import java.util.\*;

public class AuthorizeAPI {

int tokenIdStart = 10000;

HashMap<String, String> userMap = new HashMap<>();

HashMap<String, HashMap<String, String>> serviceMap = new HashMap<>();

HashMap<String, ArrayList<String>> authMap = new HashMap<>();

/\*\*\*

\* 初始化系统

\* 系统初始化后清空所有数据， 注册的用户数据、添加的服务访问权限 以及系统产生AccessToken数据将清空。

\* @return true or false

\* true 初始化成功

\* false 方法未实现

\*/

public boolean init() {

userMap.clear();

serviceMap.clear();

authMap.clear();

if (userMap.isEmpty() && serviceMap.isEmpty() && authMap.isEmpty()) return true;

return false;

}

/\*\*\*

\* 系统初始化后，用户可以通过用户注册接口注册到本系统

\* @param userId 用户标识, 用户的全局唯一的标识 ,不能为空，长度不超过16个字符

\* @param pwd 密码，鉴权使用的密码，不能为空，长度不小于6个字符, 不超过16个字符

\* @return true or false

\* true 用户注册成功

\* false 用户注册失败

\*/

public boolean registerUser(String userId, String pwd) {

if (userId == null || userId.isEmpty() || userId.length() > 16 || pwd == null || pwd.length() < 6 || pwd.length() > 16) {

return false;

}

if (userMap.containsKey(userId)) {

return false;

}

userMap.put(userId, pwd);

return true;

}

/\*\*\*

\* 删除用户

\* 从系统中删除用户

\* @param userId 用户标识, 用户的全局唯一的标识，不能为空

\* @return true or false

\* true 用户删除成功

\* false 用户删除失败

\*/

public boolean removeUser(String userId) {

if (userId == null || userId.isEmpty()) {

return false;

}

if (userMap.containsKey(userId)) {

userMap.remove(userId);

return true;

}

return false;

}

/\*\*\*

\* 添加服务访问权限

\* 添加新的服务访问权限到系统中

\* @param authId 服务访问权限编号；取值范围： 大于0

\* @param userId 用户标识, 用户的全局唯一的标识，不能为空

\* @param serviceName 服务名称，格式：不能为空，长度不超过16个字符

\* @return true or fase

\* true 添加成功

\* false 添加失败

\*/

public boolean addAuthPolicy(String authId, String userId, String serviceName) {

if (authId == null || authId.isEmpty()) return false;

if (userMap.containsKey(userId)) {

if (serviceMap.containsKey(userId)) {

HashMap<String, String> map = serviceMap.get(userId);

if (map.containsKey(authId)) return false;

else {

map.put(authId, serviceName);

return true;

}

} else {

HashMap<String, String> map = new HashMap<>();

map.put(authId, serviceName);

serviceMap.put(userId, map);

return true;

}

}

return false;

}

/\*\*\*

\* 删除服务访问权限

\* 将指定的的服务访问权限从系统中删除

\* @param authId 服务访问权限编号；取值范围： 大于0

\* @return true or false

\* true 服务访问权限删除成功

\* false 服务访问权限删除成失败

\*/

public boolean removeAuthPolicy(String authId) {

int count = 0;

Set<Map.Entry<String, HashMap<String, String>>> set = serviceMap.entrySet();

for (Map.Entry<String, HashMap<String, String>> entry : set) {

HashMap<String, String> map = entry.getValue();

if (map.containsKey(authId)) {

map.remove(authId);

count++;

}

}

return count > 0;

}

/\*\*\*

\* 请求AccessToken

\* 用户根据用户标识和密码经过鉴权获得访问AccessToken

\* @param userId 用户标识, 用户的全局唯一的标识, 不能为空

\* @param pwd 密码，用户鉴权使用的密码，不能为空

\* @return 返回AccessToken的标识

\*/

public String requestAccessToken(String userId, String pwd) {

if (userId == null || userId.isEmpty() || pwd == null || pwd.isEmpty()) return null;

if (userMap.containsKey(userId) && userMap.get(userId).equals(pwd)) {

String accessToken = userId + tokenIdStart;

tokenIdStart++;

ArrayList<String> list = new ArrayList<>();

HashMap<String, String> map = serviceMap.get(userId);

Set<Map.Entry<String, String>> set = map.entrySet();

for (Map.Entry<String, String> entry : set) {

list.add(entry.getValue());

}

authMap.put(accessToken, list);

return accessToken;

}

return null;

}

/\*\*\*

\* 访问服务

\* 使用已申请的令牌accessToken访问服务，系统根据预置的服务访问权限，判断是否允许访问访问

\* @param accessTokenId 令牌，不能为空

\* @param serviceName 服务名，不能为空

\* @return true or false

\* true 可以访问

\* false 无法访问

\*/

public boolean accessService(String accessTokenId, String serviceName) {

if (accessTokenId == null || accessTokenId.isEmpty() || serviceName == null || serviceName.isEmpty())

return false;

if (authMap.containsKey(accessTokenId)) {

ArrayList<String> list = authMap.get(accessTokenId);

return list.contains(serviceName);

}

return false;

}

/\*\*\*

\* 回收AccessToken

\* 当用户不需使用AccessToken时，用户可以主动放弃AccessToken，由系统释放删除。

\* @param accessTokenId 访问令牌标识, 不能为空

\* @return true or false

\* true 删除成功

\* false 删除失败

\*/

public boolean revokeAccessToken(String accessTokenId) {

if (accessTokenId == null || accessTokenId.isEmpty()) return false;

if (authMap.containsKey(accessTokenId)) {

authMap.remove(accessTokenId);

return true;

}

return false;

}

}