PSEUDOCODE FOR AUTHENTICATION MANAGER

0. glue code start authy manager

1. **Date Time Check**

   i. Get user's system date + time

   ii. Connect to http://172.20.82.6:8080/b4server

   iii. Get B4 server's date + time

   iv. Compare both time (user's system time to be ahead of server time)

   v. If, false → exit the application / alert message (via : pop up window or webUI or cmd line message only) ???

   vi. Else, move to step 2

2. **Checking Availability of User's Keystore in local storage**

   - check for .jks file

   CASE I : Keystore Does Not Exist At Local Storage

   a. **Get user's email id:** (thru glue-code → webUI)

      i. Glue code to open webUI to get user's email

      ii. Get user input (Email) through input buffer

   b. **Send user's email :** via http post request to B4 server (to check whether keystore is available with server or not)

      i. open http connection with request type " check key store "

      ii. set request method : Post

      iii. create data output stream → write data → close http connection

   c. **Get server's response**: create buffer reader→ input stream reader→ get response code → response code : 14 or 15

- **14** : user email id exists with b4 server → check for keystore availability & validity
- **15** : it is a new user/ new email id → Certificate N/A with server → Generate new certificate → new identification check process

3. Action as per Server **Response Code 15: To create new self-signed certificate**

i. Inform glue code to open webUI for user's form data , then, Get user's form input through buffer reader.

ii. Generate Pub & Pvt Keys (RSA algo, Sha256, Bounty Castle as provider).

iii. Generate x509 certificate with user's data & keyPairs.

iv. Sign the certificate with private key.

v. Connect to Identity Server and send certificate for authentication

vi. Identity Server sends OTP to user's email (through server code)

vii. Inform glue code to open OTP webUI window

viii. User submits OTP via webUI

ix. Get user's submitted OTP, through buffer

x. Sends OTP to Identity server using Post Request (doubt: old url is containing: device id + node id ???) … calling RM api ??? or get thru glue code ????

a. Post method: Open http connection→ Data Output Stream → write Bytes →flush → close

b. Get response code: Buffered Reader→ Input Stream Reader → get Input Stream → read response (these will be the client's and server's certificates)

xi. Get client certificate and server certificate

xii. Inform Glue code to open webUI to get :

a. Alias name

b. Keystore password

xiii. get client's private key → encode it →write on key.txt → close file

xiv. create keystore instance → load keystore → set keystore password → load: alias, private key, password, client cert & server cert into keystore

xv. store keystore in SignedClientKeyStore.jks file

xvi. connect to b4 server → send keystore via http post request in encoded form.

xvii. Create hash of keystore file

xviii. Create authenticate.txt and add hashed keystore to it.

xix. Save keystore

xx. Start as per **case II** (keystore is now available)

e. Action as per Server **Response Code 14 :**

i. b4 server sends otp to submitted email

ii. inform GC to open webUI to submit otp and get otp through buffer

iii. send otp to b4 server

iv. Server checks otp → carry out certificates validity check as per

case II (2) → flag is set true / false as per validity check.

Case II : User's Keystore available in local storage

1. **Expiry check** : 15 days → alerts . if expired → do steps to generate new self signed certificate

2. To check if there is valid certificate present in the keystore: **flag : true or false**

    a. Inform glue code to open webUI to get keystore password or option window from user.
    b. Read user selection value from webUI through buffer:

        i. **0**: **entered password OK :** now get alias name from webUI thru buffers → load keystore using password & alias → get the cert associated with the given alias → get server cert & client cert → cert expiry check → if valid , **set flag True**

        ii. **1**: **Regeneration of keystore**: get user's email thru webUI→set up http connection with b4 server→request otp from server→server send otp on email → open webUI and get user's input otp → again send otp to server → (…in progress…)
        iii. **2**: **certificate revocation** (…..in progress)
        iv. **3**: **generate new certificates** → do action as per 2(d) : creation on self-signed certificates

Only when **flag = true → Start other services/modules**