# 

**Single Sign On**

**For Cuberoot DMP Data Integration across Indiatoday products**

(User Authentication and Authorization for Indiatoday)

Icon

Description automatically generated

Table of Content

# 1)What is SSO? 3

# 2)Indiatoday and SSO 3

# 3)USERID BASED SSO APIs 4

# 4)EMAIL BASED APIs 11

# 5)Registration Process Workflow 22

# 6)Retrieve Forgot Password Workflow 23

# 7)Change Password Workflow for User Based Access 23

# 8)Change Password Workflow for Email Based Access 24

# 9)Error Code Table 26

# 10)Field Validation Error Codes 29

## What is SSO?

Single sign-on is a mechanism of accessing multiple independent software systems with common user base. SSO provides a single action of user authorization and authentication to access systems without the need to enter login information several times.

In the single sign-on approach the system is required to collect from the user the identification and user credential information necessary to support the authentication of the user to each of the secondary domains that the user may potentially require to interact with. The information supplied by the user is then used by Single Sign-On Services within the primary domain to support the authentication of the end user to each of the secondary domains with which the user actually requests to interact. SSO uses centralized authentication servers that all other applications and systems utilize for authentication purposes, and combines this with techniques to ensure that users do not actively have to enter their credentials more than once.

## Requirements of SSO

### Function Requirements

* **To built a single SSO for all the channels:**

For different set of channels we have different set of APIs written in Java. The aim is to build single set of Java APIs as a final SSO APIs for user management and data analysis.

* **Support to establish a default user profile:**

A user will be allowed to create user accounts through different channels but the user profile at the backend will be single through which a user can move to different channels without signing up.

* **To have a single user base for all the channels:**

The user bases for all the channels are not single. The aim is to merge all the user data in various databases into one Elasticsearch index.

* **To add security to the SSO APIs:**

A security layer is added to the SSO APIs to avoid insecure and unauthorized access to the huge user base.

* **To analyze the user action:**

The user action needs to be captured for analysis of user activity through various channels that would help in planning business strategies.

* **To avoid any password issues:**

**SSO and Data Management Platform**

For Cross Domain Identity and User Persona Management, SSO is very essential.

This enables merger of user personas stored across different domains for a user and global application usage.

Data Points of user available in first party storage of websites via Cuberoot DMP.

**User Data Points are in different domains**

a)Location

b)Platform Based

c)Time Unit

d)Behavioral Segments

e)Topic Segments

**Publisher Analytic Segments Engagement Time Based Segments**

a)Segment Engagement Times

b)Topic Engagement Times

c)Section Engagement Times

**Session Based Segments**

a)Session count

b)Time Since Last Session

c)Session Engagement Time

d)Loyalty Nature

e)Visitor Count.

**Machine Learning based Segments**

This covers different domains Personas

a)Emotion Persona

b)Affinities based on Creative Knowledge graph Entities

c)Refined Demographic Persona

d)Affinities based on User Knowledge Graph Entities

e)Click Segments Derived After Analysis of Clickstream. This includes boolean click segment, This also includes Click probability of multiple CTR models for that User Persona. (This complements AdManager CTR model and helps in better training of Ad Manager CTR model).

**Creative Supply for Hyper personalisation**.

Specific creatives for campaigns meant for hyper – personalisation need to be provided.

Campaign Image/Media can be optimised on requirement basis.

Color, Entity recommendations for campaigns are present in color persona and creative entity Report present in dashboards.

Creative Entities showing high affinity scores can be used for optimised campaign media.

Rest of the campaigns running on the website, are crawled on daily basis and are optimised as well.

Creatives Entity Knowledge Graph and User Persona Knowledge Graph is built based on corpus and real time data stream and affinity scores are supplied to native and Ad Manager CTR model.

These data points discussed above are supplied and are visible in reports in the dashboards for data analysis and possible predictions.

**Dashboards Link**

[https://reportingsystem.cuberoot.co](https://reportingsystem.cuberoot.co/)

username: kibana

password: rrSWGVG4WKu3Y783Rub

**Supply of Key Value pairs/Ideal Segments on Daily Basis**

1)After analysis of clickstream, for period of 1 day and thereafter, cumulative campaign data build up, Ideal segments in domains above are derived on daily basis using Targeting IQ Algorithm.

Documentation of Targeting IQ has been provided already,

These ideal KVs, Targeting segments will be uploaded in dashboard and sent by mail on daily basis for campaign optimisation in AdManager.

**Specific domain persona segments can be turned off on requirements/click stream analysis.**

**Availability of third party data sets**

Clickstreams, Knowledge Graphs from various popular website and domain, demography, region specific clickstream is also available for use.

1)Wittyfeed

2)Quint

3) Silicon India

4)IDP

5)Puthiyathalaimurai

6)Marathistar

7) Clovia

8) Womansera

9) BlogtoBollywood

10) CricBuzz

11)Social Post

12)North East Today

13)Rajasthan Patrika

14)Adda 52

15)Digit

**Here is some more short summary of third party data**

a)Third party data sources -  Some of the third party data sources - Bloomberg, Wittyfeed, Digit (9.9 media), Womansera - India's No.1 women's magazine!, Adda52 - India’s largest online gaming platform!, Silicon India, Aam Aadmi Party - Indian political party, [kyarate.com](http://kyarate.com) - ecommerce, [yaantra.com](http://yaantra.com) - affordable mobile phone repair services, [northeasttoday.com](http://northeasttoday.com) - Northeast India’s first fully digital news portal!, SRM Group - [puthiyathalaimurai.com](http://puthiyathalaimurai.com) - #1 online Tamil News portal!, Rajasthan Patrika.  
ISP Network - hathway, alliance - broadband and more.  
Hyper Local Data based on Wi-fi Hotspots like McDonalds, Costa, KFC, Airports like Mumbai T1, T2 Airport , Quadgen hotspots and more.

**Personas  Available**

 1)**Creative Personas**

Creative Personalisation -

a)Gamification and personalisation of creatives based on learnings from user corpus, web crawling.

i)Entity bindings - like Name, People, Place(User Hyper Local Data from Quadgen), Brand

2)**Emotion Personas**

Developing user emotion personas towards entities based on learnings from user corpus.

Making use of third party datasets.

a)User entity interests - Emotion towards entities

b)SSO data/user comments Analysis

c)Personality Trait Analysis

d)Color Affinities

3)**Hyper Local Personas**

a)Region specific Personas/Signals Learning and Audience Enhancement

(For example - Rajasthan Patrika, Puthiyathalaimurai.com, North East Today, Marathi Star).

 b)**Language Affinities for Vernacular Ads**

 c)Data received from hotspots like Outlets - McDonalds, Costa, KFC, Airports - Mumbai T1,T2 Airport and other domains (Using Quadgen) -

Schools

Colleges

Universities

Govt and Public Sector Offices

Hospitals

Bus Stations/Stops

Banks

Metro Stations

Public Parks

Monuments/Temples

Public Markets/Agriculture Yards

Malls

Stadiums

Convention Centres

Public Library

 Learning based on ISP Logs Data collected via Wi-Fi hotspots.

4)**Demography Personas**

Using Specific Demography oriented Publisher, SSO data, Browsing behavior learning.

(Womansera.com)

 5)**Domain specific Personas**

Like Bollywood category (Blog to Bollywood), Education category (Idp), Business category oriented Publishers (silicon india)

Linking above domain Personas for Campaign Targeting via Google AdManager, native Ad server and Audience Expansion Tools.

**Data points available in SSO from different Applications – Surveys, forms, facebook connect - facebook graph api, articles likes/share, user friends and other entities like twitter, google, whatsapp.**

**Datapoints available from Facebook Graph API**

1)

<https://graph.facebook.com/oauth/authorize?client_id=117787264903013&display=popup&redirect_uri=http://socialappsintegrator.indiatoday.com/socialsite/facebookresponse&scope=email,offline_access,user_birthday,user_about_me,user_hometown,user_location,publish_stream,user_online_presence,user_photos,publish_actions>

2) <https://graph.facebook.com/me?access_token=AAABrIHwVZA2UBAJrCoMtLNodfZCoFAZBl7BX2MKZAmKZAgNc0KlzJDWo740wvILSZBuhmkTfsgKoQ1BPhInPZBCAHvcZCdEoOUjSrL4EcniZC0gZDZD&fields=id,first_name,last_name,name,link,birthday,email,hometown,location,gender,religion,verified>

**Full List of Data points available here –**

<https://developers.facebook.com/docs/permissions/reference>

**XML format data**

<?xml version="1.0" encoding="utf-8" ?>

[-](http://jsso.indiatimes.com/sso/GetUserProfileXML?User=nitin&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b##) <ProfileDocument>

[-](http://jsso.indiatimes.com/sso/GetUserProfileXML?User=nitin&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b##) <IndiatodayUserProfile>

[-](http://jsso.indiatimes.com/sso/GetUserProfileXML?User=nitin&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b##) <General>

  <USR\_ID\_VC>suhasagg</USR\_ID\_VC>

  <EML\_VC>suhasagg@gmail.com</EML\_VC>

  <FRST\_NM\_VC>Suhas</FRST\_NM\_VC>

  <LST\_NM\_VC>Aggarwal</LST\_NM\_VC>

  <FRGT\_PSSWRD\_QSTN\_VC>what is your pet’s name? </FRGT\_PSSWRD\_QSTN\_VC>

  <FRGT\_PSSWRD\_ANSWR\_VC>billi</FRGT\_PSSWRD\_ANSWR\_VC>

  <SITE\_REG />

  <CRT\_DT />

  <M\_NAME />

  <ADDRESS />

  <CITY>Noida</CITY>

  <STATE>UP</STATE>

  <COUNTRY>India</COUNTRY>

  <PIN>411027</PIN>

  <DOB>1990-07-27</DOB>

  <GENDER>M</GENDER>

  <OCCUPATION />

  <INDUSTRY />

  <EDUCATION />

  <INCOMEGROUP />

  <REFERREL />

  <TRAVELFREQUENCY />

  <AIRTRAVEL />

  <CRT\_TIME>00:00:00</CRT\_TIME>

  <POP />

  <IMAP />

  <RESPHONE />

  <MOBILIEPHONE />

  <ONLINEUSAGE />

  <ONLINESHOPPING />

  <MARITALSTATUS />

  <SMS\_STATUS />

  <PROFILE\_STATUS />

  </General>

<Interests>

  <AUCTIONS />

  <AUTO />

  <HEALTH\_BEAUTY />

  <SPITITUALITY />

  <TOYS\_GAMES />

  <CONTEST\_FREESTUFF />

  <TRAVEL />

  <SHOPPING />

  <COMPUTER\_TECHNOLOGY />

  <BUSINESS\_FINANCE />

  <MUSIC\_MOVIES />

  <LIFESTYLE />

  <NEWS />

  <SPORTS />

  </Interests>

Data from https://www.bagittoday.com/– ecommerce portal

BagItToday

<Ownership>

  <FRIDGE />

  <GRINDER\_JUICER />

  <COLORTV />

  <CREDITCARD />

  <A\_C />

  <CELLPHONE />

  <DISHWASHER />

  <COOKINGRANGE />

  <MICROWAVE />

  <MUSICSYSTEM />

  <COMPUTER />

  <TWOWHEELER />

  <FOURWHEELER />

  <WMACHINE />

  </Ownership>

  </IndiatodayUserProfile>

  </ProfileDocument>

Cross domain SSO enables seamless integration of DMP and SSO data points from different portals belonging to India Today Group.

## 

## USERID BASED SSO APIs

UserId based APIs support the user information access with the channels that use user id of the user as his primary identifier. All the APIs has userid as one of the mandatory fields (the other being site id that is used to identify the channel that is calling the API). The site id is used to track the user activity on IndiaToday user base.

1. **Insert User Profile**

This API is used to insert a new user profile. The parameters that can be inserted and saved at SSO end are:

User, Password, PasswordQues, PasswordAns, FirstName, LastName, Email, RootSite, mname, ipaddress, city, state, country, pin, dob, gender, mobile and maritalstatus. PasswordQues and PasswordAns are used for password retrieval incase user forgets his password. The parameter mname can be used to save the middle name. The mandatory fields which must be passed in the Insert API URL are: User, Password, PasswordQues, PasswordAns, FirstName, LastName and RootSite. The siteid is also a mandatory field which must be passed in the URL, which is used to track the user activity on a particular web application.

**URL:**

[http://sso.indiatoday.com/sso/AddUser?User=suhasagg&Password=password&PasswordQues=a&PasswordAns=a&FirstName=TestUser&LastName=a&Email=as@as.as&RootSite=as&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/AddUser?User=suhasagg&Password=password&PasswordQues=a&PasswordAns=a&FirstName=TestUser&LastName=a&Email=as@as.as&RootSite=as&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

Based on the record insertion the following output is generated:

* USER\_SAVED=exist Incase user already exists.
* USER\_SAVED=true Incase user information is saved.

**Error Handling:**

Incase there occurs some error because of some invalid parameter name or because of some invalid parameter value, message USER\_SAVED=error is send in response.

1. **Change Password**

This API is used to change password. The mandatory fields are username, old password and new password.

**URL:**

[http://sso.indiatoday.com/sso/ChangePassword?User=suhasagg&Password=1234567&NewPassword=12233&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/ChangePassword?User=suhasagg&Password=1234567&NewPassword=12233&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

The API gives the following output:

* PASS\_CHANGED=true Incase the password is changed successfully.
* PASS\_CHANGED=false Incase the password is not changed.

**Error Handling:**

Incase there occurs some error because of some invalid parameter name or because of some invalid parameter value, message PASS\_CHANGED=error is send in response. If user does not exists message PASS\_CHANGED=notexist is send. If any of the passwords is passed as blank then string FieldBlank is shown in response.

1. **Check User Credentials**

This API checks if given user credentials match as per SSO database. The necessary fields are username and password.

[http://sso.indiatoday.com/sso/CheckUserCredentials?username=suhasagg&password=12233&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/CheckUserCredentials?username=suhasagg&password=12233&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

The API gives result in XML format. If credentials match then true is returns else boolean tag will have false value.

<? xml version="1.0" encoding="utf-8"?>

<boolean >true</boolean>

**Error Handling:**

If username is left blank then response is **Missing parameter: UserName.** If password is left blank the response is **Missing parameter: Password.**

1. **Check If User Already Exist**

This API is used to find out whether a given user already exists in the SSO user database.

The mandatory field is username.

**URL:**

[http://sso.indiatoday.com/sso/CheckUserExist?User=suhasagg&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/CheckUserExist?User=suhasagg&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

The API returns following output:

* USER\_EXISTS=true, if user already exists in the user database.
* USER\_EXISTS=false, if user does not exists in the user database.

**Error Handling:**

* USER\_EXISTS=error string is send as response if username is not passed in the URL.
* Invalid User string is send as response if username in invalid (contains special characters).

1. **Validate User Information**

This API is used to validate user information. The necessary fields are username and password.

[http://sso.indiatoday.com/sso/CheckValidUser?User=suhasagg&Password=12233&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/CheckValidUser?User=suhasagg&Password=12233&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

* VALID\_USER=true string is return if user credentials match.
* VALID\_USER=false string is returned if user credentials do not match.

**Error Handling:**

VALID\_USER=error is returned as response if parameters are not passed or they are invalid**.**

1. **Get Security Question and Answer**

This API is used to get the security question chosen by user during profile creation. The mandatory field is username. The API returns XML with security question as a value SecQues tag.

**URL:**

[http://sso.indiatoday.com/sso/ForgotQuestion?User=suhasagg&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/ForgotQuestion?User=suhasagg&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

<?xml version="1.0" encoding="utf-8"?>

<User>

<SecQues>whats your nick name</SecQues>

</User>

**Error Handling:**

* If username is not provided or incase it is invalid, Error Message with error code E109 is send as response.
* If username is provided does not exists, Error Message with error code E117 is send as response.

1. **Get Forgot Password**

This API is used to change the password of a user. The mandatory fields are username, security question (that can be obtained from the above API), and security answer provided by the user for the given security question and NewPassword .

**URL:**

[http://sso.indiatoday.com/sso/ForgotPassword?User=suhasagg&PasswordQues=what](http://jsso.indiatimes.com/sso/ForgotPassword?User=nitin&PasswordQues=what) is mypetsname?&PasswordAns=billi&siteid=3b7afdd57ace08aceb89fef03b7d9a7c&NewPassword=123456

**Output:**

This api on success will return “RESETPASSWORD = true”.

**Error Handling:**

* GETPASSWORD=error message is thrown as error if any one of the mandatory fields is not provided
* GETPASSWORD=false message is thrown as error username provided is blank.
* PasswordQuestion=FieldBlank message is thrown when Security Question is provided as blank.
* PasswordAnswer=FieldBlank message is thrown when Security Answer is provided as blank.
* PasswordLength=Password should be between 6 and 14 characters message is thrown if length of new password is less than 6 or more than 14 characters.

1. **Get User Profile in XML**

This API return user profile in XML format. The mandatory fields are username and site id.

**URL:**

[http://sso.indiatoday.com/sso/GetUserProfileXML?User=suhasagg&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/GetUserProfileXML?User=suhasagg&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

  <?xml version="1.0" encoding="utf-8" ?>

[-](http://jsso.indiatimes.com/sso/GetUserProfileXML?User=nitin&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b##) <ProfileDocument>

[-](http://jsso.indiatimes.com/sso/GetUserProfileXML?User=nitin&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b##) <IndiaTodayUserProfile>

[-](http://jsso.indiatimes.com/sso/GetUserProfileXML?User=nitin&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b##) <General>

  <USR\_ID\_VC>suhasagg</USR\_ID\_VC>

  <EML\_VC>suhasagg@gmail.com</EML\_VC>

  <FRST\_NM\_VC>Suhas</FRST\_NM\_VC>

  <LST\_NM\_VC>Aggarwal</LST\_NM\_VC>

  <FRGT\_PSSWRD\_QSTN\_VC>what is your pet’s name? </FRGT\_PSSWRD\_QSTN\_VC>

  <FRGT\_PSSWRD\_ANSWR\_VC>billi</FRGT\_PSSWRD\_ANSWR\_VC>

  <SITE\_REG />

  <CRT\_DT />

  <M\_NAME />

  <ADDRESS />

  <CITY>Noida</CITY>

  <STATE>UP</STATE>

  <COUNTRY>India</COUNTRY>

  <PIN>411027</PIN>

  <DOB>1990-07-27</DOB>

  <GENDER>M</GENDER>

  <OCCUPATION />

  <INDUSTRY />

  <EDUCATION />

  <INCOMEGROUP />

  <REFERREL />

  <TRAVELFREQUENCY />

  <AIRTRAVEL />

  <CRT\_TIME>00:00:00</CRT\_TIME>

  <POP />

  <IMAP />

  <RESPHONE />

  <MOBILIEPHONE />

  <ONLINEUSAGE />

  <ONLINESHOPPING />

  <MARITALSTATUS />

  <SMS\_STATUS />

  <PROFILE\_STATUS />

  </General>

[-](http://jsso.indiatimes.com/sso/GetUserProfileXML?User=nitin&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b##) <Interests>

  <AUCTIONS />

  <AUTO />

  <HEALTH\_BEAUTY />

  <SPITITUALITY />

  <TOYS\_GAMES />

  <CONTEST\_FREESTUFF />

  <TRAVEL />

  <SHOPPING />

  <COMPUTER\_TECHNOLOGY />

  <BUSINESS\_FINANCE />

  <MUSIC\_MOVIES />

  <LIFESTYLE />

  <NEWS />

  <SPORTS />

  </Interests>

[-](http://jsso.indiatimes.com/sso/GetUserProfileXML?User=nitin&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b##) <Ownership>

  <FRIDGE />

  <GRINDER\_JUICER />

  <COLORTV />

  <CREDITCARD />

  <A\_C />

  <CELLPHONE />

  <DISHWASHER />

  <COOKINGRANGE />

  <MICROWAVE />

  <MUSICSYSTEM />

  <COMPUTER />

  <TWOWHEELER />

  <FOURWHEELER />

  <WMACHINE />

  </Ownership>

  </IndiaTodayUserProfile>

  </ProfileDocument>

**Error Handling:**

If user does not exists. The then ProfileDocument tag of XML profile will not have any value.

<?xml version="1.0" encoding="utf-8" ?>

<ProfileDocument />

1. **Update User Profile**

This API is used to update an already saved profile.

[http://sso.indiatoday.com/sso/UpdateUser?User=suhasagg&PasswordQues=a&PasswordAns=a&FirstName=suhasagg&LastName=Aggarwal&Email=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/UpdateUser?User=suhasagg&PasswordQues=a&PasswordAns=a&FirstName=suhasagg&LastName=Aggarwal&Email=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a)

1. **Get User Profile**

This API returns user profile. The mandatory fields are username and site id.

[http://sso.indiatoday.com/sso/GetUserProfile?User=suhasagg&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/GetUserProfile?User=suhasagg&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

User=suhasagg

FirstName=Suhas

LastName=Aggarwal

EMail=suhasagg@gmail.com

PasswordQues= what is your pet’s name?

PasswordAns=billi

CreationDate=

RootSite=

MiddleName=

Address=

City=Noida

State=UP

Country=India

Pin=411027

DOB=27/07/1990

Gender=M

Occupation=

Industry=

Education=

IncomeGroup=

TravelFrequency=

ResidencePhone=

MobilePhone=

OnlineUsage=

OnlineShopping=

MaritalStatus=

**Error Handling:**

GETRESOURCE=invaliduser is send as response if user does not exists.

1. **Login**

This API takes login, password, ru, nru and siteid as parameters and logs in user to the system after password validation. The login can be username or the email id of the user. (This API will also be used for email based login.)

http://sso.indiatoday.com/sso/GenericLogin?login=suhasagg&passwd=1234&ru=http://www.indiatoday.com/home.php&nru=http://www.indiatoday.com/dologin.php &siteid=[ef55d27920fbf01f58c0da43430f5c6a](http://127.0.0.1:8080/sso/ActivateEmailId?hashcode=ZHRzdGM0eGNhMG54ajhsMm5zMWQzbHg2cjpuaXRpbjEyM0BnbWFpbC5jb20=&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)

**Error Handling:**

* + Site id is incorrect: The value of site id is incorrect. Redirected to login page with error code E108 in the URL.
  + Invalid Parameters: No values provided for some of the mandatory fields. Redirected to login page with error code E118 in the URL.
  + Login Credentials are incorrect: Username password is not matched. Redirected to login page with error code E119 in the URL.
  + Fetching Error – Either the user does not exist or the DB is down or there are some network errors. Redirected to login page with error code E104 in the URL.

**12. Logout**

This API logs out the user from the system and lands the user to a URL called “nru”.

**URL:**

[http://](http://www.itimes.com:8080/sso/CMSLogout?siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)sso.indiatoday.com/sso[/GenericLogout?siteid=ef55d27920fbf01f58c0da43430f5c6a](http://www.itimes.com:8080/sso/CMSLogout?siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)&nru=http://www.indiatoday.com

1. **Get Encrypted Cookie Value**

This API is used to decrypt the encrypted cookie value of **MSCSAuth** cookie set during login. This gives the login parameter value given by the user during login.

**URL:**

[http://sso.indiatoday.com/sso/CookieStrCheckXML?str=&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatimes.com/sso/CookieStrCheckXML?str=&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)

**Error Handling:**

Incase some invalid string (not encrypted by the SSO end) is passed as parameter the following XML is send in response.

<?xml version="1.0" encoding="utf-8"?>

<ERROR>Error</ERROR>

## EMAIL BASED APIs

Email based APIs support the user information access with the channels that use email id of the user as his primary identifier. All the APIs has email id as one of the mandatory fields (the other being site id that is used to identify the channel that is calling the API).

1. **Get User Profile:**

The user profile is returned in XML format, when emailed of the user and site id are provided as parameters. This API works only for activated users.

**URL:**

[http://sso.indiatoday.com/sso/GetXMLUserProfile?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/GetXMLUserProfile?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

<NewDataSet>

−

<Table>

<usr\_id\_vc>suhas</usr\_id\_vc>

<eml\_vc>suhas@gmail.com</eml\_vc>

<primary\_email\_id>suhasagg@gmail.com</primary\_email\_id>

<CRT\_DT/>

<DateRegistered/>

<site\_reg>sd</site\_reg>

<frst\_nm\_vc>suhas</frst\_nm\_vc>

<lst\_nm\_vc>AGGARWAL</lst\_nm\_vc>

<Address>qw</Address>

<Gender>M</Gender>

<city/>

<state/>

<country/>

<dob>1990-07-27</dob>

<pin/>

<frgt\_psswrd\_qstn\_vc/>

<frgt\_psswrd\_answr\_vc/>

<mobilephone/>

<Referrel/>

<UserType>1</UserType>

</Table>

</NewDataSet>

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.
  + Email id does not exist: Error message with error code E104 is thrown.

1. **Check Password Strength:**

This API can be used in order to check the strength of a password string. The parameters are password and siteid. The APIs returns XML document with password string categorized as very weak, weak, medium, strong and very strong.

**URL:**

[http://sso.indiatoday.com/sso/CheckPasswordStrength?password=suhassagg&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/CheckPasswordStrength?password=suhassagg&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

<string>very weak</string>

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.

1. **Check Password Strength of an existing user:**

This API checks password strength of an existing profile. The parameters to the API are email of the user and siteid and password. The APIs returns XML document with password string categorized as very weak, weak, medium, strong and very strong.

**URL:**

[http://sso.indiatoday.com/sso/CheckPasswordStrengthUser?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/CheckPasswordStrengthUser?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a)&password= eyoweryor

**Output:**

<string>very weak</string>

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.

1. **Check if the email id is available for login:**

This API checks whether a new email id for a profile already exists. The parameters to the API are email and siteid.

**URL:**

[http://sso.indiatoday.com/sso/isEmailIdAvailable?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/isEmailIdAvailable?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

<status>0</status>

If email id is there in the repository then status is 0, otherwise if email id is available then status 1 is returned.

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.

1. **Add User Profile:**

This API is used to insert a new user profile. The email id, password and siteid are mandatory fields and should be provided in every case. The inputs to the API are emailid, firstname, middlename, lastname, password, sitereg, ipaddress, city, state, country, secques, secans, pin, dob, gender, mobile. The parameters secques and secans are used for password retrieval incase user forgets his password. The parameter mname can be used to save the middle name.

**URL:**

[http://sso.indiatoday.com/sso/InsertUserProfile?emailid=suhasagg@gmail.com&firstname=Suhas&lastname=Aggarwal&password=123&mobile=1234567899&dob=1990/07/27&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/InsertUserProfile?emailid=suhasagg@gmail.com&firstname=Suhas&lastname=Aggarwal&password=123&mobile=1234567899&dob=1990/07/27&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

On successful insertion user profile in XML format is returned.

<NewDataSet>

−

<Table>

<usr\_id\_vc>94ttvdhdtdt340p2j8hk9wtt1</usr\_id\_vc>

<eml\_vc>suhasagg@gmail.com</eml\_vc>

<CRT\_DT>Fri Jan 22 10:55:38 IST 2010</CRT\_DT>

<DateRegistered>Fri Jan 22 10:55:38 IST 2010</DateRegistered>

<site\_reg/>

<frst\_nm\_vc/>

<lst\_nm\_vc>Aggarwal</lst\_nm\_vc>

<Address/>

<Gender/>

<city/>

<state/>

<country/>

<dob>27/07/1990</dob>

<pin/>

<frgt\_psswrd\_qstn\_vc/>

<frgt\_psswrd\_answr\_vc/>

<mobilephone>1234567899</mobilephone>

<Referrel/>

<UserType>0</UserType>

</Table>

−

<VerificationCode>

OTR0dHZkaGR0ZHQzNDBwMmo4aGs5d3R0MTpuaXRpbnF3ZUBnbWFpbC5jb20=

</VerificationCode>

−

</NewDataSet>

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.
  + Email id already present: Error message with error code E103 is thrown.

1. **Activate emailed of a registered user:**

This API is for activation of a newly registered profile. The parameters to the API are hash code and site id. The parameter autologin when passed in the API with value true gives value of MSCSAuth, MSCSAuthDetail and MSCSAuthDetails cookies in base64 encoding. The applications can set these cookies at their end to allow user auto login after registration, after getting status true.

**URL:**

[http://sso.indiatoday.com/sso/ActivateEmailId?hashcode=ZHRzdGM0eGNhMG54ajhsMm5zMWQzbHg2cjpuaXRpbjEyM0BnbWFpbC5jb20=&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://192.168.58.137:9080/sso/ActivateEmailId?hashcode=ZHRzdGM0eGNhMG54ajhsMm5zMWQzbHg2cjpuaXRpbjEyM0BnbWFpbC5jb20=&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)

**Output:**

<status>true</status>

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.
  + Hash Code provided is invalid: Error message with error code E105 is thrown.

1. **Update User Profile:**

This API is used to update a newly activated user profile. The inputs to the API are emailid, firstname, middlename, lastname, password, sitereg, ipaddress, city, state, country, secques, secans, pin, dob, gender, mobile. The email id is mandatory field and should be provided in every case.

**URL:**

[http://sso.indiatoday.com/sso/UpdateUserProfile?emailid=suhasagg@gmail.com&firstname=Suhas&lastname=Aggarwal&password=1234&dob=27071990&ipaddress=qw&sitereg=sd&gender=M&city=&state=&country=&secques=&secans=&mobile=&pin=&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/UpdateUserProfile?emailid=suhasagg@gmail.com&firstname=Suhas&lastname=Aggarwal&password=1234&dob=27071990&ipaddress=qw&sitereg=sd&gender=M&city=&state=&country=&secques=&secans=&mobile=&pin=&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

<status>true</status>

When profile is updated successfully status true is returned, otherwise status false is returned.

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.

1. **Change Old Password of a Active User:**

This API is used to change the password please specify an email id and new password to be set. The input to the API is email id of the user, his old password, new password and site id.

**URL:**

[http://sso.indiatoday.com/sso/changeOldPassword?emailid=suhasagg@indiatoday.com&oldPassword=qwerty123456&newPassword=ytrewq654321&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/changeOldPassword?emailid=suhasagg@indiatoday.com&oldPassword=qwerty123456&newPassword=ytrewq654321&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

<status>true</status>

When password is updated successfully status true is returned, otherwise status false is returned.

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.
  + Some Insertion Error: Error message with error code E101 is thrown.

1. **Add Email Mapping:**

This API is used to update current email mapping of a user. The following cases can exist for updating email mapping:

1. Use any new email id to map to an existing userid. Set a new email id in the email id parameter.  
2. The userid can be any existing userid of the same user already registered with et.com. For e.g: we can map multiple email ids to same userid, so userid value can be set as zxz12 here.

3. Primary option: can be either 0 or 1, i.e. if the user wants the new email id to be set as his primary email id, and then set the value 1 for primaryOpt parameter.

**URL:**

[http://sso.indiatoday.com/sso/AddEmailMapping?userid=12345678uyi&emailid=suhasagg@gmail.com&primaryopt=0&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/AddEmailMapping?userid=12345678uyi&emailid=suhasagg@gmail.com&primaryopt=0&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

On successful insertion API will return status will be 0 along with verification code value. Else status is -1 with blank verification code.

<emailid>

<status> 0 </status>

<VerificationCode>47848729</VerificationCode>

</emailid>

or

<emailid>

<status>-1</status>

<VerificationCode/>

</emailid>

1. **Fetch Alias for a registered user:**

Fill in an existing userid in order to fetch its primary email id.

**URL:**

[http://sso.indiatoday.com/sso/fetchAlias?userid=6xme84tdyjvuto38jftpj26a8&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatimes.com/sso/fetchAlias?userid=6xme84tdyjvuto38jftpj26a8&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)

**OUTPUT:**

<PrimaryEmailid> suhasagg@gmail.com </PrimaryEmailid>

<SecondaryEmailid> suhasagg@gmail.com </SecondaryEmailid>

<SecondaryEmailid> suhasagg@gmail.com </SecondaryEmailid>

**Error Handling:**

* + Some Insertion Error: Error message with error code E107 is thrown.

1. **Login:**

This API takes login, password, ru, nru and siteid as parameters and logs in user to the system after password validation. The login can be username or the email id of the user.

http://sso.indiatoday.com/sso/GenericLogin?login=[suhasagg@gmail.com](mailto:suhasagg@gmail.com)&passwd=1234&ru=http://www.aajtak.com/home.php&nru=http://www.aajtak.com/dologin.php &siteid=[ef55d27920fbf01f58c0da43430f5c6a](http://127.0.0.1:8080/sso/ActivateEmailId?hashcode=ZHRzdGM0eGNhMG54ajhsMm5zMWQzbHg2cjpuaXRpbjEyM0BnbWFpbC5jb20=&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)

**Error Handling:**

* + Site id is incorrect: The value of site id is incorrect. Redirected to login page with error code E108 in the URL.
  + Invalid Parameters: No values provided for some of the mandatory fields. Redirected to login page with error code E118 in the URL.
  + Login Credentials are incorrect: Username password is not matched. Redirected to login page with error code E119 in the URL.
  + Fetching Error – Either the user does not exist or the DB is down or there are some network errors. Redirected to login page with error code E104 in the URL.

**13. Logout:**

This API logs out the user from the system and lands the user to a URL called “nru”.

[http://](http://www.itimes.com:8080/sso/CMSLogout?siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)sso.indiatoday.com/sso[/GenericLogout?siteid=ef55d27920fbf01f58c0da43430f5c6a](http://www.itimes.com:8080/sso/CMSLogout?siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)&nru=http://www.indiatoday.com

**Error Handling:**

* + Site id is incorrect: Error message with error code E108 is thrown.

1. **Load Latest Site Ids:**

This API loads the latest side ids information in the memory. This is called whenever new site information is added in the database.

**URL:**

[http://sso.indiatoday.com/sso/LoadLatestSiteIds](http://jsso.indiatimes.com/sso/LoadLatestSiteIds)

1. **Insert Site Detail :**

This API is for insertion of new site details in the DB.

**URL:**

http:// sso.indiatoday.com/sso/InsertNewSiteInfo?sitename=sitename&desc=desc

Site id is calculated as md5 of site name provided.

**Error Handling:**

* + Site Already Exists: Error message with error code E120 is thrown.

1. **Get Activation Code**

This API is used to get activation code of a user when a registered email is given in the URL.

**URL:**

[http://sso.indiatoday.com/sso/GetActivationCode?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/GetActivationCode?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

<?xml version="1.0" encoding="utf-8"?>

<ActivationInfo>

<VerificationCode>Y2I1cjMwNTNmaGh3eXNwYXZldTBrNHI2dTpuaXRpbjE5ODJAZ21haWwuY29tOjEyNzgwNDY0Nzc2MTI=</VerificationCode>

<isActive>false</isActive>

<firstName>Suhas</firstName>

<lastName>Aggarwal</lastName>

</ActivationInfo>

1. **Get Available UserIds**

This API gives the list of users available for registration. The mandatory fields are userid, firstname, lastname, number of options required as reqnos, and siteid. The userid is passed as emailid with domain name Indiatoday or Aajtak

**URL:**

[http://sso.indiatoday.com/sso/GetAvailableUserIds?userid=suhasagg@indiatoday.com&firstname=suhas&lastname=aggarwal&reqdnos=12&dob=27/07/1990&siteid=b41089221688240eb535bb15b71d9d1b](http://jsso.indiatoday.com/sso/GetAvailableUserIds?userid=suhasagg@indiatoday.com&firstname=suhas&lastname=aggarwal&reqdnos=12&dob=27/07/1990&siteid=b41089221688240eb535bb15b71d9d1b)

**Output:**

<?xml version="1.0" encoding="utf-8"?>

<USEROPTIONS>

<USER>suhasagg@indiatoday.in</USER>

<USER>suhasagg@indiatoday.in</USER>

<USER>suhasagg@indiatoday.in</USER>

<USER>suhasagg@indiatoday.in</USER>

<USER>suhasagg@indiatoday.in</USER>

<USER> suhasagg@indiatoday.in </USER>

<USER> suhasagg@indiatoday.in </USER>

<USER> suhasagg@indiatoday.in </USER>

<USER> suhasagg@indiatoday.in </USER>

<USER> suhasagg@indiatoday.in </USER>

<USER> suhasagg@indiatoday.in </USER>

<USER> suhasagg@indiatoday.in </USER>

<USER> suhasagg@indiatoday.in </USER>

<USER> suhasagg@indiatoday.in </USER>

</USEROPTIONS>

**Error Handling:**

* + Domain is invalid: Error message with error code E121 is send as response.
  + Request Number is invalid: Error message with error code E104 is send as response.
  + Date of Birth is invalid: Error message with error code E112 is send as response.

1. **User Validation**

This API can be used for user validation usually when login forms implement Ajax based calls to SSO servers. The mandatory fields are username (with parameter name User), password (with parameter name Password) and site id (with parameter name siteid). The username can be email ids also. No cookies are created after user validation.

**URL:**

[http://sso.indiatoday.com/sso/GenericUserValidation?User=suhasagg&Password=suhas&siteid=b41089221688240eb535bb15b71d9d1b](http://jsso.indiatoday.com/sso/GenericUserValidation?User=suhasagg&Password=suhas&siteid=b41089221688240eb535bb15b71d9d1b)

**Output:**

VALID\_USER=true message is send as response incase valid credentials are passed.

**Error Handling:**

VALID\_USER=false message is send as response incase invalid credentials are passed.

VALID\_USER=error message is send as response incase mandatory fields are not passed.

1. **Update Inactive User**

This API can be used to activate user incase he is not activated during the activation process i.e. when activation code is passed. The API will activate all the email ids (primary as well as secondary) mapped to the user having the email id passed in the API.

**URL:**

[http://sso.indiatoday.com/sso/UpdateInactiveUser?emailid=suhasagg@gmail.com&siteid=b41089221688240eb535bb15b71d9d1b](http://jsso.indiatoday.com/sso/UpdateInactiveUser?emailid=suhasagg@gmail.com&siteid=b41089221688240eb535bb15b71d9d1b)

**Output:**

<status>true</status> XML is returned incase an inactive user is made active.

**Error Handling:**

* Error Message with error code E106 is send as response if user is already activated.
* If email id is not valid then email validity related error messages are thrown as per Field Validation Error Coded.

1. **Token Generator**

This API is used to generate token that can be used as a security token in case of secure APIs like Change Password when user forgets his old password. The mandatory field is emailid and siteid.

**URL:** [http://sso.indiatoday.com/sso/RequestChangePassword?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatoday.com/sso/RequestChangePassword?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a)

**Output:**

The following XML is returned as response with value of authtoken in authtoken tag.

<?xml version="1.0" encoding="utf-8"?>

<authtoken>a9b2b13cca472136665f864445cece22</authtoken>

**Error Handling:**

* + Error Message with error code E107 is send as response incase emailid is not registered.
  + Error Message with error code E112 is send as response incase emailid is registered but not activated.

1. **Change Old Password**

This API is used to change password. The mandatory fields are emailid, new password, authtoken (that is generated in API number 20). This API is a part of forget password case (that is discussed in this document).

**URL:**

[http://sso.indiatoday.com/sso/ChangeOldPassword?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a&password=suhasagg&authtoken=a9b2b13cca472136665f864445cece22](http://jsso.indiatoday.com/sso/ChangeOldPassword?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a&password=suhasagg&authtoken=a9b2b13cca472136665f864445cece22)

**Output:**

The following XML is returned as response with value of status tag true or false incase password is changed or not changed.

<?xml version="1.0" encoding="utf-8"?>

<status>true</status>

**Error Handling:**

* + Error Message with error code E102 is send as response incase there is some error in updation.

1. **Check Login Type**

This api is used to know the type by which the user last logged in.The logged in type can be either facebook, twitter or sso.

**URL:**

http://sso.indiatoday.com/sso/checkLoginType?emailId=suhasagg@gmail.com&callback=func&siteId=ef55d27920fbf01f58c0da43430f5c6a

**INPUT:**

Site id and jsonp function name are necessary parameters.

EmailId is optional parameter. If provided in the input , this api will provide in output the last login type of the user as mentioned in this parameter. If not, then it will provide the last login type of the user who is currently logged in.

**OUTPUT:**

The following string is returned as a json response.

func({"isUserLoggedIn":"true","LoginType":"sso","code":"200"});

**ERRORS:**

1)If user is not logged in and emailed parameter is not provided then the following error will get returned.

func({"error":"User not logged in","code":"401"});

2)If siteId is not provided then the following error will get returned.

func({"error":"Invalid site Id","code":"400"});

3) If jsonp function name is not provided as input the following error will get returned.

func({"error":"Invalid jsonp request","code":"400"});

4)If emailId provided as a parameter is not a valid sso user then the following error will get returned.

func({"error":"Not a valid user","code":"401"});

1. **Is User Logged In**

**URL:**

[http://sso.indiatoday.com/sso/IsUserLoggedIn?callback=func](http://jsso.indiatimes.com/sso/IsUserLoggedIn?callback=func)

This api is used to know whether the user is currently logged in. It checks whether there are cookies set in the browser. If yes then it returns true and if cookies are not set, it returns false.

**INPUT:**

There are no necessary parameters . callback can be given as optional parameter. If callback is given as parameter it returns a jsonp response. If callback is not provided as a parameter then the api returns a string which is either true or false.

## Registration Process Workflow

Steps of registration:

* User fill registration form of the application.
* Application will call SSO API 3 Add User Profile, and pick <VerificationCode>.
* Application will prepare some URL like ([http://www.indiatoday.com/activate?hs="](http://www.indiatoday.com/activate?hs=%22)verification code") and send this link in email to user.
* User will click on this link and the application will internally call our activation API (API number 4).
* The SSO API will send application result <status>true</status> or <status>false</status> based on the user is activated or not activated.
* The application redirects user based on the response.

Note: SSO will interact with the application only.

Diagram

Description automatically generated

Figure 6: Sequence Diagram for Registration Process

## Retrieve Forgot Password Workflow

The Indiatoday channels collectively use userid based APIs as well as email based APIs for accessing user information. In order to allow all types of users (with userid or email id as unique identifier) to change password, we need to implement the following two different processes:

### Change Password Workflow for User Based Access

The steps to get new password in case of forgot password:

1. The application will call SSO with the following API [http://sso.indiatoday.com/sso/ForgotQuestion?User=testuser&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.indiatimes.com/sso/ForgotQuestion?User=testuser&siteid=bb857c2ade1c6bcc4a9cccc12e89e39b)

The API will give password question as output. This question was selected by the user during registration process.

2. The question will be shown to user on some form at application end.

3. The user will enter the answer (this was entered by the user during registration process).

4. The application will call forgot password API (at SSO end):

[http://sso.indiatoday.com/sso/ForgotPassword?User=suhasagg&PasswordQues=what](http://jsso.indiatoday.com/sso/ForgotPassword?User=suhasagg&PasswordQues=what) is my+pet’s name?&PasswordAns=billi&siteid=3b7afdd57ace08aceb89fef03b7d9a7c

5. SSO will send user password or some error message as response after validating the security answer against security question chosen by the user.

6. The result will be used at the application end for redirecting the user accordingly.

Diagram

Description automatically generated

Figure 7: Retrieve Password for Used Based APIs

### Change Password Workflow for Email Based Access

The steps to get new password in case of forgot password:

1. The application will call SSO with the following API [http://sso.indiatoday.com/sso/RequestChangePassword?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a](http://jsso.itimes.com/sso/RequestChangePassword?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a)

The API will give authtoken as output which is the authority token.

2. The token will be mailed to the user in some link (of change password page at application end).

3. The user will click on the link and will fill email id and new password and then submit the form.

4. The application will call change password API (at SSO end):

[http://sso.indiatoday.com/sso/ChangeOldPassword?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a&password=suhasagg&authtoken=a9b2b13cca472136665f864445cece22](http://jsso.itimes.com/sso/ChangeOldPassword?emailid=suhasagg@gmail.com&siteid=ef55d27920fbf01f58c0da43430f5c6a&password=suhasagg&authtoken=a9b2b13cca472136665f864445cece22)

The authtoken parameter value should be same as passed in step 1.

5. SSO will changed password after validating authtoken and give status true/ false based on the validation.

6. The result will be used at the application end for redirecting the user accordingly.

A picture containing diagram

Description automatically generated

Figure 8: Sequence Diagram for Retrieving Forget Password

## Error Code Table

|  |  |  |
| --- | --- | --- |
| **Error Code** | **Error Type** | **Error Message** |
| **E101** | Record Insertion Error | - <Error>  <ErrCode>E101</ErrCode>  <Message>Insertion Error</Message>  <Exception>Unable to insert record.</Exception>  </Error> |
| **E102** | Record Updation Error | <Error>  <ErrCode>E102</ErrCode>  <Message>Updation Error</Message>  <Exception>Unable to update record.</Exception>  </Error> |
| **E103** | Email Id Already Exists | <Error><ErrCode>E103</ErrCode>  <Message>Emailid already exists</Message>  <Exception>Emailid already exists.</Exception>  </Error> |
| **E104** | Record Not Available Error | <Error><ErrCode>E104</ErrCode>  <Message>Fetching Error</Message>  <Exception>No user profile exists with respect to given emailid.</Exception>  </Error> |
| **E105** | Invalid Hash code Error | <Error><ErrCode>E105</ErrCode>  <Message>Invalid HashCode</Message>  <Exception>The hashcode given is invalid.</Exception>  </Error> |
| **E106** | Email Id Already Activated | <Error>  <ErrCode>E106</ErrCode>  <Message>Email Already Activated</Message>  <Exception>The email already activated.</Exception>  </Error> |
| **E107** | Record Fetching Error | <Error>  <ErrCode>E107</ErrCode>  <Message>Fetching Error</Message>  <Exception>Unable to get record.</Exception>  </Error> |
| **E108** | Invalid Site Id | <Error>  <ErrCode>E108</ErrCode>  <Message>SITE ID Error</Message>  <Exception>Invalid Siteid.</Exception>  </Error> |
| **E109** | Invalid Primary Option Value | <Error>  <ErrCode>E109</ErrCode>  <Message>Error In Insertion</Message>  <Exception>primaryopt value should be 0/1</Exception>  </Error> |
| **E110** | Email Id Already Exists | <Error>  <ErrCode>E110</ErrCode>  <Message>Error In Insertion</Message>  <Exception>This email id already exists. </Exception>  </Error> |
| **E111** | Primary Mapping Updation Error (Not in Use) | <Error>  <ErrCode>E111</ErrCode>  <Message>Error In Insertion</Message>  <Exception>Email exists as primary for this user. You can not change primaryopt value</Exception>  </Error> |
| **E112** | Invalid Date of Birth | <Error>  <ErrCode>E112</ErrCode>  <Message>Invalid Date of Birth </Message>  <Exception> The date of Birth provided is greater then the current date.</Exception>  </Error> |
| **E113** | Invalid Primary Option | <Error>  <ErrCode>E113</ErrCode>  <Message> Invalid Primary Option </Message>  <Exception> The primary option can only have values 0 or 1.  </Exception>  </Error> |
| **E114** | Invalid Mapping Parameter | <Error>  <ErrCode>E114</ErrCode>  <Message> Invalid Mapping Parameters Values</Message>  <Exception> The default emailid cannot be updated to primary/ secondary email mapping. </Exception>  </Error> |
| **E115** | Records Exists | <Error>  <ErrCode>E115</ErrCode>  <Message> Mapping Already Exists </Message>  <Exception> The email mapping already exists. </Exception>  </Error> |
| **E116** | Primary Mapping Error | <Error>  <ErrCode>E116</ErrCode>  <Message> Primary Mapping Error </Message>  <Exception> The primary email mapping cannot be changed to secondary. </Exception>  </Error> |
| **E117** | Invalid Userid | <Error>  <ErrCode>E117</ErrCode>  <Message> Invalid Userid </Message>  <Exception> The userid provided does not exists. </Exception>  </Error> |
| **E118** | Invalid Parameters | <Error>  <ErrCode>E118</ErrCode>  <Message> Invalid Parameters</Message>  <Exception> The parameters provided are not valid. </Exception>  </Error> |
| **E119** | Invalid User Credentials | <Error>  <ErrCode>E119</ErrCode>  <Message> Invalid User Credentials </Message>  <Exception> The user credentials provided are invalid. </Exception>  </Error> |
| **E120** | Site Information Exists | <Error>  <ErrCode>E120</ErrCode>  <Message> Site Information Exists </Message>  <Exception> The provided site already exists. </Exception>  </Error> |
| **E121** | Invalid Domain | <Error>  <ErrCode>E121</ErrCode>  <Message>Invalid Domain</Message>  <Exception> The domain is invalid. </Exception>  </Error> |
| **E122** | Inactive User | <Error>  <ErrCode>E122</ErrCode>  <Message>Inactive User</Message>  <Exception> The user is not active. </Exception>  </Error> |
| **E123** | Invalid Request Number | <Error>  <ErrCode>E123</ErrCode>  <Message> Invalid Request Number </Message>  <Exception>The request number is invalid.</Exception>  </Error> |

## Field Validation Error Codes

The table below gives the error code and their descriptions. These error codes are generated by the Field Validation Framework while validating the values of the parameters. Apart from generating error code in the above table the SSO APIs will generate the errors given below in case parameter values are not correct.

|  |  |  |
| --- | --- | --- |
| **Error Code** | **Error Type** | **Error Message** |
| **N101** | Null Field Value | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>N101</ErrCode>  <Message>Null Field Value for "fieldName"</Message>  <Exception>Field value cannot be null.</Exception>  </Error> |
| **B101** | Blank Field Value | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>B101</ErrCode>  <Message>Blank Field Value for "fieldName"</Message>  <Exception>Field value cannot be blank.</Exception>  </Error> |
| **U101** | Invalid UserId | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>U101</ErrCode>  <Message>Invalid userid</Message>  <Exception>Either no value provided for userid or userid has special characters. </Exception>  </Error> |
| **U102** | Invalid UserId | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>U102</ErrCode>  <Message>Invalid userid</Message>  <Exception>UserId length cannot be more than 50 characters. </Exception>  </Error> |
| **E101** | Invalid Email | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>E101</ErrCode>  <Message>Invalid emailid</Message>  <Exception>EmailId is not valid.</Exception>  </Error> |
| **E102** | Invalid Email | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>E102</ErrCode>  <Message>Invalid emailid</Message>  <Exception>EmailId cannot be more than 100 characters.</Exception>  </Error> |
| **D101** | Invalid DOB | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>D101</ErrCode>  <Message>Invalid DOB</Message>  <Exception>Date of Birth is not valid.</Exception>  </Error> |
| **FN101** | Invalid First Name | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>FN101</ErrCode>  <Message>Invalid First Name</Message>  <Exception>First Name has numbers/special characters. </Exception>  </Error> |
| **FN102** | Invalid First Name | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>FN102</ErrCode>  <Message>Invalid First Name</Message>  <Exception>First Name length cannot be more than 50 characters.</Exception>  </Error> |
| **MDN101** | Invalid Middle Name | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>MDN101</ErrCode>  <Message>Invalid Middle Name</Message>  <Exception>Middle Name has numbers/special characters.</Exception>  </Error> |
| **MDN102** | Invalid Middle Name | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>MDN102</ErrCode>  <Message>Invalid Middle Name</Message>  <Exception>Middle Name length cannot be more than 50 characters.</Exception>  </Error> |
| **LN101** | Invalid Last Name | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>LN101</ErrCode>  <Message>Invalid Last Name</Message>  <Exception>Last Name has numbers/special characters.</Exception>  </Error> |
| **LN102** | Invalid Last Name | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>LN102</ErrCode>  <Message>Invalid Last Name</Message>  <Exception>Last Name length cannot be more than 50 characters.</Exception>  </Error> |
| **MN101** | Invalid Mobile Number | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>MN101</ErrCode>  <Message>Invalid Mobile Number</Message>  <Exception>Mobile number can only contain numbers. </Exception>  </Error> |
| **MN102** | Invalid Mobile Number | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>MN102</ErrCode>  <Message>Invalid Mobile Number</Message>  <Exception>Mobile number length should be between 10 and 14 characters.</Exception>  </Error> |
| **PN101** | Invalid PIN | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>PN101</ErrCode>  <Message>Invalid Pin Number</Message>  <Exception>Pin number can only contain numbers/capital letters/spaces/hyphen. </Exception>  </Error> |
| **PN102** | Invalid PIN | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>PN102</ErrCode>  <Message>Invalid Pin Number</Message>  <Exception>Pin number length should be between 3 and 9 characters. </Exception>  </Error> |
| **OP101** | Invalid Old Password | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>OP101</ErrCode>  <Message>Invalid Old Password</Message>  <Exception>Old Password length should be between 6 and 14 characters. </Exception>  </Error> |
| **NP101** | Invalid New Password | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>NP101</ErrCode>  <Message>Invalid New Password</Message>  <Exception>New Password length should be between 6 and 14 characters. </Exception>  </Error> |
| **IG101** | Invalid Gender | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>IG101</ErrCode>  <Message>Invalid Gender</Message>  <Exception>Gender can be M or F only. </Exception>  </Error> |
| **IG102** | Invalid Gender | <?xml version="1.0" encoding="utf-8"?>  <Error>  <ErrCode>IG102</ErrCode>  <Message>Invalid Gender</Message>  <Exception>Gender length should be equal to 1 character. </Exception>  </Error> |