Proyecto II-2017

1. Escenario

En la actualidad se presenta un desafío al momento de adaptarse al entorno laboral. Influyen factores tales como el tiempo, la disponibilidad, la metodología implementada, los déficits en torno al conocimiento adquirido, y el factor más importante, la adaptabilidad.

Es un hecho que gran parte de los desarrolladores al momento de iniciar en la curva de aprendizaje laboral suelen tener el concepto errado de utilizar "una tecnología, o un grupo de tecnologías similares", pero la realidad es totalmente diferente. Un proyecto puede abarcar tanto un desarrollo REST en un módulo, como un desarrollo SOAP en otro, y lograr la interconexión entre sí, así como también consumir servicios de esos tipos mediante la construcción de APIs.

Adicionalmente, un desarrollador no debería ***"casarse"*** con un sólo tipo de Sistema Manejador de Bases de Datos, por el contrario, lo ideal sería encontrar un modelo, o patrón de diseño (Data Access Object) que le permita cumplir con los objetivos de su aplicación, logrando la abstracción con respecto al SMBD que utilice.

Por todas las razones señaladas anteriormente, se considera imprescendible que los estudiantes pertenecientes a la cátedra **Aplicaciones con La Tecnología Internet** II del **semestre lectivo II-2017** desarrollen la habilidad de construir aplicaciones capaces de gestionar entidades mediante el consumo de servicios públicos, tanto de tipo REST como SOAP, construir APIs REST con la posibilidad de ser alojadas en la nube, y construir SPAs que provean la funcionalidad de consumir los servicios dispuestos por dichos APIs.

De esta forma, será posible formar una base sólida durante su capacitación a lo largo de su carrera universitaria.

2. Requerimientos

Como se especificó anteriormente, se desean lograr tres objetivos, los cuales serán divididos por fases:

- **<u>Fase I</u>: Construcción de API Spring MVC para gestión de entidades pertenecientes a servicios públicos.**

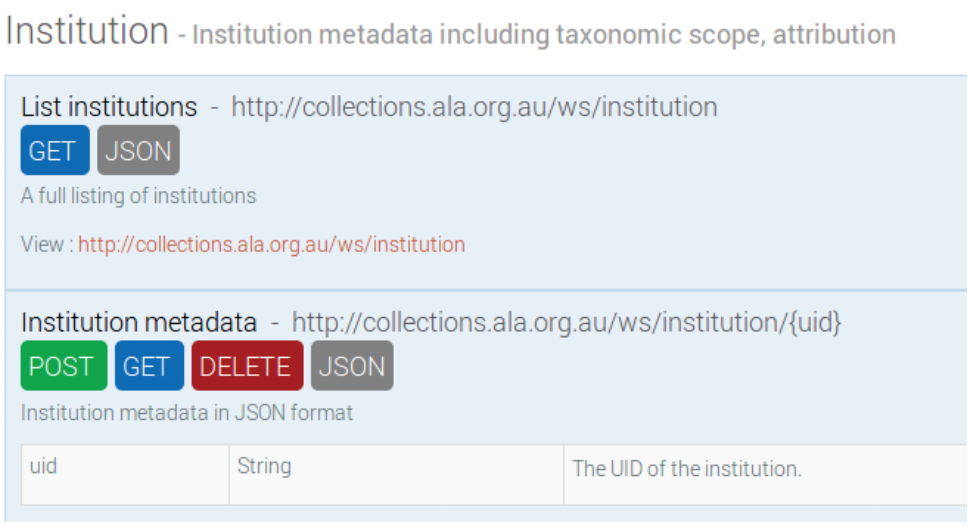
    - El proyecto debe ser construido por medio de **Maven.**
    - El entregable debe consistir en un archivo **WAR.**
    - Se debe utilizar **Spring MVC** (obligatorio).
    - Se debe utilizar **Hibernate.**
    - El patrón de diseño a ser implementado durante el desarrollo de este módulo será **D.A.O.**
    - Se debe realizar un diagrama de clases que refleje el patrón a implementar.
    - Las vistas implementadas deben ser **JSP.**
    - Las operaciones realizadas sobre el servicio a consultar, serán llevadas a cabo por medio de una instancia de cliente **Jersey**. ***Se incorporará una guía al laboratorio #3 donde se indique cómo utilizar Jersey.***
    - La aplicación debe proveer las siguientes funcionalidades:
        - Debe desplegar una interfaz principal (**index**), la cual dispondrá de las opciones:
            - **Consultar Servicio:** esta opción redireccionará a la url ***'/consultarServicio'***, en la cual se generará la instancia del cliente, para posteriormente consumir el web service y obtener los datos necesarios. En este punto, es importante señalar que tales datos serán incorporados a las entidades.
            - **Mostrar registros:** esta opción redireccionará a la url ***'/mostrarServicio'***, en la cual se desplegarán los datos contenidos en las entidades (por medio de tablas), que a su vez, corresponden a la información generada al consumir el web service.
            - **Modificar registros:** se redireccionará a la url ***'modificarServicio'***, debe permitir realizar modificaciones sobre las entidades. Por ejemplo, cambiar el nombre de algún

atributo. Además, dichos ***'atributos'*** a modificar deben ser desplegados en forma de enlaces, los cuales direccionarán a distintas url:

1. **'modificarServicio/nombreInstituto':** devolverá una lista con todos los nombres de los institutos, con la posibilidad de seleccionar uno para modificarlo.
2. **'modificarServicio/uriInstituto':** retornará una lista con todos los url de los intitutos, con su respectivo nombre, con la finalidad de seleccionarlo y modificarlo.
3. **'modificarServicio/descripcionInstituto':** retornará una lista con todos los intitutos, con su respectiva descripción, con el objetivo de seleccionarlo y modificarlo.

- La modificación de las entidades será realizada de manera local, no es necesario realizar un **POST** hacia el web service. El hecho de que la entidad almacene las modificaciones será suficiente para considerar correcta su funcionalidad.

- El web service a consumir será:
  - https://api.ala.org.au/#ws40
  - Sólo se trabajará sobre la sección ***Institución***:



- Se deben realizar dos accesos, el primero en dirección a **List institutions**, que dará como resultado tres atributos**:**

name--> corresponde al nombre de la institución.
uid **-->** Id de la institución.
uri **-->** Para obtener la descripción con respecto a la institución.

- Estos atributos serán incorporados a la entidad **Instituto.**
- El siguiente acceso se puede realizar en dos modalidades**:**
    1. Realizar el acceso mediante la uri de la institución.
    2. En su defecto, tomar cada ***'uid'*** de cada institución, para luego realizar un **GET,** pasando como parámetro el ***'uid'*****. (Ver imagen - Institution metadata).**
- De este acceso, se espera que sean incorporados los siguientes atributos a la entidad:
    1. **Acronym.**
    2. **Email.**
    3. **pubDescription.**
    4. **websiteUrl.**
- Si estos datos se encuentran vacíos, se agrega por defecto: ***"Sin información".***

- **<u>Fase II</u>: Construcción de API REST para exponer las entidades dispuestas por la aplicación de la Fase I.**

    - El proyecto debe ser construido por medio de ***NodeJS.***
    - Es obligatorio el uso de los módulos:
        - ***ExpressJS.***
        - ***Mongoose.***
    - La aplicación debe proveer las siguientes funcionalidades:
        - Partiendo de la Base de Datos generada por la implementación de la ***Fase I*** del proyecto:
            - Se debe crear un ***API REST*** que permita mostrar los datos presentes en la Base de Datos, en formato ***JSON.*** Y debe ser posible consultar dichos datos desde cualquier ángulo, por ejemplo, obtener un registro filtrando por su correspondiente campo uid, por dirección, o por algún otro campo. Y para ello se deben seguir los siguientes pasos:
                1. Se migrarán los datos pertenecientes a la Base de Datos de la ***Fase I***, a una nueva Base de Datos ***MongoDB*** creada con ***Mongoose.***
                2. Se debe crear un usuario que tenga acceso a dicho esquema.
                3. Por medio de la conexión del usuario definido, se procederán a realizar los servicios, los cuales básicamente expondrán las entidades que se encuentran en la Base de Datos, tomando en cuenta los campos que existen en la tabla ***Entidad.*** Deben tener en consideración que en caso de entidades que compartan un campo en común, deben ser mostradas ambas entidades al momento de generar la solicitud. Toda la exposición de las entidades se debe realizar en formato ***JSON.***

- **<u>Fase III</u>: Construcción de SPA que implemente una aplicación que consuma los servicios del API desarrollado durante la Fase II.**

3. Condiciones Generales

- Todo diseño creado deber ser responsive.
- Como sugerencia, se recomienda utilizar ***Bootstrap*** por medio de ***webjars*** (**Ver tutorial, laboratorio #2**).
- El proyecto será realizado en equipos de **3 personas.**
- Si un miembro del equipo no realiza ningún aporte significativo sobre el proyecto, se penalizará con ***0 puntos*** sobre la nota final de la fase.
- Una vez conformados los grupos, no existe posibilidad de cambio de integrantes entre equipos.
- Se sugiere utilizar el modelado de proyecto del ***Laboratorio #2 Spring MVC*** para la ***Fase I.***
- Como mínimo debe existir la entidad ***Institución***, el desarrollador es libre en caso de que desee crear más de 1 entidad.

4. Entregas

Fase 1

- Domingo 17 de junio del 2018 hasta las 23:59.
- Enviar a la dirección de correo: aictor94@gmail.com.
- Formato de entrega: [ATI_II] <CI>_Proyecto_Fase_I
  - Ejemplo: 23073727_Proyecto_Fase_I

Fase 2

- Viernes 06 de julio del 2018 hasta las 23:59.
- Enviar a la dirección de correo: aictor94@gmail.com.
- Formato de entrega: [ATI_II] <CI>_Proyecto_Fase_II
  - Ejemplo: 23073727_Proyecto_Fase_II

Fase 3 --> Sin definir.