CODIGO: EIF509
NOMBRE: Desarrollo de Aplicaciones Web
VALOR: 15%
ENTREGA: 23 de Octubre 2017

Escuela de Informática
CICLO: II ciclo 2017
PROFESOR: Maikol Guzmán Alán, MPM

- **Técnicas**:
    - **Leer** cuidadosamente el proyecto
    - Seguir los estándares para presentar proyectos (orden del código, documentación)

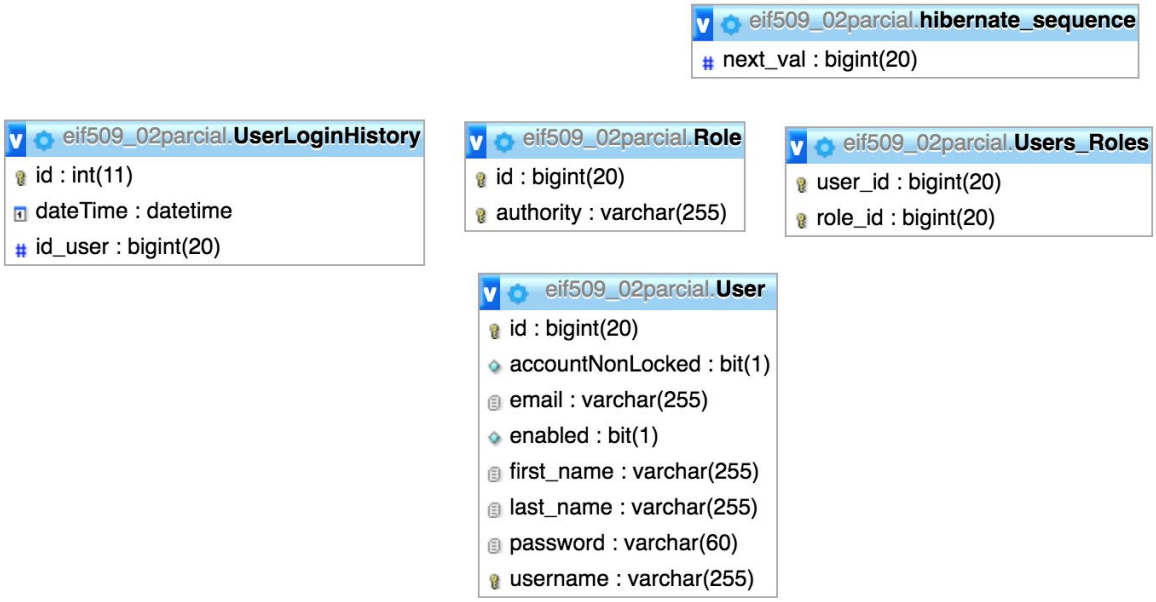
Contenido a evaluar

- Temas incluidos en el examen
    - Servicios
    - Webservice
    - AOP
    - Task / Scheduling
    - DTO
    - Security (Digest)

Descripción del examen

El examen consta de un proyecto base que tiene todas las capas necesarias, objetos, modelos y clases necesarias para desarrollar una aplicación web. El objetivo del examen es concentrarse en desarrollar ese contenido a partir de las secciones que se detallan en este enunciado.

Entidad Relación



Descripción de la aplicación

Se tiene una aplicación desarrollada en capas, el acceso a la misma es por medio de webservices de tipo rest, los entry points son dos únicamente:

- http://127.0.0.1:8080/rest/users, la cual lista los usuarios almacenados en el sistema de la siguiente forma:


```
[
    {
        "id": 1,
        "username": "admin",
        "fullName": "Mike Guzman",
        "lock": "ACTIVO"
    },
    {
        "id": 2,
        "username": "guest",
        "fullName": "Paul Bora",
        "lock": "ACTIVO"
    },
```


- http://127.0.0.1:8888/rest/history, nos retorna una lista de histórico de accesos al sistema por parte de los usuarios, esto quiere decir que por cada intento de acceso al sistema, se registra el intento en la tabla de históricos por medio, este registro se realiza **por medio de una función AOP**, el resultado esperado es:


```
[
    {
        "id": 21,
        "username": "admin",
        "dateTime": 1508527604000
    },
    {
        "id": 20,
        "username": "admin",
        "dateTime": 1508527604000
    }
]
```

[Seguridad Digest]

- Se solicita desarrollar toda la seguridad necesaria para el webservice basada en Digest
    - Implementar la configuración necesaria
    - Desarrollar las clases
        - AppSecurityConfig
        - CustomDigestAuthenticationEntryPoint
            - Real Name: Examen
            - Key: parcial2

[UserLoginHistoryServiceImp]

El objetivo de este service, es registrar los intentos de login al sistema, almacenando el usuario y la fecha y hora del intento

Se solicita lo siguiente:

1. UserLoginHistory add(UserLoginHistory userLoginHistory);
    a. Método necesario para agregar un registro al sistema

2. List<UserLoginHistory> listUserLoginHistory();
    a. Método que retorna la lista total de accesos al sistema

3. int getTotalCountById(Long id);
    a. Método que retorna el total de registro para un usuario en particular, esto quiere decir si un usuario se loguea 3 veces el resultado de este método es 3

4. boolean analyzeLoginUsers();
    a. Este método analiza todos los usuarios que tienen algún registro en el histórico de accesos.
    b. Utiliza el método **getTotalCountById** para determinar cuántas veces un usuario en específico a intentado ingresar al sistema, si la cantidad es igual o mayor a la constante **MAX_LOGING**, se bloquea el usuario.
    c. Si se logró bloquear el usuario el retorno es **true** en caso contrário es **false**

[UserLoginHistoryAspect]

El objetivo de este Aspect es evaluar cuando un usuario hace un intento de acceso, ya sea válido o inválido, y registra el intento en la tabla de históricos.

Se solicita lo siguiente:

1. pointcutAllMethods
    a. Pointcut que apunta al método **loadUserByUsername** del servicio que le corresponde
2. auditUserLoginHistory
    a. Es de tipo **@AfterReturning**
    b. Parámetros (JoinPoint joinPoint, UserDetails userDetails)
    c. La intención es utilizar el objeto **userDetails** y si viene cargado se utiliza con el el método apropiado en el service de **userLoginHistoryService** para registrar el intento de ingreso en la base de datos

[DTO]

El objetivo es crear los DTO's necesarios para el webservice

Se solicita lo siguiente:

1. Crear los DTO's necesarios

[Task / Scheduling]

El objetivo es crear un task que permita utilizar el método de analizar los registro para bloquar al usuario si tiene más de una cantidad de intentos en el sistema. Este task se va a ejecutar cada 1 minuto

Se solicita lo siguiente:

1. Crear un task que se dispare cada 1 minuto
2. El mismo debe de correr el método **analyzeLoginUsers** del service **UserLoginHistoryService**

[WebserviceController]

El objetivo de esta clase es tener los entry points al sistema mediante el webservice.

Se solicita lo siguiente:

1. Completar los métodos GETS
2. Desarrollar los mappings del modelo al DTO
    a. convertUserLoginHistoryToDto
    b. convertUserToDto
3. JSON esperados:

```
[
    {
        "id": 1,
        "username": "admin",
        "fullName": "Mike Guzman",
        "lock": "ACTIVO"
    },
    {
        "id": 2,
        "username": "guest",
        "fullName": "Paul Bora",
        "lock": "ACTIVO"
    },
```

```
[
    {
        "id": 21,
        "username": "admin",
        "dateTime": 1508527604000
    },
    {
        "id": 20,
        "username": "admin",
        "dateTime": 1508527604000
    }
]
```

Evaluación:

Detalle	Puntaje
[BÁSICO] Configuración básica y funcional de todos los frameworks Hibernate, Spring, LogBack, AOP, Test	**2**
[SEGURIDAD] Configurar el sistema para que utilice Digest en los entypoints del webservice	**4**
[SERVICE] Desarrollar los métodos necesarios en la clase **UserLoginHistoryServiceImp**	**6**
[AOP] Desarrollar el aspect necesario para registrar los accesos al sistema en la clase **UserLoginHistoryAspect**	**4**
[DTO] Crear los DTO's necesarios	**2**
[TASK / SCHEDULING] Crear el task necesario y que se ejecute cada 1 minuto para revisar a los usuarios y bloquearlos si superan un límite de intentos	**6**
[WEBSERVICE] Desarrollar los métodos necesarios para los entry points en el **WebserviceController**	**4**

Reglas:

1. El estudiante debe de clonar la base del examen desde el GitHub y trabajar desde la base presentada.
    a. GitHub Link: https://classroom.github.com/a/RPlsJfh3
2. El estudiante debe de hacer el commit en el repositorio indicado y tener claro la fecha final