

Como parte de la evaluación se va a revisar el nivel de cobertura de los test sobre el código entregado, por lo que se debe entregar un reporte y un análisis de la cobertura de las

pruebas justificando cada uno de los valores obtenidos de acuerdo al diseño elaborado para la solución.

Control de versiones (se mantienen los requerimientos de la primera entrega)

La gestión del código del obligatorio debe realizarse utilizando el mismo repositorio Git de GitHub (github.com) perteneciente a la organización de la cátedra ORT-DA1 (github.com/ORT-DA1), apoyándose en el flujo de trabajo recomendado GitFlow (nvie.com/posts/a-successful-git-branching-model).

Como clientes visuales desktop para el manejo del repositorio, pueden utilizar:

➔ SourceTree (www.atlassian.com/software/sourcetree) el cual incorpora GitFlow (www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow).
➔ Git Bash (https://git-scm.com/downloads).
➔ Visual Studio with Git (msdn.microsoft.com/es-es/library/hh850437.aspx).

Documentación

La documentación entregada debe ser en un solo documento digital (límite 20 páginas), que contenga la siguiente información ordenada e indexada:

1. **Descripción general del trabajo** y del sistema: si alguna funcionalidad no fue implementada o si hay algún error conocido (bug) deben ser descritos aquí.

2. **Descripción y justificación de diseño**. Para ello se debe incluir:
    a. Diagrama(s) de paquetes mostrando la organización general de la aplicación.
    b. Diagramas de clases: al menos uno por paquete. Los diagramas deberán ser lo más completos y formales posible respecto a lo que se desea comunicar.
    c. Diagramas de interacción para especificar los principales comportamientos del sistema.
    d. Modelo de tablas de la estructura de la base de datos.
    e. Explicación los mecanismos generales y descripción de las principales decisiones de diseño tomadas.

3. **Cobertura de pruebas unitarias** con su debido análisis y justificaciones. (De entregar evidencia la misma puede ir en anexos por fuera del límite de páginas)

4. Se debe **actualizar todos los diagramas que corresponda**, y agregar los que se considere necesario para mostrar los cambios realizados y justificar las decisiones tomadas.

Para esta segunda entrega es necesario actualizar la documentación de diseño de la primera entrega.

Importante: se espera que la descripción y justificaciones sigan un orden lógico, intercalando diagramas y explicaciones según sea necesario. Las condiciones de entrega serán evaluadas como si se le estuviese entregando a un cliente real: prolijidad, claridad, profesionalismo, etc.

Entregables

VER AL FINAL DEL DOCUMENTO "RECORDATORIO: IMPORTANTE PARA LA ENTREGA"

En el sistema de gestión: se deberá entregar un .zip que incluya:

- Código fuente de la aplicación, incluyendo el proyecto que permita compilar, probar, ejecutar y ejecutar pruebas unitarias. No se deben entregar carpetas de bin, obj, debug, etc.
- Carpeta con la aplicación compilada en release.
- Documentación digital en formato PDF (incluyendo modelado UML).
- Base de datos:
    - Entregar una base de datos vacía y otra con datos de prueba.
    - Scripts de generación de tablas y/o datos.
- En resumen: entregar el último commit subido a la rama master del repositorio

En Github: El commit final del obligatorio debe estar claramente identificado (mediante un tag con el texto "entrega_2") en la rama master en el repositorio del grupo de GitHub dentro de la organización ORT-DA1. Debe incluir:

- Código fuente de la aplicación, incluyendo el proyecto que permita compilar, probar, ejecutar y ejecutar pruebas unitarias.
- Carpeta con la aplicación compilada en release.
- Documentación digital en formato PDF (incluyendo modelado UML).
- Base de datos:
    - Entregar una base de datos vacía y otra con datos de prueba.
    - Scripts de generación de tablas y/o datos.
- No se aceptará que a master se suba un archivo comprimido con la entrega final (ej: Zip, Rar, otros), el contenido de master deberá ser la integración de la rama develop a la master.

Evaluación (30 puntos)

	Evaluación de	Pts.	Criterios de corrección
Funcionalidad	Implementación de la funcionalidad pedida y calidad de la interfaz de usuario.	7	**Excelente:** Se implementa toda la funcionalidad, tiene buena usabilidad y no hay errores importantes de funcionamiento. **Correcto:** Falta algún punto no central de la funcionalidad, existen detalles no bloqueantes de funcionamiento o la aplicación no es fácil de usar. **No suficiente:** Faltan partes importantes (por alcance o dificultad) de la funcionalidad. Existe gran cantidad de errores o funcionalidades no usables.
Diseño y Documentación	Diagramas de clases Diagramas de interacción Justificación del diseño Uso adecuado y justificado de patrones de diseño Diagramas y justificación de la solución de persistencia. Modelo de tablas. Descripción de cambios importantes sobre el diseño de la primera entrega. Calidad del diseño Informe de cobertura de los casos de prueba Claridad de la documentación Organización de la documentación (debe tener un orden lógico y un índice) Completitud de la documentación Prolijidad de la documentación	5	**Excelente**: Se presenta un documento completo en función de lo que se pide, ordenado y prolijo, que explica la solución entregada en todos su aspectos. Se utiliza la notación vista en clase, de manera formal y correcta, para presentar los aspectos importantes del diseño. Se intercalan diagramas y explicaciones, que permiten comprender el diseño de la solución, las decisiones tomadas y la motivación detrás de las mismas. **Correcto**: Se presenta un documento prolijo que describe el diseño de la solución. Se describen los aspectos más importantes, pero hay errores en la nomenclatura. Falta describir aspectos importantes. El orden de la documentación no permite seguir un hilo descriptivo. **No suficiente**: Falta detallar parte importante de la solución. No se usa la notación vista en clase, o se usa en forma equivocada o incompleta en repetidas ocasiones. Se presenta la documentación como una sucesión de diagramas sin explicación.
Código	Desarrollo guiado por las pruebas (TDD) y técnicas de refactoreo de código Buenas prácticas de estilo y codificación y su impacto en la mantenibilidad (Clean Code) Correcto uso de las tecnologías Claridad del código Concordancia con el diseño Implementación de acceso a la base de datos	8	**Excelente:** El código es prolijo, fácil de entender y cumple con los puntos cubiertos en el curso sobre Clean Code. Lo construido coincide con lo detallado en la documentación de diseño. Se evidencia el uso de TDD mediante los commits en el repositorio. Hay buena cobertura de casos de prueba. El acceso a la base de datos está bien implementado y se evitan las operaciones con grandes volúmenes de datos en memoria.

			Correcto: El código es prolijo en general, aunque presenta detalles a mejorar. Lo construido se corresponde en términos generales con lo presentado en el documento. Hay pruebas unitarias, aunque no se evidencie el uso de TDD. **No suficiente:** El código es desprolijo o difícil de entender. No cumple en repetidos casos lo propuesto por Clean Code. No hay pruebas unitarias.

Defensa

La defensa del trabajo intenta:

➔ Evaluar el conocimiento general de los integrantes del grupo sobre la solución propuesta. Todos los integrantes deben conocer toda la solución.

➔ Verificar el aporte individual al trabajo por parte de cada uno de los integrantes del equipo y en función de los resultados, se podrán otorgar distintas notas a los integrantes del grupo.

Se espera que cada uno de los integrantes haya participado en la codificación de parte significativa del obligatorio. El mecanismo de defensa se determinará al momento de la entrega pudiendo ser escrita o en el laboratorio.

NOTA: El incorrecto funcionamiento de la instalación puede significar la no corrección de la funcionalidad. En el caso de defensa en el laboratorio, durante la defensa cada grupo contará con 15 minutos para la instalación de la aplicación. Luego de transcurridos los mismos se restan puntos al trabajo.

Información Importante

Lectura de obligatorio: 21-05-2020
Plazo máximo de entrega: 25-06-2020
Defensa: A definir por el docente
Puntaje mínimo / máximo: 12 / 30 puntos

LA ENTREGA SE REALIZA EN FORMA ONLINE EN ARCHIVO NO MAYOR A 40MB EN FORMATO ZIP, RAR O PDF.

IMPORTANTE:

- Inscribirse.
- Formar grupos de hasta dos personas.
- Subir el trabajo a Gestión antes de la hora indicada, ver hoja al final del documento: “RECORDATORIO”.

RECORDATORIO: IMPORTANTE PARA LA ENTREGA

➢ Obligatorios (Cap.IV.1, Doc. 220)

La entrega de los obligatorios será en formato digital online, a excepción de algunas materias que se entregarán en Bedelía y en ese caso recibirá información específica en el dictado de la misma.

Los principales aspectos a destacar sobre la **entrega online de obligatorios** son:

1. La entrega se realizará desde gestion.ort.edu.uy

2. Previo a la conformación de grupos cada estudiante deberá estar inscripto a la evaluación. **Sugerimos realizarlo con anticipación.**

3. **Uno de los integrantes del grupo de obligatorio será el administrador del mismo** y es quien formará el equipo y subirá la entrega

4. Cada equipo debe entregar **un único archivo en formato zip o rar** (los documentos de texto deben ser pdf, y deben ir dentro del zip o rar)

5. El archivo a subir debe tener **un tamaño máximo de 40mb**

6. Les sugerimos **realicen una 'prueba de subida' al menos un día antes**, donde conformarán el 'grupo de obligatorio'.

7. **La hora tope para subir el archivo será las 21:00** del día fijado para la entrega.

8. La entrega se podrá realizar desde cualquier lugar (ej. hogar del estudiante, laboratorios de la Universidad, etc)

9. Aquellos de ustedes que presenten alguna dificultad con su inscripción o tengan inconvenientes técnicos, por favor pasar por la oficina del Coordinador o por Coordinación adjunta antes de las 20:00hs. del día de la entrega.

Si tuvieras una situación particular de fuerza mayor, debes dirigirte con suficiente antelación al plazo de entrega, al Coordinador de Cursos o Secretario Docente.