IIC2173 2024-2
Interrogación II

Nombre:

Número de alumno:

Fecha:

Instrucciones

- Tiene **2 horas y 5 minutos** para completar este examen.
- Tendrán **15 min** para escanear las hojas de este examen y subirlas a Canvas en la tarea I2. **NO** habrá excepciones a este plazo de escaneo y subida. Posteriormente, entregue sus hojas al profesor/ayudante para su archivado.
- Recuerde nombrar todas sus hojas.
- Solo se responderán preguntas realizadas en la sala, en los momentos destinados para ello.
- La prueba es estrictamente individual. No copie. La idea es tratar de evaluar los conceptos y que aprenda los principales contenidos del curso.
- Este curso se adscribe a la política de integridad académica del DCC, que se encuentra en el programa del curso. Esperamos que respeten la política de integridad académica, y los exámenes serán comparados con diversos métodos para revisar casos de copia.
- Puede usar las hojas que necesite para completar la interrogación, pero deben ser precisos en sus respuestas para facilitar el entender qué quieren decir.

"Cuando todo parece ir en tu contra, recuerda que el avión despega contra el viento, no a favor."

(Henry Ford, empresario)

1) Preguntas de Desarrollo (20p)

Escriba sus respuestas en el espacio, máximo 8 líneas en preguntas a y b. De escribir más, solo se corregirán hasta las líneas permitidas.

1.a) (10p) Asuma que tiene un sistema distribuido con distintos servicios replicados horizontalmente. Para cada uno de estos servicios y réplicas, implementó un flujo de caché para facilitar las respuestas repetidas hacía sus usuarios, pero con el paso del tiempo se dio cuenta que muchas veces, a estos usuarios, les llegan datos antiguos o caducados por problemas del tiempo que se demora la sincronización entre los distintos servidores, generando problemas de inconsistencia luego cuando quiere actualizar los datos.

Indique una solución que podría aplicar para este problema. Para esto debe dar un ejemplo inicial donde se observe el problema, luego identificar y justificar su solución, y finalmente indicar otro ejemplo usando su solución para mostrar que sería correcta.

> ***Respuesta:*** *Se espera que los alumnos sean capaces de poder pensar en un caso donde ocurra este problema de consistencia debido al uso de caché (puede estar afectando solo un servicio o todos), para luego trabajar sobre ese ejemplo en particular.*
>
> - *3 ptos por nombrar un caso correctamente*
> - *3 ptos por indicar una solución percibible correctamente para su caso (o para el problema en general)*
> - *3 ptos por justificar correctamente la solución*
> - *1 ptos por ejemplificar como su solución resuelve su problema planteado (u otro con el problema de la consistencia)*

1.b) (10p) Comente las diferencias entre autorizar usuarios mediante el uso de cookies y JWTs en headers. ¿Cuáles son las diferencias a nivel de performance, escalabilidad y seguridad de implementar uno u otro método? ¿Cómo cambia el escenario en comunicaciones M2M?

> ***Respuesta:*** *Se aceptan dos posibles soluciones dependiendo de cómo se tome el problema:*
>
> - *Caso donde se considera cookies y JWT como guardar datos sin encriptar vs encriptados:*
> - *Performance: Cookies*
> - *Escalabilidad: Cookies*
> - *Seguridad: JWT*
> - *M2M: Si la seguridad es importante, para evitar ataques MITM se puede usar JWT, en otro caso se puede usar Cookie*
> - *Caso donde se usa una cookie de sesión y un JWT (JWT tiene datos en su token, pero cookies debe buscar los datos el backend):*
> - *Performance: JWT*
> - *Escalabilidad: JWT*
> - *Seguridad: En general JWT permite firma pública y mediante JWK validar información, y establecer no repudio entre sistemas, por lo que es mejor ya que evita modificaciones del token. Si se indica la seguridad solo en la información Cookies es mejor ya que no expone los datos.*
> - *M2M: Si no hay problema con transparentar la información es mejor JWT, en otro caso se puede asumir que no habría espera al usar cookies*

Los puntajes se distribuyen por 3 ptos por Performance, Escalabilidad y Seguridad; 1 pto por M2M

2) Preguntas de Alternativas (6p)

Escoja la alternativa correcta para cada pregunta.

2.a) (2p) Indique la respuesta correcta con respecto a una arquitectura monolítica

a) En un monolito la seguridad es lo suficientemente alta por si sola como para tener que implementar otras medidas
b) Al tener sus funciones dentro de un mismo sistema, se ve un incremento de la performance en comparación a si se distribuyeran en la red
c) Es sumamente escalable, ya que puedo escalar verticalmente tanto como sea necesario
d) Esta arquitectura no me sirve si necesitara mejorar la integración con otros sistemas externos

2.b) (2p) Indique la respuesta correcta con respecto a los problemas de seguridad vistos en clases:

a) Los troyanos son virus que son difíciles de sacar porque se replican a sí mismos en las máquinas que infectan sin intervención humana
b) Una fuente de vulnerabilidad a nivel de código sería no validar mis inputs
c) Los sistemas no son 100% seguros debido únicamente a que es común usar malas prácticas en seguridad
d) Podemos defendernos de todos los tipos de phishing implementando múltiples formas de autenticación (MFA)

2.c) (2p) Cuál(es) de las siguientes afirmaciones es(son) verdadera(s):

I. El caché me puede servir solo para mejorar la performance
II. El caché se puede encontrar en distintas capas de las aplicaciones dependiendo de lo que se quiera guardar
III. Un caché en el lado del cliente sólo me podrá guardar los archivos HTML, CSS, JS e imágenes de las páginas Web

a) Solo I
b) Solo II
c) I y III
d) Todas son correctas

3) Preguntas de Resolución de Casos (34p)

***Escoja uno de los dos casos disponibles**, y responda sus preguntas a, b y c. Luego, responda las preguntas comunes d,e, f y g con el caso que escogió. Hacer dos preguntas **no** tiene bonus disponible y solo se corregirá la que indique, si intenta hacer las dos. **Si no indica alguna, no se considerará respondida**.*

Caso I

Dado sus grandes avances y aprendizajes dentro del curso de Arquitectura de Sistemas de Software, son contactados por nuevos socios que les presentan una idea para que sean fundadores de una nueva aplicación de noticias llamada *News Hunter Unlimited.* Y como están en etapas muy tempranas les piden idear los diseños iniciales en base a los requisitos e ideas que han ido encontrando con analistas que financiarán el proyecto.

En primera instancia, se quiere crear la plataforma para que distintos caza-noticias de todo el mundo (personas naturales) puedan ir subiendo noticias y datos a la aplicación, para que luego de ser revisados por grupos de reporteros y otros sistemas se puedan mostrar aquellas noticias reales a los usuarios de la plataforma según su zona geográfica.

Por un lado, los caza-noticias serán personas certificadas por la plataforma, por lo que tendrán un usuario y contraseña con los que podrán acceder a una sección del sistema para subir noticias de forma ordenada que vayan encontrando, y así poder guardarlas en formato JSON. Luego de esto las solicitudes de noticias subidas deberán ser filtradas y procesadas aprovechando su formato, donde primeramente se revisará que no se haya subido previamente para que luego se guardará para ser revisado manualmente por un validador, dejando una tarea para iniciar ese proceso. Se espera que esta parte del sistema reciba muchas noticias constantemente de distintas partes del mundo, además de que no se quiere que personas no certificadas puedan subir estas solicitudes para no sobrecargar el sistema.

Aparte de esto, los validadores serán personas que trabajan para *News Hunter Unlimited* que, entrando en un portal especial del sistema, deberán de ser capaces de ver las noticias cercanas a su zona geográfica que se han subido, y con apoyo de una IA que les filtrarán las noticias bien redactadas de las que no lo están, deberán verificar su validez manualmente para poder aprobarlas. Para esto, cuando el módulo anterior suba una solicitud, dejará una tarea de revisión para que un sistema de workers la tome y ejecute la revisión con la IA para descartar los casos que no serán revisados. Una vez se aprueba una solicitud de noticia, esta deberá ser guardada en una base de datos relacional escalable para guardar tanto su metadata como su información. Se espera que esta parte de la aplicación no permita accesos a usuarios sin los permisos necesarios.

Finalmente, los lectores de la aplicación (usuarios normales sin necesidad de iniciar sesión) deberán poder leer las noticias aprobadas de su zona geográfica rápidamente.

3.1.a) (4p) La aplicación en general debe guardar muchos datos en distintas partes. Por cada sección de cada tipo de usuario indique el tipo principal de base de datos que usaría y justifique su decisión (una por cada una). En particular para cada sección indique: nombre del tipo de base de datos, los datos que guarda, y el por qué es adecuado.

Respuesta:

- *Sección Caza-Noticias:*
  - *Guarda archivos JSON.*
  - *Base de datos de documentos para guardar en formato JSON.*
- *Sección Validadores:*
  - *Guarda archivos JSON u otro formato. Esta base de datos que soporte actualizaciones de las IAs.*
  - *Puede ser la misma que para la sección Caza-Noticias si lo especifican.*
  - *Posibles: Relacional, Documentos u otra bien justificada.*
- *Sección Lectores:*
  - *Guarda noticias con su metadata para filtros de búsqueda.*
  - *Relacional, se pide del enunciado.*

3.1.b) (4p) Los usuarios lectores deben ser capaces de ver sus noticias rápidamente según su zona geográfica. ¿Qué herramientas de arquitectura vistas en clases podría usar para asegurar esto? Indique una única elección y justifique el por qué sería ideal y en qué parte/s del sistema se encontraría/n.

Respuesta: *Se tomará como válido cualquier sistema que pueda generar una consistencia a la vez que rapidez a nivel de zona geográfica. En particular se espera el uso de caché mediante un proxy que permita detectar la zona geográfica, el uso de un CDN, o un componente que se justifique que claramente solo se encarga de entregar datos por zona. Iría en la sección de lectores entre los usuarios y el servidor.*

- *1 pto por nombrar una solución que se vea que pueda ayudar a resolver el problema*
- *2 ptos por justificar la solución correctamente*
- *1 pto por indicar la parte del sistema en donde lo implementarían*

3.1.c) (4p) Como se esperan muchas noticias por segundo mundialmente de los distintos caza-noticias (alrededor de 100GB por minuto). Indique que solución o soluciones podría implementar en su sistema para asegurar poder aceptarlas y guardarlas. En particular, debe indicar lo que usaría, en dónde lo implementaría y el por qué es una solución correcta.

Respuesta: *Se espera el uso de partitioning o sharding (en particular sharding) junto a una replicación por zona del(de los) servicio(s) de la sección de Caza-Noticias. Se tendría alta cantidad de writes y pocos reads, por lo que un Leader Follower no serviría.*

- *1 pto por indicar la solución que se vea que pueda ayudar a resolver el problema*
- *1 pto por indicar donde se implementaría en el sistema correctamente*
- *2 ptos por justificar la solución correctamente*

Caso II: Ocumore

A medida que el progreso humano avanza a pasos veloces, la cantidad de conocimiento nuevo a transmitir aumenta sustancialmente. Esto se refleja en el conocimiento estructurado disponible en internet accesible ya sea de forma abierta o de pago, presente en sitios de enseñanza, documentación, libros, papers y otros. No obstante, el enorme volumen de datos hace que el acceso a este conocimiento se vea dificultado por la necesidad de agregar todo este contenido de una forma accesible.

Se ha unido a LegitLearning, un startup spinoff de LegitBusiness, la que tomará la misión de modelar Ocumore, un sistema capaz de tomar esta información desde todas estas fuentes y generar micro cursos a partir de una solicitud generada por una persona, con material ordenado de tal forma que contenga documentos explicando los diversos tópicos del micro curso, un resumen de las fuentes ocupadas y una copia de estas accesible dentro del curso.

Para recoger estas copias de material, deberá existir un enjambre de instancias de workers con acceso a internet, dispuestas a ejecutar una serie de instrucciones para poder recoger material desde cualquier sitio deseado. Este acceso podrá ser mediante diversos protocolos, como Web, DAV, File transfer protocol, SFTP, entre otros.

Para autorizar el acceso a los recursos, deberá existir un sistema capaz de gestionar y proporcionar a los workers, bajo demanda, la capacidad de acceder a los sistemas deseados.

La primera parte clave de uso del sistema de construcción es el solicitante del curso. Este tendrá dos formas principales de interacción:

- Modo manual: En este modo, el operador que desea construir un micro curso deberá suministrar una serie de identificadores (DOIs, URLs, ids) para conseguir los documentos fuente. Junto con esto, el operador subirá manualmente los documentos que explican el curso y ordenará este curso en módulos de aprendizaje
- Modo AI: Mediante una interfaz interactiva, el operador conversará con un sistema de AI. Este sistema de AI (que viene preconstruido) recibirá instrucciones más abiertas y operará con los sistemas de la misma forma que el humano, pero de modo automático. Podrá ir explorando fuentes y solicitar información de diversos sitios mediante otros componentes de Ocumore

Posteriormente, recuperadas las fuentes y construidos (o recibidos) los documentos con los tópicos del micro curso, estos son enviados a través a un LMS (como Canvas) mediante la interfaz LTI o la interfaz SCORM.

Todo este proceso, desde que los requisitos están claros, debe completarse en menos de 10 minutos, para un curso de 300 recursos

Así, si un profesor de física quisiera ensamblar un curso, podría solicitar a la AI que busque papers de física, consiga videos en Youtube, consulte su curso anterior de Canvas, busque datos de experimentos sencillos en su computador de laboratorio y se arme un curso en Canvas de forma automática

3.2.a) (4p) El operador podría requerir delegar acceso a plataformas a las que solo el tenga acceso. ¿Como podría manejar este problema considerando seguridad y performance?

> ***Respuesta:*** *OAuth2. Una implementacion correcta permite todas estas demandas. El alumno debe describir una implementación adecuada de OAuth*

3.2.b) (4p) ¿Qué base(s) de dato(s) usaría para apoyar el guardado de datos de este sistema y en que partes lo haría? ¿Si usa más de una base, como compondría sus datos en consultas más complejas?

> ***Respuesta:*** *Hay varias justificaciones, pero debería haber una relacional transaccional al menos. Debería complementarse con algo capaz de guardar archivos, como una documental o un datastore con el protocolo S3*

3.2.c) (4p) ¿Que tácticas de performance y escalabilidad usaría en este sistema? Mencione al menos 4 tácticas, donde las usaría y porque

> ***Respuesta:*** *Depende del alumno, pero deberían ser tácticas del Bass Kazman. Puede que no coincidan los nombres pero se puede buscar una similitud al menos*

Preguntas comunes a los casos

3.d) (3p) Indique las características arquitectónicas que Ud. considere son fundamentales en este problema. Enumérelas en orden de prioridad y justifique brevemente.

Respuesta: Caso I

Aplican directamente:

- *Performance: Se pide explicito en el enunciado la velocidad*
- *Escalabilidad: Soporte de muchas noticias*
- *Elasticidad: Soporte de noticias cuando ocurra un evento particular*
- *Seguridad: Se pide del enunciado*
- *Confiabilidad: Noticias iguales a todos*
- *Otra bien justificada*

No aplican:

- *Integrabilidad: No se habla de conexión con otros sistemas*
- *Mantenibilidad: No se habla ser capaces de manejar y corregir errores*
- *Flexibilidad: No se habla de extender el sistema con nuevas funcionalidades*

Respuesta: Caso II

Aplican directamente:

- *Confiabilidad: documentación, mucho guardado de archivos*
- *Escalabilidad: Muchos workers, muchas fuentes, aunque no se especifica que tan rápido debe escalar*
- *Integrabilidad: Literal con los demás sistemas*

Podría ser:

- *Elasticidad bien justificada*

3.e) (4p) Indique los estilos/patrones arquitectónicos que va a emplear para solucionar el problema. Indique explícitamente qué aspecto del problema ataca cada estilo (responsabilidad).

Respuesta: Caso I

Aplican directamente:

- *En capas: Diferencias entre partes de la aplicación por tipo de usuario*
- *EDA: Uso de broker de eventos para manejar grandes cantidades de requests o tareas con workers*
- *Pub/Sub: Subida masiva de noticias*
- *Pipe & Filter: Filtrado de noticias al momento de subirse*
- *En lotes secuenciales: Flujo de subida o de aprobación*
- *Otra bien justificada*

- *Nota: Cliente servidor solo aplica si se da un contexto específico de la aplicación, por ejemplo, solo para la sección donde los usuarios normales accederán a los datos a través de una única API.*

No aplican:

- *Pizarra: No se espera un motor de procesos de muchas fuentes de datos*
- *Main y subrutinas: No se requiere que sea bloqueante*
- *Código Móvil: No se pide ejecución en múltiples ambientes*

Respuesta: Caso II

Aplican directamente:

- *Mediator*
- *EDA mediante workers*
- *Cli/Sv, especificado en la UI y en los consumidores*

No suman

- *Capas: Innecesario y restringe flexibilidad, escalabilidad*
- *PubSub: Debe haber un mediator*
- *Pipe and filter: Complejiza innecesariamente el ensamblador de cursos, no se pide modificar mucho ni pretrabajar la data*

No aplican:

- *Pizarra*
- *Main y subrutinas*

3.f) (13p) Confeccione un diagrama de componentes UML indicando los componentes arquitectónicos y sus relaciones, además de sus estilos especificados previamente, tal que reflejen una solución al problema planteado. Añada tanto detalle como haga falta para que quede claro cómo soluciona el problema planteado. Diséñelo en una hoja aparte.

Common - Respuesta:

Si no es un diagrama de componentes UML no se corrige. Como puede haber muchas soluciones al problema, se analizará en base a descuentos:

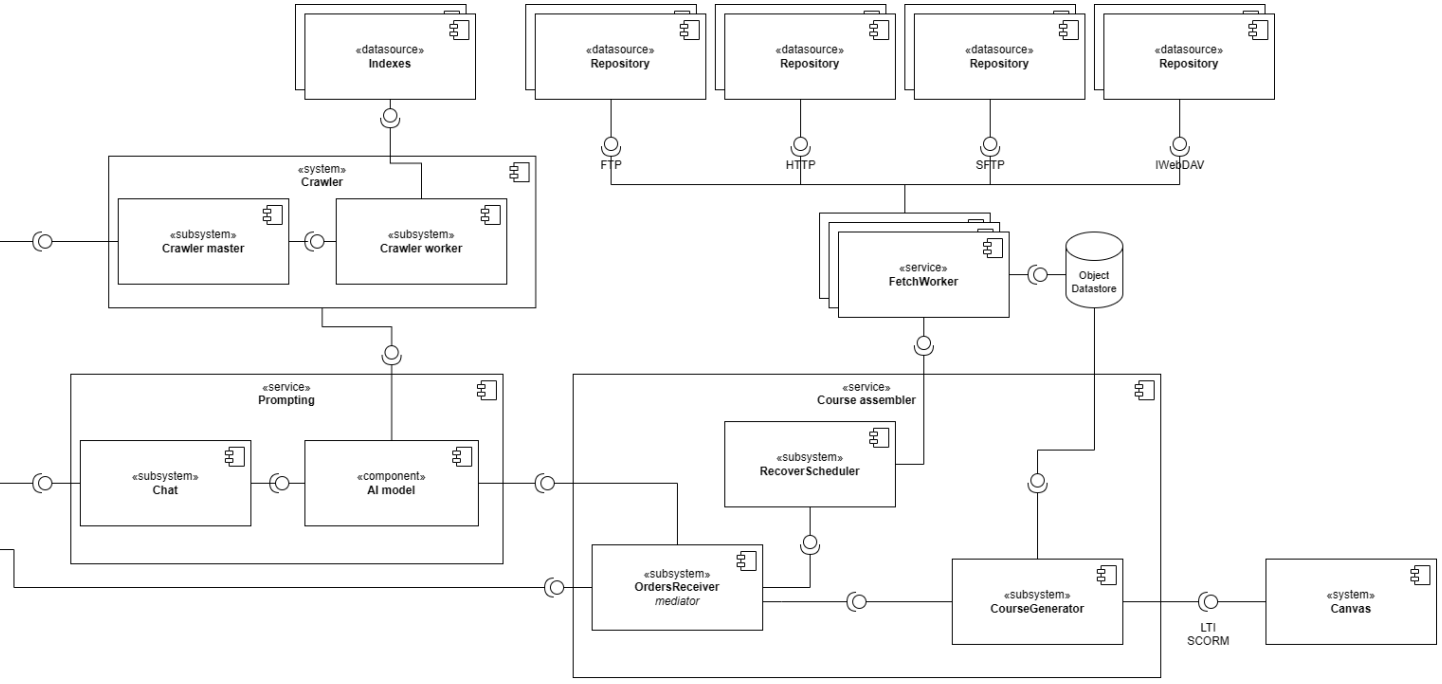
- *(-3 c/u) No usan los estilos que justificaron previamente*
- *(-1 c/u) Falta un requisito funcional*
- *(-1 c/u) Interfaces mal implementadas*
- *(-1 c/u) Componentes desagregados (sin conectores) sin justificación aparente*
- *(-6) No hay interfaces en el diagrama*

Nota1: Para ayudar a entender el Caso I se adjunta un diagrama de referencia en el siguiente link.

Nota2: Para ayudar a entender el Caso I se adjunta un diagrama de referencia en la última hoja.

3.g) (2p) Identifique 2 falencias de la solución de su diagrama y comente como las solucionaría y las implicancias de estas soluciones. No se corregirán aquellas que apunten al formato de cómo se diagramó o si faltó explotar algún componente. Escriba esta respuesta junto a su diagrama si le falta espacio.

> ***Respuesta:*** *Se esperan trade-off asociados a su solución, puede ser por una decisión de implementar un estilo u herramienta o por el mismo diseño de su sistema. Sobre la distribución de puntaje:*
>
> - *(0.5 ptos) Identificar un trade-off correctamente*
> - *(0.5 ptos) Identificar correctamente una posible solución aplicable a su sistema*


«datasource»
Indexes
«datasource»
Repository
«datasource»
Repository
«datasource»
Repository
«datasource»
Repository
FTP
HTTP
SFTP
IWebDAV
«system»
Crawler
«subsystem»
Crawler master
«subsystem»
Crawler worker
«service»
FetchWorker
Object
Datastore
«service»
Prompting
«subsystem»
Chat
«component»
AI model
«service»
Course assembler
«subsystem»
RecoverScheduler
«subsystem»
OrdersReceiver
mediator
«subsystem»
CourseGenerator
LTI
SCORM
«system»
Canvas