List Algorithms Used in Application:

|  |  |  |
| --- | --- | --- |
| **Algorithm 1:** UpdateCredentails | | |
|  | **Input:** | set of devices *D* |
| **Output:** | set of devices *P* with update status 1 for SUCCESS or 0 for FAILED |
|  | CONSTANT*cpdays* := current date + 7 (expire within 7 days)  CONSTANT*cexpdt* := current date + 90 (set next expiry date)  CONSTANT*ctry* := 10 (try 10 times to update failed)  copy *D* to *P*  **for each** device*x* **in** *P*  expiry date *EXPD* := *P*[*x*]  number of try *NTRY* := *P*[*x*]  update status *ustat* := *P*[*x*]  **if** *EXPD* **<** *cpdays* ***or*** *NTRY <= ctry* **then**  Device Status API Name *DSAPI* := *P*[*x*] // API: /device/active  Device Credentials Update API Name *DUAPI* := *P*[*x*]  // API: /device/setpubcred  Device Stat *DS* := device API call *DSAPI*  **If** *DS* = ACTIVE **then**  New User Name *UM*:= getUserName()  New Password *NP*:= getPassword()  Credentials Update Stat *CUS*:= device API call DUAPI with payload (*UM*, *NP*)  **if** *CUS* is success **then**  *P* := *P* U {*x*,*ustat* =1} U {*x*, *EXPD* **=***cexpdt*}  // update with status and next password expire date as 90 days later  **else**  *P* := *P* U {*x*,*ustat* =0} U {*x*, *NTRY* **=** *NTRY* +1}  // credentials update failed status with try number  **end if**  **end if**  **end for**  **return** *P* | |

|  |  |  |
| --- | --- | --- |
| **Algorithm 2:** SetDeviceCredentials | | |
|  | **Input:** | New User Name for public as *NU* and New Password public as *NP,* New User Name for private as *LNU* and New Password private as *LNP* |
| **Output:** | Status of the device credentials update |
|  | Detected IP of the request as *RIP :=* request.ip  Device switch stat as SWSTAT := switching stat from configuration file  Device private credentials update by *PVTUPD* := read from configuration file  *RIP* := requestCheckIP(*RIP*)  **if** *NU* is null or *NP* is null **then**  **return** *Null*  **else**  **if**  requestCheckIP(*RIP*) = ‘PUB’ **and** *SWSTAT* = ‘CMan’ **then**  Public Access Credentials file *CF* := **Open** the File in write mode  **clear** content of *CF*  **write** first line into *CF* with the value of *NU*  go to **next** line  **write** second line into *CF* with the value of *NP*  **close** the file *CF*  **if** *PVTUPD =* ‘YES’ **then**  Private Access Credentials file *CF* := **Open** the File in write mode  **clear** content of *CF*  **write** first line into *CF* with the value of *LNU*  go to **next** line  **write** second line into *CF* with the value of *LNP*  **close** the file *CF*  **end if**  **return** SUCCESS  **else**  **return** *Null*  **end if**  **end if** | |

|  |  |  |
| --- | --- | --- |
| **Procedure 1:** getIPParts () | | |
|  | **Input:** | Part number of IP as *PN,*IP as *rIP* |
| **Output:** | Mentioned part of the IP |
|  | *ip\_part1* := substring of *rIP* till first ‘.’  *ip\_part2* := substring of *rIP* from first ‘.’ to second ‘.’  *ip\_part3* := substring of *rIP* from second ‘.’ to third ‘.’  *ip\_part4* := substring of *rIP* from third ‘.’ to end of string  **if** *pn* = 1 **then**  **return** *ip\_part1*  **else if** *pn* = 2 **then**  **return** *ip\_part2*  **else if** *pn* = 3 **then**  **return** *ip\_part3*  **else if** *pn* = 4 **then**  **return** *ip\_part4*  **else**  **return** *null*  **end if** | |

|  |  |  |
| --- | --- | --- |
| **Procedure 2:** requestCheckIP() | | |
|  | **Input:** | Request IP *rIP* |
| **Output:** | Return IP status PVT=PRIVATE or PUB=PUBLIC |
|  | Part 1 of the IP as *IPP1* := getIPParts (1, *rIP* )  Part 2 of the IP as *IPP2* := getIPParts (2, *rIP* )  Part 3 of the IP as *IPP3* := getIPParts (3, *rIP* )  Part 4 of the IP as *IPP4* := getIPParts (4, *rIP* )  // Class A Private Range: 10.0.0.0 to 10.255.255.255  **if** *IPP1* **=** 10 **then**  **return** 'PVT'  // Class B Private APIPA Range: 169.254.0.0 to 169.254.255.255  **else if** *IPP1* **=** 169 and *IPP2* = 254 **then**  **return** 'PVT'  // Class B Private Range: 172.16.0.0 to 172.31.255.255  **else if** *IPP1* **=** 172 and (*IPP2* between 16 and 31) **then**  **return** 'PVT'  // Class C Private Range: 192.168.0.0 to 192.168.255.255  **else if** *IPP1* **=** 192 and *IPP2* = 168 **then**  **return** 'PVT'  // Special IP Addresses : IP Range: 127.0.0.1 to 127.255.255.255  **else if** *IPP1* **=** 127 and*IPP4* **>=** 1 **then**  **return** 'PVT'  **else**  **return** 'PUB'  **end if** | |