PROGRAMACION ORIENTADA A OBJETOS II
PROGRAMA 2

Para este programa, deberá completar los servicios REST que se comenzaron a desarrollar para la práctica 9, los cuales deben poder ser usados por alguna implementación de la interface DaoConcursos del Programa 1, es decir, los servicios REST a generar, deben proveer la información requerida por los métodos indicados en la interface DAOConcursos (no es necesario todos los métodos de los DAOs individuales del Programa 1).

Entonces, tomando en cuenta de la práctica que el URL Base de los servicios a proporcionar es http://localhost:8080/RESTConcursos-XXXXXXXX/servicios (donde XXXXXXXX se sustituye por su matrícula, el programa 2 deberá contener un servicio REST por cada una de las tablas (entidad, municipio, institucion, persona, datos_estudiante, sede, concurso, equipo, sede_concurso y equipo_sede_concurso) que componen a la base de datos controlconcursos, cada uno representado por la ruta indicada más adelante.

INSTRUCCIONES PARA INICIALIZAR EL PROYECTO DEL PROGRAMA 2

ES IMPORTANTE QUE LAS SIGUIENTES INSTRUCCIONES LAS SIGA EN EL ORDEN INDICADO:

1. Entrar a la página cuyo URL les fue proporcionado al aceptar la tarea, en tal página dé click en el botón Code y copie el URL que aparece en el cuadro de texto de nombre Clone with HTTPS (comienza con https://)
2. Abre IntelliJ IDEA e indica que harás un clon local de tu repositorio:
   - Si no tienes ningún repositorio abierto selecciona la opción Get From Version Control de la Ventana de Bienvenida, o si tienes un proyecto abierto, puedes entrar al menú VCS y seleccionar la opción Get From Version Control
   - En el cuadro de diálogo que aparece:
     - Selecciona Git
     - Pega el URL que copiaste en el paso 1 en el cuadro de texto URL
     - Selecciona en Directory la carpeta donde lo colocarás, es importante que crees una nueva carpeta o se colocará (da click en el icono de carpeta , navega a donde lo quieres colocar y da click en el icono de New Folder para crear una nueva carpeta)
     - Da click en Clone
     - Si te pide usuario y clave de Github proporciona esos datos
     - Después de unos segundos tendrás listo tu un clon de tu repositorio listo para trabajar en Intellij IDEA

MODIFICACIÓN DE LOS ARCHIVOS PROPORCIONADOS (EN CUANTO A CONFIGURACIÓN)

Una vez que tengas el repositorio local, el trabajo principal consiste en crear las clases para proporcionar los servicios REST descritos en la siguiente sección. Aseguráte de una vez creados los servicios REST, poner en el método agregaRecursosREST de la clase AplicacionConfig código para agregar las clases que representan a los servicios REST al Set que recibe como argumento.

Antes de hacer los cambios en el código haga las siguientes modificaciones:

1. En el archivo index.html, incluido dentro de la subcarpeta webapp en la sección main del proyecto, deberá sustituir las NNNNNNNNNNNN por su nombre completo y las XXXXXXXX por su matrícula, lo cual

también deberá realizar en el archivo datosmysql.properties ubicado dentro de la subcarpeta resources en la sección main
2. En el archivo build.gradle modifique el valor de version para que se use su matrícula

3. Asegurese de estar usando la versión 8 de Java (en el menu File-> Project Structure, Sección Project, Opción Project SDK)
4. Agregue una configuración de ejecución para poder poner a correr el Glassfish, para ello:
- Dé click en Add Configuration...
- D+e click en el +
- Seleccione GlassFish Server -> Local
- En el cuadro de diálogo que aparece seleccione domain1 en Server Domain, marque Preserve Sessions Across Redeployment, en la pestaña Deployment dé click en +, seleccione Artifact y seleccione RESTConcursos-XXXXXXXX.jar (donde las XXXXXXXX ya debieran aparecer como su matricula), de click en OK, regrese a la pestaña Server y seleccione Redeploy en On 'Update' Action, de click en OK

TRABAJO A REALIZAR EN CUANTO A CÓDIGO

De acuerdo a lo indicado para el Programa 1, la interface ***DaoConcursos*** contiene los siguientes métodos:

1. **public List<Carrera> obtenInstituciones()**
2. **public boolean agregaInstitucion(Institucion dato)**
3. **public boolean eliminaInstitucion(long idInstitucion)**
4. **public boolean actualizaInstitucion(Institucion dato)**

5. **public List<Entidad> obtenEntidades()**

6. **public List<Municipio> obtenMunicipios(long idEntidad)**

7. **public List<Persona> obtenPersonas()**
8. **public boolean agregaPersona(Persona dato)**
9. **public boolean eliminaPersona(String emailPersona)**
10. **public boolean actualizaPersona(Persona p)**
11. **public List<String> obtenCorreosDeInstitucion(long idInstitucion, String tipo)**

Los métodos en verde son métodos que deben no corresponde a los 5 métodos que ya pueden realizar los servicios REST por heredar de RESTAbstracto

12. **public DatosEstudiante obtenDatosEstudiante(String emailEstudiante)**
13. **public boolean agregaDatosEstudiante (DatosEstudiante dato)**
14. **public boolean eliminaDatosEstudiante (String emailPersona)**
15. **public boolean actualizaDatosEstudiante (DatosEstudiante dato)**

16. **public List<Sede> obtenSedes()**
17. **public boolean agregaSede(Sede dato)**
18. **public boolean eliminaSede(long idSede)**
19. **public boolean actualizaSede(Sede dato)**
20. **public List<Sede> obtenSedesDisponibles(long idConcurso)**

21. **public List<Concurso> obtenConcursos()**
22. **public boolean agregaConcurso(Concurso dato)**
23. **public boolean eliminaConcurso(long idConcurso)**
24. **public boolean actualizaConcurso(Concurso dato)**

25. **public List<Equipo> obtenEquipos()**
26. **public boolean agregaEquipo(Equipo dato)**
27. **public boolean eliminaEquipo(long idEquipo)**
28. **public boolean actualizaEquipo(Equipo dato)**
29. **public List<Equipo> obtenEquiposDisponibles(long idSedeConcurso, long idInstitucion)**

30. **public List<SedeConcursoExtendida> obtenSedesAsignadas(long idConcurso)**
31. **public boolean agregaSedeConcurso(SedeConcurso dato)**
32. **public boolean eliminaSedeConcurso(long idSedeConcurso)**

33. **public List<EquiposSedeConcursoExtendida> obtenEquiposRegistrados(long idConcurso, long idInstitucion)**
34. **public boolean registrarEquipoSedeConcurso(EquiposSedeConcurso dato)**
35. **public boolean cancelarEquipoSedeConcurso (long idEquipoSedeConcurso)**

Para todos estos métodos debe crear servicios REST que realicen el trabajo correspondiente.

Los métodos 1 a 4 deberían estar en un servicio REST de ruta **institucion**. El método 5 debería estar en un servicio REST de ruta **entidad**. El método 6 debería estar en un servicio REST de ruta **municipio**. Los métodos 7 a 11 deberían estar en un servicio REST de ruta **persona**. Los métodos 12 a 15 deberían estar en un servicio REST de ruta **datosestudiante**. Los métodos 16 a 20 deberían estar en un servicio REST de ruta **sede**. Los métodos 21 a 24 deberían estar en un servicio REST de ruta **concurso**. Los métodos 25 a 29 deberían estar en un servicio REST de ruta **equipo**. Los métodos 30 a 32 deberían estar en un servicio REST de ruta **sedeconcurso**. Los métodos 33 a 35 deberían estar en un servicio REST de ruta **equipossedeconcurso**.

Las clases de entidad a generar deberán estar en el paquete mx.edu.uaz.ingsoftware.poo2.entidades y los servicios en el paquete mx.edu.uaz.ingsoftware.poo2.servicios.

Algunos de estos métodos ya tienen prácticamente la funcionalidad deseada, puesto que los servicios REST heredan de la clase RESTAbstracto, mientras que otros requieren de la generación de código más elaborado para realizar el trabajo que se requiere. La ruta específica que debe tener el servicio REST asociado con los métodos de la interface DAOConcursos son los siguientes (urlbase es el URL Base de los servicios REST, indicado ya previamente):

Método de la Interface DAOConcursos	**Ruta del Servicio REST**	**Método HTTP a usar**
`obtenInstituciones`	`urlbase/institucion`	`GET`
`agregaInstitucion`	`urlbase/institucion`	`POST`
`actualizaInstitucion`	`urlbase/institucion/{idinst}`	`PUT`
`eliminaInstitucion`	`urlbase/institucion/{idinst}`	`DELETE`
`obtenEntidades`	`urlbase/entidad`	`GET`
`obtenMunicipios`	`urlbase/municipio/{identidad}`	`GET`

obtenPersonas	urlbase/persona	GET
agregaPersona	urlbase/persona	POST
actualizaPersona	urlbase/persona/{emailpersona}	PUT
eliminaPersona	urlbase/persona/{emailpersona}	DELETE
obtenCorreosDeInstitucion	urlbase/persona/correos/{idinst}/{tipopersona}	GET
obtenDatosEstudiante	urlbase/datosestudiante	GET
agregaDatosEstudiante	urlbase/datosestudiante	POST
actualizaDatosEstudiante	urlbase/datosestudiante/{email}	PUT
eliminaDatosEstudiante	urlbase/datosestudiante/{email}	DELETE
obtenSedes	urlbase/sede	GET
agregaSede	urlbase/sede	POST
actualizaSede	urlbase/sede/{idsede}	PUT
eliminaSede	urlbase/sede/{idsede}	DELETE
obtenSedesDisponibles	urlbase/sede/disponibles/{idconc}	GET
obtenConcursos	urlbase/concurso	GET
agregaConcurso	urlbase/concurso	POST
actualizaConcurso	urlbase/concurso/{idconcurso}	PUT
eliminaConcurso	urlbase/concurso/{idconcurso}	DELETE
obtenEquipos	urlbase/equipo	GET
agregaEquipo	urlbase/equipo	POST
actualizaEquipo	urlbase/equipo/{idequipo}	PUT
eliminaEquipo	urlbase/equipo/{idequipo}	DELETE
obtenEquiposDisponibles	urlbase/equipo/disponibles/{idsedeconcurso}/{idinst}	GET
obtenSedesAsignadas	urlbase/sedeconcurso/asignadas/{idconcurso}	GET
agregaSedeConcurso	urlbase/sedeconcurso	POST
eliminaSedeConcurso	urlbase/sedeconcurso/{idsedeconc}	DELETE
obtenEquiposRegistrados	urlbase/equipossedeconcurso/registrados/{idconc}/{idinst}	GET
registrarEquipoSedeConcurso	urlbase/equipossedeconcurso	POST
cancelarEquipoSedeConcurso	urlbase/equipossedeconcurso/{idequiposedeconc}	DELETE

Para que los servicios REST funcionen correctamente, es necesario que las clases de entidad correspondiente tengan las anotaciones necesarias, para la práctica 9 ya se les había proporcionado 5 de ellas (Equipo, Institucion, Municipio, Persona y Sede) y en el proyecto que se les proporciona de base para el Programa 2, ya están incluidas, solo requiere completar las anotaciones en las otras 5 clases de entidad (Concurso, DatosEstudiante, SedeConcurso, EquiposSedeConcurso y Entidad). Para las clases extendidas (SedeConcursoExtendidad y EquiposSedeConcursoExtendida solo es necesario poner la anotación @XmlRootElement a la clase (dado que esas no representan a una tabla de la base de datos).

Al igual que para los servicios realizados en la práctica 9, las operaciones para agregar, actualizar o eliminar deberán primero validar si es posible realizar la operación solicitada, de acuerdo a las reglas indicadas en el Programa 1, si no es posible, regresará "false".

NOTA: El servicio REST asociado con el método obtenCorreosDeInstitucion deberá regresar un String compuesto por todos los correos de la institución y tipo de persona indicados; separados por un punto y coma.

La calificación de cada servicio es el siguiente:

Método de la Interface DAOConcursos	**Ruta del Servicio REST**	**Puntos**
obtenInstituciones	urlbase/institucion	1
agregaInstitucion	urlbase/institucion	1
actualizaInstitucion	urlbase/institucion/{idinst}	1
eliminaInstitucion	urlbase/institucion/{idinst}	1
obtenEntidades	urlbase/entidad	3
obtenMunicipios	urlbase/municipio/{identidad}	1
obtenPersonas	urlbase/persona	3
agregaPersona	urlbase/persona	5
actualizaPersona	urlbase/persona/{emailpersona}	5
eliminaPersona	urlbase/persona/{emailpersona}	5
obtenCorreosDeInstitucion	urlbase/persona/correos/{idinst}/{tipopersona}	4
obtenDatosEstudiante	urlbase/datosestudiante	3
agregaDatosEstudiante	urlbase/datosestudiante	3
actualizaDatosEstudiante	urlbase/datosestudiante/{email}	3
eliminaDatosEstudiante	urlbase/datosestudiante/{email}	2
obtenSedes	urlbase/sede	3
agregaSede	urlbase/sede	4
actualizaSede	urlbase/sede/{idsede}	4
eliminaSede	urlbase/sede/{idsede}	4
obtenSedesDisponibles	urlbase/sede/disponibles/{idconc}	2.5
obtenConcursos	urlbase/concurso	3
agregaConcurso	urlbase/concurso	4
actualizaConcurso	urlbase/concurso/{idconcurso}	4
eliminaConcurso	urlbase/concurso/{idconcurso}	2
obtenEquipos	urlbase/equipo	3
agregaEquipo	urlbase/equipo	6
actualizaEquipo	urlbase/equipo/{idequipo}	6
eliminaEquipo	urlbase/equipo/{idequipo}	4
obtenEquiposDisponibles	urlbase/equipo/disponibles/{idsedeconcurso}/{idinst}	2.5

obtenSedesAsignadas	urlbase/sedeconcurso/asignadas/{idconcurso}	2.5
agregaSedeConcurso	urlbase/sedeconcurso	3
eliminaSedeConcurso	urlbase/sedeconcurso/{idsedeconc}	3
obtenEquiposRegistrados	urlbase/equipossedeconcurso/registrados/{idconc}/{idinst}	2.5
registrarEquipoSedeConcurso	urlbase/equipossedeconcurso	3
cancelarEquipoSedeConcurso	urlbase/equipossedeconcurso/{idequiposedeconc}	3

TOTAL DE PUNTOS: 110

PUNTOS EXTRA: 10%

- Si durante el desarrollo del programa realizan commits atómicos (es decir, que representan el cambio de una unidad de código, en este caso un método) que describan de manera adecuada el cambio realizado obtendrán 10 puntos extra

RECUERDEN QUE LOS TRABAJOS DE PROGRAMACION NO SON REEMPLAZABLES LO CUAL SIGNIFICA QUE SI NO SON ENTREGADOS NO HABRA FORMA DE RECUPERAR EL PORCENTAJE DE LA CALIFICACION FINAL QUE LES CORRESPONDE.

La fecha y hora límite para hacer el push final es a las 23:59 hrs del 16 de noviembre del 2020. NO SE ACEPTARÁN PROGRAMAS ENVIADOS POR CORREO ELECTRÓNICO. El nombre que utilice para las clases, atributos y métodos deben seguir las convenciones mencionadas en clase, si usa identificadores tales como xxx, xya, o similares el programa se le asignará una calificación reprobatoria. Finalmente, si se reciben programas copiados, todas las personas que así lo entreguen recibirán calificación reprobatoria **EN LA MATERIA, NO SOLAMENTE EN EL PROGRAMA**.

Una vez verificado que los servicios REST funcionan, empaquen el directorio que contiene su proyecto en un archivo ZIP o RAR habiéndole dado **Clean** antes de empacarlo. El archivo ZIP/RAR debe ser nombrado usando tu apellido paterno y nombre, así por ejemplo el instructor subiría el archivo con nombre SolisRoberto.zip