Projektarbete - Java Webservices 2016

Beskrivning

Under detta projektarbete ska ni exponera ert ärendehanteringssystem som ni byggde under datalagringskursen. Ni kommer att bygga ett webb-api som använder sig av JAX-RS för att exponera ert system över HTTP.

Projektet är betygsgrundande och har två svårighetsgrader **godkänd** och **väl godkänd**. Nivå godkänd gör ni tillsammans som grupp medan nivå väl godkänd gör ni individuellt.

Projektet kommer att genomföras i grupper. Grupperna ska bestå av **minst 3** och **max 5** personer.
Projektet ska redovisas 09:30 **onsdagen den 13 april för nivå Godkänd** och 09:30 **fredagen den 15 april för nivå Väl Godkänd**.

Arkitektur

Allmänt

Koden **MÅSTE** vara formaterad korrekt och får **INTE** innehålla några System.out (förutom i en Main-klass om ni har en sådan). All funktionalitet ska exponeras som webb-tjänster. Det är inte ett krav att använda JPA (eller Spring Data JPA) som lagringsteknik men detta rekommenderas starkt. Kravet är att all data ska sparas i en relationsdatabas (förslagsvis MySQL). JAX-RS måste användas i webb-api:et (använd valfri implementation). I övrigt **MÅSTE** koden vara väl strukturerad och följa en bra objektorienterad design.
Alla beroenden ska hanteras med hjälp av Maven. Ett förslag är att ha två projekt. Ett projekt för datalagret och ett för webbtjänstlagret.

Webbtjänstlager (JAX-RS)
Detta lager kommer att exponera er affärslogik (era service-klasser ni byggde tidigare) som webbtjänster. Detta lager kommer att hantera övergången till och från JSON-formatet. Det betyder att om användaren exempelvis gör en POST med JSON som body kommer detta att göras om till ett Java-objekt som sedan skickas vidare till service/data-lagret. Omvänt betyder det att när webbtjänst-lagret anropar service/data-lagret så kommer webbtjänstlagret att översätta resultatet till JSON som sedan returneras till klienten m.h.a. HTTP.

Krav för Godkänd

(Denna nivå görs som grupp och är obligatorisk för att få betyg i kursen)

Godkända format för endpoints: application/xml **eller** application/json. JAX-B är tillåtet.

User *- en användare i systemet som tillhör ett team*

Funktioner:
- Skapa en User
- Uppdatera en User
- Ta bort* en User
- Hämta en User baserat på user id (inte entity id)
- Söka efter en User baserat på förnamn *eller* efternamn *eller* användarnamn - Hämta alla User som ingår i ett visst team

Team *- en gruppering av User*

Funktioner:
- Skapa ett team
- Uppdatera ett team
- Ta bort* ett team
- Hämta alla team
- Lägga till en User till ett team

Work item - *ett ärende som tilldelas en User*

Funktioner:

- Skapa en work item
- Ändra status på en work item
- Ta bort* en work item
- Tilldela en work item till en User
- Hämta alla work item baserat på status
- Hämta alla work item för ett Team
- Hämta alla work item för en User
- Söka efter work item som innehåller en viss text i sin beskrivning

Issue - en anmärkning som kan ges en work item när den inte accepteras

- Skapa en Issue
- Uppdatera en Issue
- Lägga till en Issue till en work item
- Hämta alla work item som har en Issue

** När ni tar bort en entitet behöver ni fundera på hur detta ska påverka eventuellt relaterade entiteter, dvs, vilken/vilka cascade type(s) ska användas*

Krav för Väl Godkänd

Förutom nivå Godkänd ska du:

- se till att alla endpoints har stöd för JSON utan att JAX-B används. Du använder dig uteslutande av MessageBodyReader och MessageBodyWriter tillsammans med exempelvis Gson för att serialisera och deserialisera data från och till dina endpoints
- ha minsta ett eget Exception som subklassar WebApplicationException som kastas på passande ställe
- ha minst ett Exception som mappas till en Response m.h.a. en ExceptionMapper
- använda JAX-RS Client för att testa alla dina endpoints. Du tar själv redan på hur denna fungerar

Lycka till,
Anders