Om te beginnen staan er in elke (behalve de tussentabel UserRole) 3 velden die een simpele auditrol vervullen: aan status kan een teken worden meegeven dat correspondeert met een bepaalde elders geselecteerde status, lastUpdateBy en lastUpdateTimestamp houden bij wie en wanneer de laatste wijziging aan een bepaald record is aangemaakt.

De **user** tabel bevat enkele standaard velden voor een gebruiker. Daarnaast hebben we hier een veld 'department' aan toegevoegd als String. Aangezien WantedSubscriber deze ook gebruikt moeten we overwegen om hier een apart model voor te maken hiernaar te verwijzen met zijn id. Ook is er salt-string die instaat voor een betere beveiliging.

De **Role** tabel bevat enkel een roleId en een roleName.

De tabel **UserRole** fungeert enkel als tussentabel op de many to many relatie tussen Role en User te kunnen leggen. De 3 eerder vernoemde kolommen toevoegen heeft mogelijk geen zin aangezien deze niet aangesproken kunnen worden in Hibernate aangezien er voor deze tabel geen klasse wordt aangemaakt. Foreign keys gelegd naar User- en Role-tabellen

De **Picture** tabel heeft een pictureId, legt een fk-link met de Project-tabel en slaat een URL op. Deze laatste is momenteel ingesteld op 512 tekens, maat dit moet besproken worden.

De **Movie** tabel heeft is ongeveer hetzelfde opgebouwd, maar url is vervangen door youTubeId om redundante informatie te vermijden.

**Project** bevat de belangrijkste informatie van het project, gaande van een id, titel en description tot dingen zoals belangrijke data, een status in de vorm van een enum en 2 boolean flags. Fk’s naar Category en User.

De **Like** tabel bevat een id en 2 fk’s naar User en Project. Verder niets significants.

**Category** bevat ook niets meer dan een id en een description.

**Subscriber** bevat net zoals Like enkel een id en 2 fk’s naar User en Project.

**WantedSubscriber** heeft naast een id en een fk naar Project de eerder vernoemde verwijzing naar department en een counter.