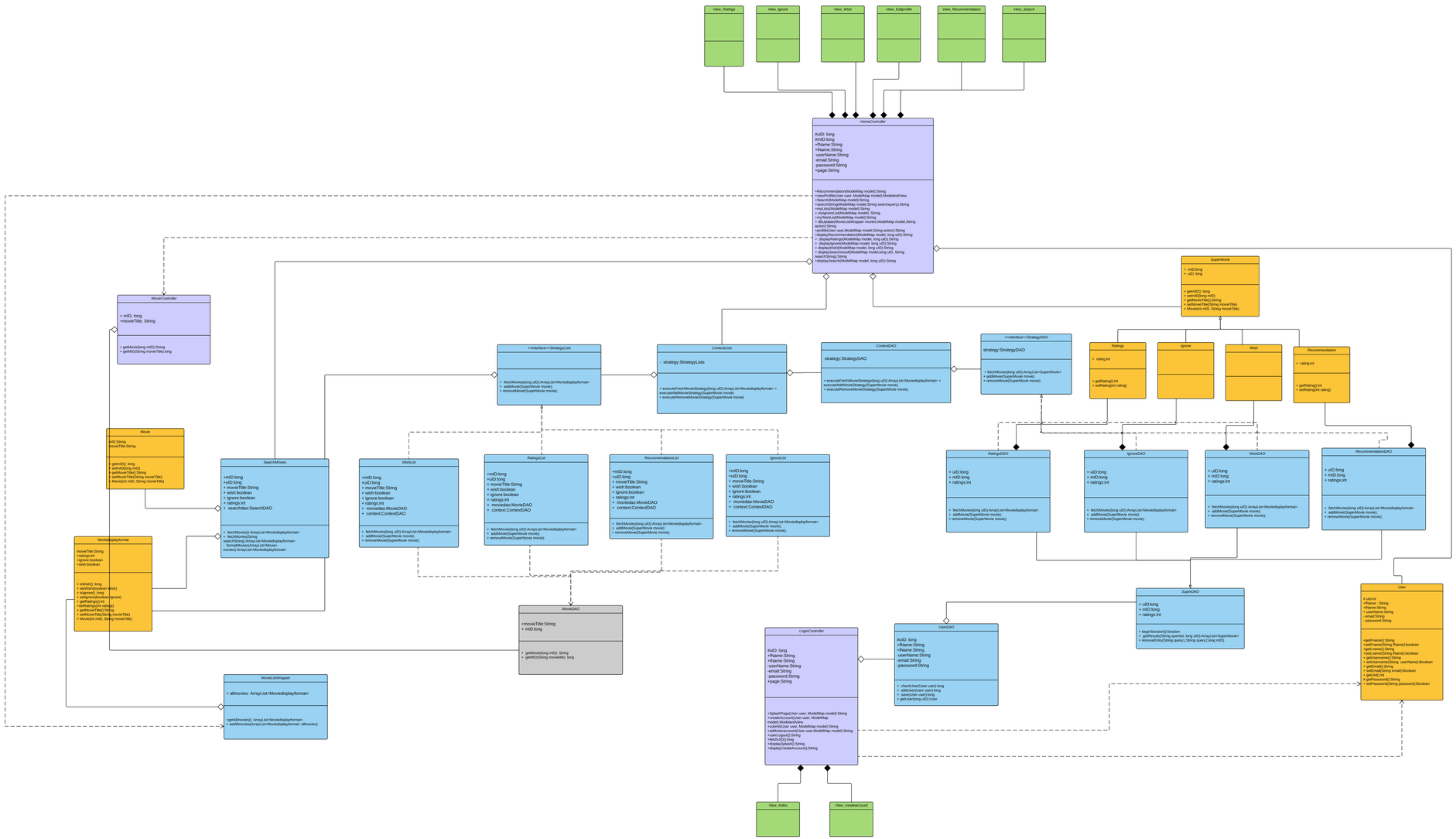
View_Ratings
View_Ignore
View_Wish
View_Editprofile
View_Recommendation
View_Search
HomeController
#uID: long
#mID:long
+fname:String
+lname:String
-userName:String
-email:String
-password:String
+page:String
+Recommendation(ModelMap model):String
+ viewProfile(User user, ModelMap model):ModelandView
+Search(ModelMap model):String
+search(String|ModelMap model,String searchquery):String
+myList(ModelMap model):String
+ myignoreList(ModelMap model): String
+myWishList(ModelMap model):String
+displayMovieList(MoviesWrapper movies,ModelMap model,String action):String
+profile(User user,ModelMap model,String action):String
+displayRating(ModelMap model, long uID, long uID):String
+displayRating(ModelMap model, long uID):String
+ displayIgnore(ModelMap model, long uID):String
+ displayWish(ModelMap model, long uID):String
+ displaySearchresult(ModelMap model,long uID, String searchString):String
+displaySearch(ModelMap model, long uID):String
MovieController
+ mID: long
+movieTitle: String
+ getMovie(long mID): String
+ getMID(String movieTitle):long
SuperMovie
+ mID:long
+ uID: long
+ getmID(): long
+ setmID(long mID)
+ getMovieTitle():String
+ setMovieTitle(String movieTitle)
+ Movie(int mID, String movieTitle)
<<Interface>>StrategyLists
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
ContextLists
- strategy:StrategyLists
+ executeFetchMovieStrategy(long uID): ArrayList<Moviedisplayformat> +
execute(AddMovieStrategy(SuperMovie movie)
+ execute(RemoveMovieStrategy(SuperMovie movie)
ContextDAO
-strategy:StrategyDAO
+ executeFetchMovieStrategy(long uID): ArrayList<Moviedisplayformat> +
execute(AddMovieStrategy(SuperMovie movie)
+ execute(RemoveMovieStrategy(SuperMovie movie)
<<Interface>>StrategyDAO
strategy:StrategyDAO
+ fetchMovies(long uID):ArrayList<SuperMovie>
+ addMovie(SuperMovie movie)
+ removeMovie(SuperMovie movie)
Ratings
+ rating:int
+ getRating():int
+ setRating(int rating)
Ignore
Wish
Recommendation
+ rating:int
+ getRating():int
+ setRating(int rating)
Movie
mID:String
movieTitle:String
+ getmID(): long
+ setmID(long mID)
+ getMovieTitle():String
+ setMovieTitle(String movieTitle)
+ Movie(int mID, String movieTitle)
SearchMovies
+mID:long
+uID:long
+ movieTitle:String
+ wish:boolean
+ ignore:boolean
+ ratings:int
+ searchdao:SearchDAO
+ fetchMovies():ArrayList<Moviedisplayformat>
+ fetchMovies(String searchString):ArrayList<Moviedisplayformat>
+ formatMovies(ArrayList<Movie> movies):ArrayList<Moviedisplayformat>
Moviedisplayformat
movieTitle:String
+ratings:int
+ignore:boolean
+wish:boolean
+ isWish(): long
+ setWish(boolean Wish)
+ isIgnore(): long
+ setIgnore(boolean Ignore)
+ getRatings():int
+ setRatings(int Ratings)
+ getMovieTitle():String
+ setMovieTitle(String movieTitle)
+ Movie(int mID, String movieTitle)
WishList
+mID:long
+uID:long
+ movieTitle:String
+ wish:boolean
+ ignore:boolean
+ ratings:int
+ moviedao:MovieDAO
+ context:ContextDAO
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
RatingsList
+mID:long
+uID:long
+ movieTitle:String
+ wish:boolean
+ ignore:boolean
+ ratings:int
+ moviedao:MovieDAO
+ context:ContextDAO
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
RecommendationList
+mID:long
+uID:long
+ movieTitle:String
+ wish:boolean
+ ignore:boolean
+ ratings:int
+ moviedao:MovieDAO
+ context:ContextDAO
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
IgnoreList
+mID:long
+uID:long
+ movieTitle:String
+ wish:boolean
+ ignore:boolean
+ ratings:int
+ moviedao:MovieDAO
+ context:ContextDAO
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
MovieDAO
+movieTitle:String
+ mID:long
+ getMovie(long mID): String
+ getMID(String movietitle): long
MoviesWrapper
+ allmovies: ArrayList<Moviedisplayformat>
+getAllmovies(): ArrayList<Moviedisplayformat>
+ setAllmovies(ArrayList<Moviedisplayformat> allmovies)
RatingsDAO
+ uID:long
+ mID:long
+ ratings:int
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
IgnoreDAO
+ uID:long
+ mID:long
+ ratings:int
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
WishDAO
+ uID:long
+ mID:long
+ ratings:int
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
RecommendationDAO
+ uID:long
+ mID:long
+ ratings:int
+ fetchMovies(long uID):ArrayList<Moviedisplayformat>
+ addMovie(SuperMovie movie);
+ removeMovie(SuperMovie movie);
SuperDAO
+ uID:long
+ mID:long
+ ratings:int
+ beginSession():Session
+ getResults(String queried, long uID):ArrayList<SuperMovie>
+ removeEntry(String query1,String query2,long mID)
User
uid:int
+Name : String
+ fname:String
+ username:String
- email:String
- password:String
+ getName():String
+ setName(String fname):boolean
+ getLname():String
+ setLname(String lfname):boolean
+ getUsername():String
+ setUsername(String username):Boolean
+ getEmail():String
+ setEmail(String email):Boolean
+ getUID():int
# getPassword():String
+ setPassword(String password):Boolean
LoginController
#uID: long
+fname:String
+lname:String
-userName:String
-email:String
-password:String
+page:String
+SplashPage(User user, ModelMap model):String
+createAccount(User user, ModelMap model):ModelandView
+ viewProfile(User user, ModelMap model):String
+addUserAccount(User user,ModelMap model):String
+userLogout():String
+fetchUID():long
+DisplaySplash():String
+DisplayCreateAccount():String
UserDAO
#uID: long
+fname:String
+lname:String
-userName:String
-email:String
-password:String
+ checkUser(User user):long
+ addUser(User user):long
+ save(User user):long
+ getUser(long uID):User
View_index
View_createaccount