FULL Photo Album test specification

|  |  |
| --- | --- |
| Atom | Scope |
| Photo | 5 |
| Name | 4 |
| Password | 4 |
| User | 4 |
| Group | 4 |

|  |
| --- |
| Enumeration |
| Report |
| ALBUM\_FULL |
| PHOTO\_EXISTS |
| OK |
| NOT\_AUTH |
| ALREADY\_IN |
| MISSING\_USERS |
| NO\_PHOTO |
| REM\_OWNER |
| REM\_OWNER\_GROUP |
| NO\_GROUP |
| AUTH\_FAILED |

|  |  |
| --- | --- |
| State variable | Type |
| photoAt | seq Photo |
| ownerName | Name |
| loggedIn | set Name |
| groups | Name -> Group |
| users | Name -> User |
| members | Group -> User |
| passwords | User -> Password |
| groupPhotos | Photo -> Group |
| ownerGroupName | Name |

|  |  |
| --- | --- |
| Invariant | Name |
| #photoAt.elems = #photoAt | Disjoint |
| #photoAt <= 3 | Bounded Size |
| lone loggedIn | At most one user logged in |
| members in ran[groups] -> ran[users] | members can only involve users and groups in the system |
| passwords in ran[users] -> one Password | users must have exactly one password |
| groupPhotos in Photo -> ran[groups] | group photos are valid |

|  |  |
| --- | --- |
| Operation |  |
| init |  |
| OWNER : User, OWNER\_PASSWD : Password, OWNER\_NAME : Name, OWNER\_GROUP\_NAME : Name, OWNER\_GROUP : Group | true |
| photoAt | none -> none |
| ownerName | OWNER\_NAME |
| loggedIn | none |
| passwords | OWNER -> OWNER\_PASSWD |
| users | OWNER\_NAME -> OWNER |
| groups | OWNER\_GROUP\_NAME -> OWNER\_GROUP |
| members | OWNER\_GROUP -> OWNER |
| ownerGroupName | OWNER\_GROUP\_NAME |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Operation |  |  |  |  |
| addPhoto |  |  |  |  |
| p:Photo, owner\_group:Group,  report!:Report | ownerName in loggedIn | | | ownerName !in loggedIn |
| #photoAt < 3 | | #photoAt >= 3 | true |
| p !in ran[photoAt] | p in ran[photoAt] | true | true |
| photoAt | photoAt.add[p] | = | = | = |
| groupPhotos | groupPhotos + p -> owner\_group | = | = | = |
| report! | OK | PHOTO\_EXISTS | ALBUM\_FULL | NOT\_AUTH |

|  |  |  |
| --- | --- | --- |
| Operation |  |  |
| viewPhotos |  |  |
| result!:set Photo, report!:Report | one loggedIn | no loggedIn |
| result! | photoAt.elems | none |
| report! | OK | NOT\_AUTH |

|  |  |  |  |
| --- | --- | --- | --- |
| Operation |  |  |  |
| removePhoto |  |  |  |
| i : Int, report!:Report | ownerName in loggedIn | | ownerName !in loggedIn |
| i in dom[photoAt] | i !in dom[photoAt] | true |
| photoAt | photoAt.delete[i] | = | = |
| report! | OK | NO\_PHOTO | NOT\_AUTH |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Operation |  |  |  |  |
| login |  |  |  |  |
| n:Name,p:Password,report!:Report | no loggedIn | | | some loggedIn |
| n in dom[users] | | n !in dom[users] | true |
| passwords[users[n]] = p | passwords[users[n]] != p | true | true |
| loggedIn | loggedIn + n | = | = | = |
| report! | OK | AUTH\_FAILED | AUTH\_FAILED | ALREADY\_IN |

FineFit found a bug in the updateUser specification. Originally the yellow colored cell contained '=' but this is obviously a mistake because when we add a new user we must also add their password.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Operation |  |  | |  |
| updateUser |  |  | |  |
| n:Name,p:Password,u:User,report!:Report | ownerName in loggedIn | | | ownerName !in loggedIn |
| n in dom[users] | | n !in dom[users] | true |
| true | | u in User - ran[users] | true |
| users | = | | users + n -> u | = |
| passwords | passwords ++ (users[n] -> p) | | passwords + (users[n] -> p) | = |
| report! | OK | | OK | NOT\_AUTH |

FineFit found a bug in the updateGroup specification. Originally the yellow colored cell contained '=' but this is a mistake because when we add a new group we must also add its members to the members variable.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operation |  | |  |  |  |
| updateGroup |  | |  |  |  |
| n:Name,nu:Name,g:Group,report!:Report | ownerName in loggedIn | | | | ownerName !in loggedIn |
| nu in dom[users] | | | nu !in dom[users] | true |
| n in dom[groups] | n !in dom[groups] | | true | true |
| true | g in Group - ran[groups] | | true | true |
| groups | groups | groups + n -> g | | = | = |
| members | members + groups[n] -> users[nu] | members + g -> users[nu] | | = | = |
| report! | OK | OK | | MISSING\_USERS | NOT\_AUTH |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ignore Operation |  |  |  |  |
| removeUser |  |  |  |  |
| n:Name, report!:Report | ownerName in loggedIn | | | ownerName !in loggedIn |
| n in dom[users] | n !in dom[users] | | true |
| n != ownerName | n = ownerName | true | true |
| users | (Name-n) <: users | = | = | = |
| members | members :> (User -users[n]) | = | = | = |
| report! | OK | REM\_OWNER | MISSING\_USERS | NOT\_AUTH |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Operation |  | |  |  |  |
| removeGroup |  | |  |  |  |
| n : Name, report! : Report | ownerName in loggedIn | | | | ownerName !in loggedIn |
| n != ownerGroupName | | | n = ownerGroupName | true |
| n in dom[groups] | n !in dom[groups] | | true | true |
| groupPhotos | groupPhotos :> (Group - n) | = | | = | = |
| photoAt | squash[photoAt :> groupPhotos.(Group-n)] | = | | = | = |
| report! | OK | NO\_GROUP | | REM\_OWNER\_GROUP | NOT\_AUTH |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Operation |  |  |  |  |
| updatePhotoGroup |  |  |  |  |
| i : Int, n : Name, report! : Report | ownerName in loggedIn | | | ownerName !in loggedIn |
| i in dom[photoAt] | | i !in dom[photoAt] | true |
| n in dom[groups] | n !in dom[groups] | true | true |
| groupPhotos | groupPhotos ++ photoAt[i] -> groups[n] | = | = | = |
| report! | OK | NO\_GROUP | NO\_PHOTO | NOT\_AUTH |