Mobile Application Store Authentication Service Design Document

CSCI E-97 Assignment 4

Mobile Application Store

Authentication Service Design Document

Date: 11/15/2013

Author: Siddharth SINGH

Reviewer(s): Hemant BAJPAI

Introduction

This document provides problem description in overview section, defines requirement and use cases, design in form of UML class diagram, class dictionary details out methods and association of classes used in the implementation and explains how application is tested.

Overview

Mobile application store is source for application, ringtones and wallpapers. Authentication service provides ways to control access to various users according to their role and permissions. Authentication service manages login/logout, users and their roles and permissions and services. Each user has a set of roles or permissions and user can perform designated tasks in the mobile store defined by their roles and permissions.

Requirements

1. Create Service: Create service give its id name and description.
2. Create Permission: Create permission in authentication service. A permission is associated with a service. Create a permission given its id name and description. Attach it to an existing service id of is given.
3. Create Role: A role composes set of permissions or nested roles. Create role given its id, name and description.
4. Add entitlement to role: An entitlement is either a role or permission. Associate the given entitlement to an existing role.
5. Create User: Create user in authentication service. A user has an id, name and description.
6. Add credential to user: Associate a login name and password with the given user.
7. Add entitlement to user: Assign role and permission to user.

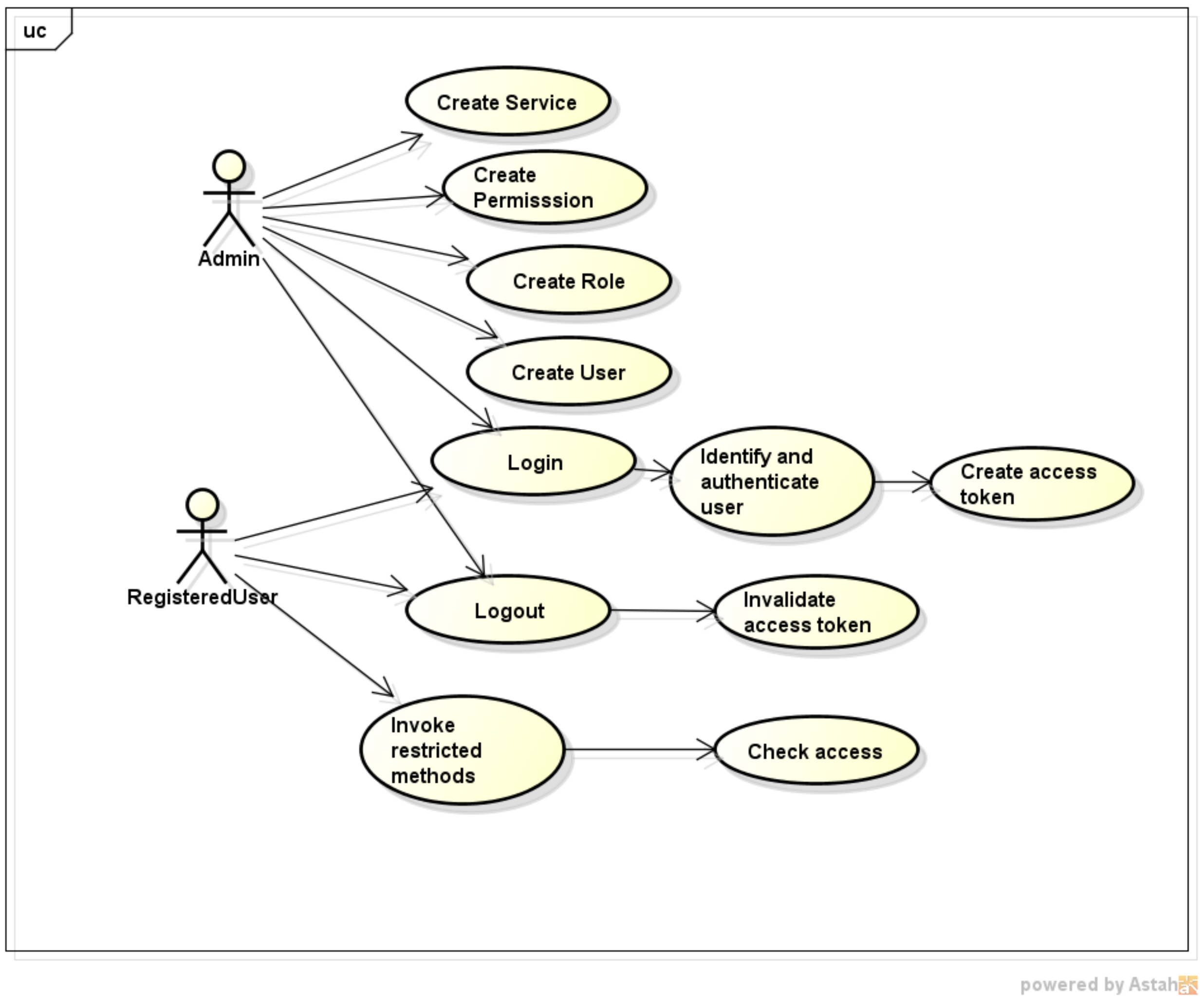
8. Login: login using a login name and password.
9. Logout: Logout form mobile application store service.
10. Check Access: Checks if a user has a permission to invoke given service.

Use Cases

There are two types of users:

1. An administrator can create services, users, roles, permissions and can invoke restricted methods.
2. Non admin users can invoke allowed restricted methods.

Both admin and non admin users logs in into the system. Authentication service identifies the user and authenticates them to invoke certain services or methods. Authentication service assigns a access token to logged in user. Access token has a valid or invalid state and an expiration time. Token becomes invalid when user logs out or when token expires. Authentication service throws exception if login fails, if user tries to invoke a restricted method for which user is not granted the permission or when user tries to invoke restricted method and access token has become invalid.



Implementation

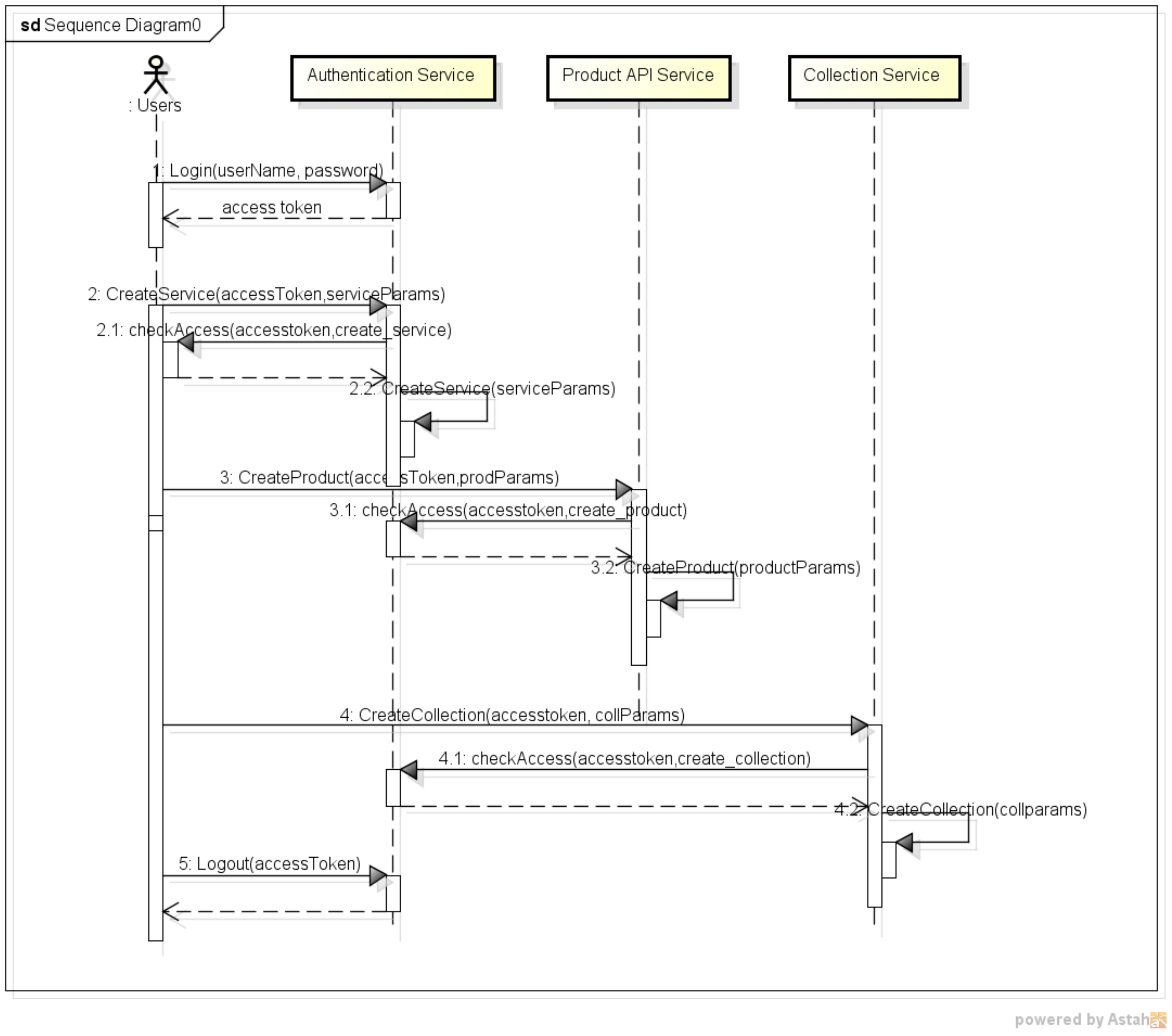
Authentication service keeps maps of service ids and service objects, entitlement ids and entitlement objects, user ids and user objects and access token uuid and access token object. These objects are created first by logging in as an admin user. All IDs except access tokens are in upper case so that corresponding searches are case insensitive. When credential is added to a user authentication service encrypts the password using MD5 hashing and stores the encrypted password. When user logs in password is encrypted and matched against stored encrypted password. If username or password is invalid application throws AuthenticationException. When login is successful a access token is returned to the user. Access token is associated with the user and user has set of roles or permissions. A role composes set of permissions and nested roles. Composite pattern allows Role to compose entitlements which derived by both role and permissions. An iterator class is associated with role in order to iterate through all the nested roles and permissions. This access token is used by the user to invoke restricted methods. Authentication service checks if permission is granted to user associated with given access token to invoke the restricted method. If user tries to invoke a restricted method for which user does not have the permission the authentication service throws AccessDeniedException. If access token is invalid (user logged out or token has expired) the authentication service throws InvalidAccesstokenException.

User, Role, Permission, AccessToken and Service implements visitable interface. They define accept() function which takes a visitor object. PrintInfoVisitor implements visitor implements and visits above mentioned objects and prints their information on standard output.

Visitor pattern enables implementation to print authentication service objects to standard output. user, service, role, permission and access token implements visitable interface by defining accept() function and accepts a visitor object. In the implementation of accept() each class visit() method of visitor and passes itself. PrintInfoVisitor is the concrete visitor class which implements visit() function for each vistable class and prints their information on standard output.

Sequence Diagram

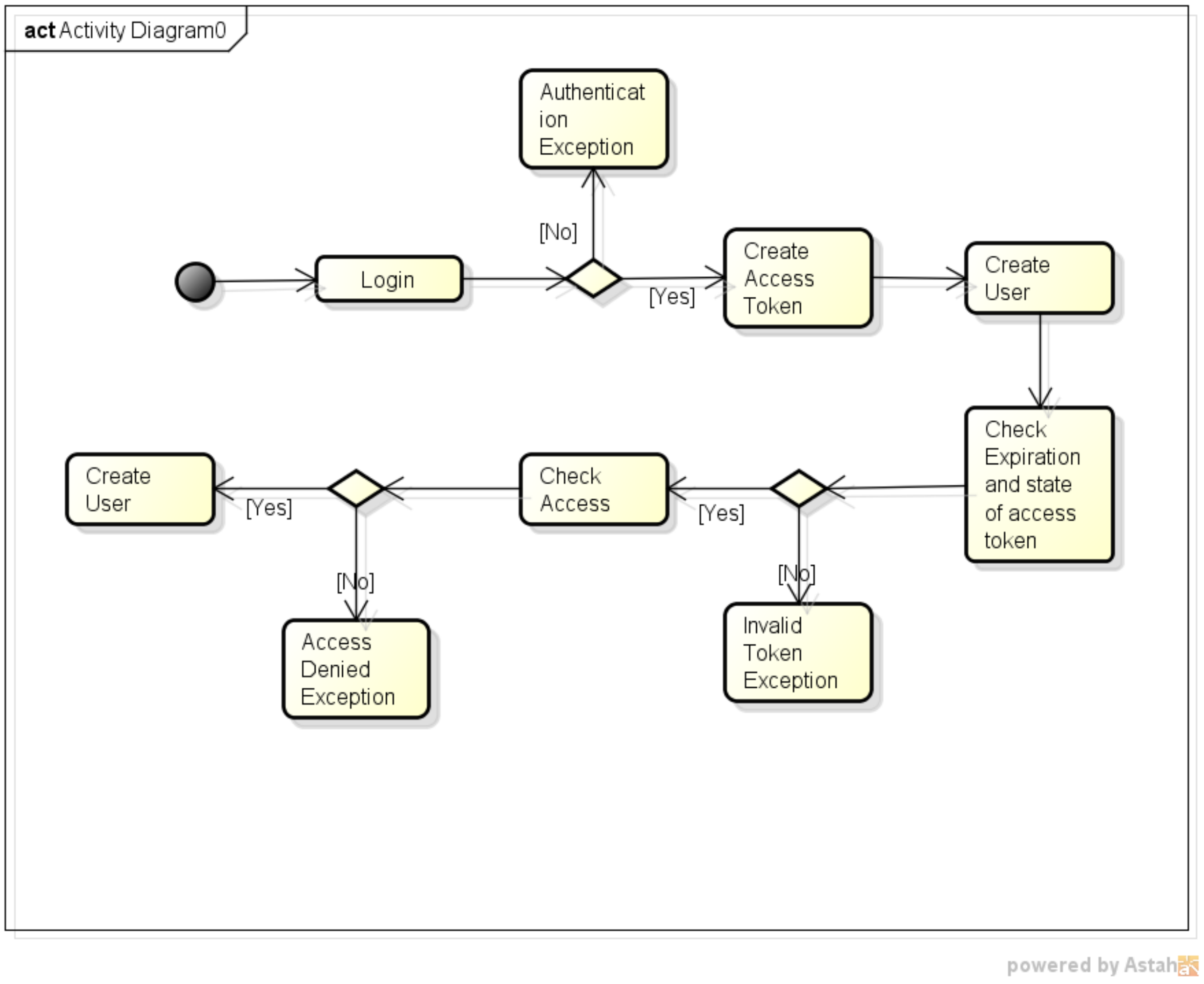
1. User logs in into the mobile application store by providing user name and password.
2. Authentication service checks user's credentials and responds back with an access token.
3. User uses the access token to invoke restricted methods by passing token as one of the parameters of restricted methods. In the following sequence diagram three restricted methods are shown (create service, create product and create collection). Other restricted methods in mobile application store service have same workflow.
4. Authentication service checks access on the token and authenticate user to invoke restricted methods.
5. User logs out and corresponding access token is invalidated ay authentication service.


sd Sequence Diagram0
: Users
Authentication Service
Product API Service
Collection Service
1: Login(userName, password)
access token
2: CreateService(accessToken,serviceParams)
2.1: checkAccess(accesstoken,create_service)
2.2: CreateService(serviceParams)
3: CreateProduct(accessToken,prodParams)
3.1: checkAccess(accesstoken,create_product)
3.2: CreateProduct(productParams)
4: CreateCollection(accesstoken, collParams)
4.1: checkAccess(accesstoken,create_collection)
4.2: CreateCollection(collparams)
5: Logout(accessToken)
powered by Astah

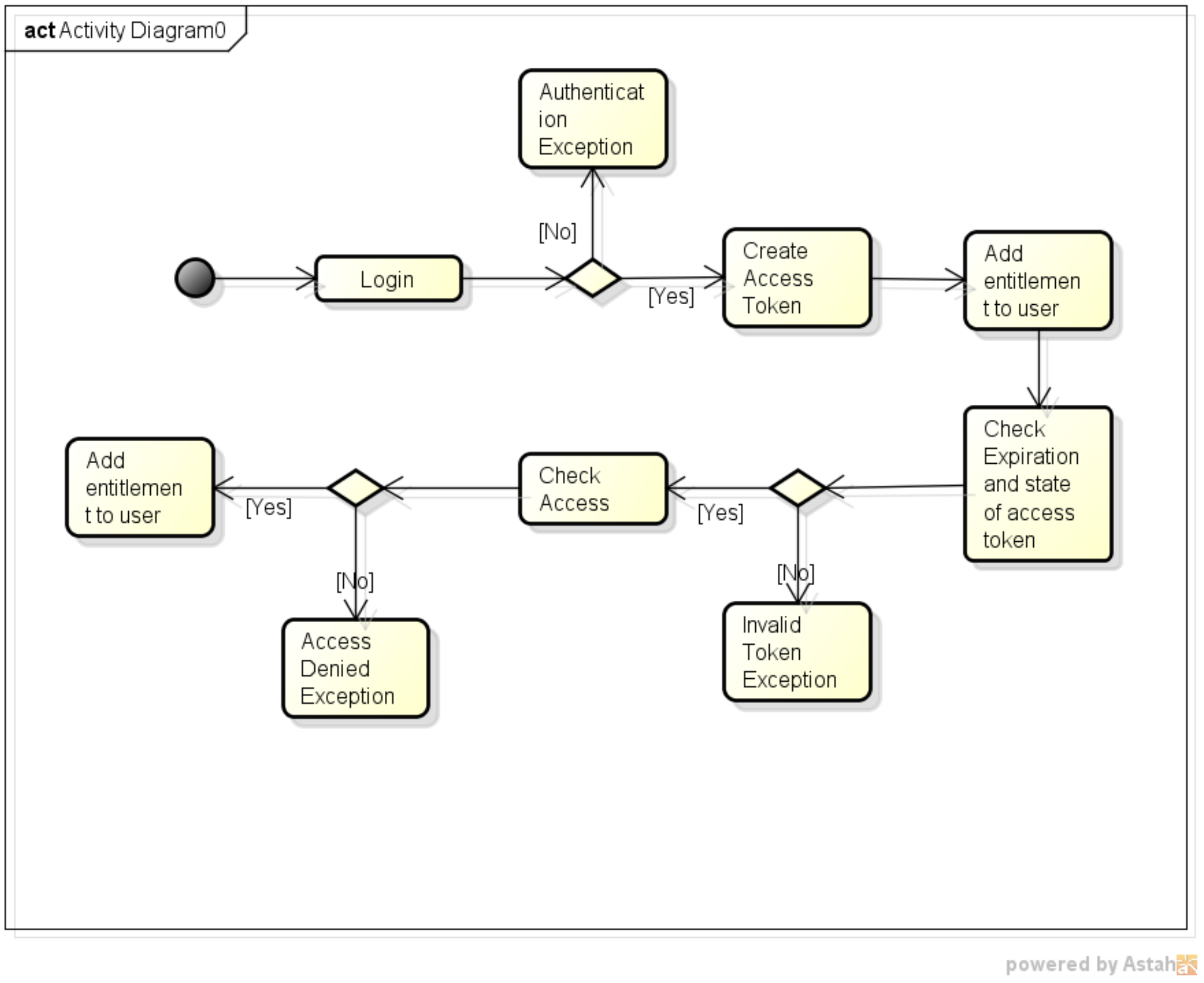
Activity Diagram

User logs in by passing username and password. If login fails authentication service throws AuthenticationException to client otherwise returns an access token. User sends the access token with request to invoke restricted methods. Authentication service checks if token is valid, if not throws InvalidTokenException. Authentication service checks if user associated with access token has permission to invoke the method. If not then throws AccessDeniedException to client otherwise invokes the method.

Create user Activity diagram

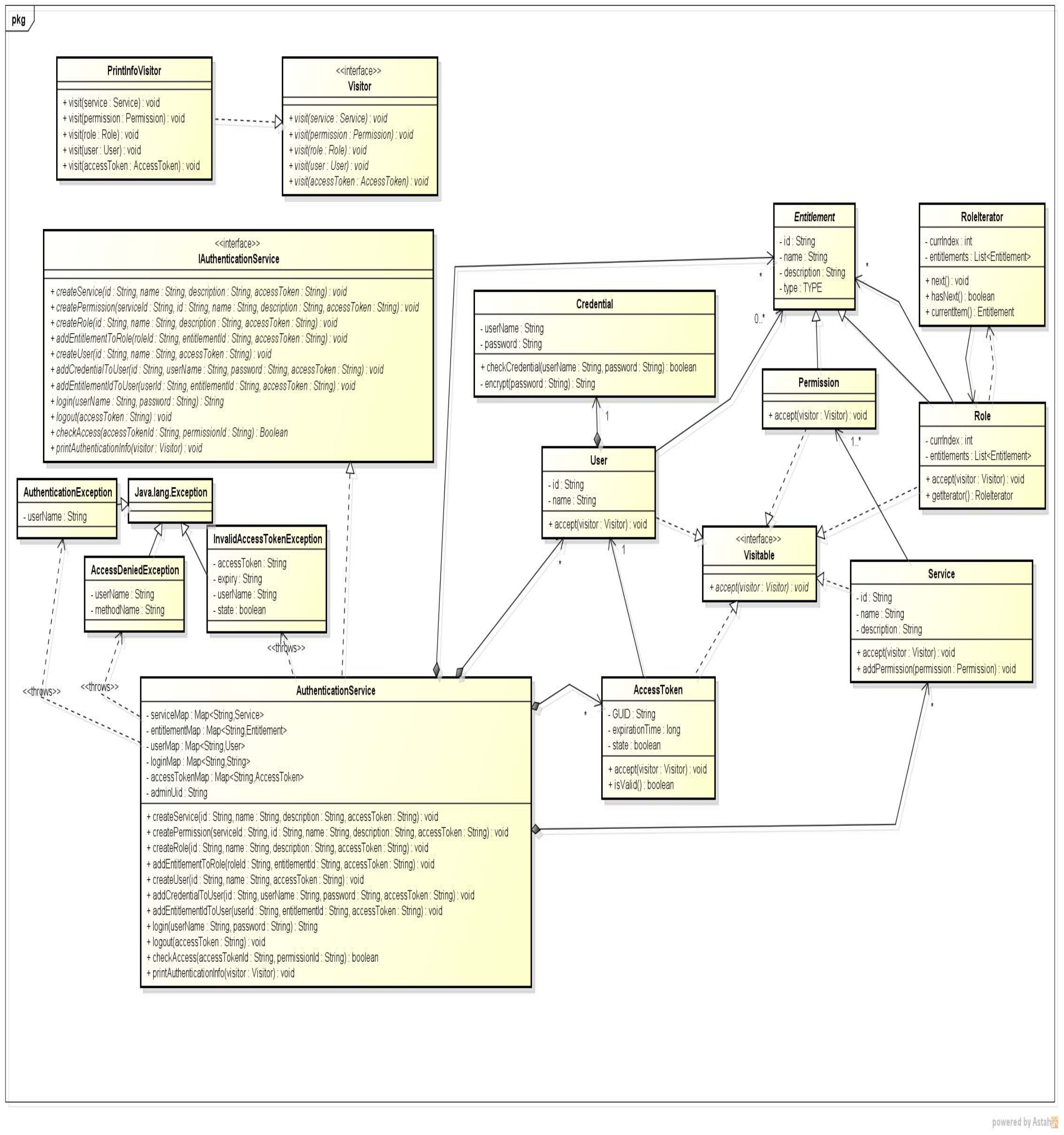


Add entitlement to user Activity diagram



Class Diagram

Please see AuthenticationService_classDiagram.png or AuthenticationService _classDiagram.asta for clear class diagram.


pkg
PrintInfoVisitor
+ visit(service : Service) : void
+ visit(permission : Permission) : void
+ visit(role : Role) : void
+ visit(user : User) : void
+ visit(accessToken : AccessToken) : void
<<interface>>
Visitor
+ visit(service : Service) : void
+ visit(permission : Permission) : void
+ visit(role : Role) : void
+ visit(user : User) : void
+ visit(accessToken : AccessToken) : void
<<interface>>
IAuthenticationService
+ createService(id : String, name : String, description : String, accessToken : String) : void
+ createPermission(serviceId : String, id : String, name : String, description : String, accessToken : String) : void
+ createRole(id : String, name : String, description : String, accessToken : String) : void
+ addEntitlementToRole(roleId : String, entitlementId : String, accessToken : String) : void
+ createUser(id : String, name : String, accessToken : String) : void
+ addCredentialToUser(id : String, userName : String, password : String, accessToken : String) : void
+ addEntitlementIdToUser(userId : String, entitlementId : String, accessToken : String) : void
+ login(userName : String, password : String) : String
+ logout(accessToken : String) : void
+ checkAccess(accessTokenId : String, permissionId : String) : Boolean
+ printAuthenticationInfo(visitor : Visitor) : void
Entitlement
- id : String
- name : String
- description : String
- type : TYPE
RoleIterator
- currIndex : int
- entitlements : List<Entitlement>
+ next() : void
+ hasNext() : boolean
+ currentItem() : Entitlement
Credential
- userName : String
- password : String
+ checkCredential(userName : String, password : String) : boolean
- encrypt(password : String) : String
Permission
+ accept(visitor : Visitor) : void
Role
- currIndex : int
- entitlements : List<Entitlement>
+ accept(visitor : Visitor) : void
+ getIterator() : RoleIterator
User
- id : String
- name : String
+ accept(visitor : Visitor) : void
<<interface>>
Visitable
+ accept(visitor : Visitor) : void
Service
- id : String
- name : String
- description : String
+ accept(visitor : Visitor) : void
+ addPermission(permission : Permission) : void
AuthenticationException
- userName : String
Java.lang.Exception
AccessDeniedException
- userName : String
- methodName : String
InvalidAccessTokenException
- accessToken : String
- expiry : String
- userName : String
- state : boolean
<<throws>>
<<throws>>
<<throws>>
AuthenticationService
- serviceMap : Map<String,Service>
- entitlementMap : Map<String,Entitlement>
- userMap : Map<String,User>
- loginMap : Map<String,String>
- accessTokenMap : Map<String,AccessToken>
- adminUid : String
+ createService(id : String, name : String, description : String, accessToken : String) : void
+ createPermission(serviceId : String, id : String, name : String, description : String, accessToken : String) : void
+ createRole(id : String, name : String, description : String, accessToken : String) : void
+ addEntitlementToRole(roleId : String, entitlementId : String, accessToken : String) : void
+ createUser(id : String, name : String, accessToken : String) : void
+ addCredentialToUser(id : String, userName : String, password : String, accessToken : String) : void
+ addEntitlementIdToUser(userId : String, entitlementId : String, accessToken : String) : void
+ login(userName : String, password : String) : String
+ logout(accessToken : String) : void
+ checkAccess(accessTokenId : String, permissionId : String) : boolean
+ printAuthenticationInfo(visitor : Visitor) : void
AccessToken
- GUID : String
- expirationTime : long
- state : boolean
+ accept(visitor : Visitor) : void
+ isValid() : boolean
1
1
1..*
0..*
*
*
*
*
*
powered by Astah

Class Dictionary

IAuthenticationService<<interface>>

Methods

Method Name	Signature	Description
createService	void createService(String id,String name,String description,String accessToken) throws InvalidAccessTokenException, AccessDeniedException;	Creates Service with given id, name and description. Throws exception if access token is invalid or user does not have permission to invoke the method.
createPermission	void createPermission (String serviceId,String id,String name,String description,String accessToken) throws InvalidAccessTokenException, AccessDeniedException;	Creates permission with given id, name and description. Attaches permission with service if serviceId as identifier. Throws exception if access token is invalid or user does not have permission to invoke the method.
createRole	void createRole(String id,String name,String description,String accessToken) throws InvalidAccessTokenException, AccessDeniedException;	Creates role with given id, name and description. Throws exception if access token is invalid or user does not have permission to invoke the method.
addEntitlementToRole	void addEntitlementToRole(String roleId,String entitlementId,String accessToken) throws InvalidAccessTokenException, AccessDeniedException;	Adds entilement (role or permission) to a role. Throws exception if access token is invalid or user does not have permission to invoke the method.
createUser	void createUser(String id,String name,String accessToken) throws InvalidAccessTokenException, AccessDeniedException;	Creates user with given id and name. Throws exception if access token is invalid or user does not have permission to invoke the method.
addCredentialToUser	void addCredentialToUser(String id,String userName,String password,String accessToken) throws InvalidAccessTokenException, AccessDeniedException;	Associates and user name and password with a user. Throws exception if access token is invalid or user does not have permission to invoke the method.
addEntitlementIdToUser	void addEntitlementIdToUser(String userId,String entitlementId,String accessToken) throws InvalidAccessTokenException, AccessDeniedException;	Adds entilement (role or permission) to a user. Throws exception if access token is invalid or user does not have permission to invoke the method.
login	String login(String userName,String password) throws AuthenticationException;	Logs in into application store using user name and password. Returns access token on success. Throws exception if authentication fails.
logout	void logout(String accessToken) throws InvalidAccessTokenException;	Invalidates the access token. Throws exception if access token is invalid.
checkAccess	boolean checkAccess(String	Checks if user associated with the

	accessTokenId,String permissionId)throws InvalidAccessTokenException, AccessDeniedException;	access token has permission to invoke method with a given permission id. Throws exception if access token is invalid or user does not have permission to invoke the method with the given permission Id.
printAuthenticationInfo	void printAuthenticationInfo(Visitor visitor);	Calls accept() function for user,role, permission, service, access tokens of visitable interface to print object's information on standard output.

AuthenticationService

This class implements IAuthenticationService interface. It keeps map of services, entitlements, users and access tokens with key as their identifiers. The keys of these maps other than access tokens are stored in upper case to make search case insensitive.

Properties

Property Name	Type	Description
adminUid	String	Access token of admin user.

Associations

Association Name	Type	Description
serviceMap	Map<String,Service>	map of service id and service object
entitlementMap	Map<String, Entitlement>	map of entitlement id and entitlement object
userMap	Map<String,User>	map of user id and user object
accessTokenMap	Map<String,AccessToken>	map of access token and access token object

Visitable<<interface>>

Declares method to accept visitor. The interface is implemented by user, service, role, permission and access token.

Methods

Method Name	Signature	Description
accept	void accept(Visitor visitor)	Accepts visitor and calls visit() of visitor.

Entitlement<<abstract>>

This acts as base class for roles and permissions. Entitlement has and id , name description and type. A type is either role or permission.

Properties

Property Name	Type	Description
id	String	Entitlement id
name	String	Entitlement name
description	String	Entitlement description
type	TYPE	ROLE or PERMISSION

Role

Role is subclass of Entitlement and implements visitable interface. Role class exhibits composite pattern by containing permissions and nested roles which are entitlements. It exposes getiterator() function which returns iterator in order to traverse though child elements of role.

Associations

Association Name	Type	Description
entitlements	List<Entitlement>	List of all roles and permission under the role.

Methods

Method Name	Signature	Description
addEntitlement	void addEntitlement(Entitlement entitlement)	Associates entitlement with the role.
getIterator	RoleIterator getIterator()	returns iterator in order to traverse though child elements of the role.

RoleIterator

Iterator class for iterating roles.

Properties

Property Name	Type	Description
currIndex	int	Current index of the flattened list in the iterator.
visited	Set<Entitlement>	Keep track of visited nodes in depth first search of iterating in role tree.

Associations

Association Name	Type	Description
parentRole	Entitlement	Parent Role object on which iterator is operating.
entitlements	List< Entitlement >	List of all entitlements in role tree rooted at parentRole.

Methods

Method Name	Signature	Description
hasNext	Boolean hasNext()	Returns true if iterator has more items to iterate otherwise false.
currentItem	Entitlement currentItem()	Returns current item in the iterator.
next	Void next()	Goes to next item in the iterator.

Permission

Permission is subclass of Entitlement and implements visitable interface. Each permission object represents a service which can be performed if user has the permission.

User

User implements visitable interface. It represents user of mobile application store. A user has an id, name and credential (user name and password). User has set of permissions.

Properties

property Name	Type	Description
id	String	User id
name	String	Name of user
credential	Credential	Login and password of user

Associations

Association Name	Type	Description
entitlements	List<Entitlement>	List of all roles and permission under the role.

Credential

This class encapsulates login and password of a user. Password is stored in MD5 encrypted form. It authenticates user's credential by encrypting the password and matching it with stored encrypted password.

Properties

Property Name	Type	Description
userName	String	Login name
password	String	Encrypted password

Methods

Method Name	Signature	Description
checkCredential	boolean checkCredential(String userName,String password)	Encrypts the password and checks it with stored encrypted password. Returns true if it matches otherwise returns false.
encrypt	String encrypt(String passwd)	Returns MD5 encrypted password.

AccessToken

This class represents access token which are assigned to logged in users. Access token has a GUID, expiration time and a Boolean state. Expiration time is two hours from the time user logged in and beyond that access token becomes invalid. Access also becomes invalid when user logs out. Each access token is associated with a user.

Properties

Property Name	Type	Description
GUID	String	GUID of access token
user	String	Associated user.
expirationTime	long	Represents expiration time in milliseconds
state	Boolean	False if access token is invalid

Methods

Method Name	Signature	Description
isValid	boolean isValid()	Returns false if access token is expired or state is invalid, otherwise returns true.

Service

This class represents a service from mobile store application. A service has id, name and description and set of permissions defined in the service.

Properties

Property Name	Type	Description
id	String	service id
name	String	service name
description	String	service description

Associations

Association Name	Type	Description
permissions	List<Permission>	Set of permissions defined in the service.

Visitor<<interface>>

Declares methods to visit user, service, role, permission and access token. All these classes implements visitable interface to accept visitor object.

Methods

Method Name	Signature	Description
visit	void visit(Service service)	Visits service object.
visit	void visit(Permission permission)	Visits permission object.
visit	void visit(Role role)	Visits role object.
visit	void visit(User user)	Visits user object.
visit	void visit(AccessToken accessToken)	Visits accessToken object.

PrintInfoVisitor

Concrete implementation of visitor interface> the purpose of the visitor class is to print the objects information on standard output.

AccessDeniedException

Represents exception class when user is not permitted to invoke any restricted method.

Properties

Property Name	Type	Description
userName	String	User name who is trying to invoke the method
methodname	String	Name of the method user is invoking

AuthenticationException

Represents exception class when user's credentials are wrong during login.

Properties

Property Name	Type	Description
userName	String	User name who is trying to log in.

InvalidAccessTokenException

Represents exception class when user tries to invoke a method but token is invalid. Token is either expired or user has logged out or user has passed an invalid token.

Properties

Property Name	Type	Description
tokenId	String	GUID
userName	String	User name who is trying to invoke the method
expiry	long	Represents expiration time in milliseconds
state	Boolean	State of accessToken

Implementation Details

Authentication service keeps a map of service, user, entitlement and access token. Each access token is associated with a user. User has set of roles and permissions. A role can have one or more permissions and nested sub roles. **Iterator** patter has been used to iterate through child elements of role.

Each user has a credential (login name and password). Credential objects is composed by user and it keep user name and encrypted password. Authentication service has map of login name and user id. In Login() authentication service fetches the user and checks if password matches with stored password in credential object. If login fails service throws AuthenticationServiceException otherwise returns access token.

In checkAccess() authentication service first checks if token is valid and then fetches user from token and iterates through all the roles and permission of the user. If any of the permission matches with permission id sent as argument then it return true otherwise throws AccessDeniedException.

Logout invalidates state of access token.

CreateService, CreateUser, CreatePermission, CreateRole adds objects to corresponding maps.

addEntitlementToRole fetches role from map and appends entitlement to it.

addEntitlementTouser fetches user from map and appends entitlement to it.

Visitor pattern enables implementation to print authentication service objects to standard output. user, service, role, permission and access token implements visitable interface by defining accept() function and accepts a visitor object. In the implementation of accept() each class visit() method of visitor and passes itself. PrintInfoVisitor is the concrete visitor class which implements visit() function for each vistable class and prints their information on standard output.

Testing

CheckAccess() implementation is augmented to allow admin user to perform any services in the store application. In test driver class client first logs in as admin user and creates services, roles, permissions, users and add entitlements to roles and users . The data is given in a csv file which test driver parses and invokes corresponding methods defined in authentication service to perform above mentioned tasks.

Once data is entered into the authentication service test drive logs in different types of user to test different scenario.

Test Cases :

Valid Cases:

1. Enter authentication data from authentication.csv
2. Login product admin user
3. Login collection admin user
4. Login product dev user
5. Login authentication admin user
6. Import countries passing product admin token
7. Import devices passing product admin token
8. Import products passing product dev token
9. Import collections passing collection admin token

Exceptional Cases:

1. Login using wrong password
2. Create user passing product admin token
3. Create collection passing authentication admin token
4. Create user passing collection admin token
5. Logout and use same token to run a restricted service

All test cases are run in one go. In exceptional cases exception is caught so that all cases can run.

Risks

Add user and add credentials are two separate interfaces. So it may be possible that there may be no credentials for few users. A user without credential defies the business logic of the authentication service. In my opinion combining the two interfaces into one is a good idea.

Admin user leaves security hole in the service. A more careful and secured design is recommended to implement administrator tasks.