Szoftverarchitektúrák

12. Sorozat portál
Rendszerterv

Balázs Zoltán (X0ELSN)
Kiss Zoltán (BUS1FJ)

Tartalomjegyzék

1 Rendszerterv ... 3
1.1 Architektúra ... 3
1.2 Komponens diagram ... 3
1.3 Perzisztens réteg ... 5
2 Implementáció ... 7
2.1 Entitások ... 7
2.2 Főbb osztályok ... 7
2.2.1 common ... 7
2.2.2 ejb ... 8
2.2.3 web ... 8
3 Továbbfejlesztési lehetőségek ... 10
3.1 Hiányzó funkciók ... 10
3.2 Fejlesztési lehetőségek ... 10

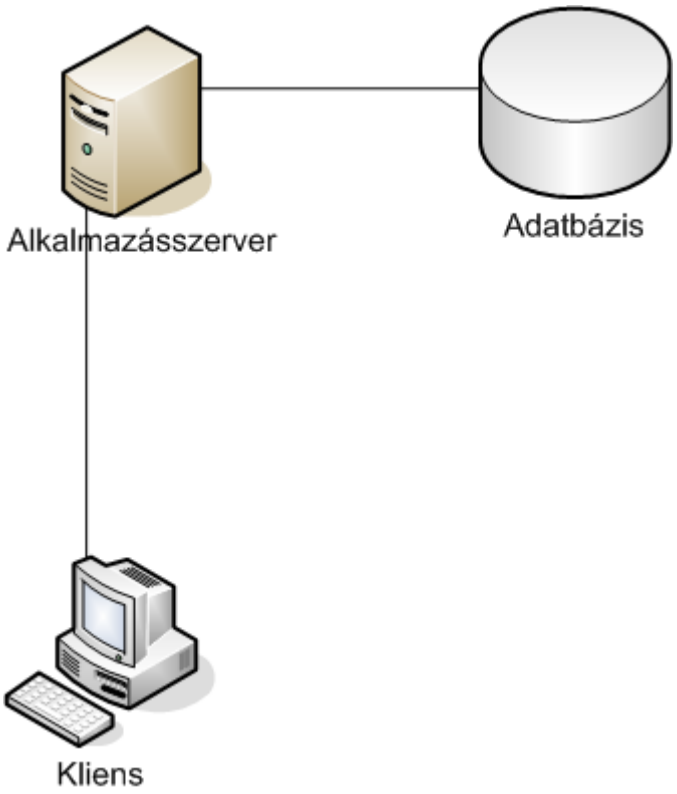
1 Rendszerterv

1.1 Architektúra

A rendszer többrétegű architektúrát valósít meg. Összesen négy logikai réteg különböztethető meg egymástól:

1. *kliens*: A felhasználók a böngészőn keresztül érik el a szerveren futó GWT webes alkalmazást. A felhasználó letölti a weboldalhoz kapcsolódó Javascripteket. A Javascript távoli eljáráshívással hívja a GWT alkalmazás szerveroldali komponenseit.
2. *gwt szerver oldal*: A GWT szerveroldali komponensek JNDI lookup használatával megkeresik a szükséges EJB szolgáltatásokat.
3. *üzleti logika*: Az EJB session beanek implementálják a szolgáltatásokat, és használják a DAO réteget.
4. *adatkezelés*: A DAO réteg végzi az adatok perzisztálását

A 2., 3. és 4., réteg fizikailag ugyanazon a szerveren található meg, ugyanabban az alkalmazásban. Az adatbázis az alkalmazás szempontjából lényegtelen, hogy különböző szerveren található-e, az ábrán ezért mint külön architektúrális elem lett szemléltetve.



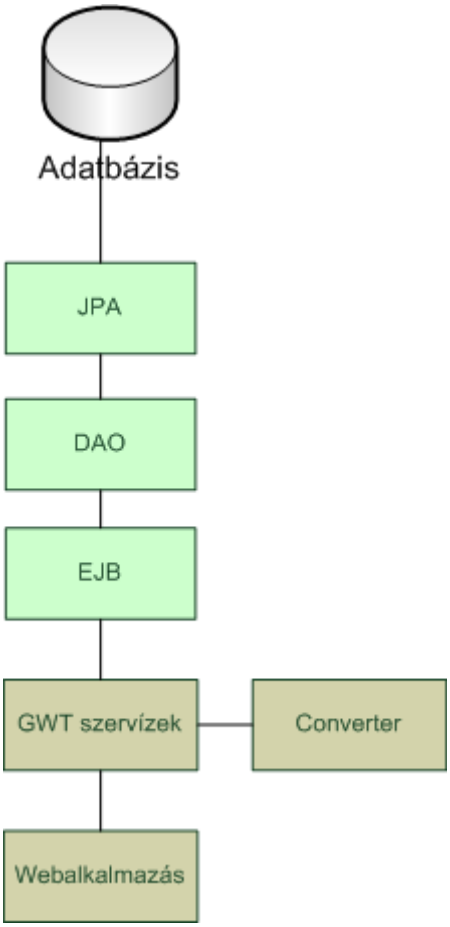
A rendszer architektúrája

1.2 Komponens diagram

A rendszer komponensei a következők:

- Webalkalmazás: maga a felhasználói felület, a felhasználó böngészőjében megjelenő tartalom

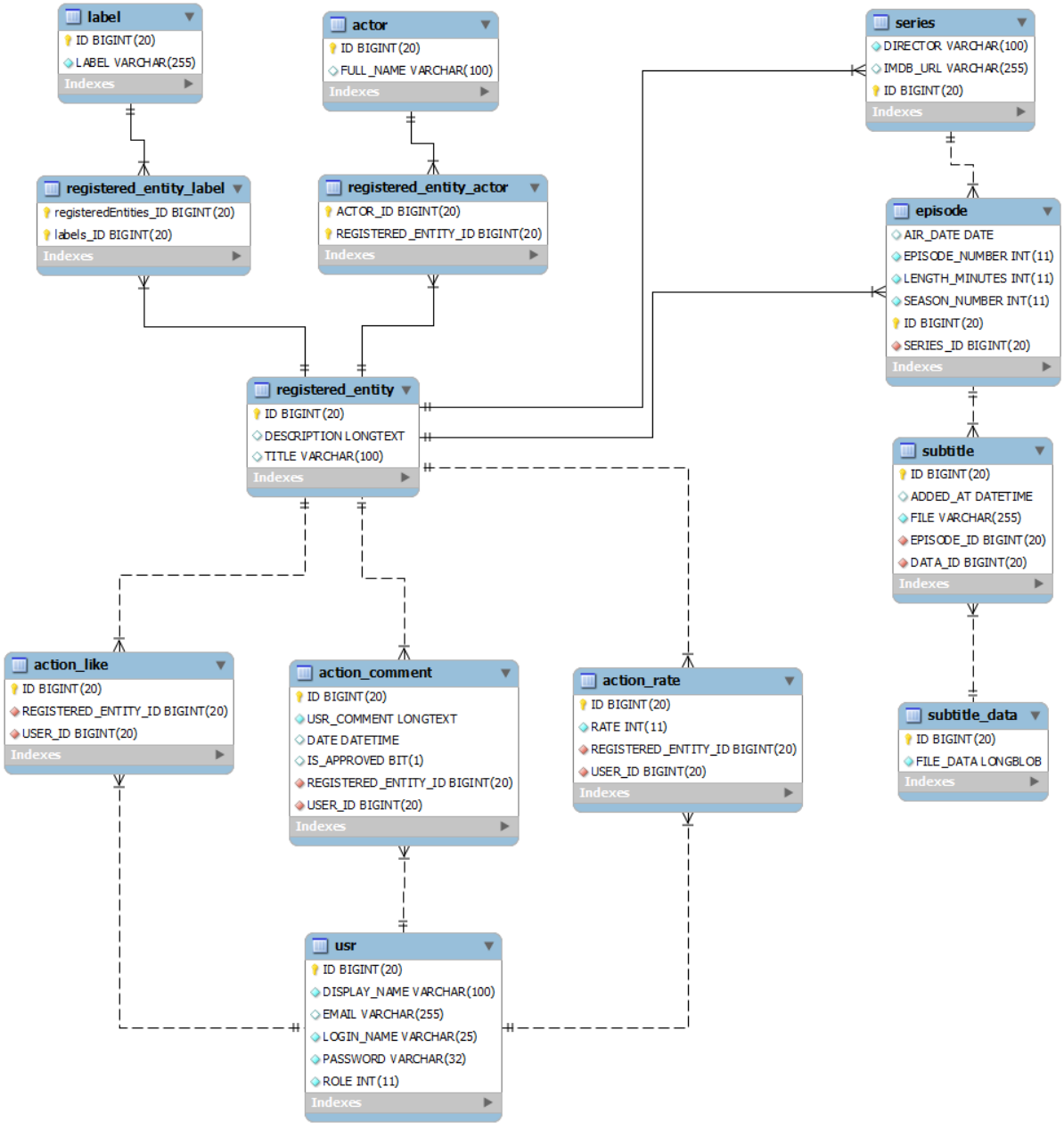
- GWT szervízek: a webalkalmazás része, de funkcionálisan jól elkülönül. Az EJB hívását oldják meg a GWT oldalról.
- Converter: Mivel a GWT úgy működik, hogy JavaScript-et generál a Java források alapján, ezért nem tudja lefordítani az annotált JPA entitás osztályokat. Ezért szükség van arra, hogy GWT oldalon létrehozzuk a JPA entitások annotálatlan másolatait (DTO-kat). A DTO-k és a JPA entitások közötti konverziót végzi ez az osztály.
- EJB: a rendszer funkcionalitását nyújtja.
- DAO: az adatok kezeléséért felelős funkciókat valósítja meg.
- JPA: a DAO réteg JPAt használ az adatbázis eléréséhez. Jelen esetben Hibernate-et használunk JPA implementációként.
- Adatbázis: az adatbázis JPA-val van elfedve a könnyű hordozhatóság elősegítése miatt.



A rendszer komponensei

1.3 Perzisztens réteg

Az adatbázis séma a következő:



Tábla neve	**Tárolt adatok jellege**
LABEL	Címkék
ACTOR	Színészek
REGISTERED_ENTITY_LABEL	Kapcsolótábla
REGISTERED_ENTITY_ACTOR	Kapcsolótábla
REGISTERED_ENTITY	A sorozat és az epizódok közös őse
SERIES	Sorozatok
EPISODE	Epizódok
SUBTITLE	Feliratok
SUBTITLE_DATA	Felirat fájl tartalma
USR	Felhasználók
ACTION_LIKE	Lájkolás
ACTION_COMMENT	Kommentelés
ACTION_RATE	Értékelés

2 Implementáció

2.1 Entitások

Az implementáció során Object-Relation Mapping technológiaként JPA-t használtunk. A JPA entitásaink nevei és funkciói nagyon hasonlóak a nekik megfelelő adatbázisbeli táblák nevéhez.

A JPA entitások a common projektben találhatóak meg. Az entitások alapvetően JavaBean-ek, a tulajdonságok neveinél törekedtünk az egyértelműségre, így azok különösebb magyarázatra nem szorulnak.

- Actor: A színészek objektuma
- Comment: A kommenteléseket reprezentálja
- EntityBase: Minden entitás őse. Megvalósítja azt a viselkedést, hogy minden entitás csak akkor számít ekvivalensnek, ha az ID-jűk megegyezik.
- Episode: A sorozatok epizódjai
- Label: A címkék
- Like: Lájkolás
- Rate: Értékelés
- RegisteredEntity: A sorozat és epizód közös őse. Azok a kapcsolatok és tulajdonságok itt vannak leírva, amik egyaránt érvényesek mindkét osztályra
- Series: A sorozatok
- Subtitle: A feliratokat reprezentáló entitás
- SubtitleData: A felirat fájlokat reprezentálja. Azért lett kiemelve a Subtitle entitásból, mert a Hibernate nem támogatja a primitív típusok Lazy fetching-jét.
- User: A felhasználókat reprezentálja

Látható, hogy pár táblának nincsen megfelelteltve entitás, ilyenek pl a kapcsolótáblák. Ezeket a táblákat a JPA automatikusan legenerálja és karban tartja.

2.2 Főbb osztályok

2.2.1 common

A common projektben levő osztályok olyan osztályok, amelyekre egyaránt szüksége van az összes többi projektnek. Ilyen osztályok például a JPA entitások is.

2.2.1.1 data.persistent csomag

Itt találhatóak az előbb már specifikált JPA entitás osztályok

2.2.1.2 service csomag

Az EJB szervízzel kapcsolatos osztályok és interfészek vannak itt. Például az EJB által dobható Exception leszármazottak és az EJB publikus interfésze.

SeriesPortal interfész: definiálja az EJB publikus interfészét. Metódusai segítségével az összes specifikációban elvárt funkció megvalósítható. Főbb metódusok:

- approveComment: Komment jóváhagyása, adminisztrátori funkció

- changeUserPassword: felhasználó jelszavának módosítása
- comment: kommentelés
- downloadSubtitle: felirat fájlt leíró SubtitleData lekérése
- like: lájkolás
- listSeriesPaged: sorozatok név szerint rendezett lapozokra bontott listázása
- listTopRatedSeries: a legjobbra értékelt sorozatok lapozott listázása
- listUnapprovedComments: jóvá nem hagyott kommentek listázása
- login: felhasználó bejelentkeztetése
- rate: sorozat vagy epizód értékelése
- register: új felhasználó regisztrálása
- save(Episode): (új) epizód mentése
- save(Series): (új) sorozat mentése
- searchByActors: színészek szerinti keresés
- searchByDescription: keresés leírás alapján
- searchByLabels: keresés címkék alapján
- searchByTitle: keresés cím alapján
- searchByDirector: keresés rendező alapján
- uploadSubtitle: felirat feltöltése

2.2.2 ejb

Az ejb projektben találhatók azok az osztályok, amelyek az EJB funkcionalitásáért felelősek. Az EJB funkcionalitását a common projekt-beli SeriesPortal interfész definiálja.

Gyakorlatilag két ilyen osztály van, a SeriesPortalImpl és a SeriesPortalDao.

2.2.3 web

A web projektben szereplő osztályok feladata a megjelenítés, a bevitt adatok formátumának ellenőrzése. A projekthez tartoznak még többek között képek, leíró xml-ek és egyéb erőforrások.

A web projekt GWT technológiával készül, emiatt az osztályokat három fő csomagban helyezzük el. A client nevű csomagban szerepelnek azok az osztályok, amelyeket csak kliens oldalon használunk fel. A server nevű csomagba kerülnek azok az osztályok, amelyeket csak a GWT szerver oldal használ. Végül a shared csomagba kerülnek azok az osztályok, amelyeket mindkét oldalon felhasználunk.

2.2.3.1 client csomag

A client csomagban kapnak helyet az RPC hívásokhoz szükséges szinkron és aszinkron szolgáltatás interfészek. Ebből az alkalmazás elkészítéséhez kettőt hozunk létre:

- UserService, UserServiceAsync: a felhasználók kezeléséhez kapcsolódó szolgáltatás szinkron, illetve aszinkron interfésze.
- RegisteredEntityService, RegisteredEntityServiceAsync: a sorozathoz és epizódokhoz kapcsolódó szolgáltatások szinkron és aszinkron interfésze.

A client.ui csomagba kerülnek azok az osztályok, amelyek a webes felület kialakításáért, illetve a felhasználók akcióinak kezeléséért felelősek. A főbb osztályok, és szerepeik:

- WebMainPanel: Az alkalmazás fő panelje, ő tartalmaz minden további elemet.

- ContentPanel: Az aktuálisan elkért tartalmat jeleníti meg. DeckPanel ősből származik, azaz több panelt tartalmazhat, amelyek közül, mindig csak egy látható.
- ApprovablePanel: Csomagoló panel, ami az átadott másik panel alá egy, esetleg két gombot helyez el. Tipikusan formok esetében használjuk.
- ApprovableCancelDialogBox: Csomagoló panel, egy felugró dialógusablakhoz. Két gombot tartalmaz. A bejelentkezéshez használt LoginPanelt és a regisztrációhoz használt RegisterPanelt csomagoljuk vele.
- ListPanel: A ContentPanel által megjelenített egyik felület. Sorozatok listázására alkalmas. A hozzá tartozó felületen megadható, hogy hogyan szeretnénk rendezni a megjelenő listákat (abc szerint, értékelés szerint, stb.)
- RegisteredEntityPanel: egy sorozat, vagy egy epizód teljes megjelenítését végzi. A ContentPanel része.

A client csomag részét képezi továbbá egy saját komponens (MultipleTextInput), amelynek segítségével szövegek listáját adhatja meg a felhasználó. Segítségével lehetőség nyílik pl. a sorozathoz kapcsolódó színészek egyszerű megadására. Sajnos hasonló, előre legyártott komponens nincs még a GWT-ben.

Végül a client csomaghoz tartozik még a WebImageBundle osztály, amely a megadott képekből egy bundle-t készít, és azokat elérhetővé teszi, a felület létrehozásakor.

2.2.3.2 server csomag

A server csomagba kerül a ServiceLocator osztály, amely az session bean interfészének megszerzésére szolgál.

Emellett az említett két szolgáltatás (a felhasználókhoz és sorozatokhoz, epizódokhoz kapcsolódóak) megvalósítása kerül ide. A nevük UserServiceImpl és RegisteredEntityServiceImpl.

A server.converter csomagba kerülnek azok az osztályok, amik az annotált EJB entitások és a GWT kliensoldalon használt objektumok közötti átalakítást elvégzik. Erre azért van szükség, mert a GWT kliens oldalán nem szerepelhetnek ilyen osztályok, mivel az a fordító nem képes ezeket Javascriptté alakítani.

2.2.3.3 shared csomag

A shared csomagba kerülnek az EJB entitásoknak megfelelő adattároló osztályok, illetve kliens és szerver rétegben közösen használt kivételek. Az adattároló osztályok pontosan megfelelnek az EJB rétegben bemutatott entitásoknak, kettejük között a konverziót a Converter osztály végzi a GWT szolgáltatás szerveroldali részén.

3 Továbbfejlesztési lehetőségek

3.1 Hiányzó funkciók

Időhiány miatt a specifikált funkciók közül kimaradt a webalkalmazásból az adatbázisban történő keresés, a kommentelés és a feliratok fel- és letöltése. Az EJB-ben azonban a teljes funkcionalitás implementálva lett.

3.2 Fejlesztési lehetőségek

Jelenleg nincs lehetőség a tárolt sorozatok és epizódok megadott adatainak módosítására, törlésére. Utóbbi adminisztrátori feladat kellene, hogy legyen.

Szintén szükség lenne arra, hogy új sorozatok és epizódok létrehozása csak adminisztrátori jóváhagyással történhessen.

Sokkal kellemesebbé tenné az alkalmazás használatát, ha a sorozatokhoz, epizódokhoz képet is fel lehetne tölteni, listázásakor ezek közül egy, plakátként megjelenne.

Szükség lenne az adatbázis séma bővítésére is, ugyanis egy sorozathoz több rendező tartozhat, illetve nem tároljuk a sorozatok műfaját.