Completamento delle specifiche

* La lunghezza massima per i seguenti campi è:
  + Percorso in cui è conservata un’immagine : 512 caratteri
  + Nome utente : 128 caratteri
  + E-mail : 128 caratteri
  + Titolo di un’immagine e di un album : 128 caratteri
  + Descrizione di un’immagine : 1024 caratteri
* La pagina di creazione dei nuovi album è accessibile dalla HOME\_PAGE, che porta ad una ALBUM\_EDIT\_PAGE con nome vuoto, in cui inserire le immagini di quell’utente non ancora associate a nessun album
* In ALBUM\_PAGE sarà presente un pulsante per modificare l’album corrente, che porterà ad ALBUM\_EDIT\_PAGE, la quale avrà il campo “Titolo” già riempito con il nome precedente, e che sarà modificabile per cambiare il titolo dell’album
* In caso di parametri nella request errati, il comportamento varia, ma è solitamente una redirezione alla HOME\_PAGE in caso di errori difficilmente reversibili, o alla schermata valida più vicina alla richiesta errata. Ad esempio, se nella ALBUM\_PAGE l’id fosse errato, il comportamento di default è aprire l’album più recente dell’utente
* Formati d’immagine ammessi: png – jpg – webp
* Le immagini sono salvate all’interno della cartella webapp del server, in quanto non è richiesto l’upload di immagini da parte dell’utente.  
  Questa scelta facilita l’implementazione, permettendo l’uso di getResourceAsStream anche per inviare le immagini, anche se non sarebbe utilizzabile in un’applicazione reale, in quanto il server dovrebbe esser fatto ripartire per osservare i cambiamenti in questa cartella

Schema E/R

L’entità “Commento” è debole rispetto all’entità Immagine in quanto la creazione di un commento non può avvenire se non associata a una specifica immagine. Inoltre, abbiamo supposto che, all’eventuale cancellazione di un’immagine (non richiesta nel progetto), tutti i suoi commenti vengano a loro volta eliminati.

Album non debole rispetto a utente perché…

Componenti

* Data Access Objects
  + UserDAO
    - createUser
    - getUserFromUsername
    - checkCredentials
  + ImageDAO
    - getImageFromId
    - getImagesInAlbum
    - getImagesOfUser
  + AlbumDAO
    - addImageToAlbum
    - createAlbum
    - getAlbumFromId
    - getAllAlbums
  + CommentDAO
    - createComment
    - getAllCommentsForImage
    - getCommentFromProgressiveImageId
* Oggetti di modello (Beans)
  + Album
  + Comment
  + Image
  + User
* Viste e componenti
  + LoginPage
    - Login form
    - Signup form
  + HomePage
    - User album list
    - Other users’ album list
  + AlbumPage
    - Thumbnail list
    - Full-sized image
    - Comment list
    - New comment form
      * Text box
  + AlbumEditPage
    - Album Edit Form
      * Title
      * N checkboxes (one for each image uploaded by the user)
* Controller (Servlets)
  + SignupCheck
  + LoginCheck
  + Logout
  + CreateNewComment
  + CreateEditAlbum
  + GoToLoginPage
  + GoToHomePage
  + GoToAlbumPage
  + GoToAlbumEditPage
* Filtri
  + LoggedFilter