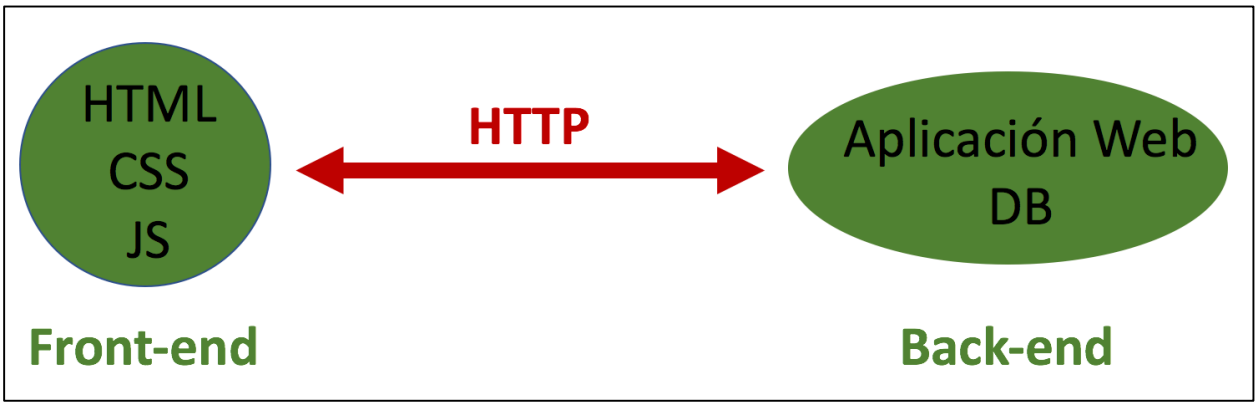
Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2513 – Tecnologías y Aplicaciones Web

Interrogación 1 - Pauta

Instrucciones: Sea preciso: no es necesario escribir extensamente pero sí ser preciso. En caso de ambigüedad, utilice su criterio y explicite los supuestos que considere convenientes. Esta interrogación fue diseñada para durar 100 minutos.

1. (0.6 pts) Realice un diagrama que grafique los principales actores y componentes del ecosistema Web. Detalle brevemente el rol de cada uno. Considere un nivel de detalle similar al que se utilizó en clases.



- HTML: estructura de la interfaz de usuario
- CSS: estilo de la interfaz de usuario
- JS: comportamiento de la interfaz de usuario
- HTTP: comunicación entre Front-end y Back-end
- Aplicación Web: lógica de la aplicación. Construye documentos utilizados por Front-end.
- DB: persistencia de los datos

2. (0.4 pts) Explique 3 beneficios y 2 desventajas de una aplicación Web.

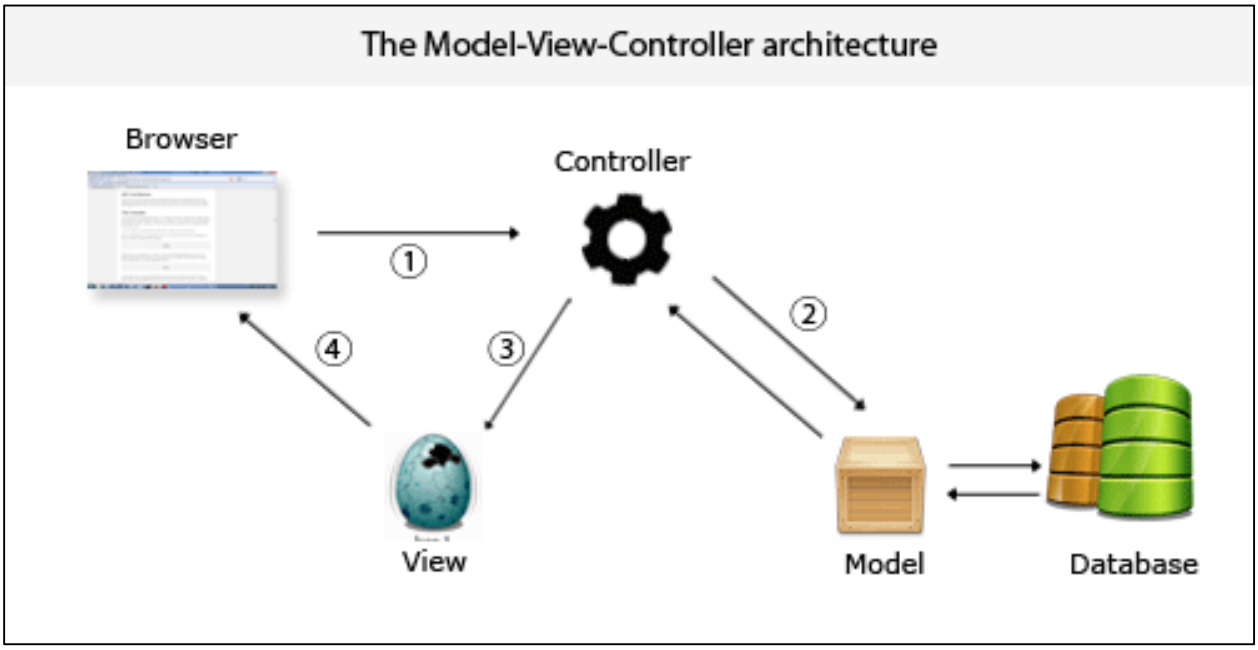
Beneficios:

- Multiplataforma
- Sin instalación, fácil distribución de actualizaciones
- Disponibilidad global a través de internet
- Bajos requerimientos de hardware en dispositivos de usuarios

Desventajas:

- Fuertemente dependiente de internet
- Velocidad de respuestas limitadas como mínimo por internet
- *Stack* mínimo de tecnologías complejo
- Expuesta, por lo que es más vulnerable

3. (0.6 pts) Con la ayuda de un diagrama explique cómo una aplicación web con arquitectura MVC procesa una petición HTTP. Detalle las responsabilidades de cada componente y cómo interactúan entre ellos.



- Model: representa las entidades de la aplicación web y modera la persistencia de los datos.
- View: construye documentos con información servida por el servidor, para ser enviados al solicitante.
- Controller: responde a eventos de una solicitud. Conecta modelos y vistas para atender a las peticiones.
- Database: persistencia de datos.

4. (0.5 pts) ¿Cómo se ve representada la arquitectura MVC en el *framework Ruby on Rails*? Relacione su respuesta con 3 de las gemas fundamentales incluidas en la gema *rails*.

Ruby on Rails promueve la utilización de las tres capas de la arquitectura MVC, al ofrecer convenciones para trabajar sobre estas. Un claro indicio de esto es la presencia de 3 carpetas en la estructura del *framework*: *models*, *views* y *controllers*. Respetar las convenciones dentro de estas 3 carpetas facilita la interacción descrita por la arquitectura MVC. Las gemas de *RoR* que se relacionan con los componentes MVC son:

- *activemodel* provee herramientas para ser utilizadas por los componentes de *Model*.
- *activerecord* transforma la información del componente *Database* a objetos utilizables en el componente *Model*.
- *actionview* ofrece herramientas para construir vistas del componente *View*, a partir de prototipos que se completan con información proveniente del controlador.
- *actionpack* se utiliza en la componente de *Controller*, al gestionar y responder las peticiones web.

5. (0.8 pts) Responda si las siguientes afirmaciones son verdaderas (**V**) o falsas (**F**). Justifique todas sus respuestas.

F: El método POST del protocolo HTTP es idempotente.

- La intención del método POST es crear nuevos recursos, por lo que diferentes invocaciones generan diferentes recursos (contrario al concepto de idempotencia).

F: El *status code* "403 - *Forbidden*" del protocolo HTTP representa que el usuario de la petición no está autenticado.

- Esta descripción corresponde al *status code* "401 – *Unauthorized*". El *status code* "403 - *Forbidden*" corresponde a una correcta autenticación por parte del usuario, pero que no tiene los permisos necesarios para acceder al recurso que solicita.

F: Cualquier método del protocolo HTTP debería retornar un *status code* 201.

- El *status code* 201 representa una correcta creación de un recurso. El método GET nunca debería crear un recurso en la aplicación.

F: El *header* "*Content-Type*" de una petición HTTP especifica los formatos en que se desea recibir la respuesta.

- Esta descripción corresponde al *header* "*Accept*". El *header* "*Content-Type*" especifica en qué formato la petición o la respuesta envían la información.

F: Según el protocolo HTTP, puede tener sentido que una petición GET envíe parámetros a través del *body*.

- El método GET se utiliza para obtener información de los recursos. Por otra parte, el *body* de una petición HTTP se utiliza para enviar datos asociados a los recursos. Por esta razón no tiene sentido que una petición GET envíe parámetros a través del *body*.

6. (0.3 pts) ¿Qué beneficios conlleva una correcta utilización del *status code* "304 - *Not Modified"* del protocolo HTTP para una aplicación Web? ¿Qué *headers* son necesarios para esto?

El *status code* "304 - *Not Modified*" mejora el desempeño de la aplicación Web, al evitar transferencia de datos innecesarios. Los headers que se deben enviar en la petición para obtener esta respuesta son "*If-Modified-Since*" o "*If-None-Match*".

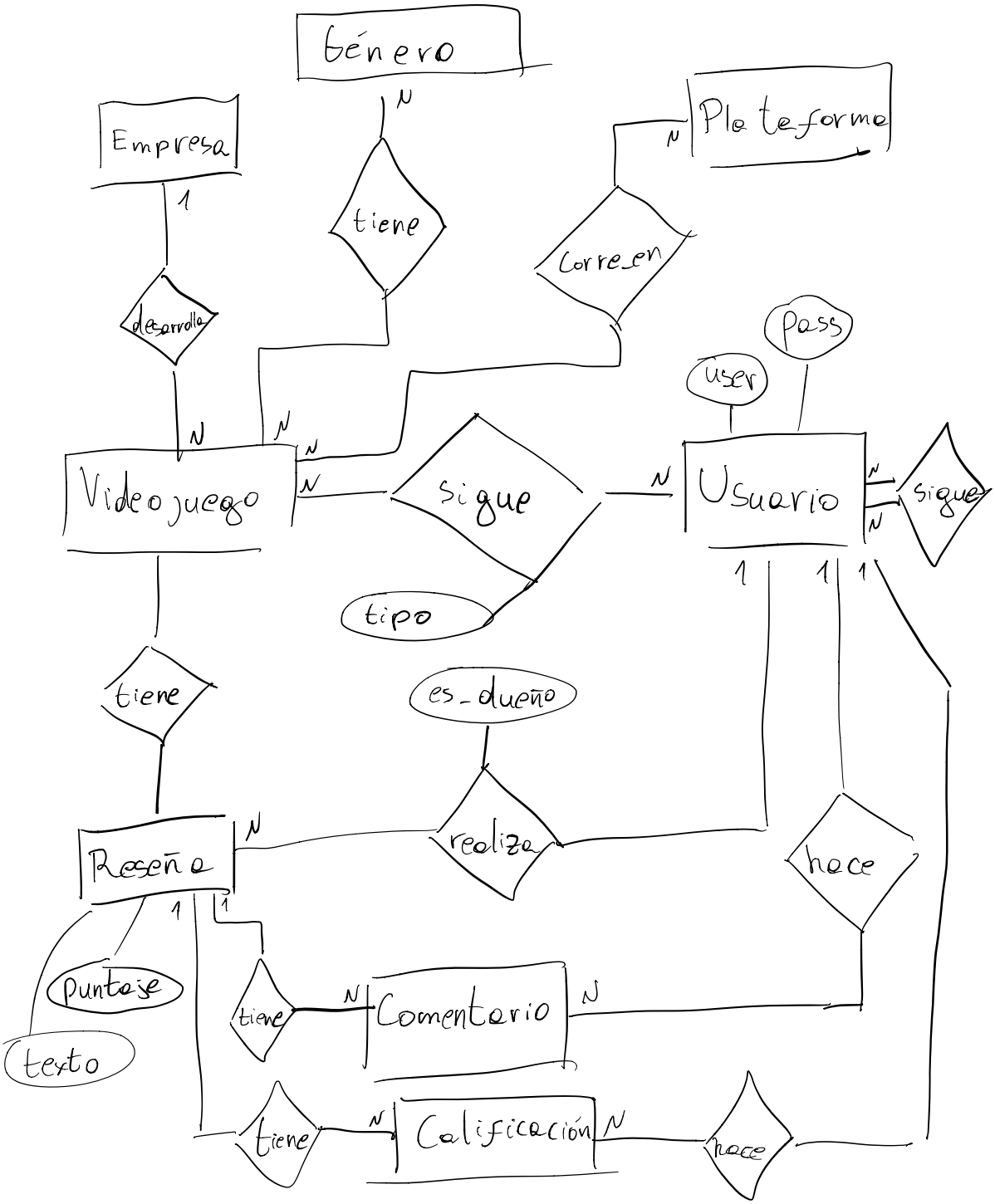
7. (0.3 pts) Explique a qué se refiere *Duck Typing*. De un ejemplo en el contexto de *Ruby*.

Duck Typing se refiere a que, en lugar de conocer con precisión el tipo de un objeto, nos preocupamos de chequear si sus capacidades (métodos) son las necesarias para un cierto propósito. Un método útil para este propósito en *Ruby* es *respond_to?*.

8. (2.5 pts) Considerando la siguiente descripción:

> Se desea realizar una aplicación Web para hacer reseñas de videojuegos. El objetivo de esta plataforma es generar comunidad entre jugadores y desarrolladores. El plan de negocio de los fundadores es ofrecer métricas y estadísticas en tiempo real a los desarrolladores de videojuegos. En este sistema los videojuegos se organizan por empresas que los desarrollan, plataformas en que se pueden jugar y géneros a los que pertenecen. Las reseñas deben ser públicas, pero solamente usuarios registrados pueden publicar o calificar estas. También se debe indicar si el usuario es dueño del videojuego que está evaluando, para así hacer más válida su opinión. Distintos usuarios registrados pueden dejar comentarios sobre una reseña. Además, los usuarios pueden seguir la actividad de otros usuarios o juegos, lo que significa que se les notificará cada vez se agregué una reseña relacionada.

Realice un diagrama entidad-relación que represente el modelo del problema. Explique cómo se podrían implementar todas las funcionalidades descritas a partir de su diagrama.


Género
Empresa
Plataforma
tiene
desarrolla
corre_en
pass
user
Videojuego
sigue
Usuario
sigue
tipo
tiene
es_dueño
Reseña
realiza
hace
puntaje
tiene
Comentario
texto
tiene
Calificación
hace

Funcionalidades:

- Login / tipos de usuario
- CRUD juegos / relacionar con entidades
- CRUD reseñas
- Comentar
- Seguir juego, usuario
- Estadísticas en base a métricas
- Calificaciones

* Métrica podría ser una entidad o no