Diseño de Sistemas. Examen Final. 08/06/2018.

Apellido y Nombre: **Legajo:**

Planning.com

Contexto general

Existen diversos sitios web populares que permiten reservar alojamiento para los viajes de ocio o de placer, otros que permiten reservar transporte, otros brindan información sobre lugares a visitar, localidades, monumentos, museos u otros hitos. Supongamos que tengo claro que voy a hacer un viaje, tengo claro qué localidades quiero recorrer (suponiendo que son varias) y que ya sé en qué medio viajar. O por lo menos… puedo suponer todo esto.

¿Has reservado alguna vez un alojamiento en Bariloche, otro en El Bolsón, otro en Esquel? ¿Compraste los vuelos y los micros entre las ciudades?

Ahora surge la pregunta: ¿qué sitios intermedios existen para conocer? ¿Los conozco estando desde Bariloche o desde El Bolsón? Para ir a Lago Puelo, ¿lo hago un día que haga noche en El Bolsón? ¿O cuando estoy directamente camino a Esquel hago una parada intermedia? A partir de estas preguntas surge **Planning.com**, el Planificador Inteligente de Viajes.

Decisiones preliminares

El sistema se encuentra en su etapa preliminar de diseño y se han definido algunas características:

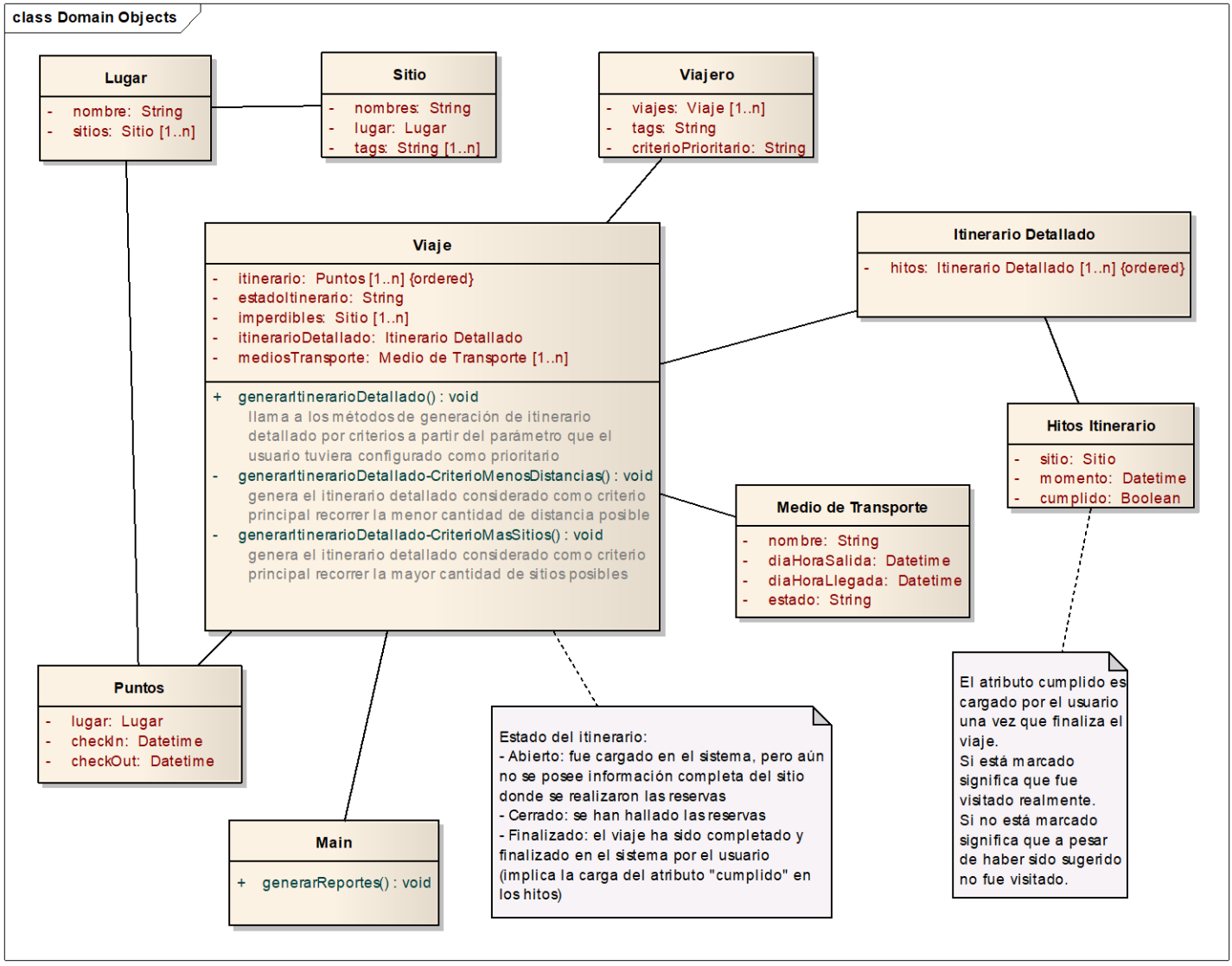
- Glosario de términos:
  - ***Lugar***: localidades
  - ***Sitios***: lugares de interés para un visitante (monumentos, museos, etc.)
  - ***Itinerario***: recorrido que incluye diversas localidades ordenadas
  - ***Itinerario Detallado***: recorrido que incluye diversos sitios ordenados teniendo en cuenta los lugares de visita.
- Se trabajará en base a ***tags***. Cada sitio tendrá una cantidad (no determinada) de tags. El usuario seleccionará tags como sus preferencias que posteriormente se matchearán entre tags de sitios y tags de preferencias de usuario. Por ejemplo: un museo de arte podría tener un tag “arte”; un parque de diversiones, “adrenalina”.
- El uso de esta aplicación requiere conocer el itinerario. En una primera etapa se prevé trabajar con sitios conocidos de reservas de alojamiento que puedan brindar una *API a través de la cual acceder a las* ***reservas***.
- El usuario podrá indicar en el sistema algún sitio como “***imperdible***” (¿quién iría a París y no diría que la Torre Eiffel es imperdible?), o sea, lugares que el sistema obligatoriamente deberá indicar en el itinerario detallado. Entre los restantes sitios, el sistema armará un itinerario detallado de manera inteligente.
- El sistema deberá, a partir del itinerario del usuario, conociendo los “imperdibles”, los tags de preferencias de éste y el medio de movilidad escogido, generar el itinerario detallado (llamaremos a esto BuilderMain)**.**
- El ***BuilderMain*** deberá estar lo más desacoplado posible del sistema en general dado que será trabajado por un equipo ad-hoc y sufrirán transformaciones rápidamente.
- El software tendrá acceso a *incidencias de los medios de transporte*: en caso que un medio de transporte modificara sus horarios, tuviera una cancelación o se produjera cualquier modificación que impacte sobre el itinerario del viaje, el itinerario deberá ser recalculado.
- Se prevé un espacio de Analitycs. En principio está pensando generar dos ***reportes***:
  - Sitio a sitio: ranking de relaciones “sitio origen - sitio destino” más presentes en los viajes. Los datos serán vendidos a las empresas de turismo que ofrecen transporte de media y corta distancia.

- Incumplimientos: ranking de sitios que ostentan el porcentaje más bajo de atributo cumplido no marcado. Los datos alimentarán el listado de mejora continua del BuilderMain.

- Cuando se detecta que un itinerario ha sido ***cerrado*** (o sea, se terminó la edición), automáticamente debe ejecutarse el BuilderMain para dicho itinerario.
- Cuando un usuario accede al sistema, y si el itinerario está ***abierto***, se le deben ofrecer ***banners*** de nuevos lugares posibles para visitar. Esto será manejado por una *aplicación desarrollada externamente* denominada AVC (acrónimo de Atracción de Visitantes a las Ciudades) que estará alojada en el propio web server del sistema, con un File Server donde se alojan los archivos de los banners.
- Cuando un ***hito*** del itinerario es "***cumplido***" se debe permitir notificar a todos los administradores de sitio que lo deseen.

Punto 1 – Modelo de Objetos

1.1. A partir del siguiente modelo de objetos, y teniendo en cuenta un apropiado balanceo entre los atributos de mantenibilidad y performance, indique qué modificaciones realizaría sobre el mismo.
Si utiliza patrones de diseño, indíquelos y justifique su uso. Puede complementar con código fuente, pseudo-código, prosa u otros diagramas (diagrama de secuencia, de estados, entre otros).



Punto 2 – Persistencia

2.1. Indicar qué estrategias de persistencia utilizaría. ¿En qué casos utilizaría un modelo relacional y en qué casos un modelo no relacional? **Justificar.**

2.2. Para aquellos elementos que considera necesario persistir en un modelo relacional:

- Realizar el DER, indicando claves primarias y foráneas.
- Indicar las restricciones necesarias.
- Detallar las estrategias de mapeo utilizadas. Explicar por qué fueron elegidas.

Punto 3 – Arquitectura

Proponer una arquitectura lógica del sistema a alto nivel, y comunicarla utilizando prosa y/o diagramas. Debe quedar en claro: componentes de alto nivel, sistemas externos y comunicación entre componentes.
Si utiliza algún patrón o estilo arquitectónico, justifique su uso y explique en qué beneficia a toda su solución propuesta.
Restricción impuesta: la compañía no acepta el uso del patrón Model-View-Controller (MVC)

Condiciones de aprobación: Sumar como mínimo 60 puntos y no menos del 50 % en cada sección:
(A) otorga 40 puntos, (B) 35 puntos y (C) 25 puntos.