Integration of

Central Food grains Storage Portal [CFSP] with State storage portal

Web Services Integration Document

Food and Public Distribution System
National Informatics Centre HQ
CGO Complex, Lodhi Road, New Delhi

Document Control Record

Title	API Integration Document for Central Food grain Storage Portal (CFSP)			
Description	API Document explaining the web services for exchange of data between CFSP (Central Server) and respective State storage application / portal.			
Maintained by	Food & Public Distribution System, National Informatics Centre			
Version	Submission Date	Change Record / Reason for change	Author(s)	Status
1.0	11-Apr-2022	Initial	1. Anita 2. AJK Jose	Initial
1.0	17-Jun-2022	Json Format change for Truck Stack inflow and Truck Stack Outflow	1.Anita 2. AJK Jose	

Reviewers:

Reviewer(s)	Version	Review Remarks
AJK Jose (ajk.jose@nic.in)	1.0	Released in April 2022

Distribution:

Version	Distribution List	Location
1.0	FCI and States / UTs	New Delhi & States

Contents

1. Introduction 1

2. Scope 2

2.1 APIs for Data Receiving& Sharing 2

2.1.1 Web Service for State storage application to push the profile of a given depot. 3

2.1.1.1 Authorization Setup 3

2.1.1.2 Input JSON Parameters 3

2.1.1.3 Request/Response JSON Parameters 6

2.1.2 Web Service for State storage application to push the details of Stack Wise Stock Position details. 7

2.1.2.1 Authorization Setup 7

2.1.2.2 Input JSON Parameter 7

2.1.2.3 Request/Response JSON Parameter 9

2.1.3 Web Service for State application to push the inward details of Truck and inflow into stack. 10

2.1.3.1 Authorization Setup 11

2.1.3.2 Input JSON Parameter 11

2.1.3.3 Request/Response JSON Parameter 12

2.1.4 Web Service for State application to push the outward details of Truck and outflow from stack. 14

2.1.4.1 Authorization Setup 14

2.1.4.2 Input JSON Parameter 14

2.1.4.3 Request/Response JSON Parameter 16

2.1.5 Web Service for State application to push the details of Stack Gain/Loss Summary. 17

2.1.5.1 Authorization Setup 18

2.1.5.2 Input JSON Parameter 18

2.1.5.3 Request/Response JSON Parameter 19

2.1.6 Web Service for State application to push the details of Infestation details. 20

2.1.6.1 Authorization Setup 21

2.1.6.2 Input JSON Parameter 21

2.1.6.3 Request/Response JSON Parameter 23

2.1.7 Web Service for State application to push the details of Treatment details. 24

2.1.7.1 Authorization Setup 24

2.1.7.2 Input JSON Parameter 24
2.1.7.3 Request/Response JSON Parameter 26
3. APIs for Master 29
4. API's URL and Token Generation 34
5. Sample URL's 35

1. Introduction

Food Corporation of India (FCI) and the State Agencies, on behalf of Government of India, extend price support for procurement of wheat, paddy and coarse grains. All the food grains conforming to the prescribed specifications are bought by the public procurement agencies (including State governments under DCP procurement) at the Minimum Support Price (MSP).

FCI has developed and implemented DOS application to cater to end-to-end automation of depot operations, involving various depot operations activities, viz. storage, preservation, receipt and issue of food grains etc. DOS application is already functional in its owned godowns and has also been extended to the hired godowns.

Further, as per the vision of the DFPD, an ecosystem of online storage management applications is being developed in all DCP States which will ensure implementation of certain standards related to storage management. FCI intends to establish a central portal for online storage management in all FCI (owned and hired) and State Government (owned and hired) across the country to ensure adherence to SOP compliance for storage godowns.

Accordingly, with an objective of streamlining and bringing uniformity to the management practices across the country, and analysis of the various storage Management Portals of the States was recently conducted by FCI. Efforts have been made to identify the minimum storage specifications (MSS) that must necessarily be captured by the State storage management portals, so as to ensure uniformity and interoperability among them. The MSS are mentioned as under:

- Capability to compute Storage capacity
- Depict Storage point-wise stock position
  - Crop-year-wise break up of stocks held and depiction of opening Balance (OB), Issue and Closing Balance (CB)
- Stack-wise, Truck-Wise Linkages
  - Stack-wise details of stock-position, Truck-wise information
- Quality Parameters
  - Infestation details, Treatment details

Govt. of India has desired that various storage management operations, quality parameters and movement details shall be captured using different State portals. The operations are to be standardized with respect to the minimum storage specifications.

Further, these State portals are to integrate with the central portal, in order to ensure that the storage operations are carried out with complete transparency, efficiency and provide consolidated information to ensure storage losses and deficiencies of food grains in the storage units. This will facilitate enhanced decision-making and policy making.

2. Scope

These web services provide a common data format to the States using which State storage application will push the data as per the parameter defined in the common data format.

2.1 APIs for Data Receiving& Sharing

There are 6 Transactional Web Services which facilitates State storage portals to send details related to Minimum Storage specifications, to the central portal.

Sl. No.	Service Name	Service Purpose	Service Method
1.	Depot Profile	Profile of the depot along with the storage capacity of the depot.	POST
2.	Stack Profile	Profile of the stack.	POST
3.	Inflow	Details of receipt.	POST
4.	Outflow	Details of issue / dispatch.	POST
5.	Stack – Gain – Loss	Details of the loss or gain of the stack at the time of stack getting killed.	POST
6.	Infested Stacks	Details of the infestation of the respective stack.	POST
7.	Treated Stacks	Details of the treatment carried out on the respective stacks.	POST

There are 6 Web Services related to Masters provided by the central portal:

Sl. No.	Service Name	Service Purpose	Service Method
1.	Commodity	Commodity Master	GET
2.			
3.	Bag Type	Bag Type	GET
4.	State Master	State Master	GET
5.	District Master	District Master	GET
6.	Tehsil Master	Tehsil Master	GET
7.	Village Master	Village Master	GET

For the sake of uniformity, relevant Masters like Commodity, State code, LGD District Code, LGD Tehsil Master, LGD Village master, etc., shall be defined by CFSP Central Server. This master details and the relevant ID's/Code can be accessed from the website of CFSP using the GET method based services. The website https://lgdirectory.gov.in may be used to download the latest LGD codes of State, District, Sub-district, village etc.,

2.1.1 Web Service for State storage application to push the profile of a given depot.

DEPOT PROFILE

This CFSP API will consume the details of the depot profile pushed by State storage application to central CFSP Server.

Frequency: **As and when a depot is getting hired or dehired or created in the storage application of the State.**

2.1.1.1 Authorization Setup

Header	Value
Client Id	To be provided by CFSP team
Client Password	To be provided by CFSP team

2.1.1.2 Input JSON Parameters

S. No.	Parameter	Data Type	Description/ Purpose	Required?
1.	lgd_state_code	integer	LGD State Code of depot / godown	Y
2.	lgd_district_code	integer	LGD District code of depot / godown	Y
3.	depot_status	integer	Whether depot is active or inactive? 1. Active 2. Inactive	Y
4.	depot_code	character varying(99)	Depot Code; as maintained in the State storage application.	Y
5.	depot_name	character varying(99)	Depot Name	Y

S. No.	Parameter	Data Type	Description/ Purpose	Required?
6.	declared_capacity (in MTs)	Integer	Declared storage capacity of the depot / godown in MTs.	Y
7.	ownership_group_type (Depot Type)	Integer	1. FCI 2. State Government 3. CWC 4. SWC 5. PEG 99. Others	Y
8.	owner_name	character varying(99)	Name of agency or the individual or group owning the depot / godown.	Y
9.	ownership_type	Integer	1. Owned 2. Hired	Y
10.	hired_by	Integer	Applicable only if Ownership type is Hired. The entity who has hired the depot / godown. 1. FCI 2. State Government 3. CWC 4. SWC 5. PEG 99. Others	Y/O
11.	hired_from	Integer	Applicable only if Ownership type is Hired. The entity from whom the depot / godown has been hired. 1. FCI 2. State Government 3. CWC 4. SWC 5. PEG 99. Others	Y/O
12.	hired_from_date (DD-MM-YYYY)	character varying(10)	Date of hiring of the depot / godown. Mandatory if Ownership type is Hired. If the depot is owned, then the date from which the depot is operational may be provided. *The hired from date should be less than or equal to system date.*	Y/O
13.	hired_upto_date (DD-MM-YYYY)	character varying(10)	Date up to which the depot /godown has been hired. Mandatory if Ownership type is Hired. *The hired upto date should be greater than or equal to system date.*	Y/O

S. No.	Parameter	Data Type	Description/ Purpose	Required?
14.	depot_address	character varying(99)	Address of the depot / godown.	Y
15.	latitude	character varying(20)	GIS	Y/O
16.	longitude	character varying(20)	GIS	Y/O
17.	pin_code	Integer	Pin Code of the depot	Y
18.	electronic_weighbridge_count	Integer	Count of electronic weigh bridges. 0. Not Available 1. if one WB available 2. if two WBs Available 3. if three … and so on	Y
19.	electronic_weighbridge_make	character varying(99)	OEM/ Make of the electronic weighbridge. If more than one, the makes may be provided as comma separated values.	Y/O
20.	rail_sided	integer	Whether the depot / godown is rail sided or not? 1. Yes 2. No	Y
21.	rail_siding_count	Integer	Count of rail sidings. 0. Not Available 1. if one rail siding available 2. if two rail sidings available 3. if three … and so on	Y
22.	lgd_subdistrict_code	integer	LGD Sub-district Code	O
23.	lgd_block_code	integer	LGD Block Code	O
24.	lgd_village_code	integer	LGD Village code	O
25.	new_update	integer	1. New / First time sending of the data 2. Update / any parameter is updated	Y
26.	data_date (DD-MM-YYYY)	character varying(10)	The date to which the data pertains to.	Y
27.	transaction_reference_id	character varying(12)	Transaction reference number to be tagged with every record pushed by State application (2 digit LGD state code (If LGD state code is single digit, prefix with ‘0’ + 10 digit running number / random number). Should not be duplicate.	Y

2.1.1.3 Request/Response JSON Parameters

Web Service Type: Transaction Service -1
Web Service Name: DEPOT PROFILE
Method: POST
URL:<domain>/cfsp /depot_profile/{client_id}
Note: <domain> may be kept as a variable, since domain name may change.

Request JSON:

```
[ {
"lgd_state_code":11,
"lgd_district_code":345,
"depot_status":1462,
"depot_code":"12345",
"depot_name":"Sohan",
"declared_capacity":125,
"ownership_group_type":1,
"owner_name":"Suresh",
"ownership_type":1,
"hired_by":3,
"hired_from":4,
"hired_from_date":"09-03-2022",
"hired_upto_date":"31-03-2022",
"depot_address":"Vasdhra enclave Delhi",
"latitude":"356",
"longitude":"123,
"pin_code":110035,
"electronic_weighbridge_count":2,
"electronic_weighbridge_make":"56,89,336,452",
"rail_sided":1,
"rail_siding_count":1,
"lgd_subdistrict_code":896,
"lgd_block_code":253,
"lgd_village_code":745,
"new_update":1,
"data_date":"14-03-2022",
"transaction_reference_id":“111234567891”
} ]
```

Response JSON:

```
{
"data": {
"client_id": "1",
"transaction_id": "111234567891",
"service_code": "DEPOTPROFILE",
"transaction_status": "SUCCESS",
"transaction_remarks": "Depot Profile Uploaded",
"data_landing_timestamp": "2022-03-31 09:47:37.895",
"acknowledgement_no": "310320221232125"
},
"message": "Depot Profile saved successfully.",
"status": 200
}
```

2.1.2 Web Service for State storage application to push the details of Stack Wise Stock Position details.

STACK PROFILE

This CFSP API will consume the details of the stacks in a given depot. The stack details shall be pushed by State storage application to central CFSP Server.

Frequency: **Daily**

2.1.2.1 Authorization Setup

Header	Value
Client Id	Same client id which is specified by CFSP
Client Password	Same client password which is specified by CFSP

2.1.2.2 Input JSON Parameter

S. No.	Parameter	Data Type	Description/Purpose	Required?
1.	lgd_state_code	integer	LGD State Code of depot / godown	Y
2.	lgd_district_code	integer	LGD District code of depot / godown	Y
3.	depot_status	integer	Whether depot is active or inactive? 1. Active 2. Inactive	Y
4.	depot_code	character varying(99)	Depot Code; as maintained in the State storage application.	Y
5.	depot_name	character varying(99)	Depot name.	Y
6.	stack_number	character varying(99)	Stack Number or Stack ID	Y
7.	stack_capacity	double	Declared capacity of the stack	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
	(in Qtl.Kgs.)			
8.	stack_created_on (DD-MM-YYYY)	character varying(10)	Date on which the stack was created.	Y
9.	stack_formed_on (DD-MM-YYYY)	character varying(10)	Date on which the stack was fully formed.	Y
10.	stack_formation_quantity (in Qtl.Kgs.)	double	Quantity of stocks stored in the fully formed stack as on stack formation date.	Y
11.	storage_type	integer	The type of storage within which the stack exists. 1. Covered 2. CAP(Scientific) 3. CAP(Non-Scientific)	Y
12.	commodity	Integer	The commodity which is stored in the stack. The code of commodities can be obtained through the master service of commodity. The commodity codes are maintained at the central server. Wheat Rice-Grade A(RRA) Rice-Common(RRC) Rice-Parboiled(PBR) Fortified Rice Kernel(FRK) Paddy	Y
13.	marketing_season	Integer	The season to which the stacked commodity belongs to. 1. KMS (Kharif marketing season) 2. RMS (Rabi marketing season)	Y
14.	crop_year	character varying(9)	2021-2022 (Financial Year)	Y
15.	crop_type_id	integer	1. Kharif crop 2. Rabi crop	Y
16.	bag_type	integer	The type of bag within which the commodity is stored in the stack. Master available. 1. SBT (580) 2. SBT 3. HDPE	Y
17.	categorization_type	integer	Category of the commodity / stock stored in the stack	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
			1. Issuable – Category A 2. Issuable – Category B 3. Issuable – Category C 4. Issuable – Category D 8. Non-Issuable – Feed 1 9. Non-Issuable – Feed 2 10. Non-Issuable – Feed 3 14. Industrial Use 15. Manure 16. Fit for dumping	
18.	opening_balance (in Qtl.Kgs.)	double	Quantity in the stack at the opening of the day i.e., data_date.	Y
19.	closing_balance (in Qtl.Kgs.)	double	Quantity in the stack at the closing of the day i.e., data_date.	Y
20.	transaction_reference_id	character varying(12)	Transaction reference number to be tagged with every record pushed by State application (2 digit LGD state code (If LGD state code is single digit, prefix with '0' + 10 digit running number / random number). Should not be a duplicate.	Y
21.	data_date (DD-MM-YYYY)	character varying(10)	The date to which the data pertains to.	Y
22.	bag_count	Integer	The number of bags deposited into the stack.	Y

2.1.2.3 Request/Response JSON Parameter

Web Service Type: Transaction Service - 2
Web Service Name: STACK PROFILE
Method: POST
URL:<domain>/cfsp /stack_profile/{client_id}
Note: <domain> may be kept as a variable, since domain name may change.

Request JSON:

```
[ {
"lgd_state_code":11,
"lgd_district_code":123,
"depot_status":1,
```

"depot_code":" 1235",
"depot_name":" Sohan",
"stack_number":" 58796",
"stack_capacity":526.52,
"stack_created_on":"05-03-2022",
"stack_formed_on":"06-03-2022",
 "stack_formation_quantity":526.52,
"storage_type":2,
"commodity":3,
"marketing_season":"2022-2022",
"crop_year":"2022 ",
"crop_type_id":2,
"bag_type":3,
"categorization_type":1,
"opening_balance":1234.56,
"closing_balance":222.22,
"transaction_reference_id":“111234567891”,
"data_date":"14-03-2022",
"bag_count ":1
}]

<u>Response JSON:</u>

{
"data": {
"client_id": "1",
"transaction_id": "111234567891",
"service_code": "STACKPROFILE",
"transaction_status": "SUCCESS",
"transaction_remarks": "Data Uploaded",
"data_landing_timestamp": "2022-03-31 09:47:37.895",
"acknowledgement_no": "310320221232125"
},
"message": "Stack Profile saved successfully.",
"status": 200
}

2.1.3 Web Service for State application to push the inward details of Truck and inflow into stack.

INFLOW

This CFSP API will consume the details of Truck-wise information and the inflow into the stacks. The data shall be pushed by State storage application to central CFSP Server.

Frequency: For every day / date of inflow into the depot

2.1.3.1 Authorization Setup

Header	Value
Client Id	Same client id which is specified by CFSP
Client Password	Same client password which is specified by CFSP

2.1.3.2 Input JSON Parameter

S. No.	Parameter	Data Type	Description/Purpose	Required?
HEADER				
1.	lgd_state_code	integer	LGD State Code of depot / godown in which inflow has happened.	Y
2.	lgd_district_code	integer	LGD District code of depot / godown in which inflow has happened.	Y
3.	depot_status	integer	Whether depot is active or inactive? 1. Active 2. Inactive	Y
4.	depot_code	character varying(99)	Depot Code; as maintained in the State storage application.	Y
5.	depot_name	character varying(99)	Depot name.	Y
6.	transaction_reference_id	character varying(12)	Transaction reference number to be tagged with every record pushed by State application (2 digit LGD state code (If LGD state code is single digit, prefix with '0' + 10 digit running number / random number). Should not be a duplicate.	Y
7.	data_date (DD-MM-YYYY)	character varying(10)	The date to which the data pertains to.	Y
CHILD DATA				
8.	truck_number	character varying(20)	Truck number.	Y
9.	truck_chit	character varying(20)	Truck chit reference or similar reference, generated by the State application to be provided.	Y/O
10.	net_quantity (in Qtl.Kgs.)	Double	Net weight of the commodity received at the weigh bridge i.e. gross weight of the truck minus (tare weight of the truck + tare weight of the bags)	Y
11.	inflow_source	Integer	Source from which, the stock is received in the stack/depot.	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
			1. Procurement centre 2. Miller 3. Inter-depot transfer 4. Other	
12.	stack_number	character varying(99)	Stack Number or Stack ID in which quantity is off-loaded.	Y
13.	inflow_quantity (in Qtl.Kgs.)	Double	Net Quantity of the commodity deposited in the stack during every receipt into the respective stack / stack under consideration.	Y
14.	commodity	Integer	The commodity which is stored in the stack. The code of commodities can be obtained through the master service of commodity. The commodity codes are maintained at the central server. Wheat Rice-Raw-Grade A (RRA) Rice-Raw-Common (RRC) Rice-Parboiled (PBR) Fortified Rice (FR) Paddy	Y
15.	marketing_season	Integer	The season to which the stacked commodity belongs to. 1. KMS (Kharif marketing season) 2. RMS (Rabi marketing season)	Y
16.	crop_year	character varying(9)	2021-2022 (Financial Year)	Y
17.	crop_type_id	integer	1. Kharif crop 2. Rabi crop	Y
18.	bag_type	integer	The type of bag within which the commodity is stored in the stack. Master available. 1. SBT (580) 2. SBT 3. HDPE	Y
19.	bag_count	Integer	The number of bags deposited into the stack.	Y

2.1.3.3 Request/Response JSON Parameter

<u>Web Service Type:</u> Transaction Service - 3

Web Service Name: INFLOW
Method: POST
URL:<domain>/cfsp /inflow/{client_id}
Note: <domain> may be kept as a variable, since domain name may change.

Request JSON:

```
[
[
 {
   "lgd_state_code": 1,
   "lgd_district_code": 1,
   "depot_status": 1,
   "depot_code": "12345",
   "depot_name": "ABC",
   "data_date": "30-06-2021",
   "truckstackinflowdetails": [
    {
     "truck_number": "ABC",
     "truck_chit": "ABC",
     "net_quantity": 1,
     "inflow_source": 1,
     "stack_number": "ABC",
     "inflow_quantity": 1,
     "commodity": 1,
     "marketing_season": 1,
     "crop_year": "2021-2022",
     "crop_type_id": 1,
     "bag_type": 1,
     "bag_count": 1
    }
   ],
   "transaction_reference_id": "013212554875"
 }
]
```

Response JSON:

```
{
"data": {
"client_id": "1",
"transaction_id": "111234567",
"service_code": "TRUCK-STACK-INFLOW",
"transaction_status": "SUCCESS",
"transaction_remarks": "Truck Stack inflow uploaded",
"data_landing_timestamp": "2022-02-08 09:47:37.895",
"acknowledgement_no": "080220221232125"
},
"message": "Truck Stack inflow data saved successfully.",
"status": 200
}
```

2.1.4 Web Service for State application to push the outward details of Truck and outflow from stack.

OUTFLOW

This CFSP API will consume the details of Truck-wise information and the outflow from the stacks. The data shall be pushed by State storage application to central CFSP Server.

Frequency: **For every day / date of outflow from the depot**

2.1.4.1 Authorization Setup

Header	Value
Client Id	Same client id which is specified by CFSP
Client Password	Same client password which is specified by CFSP

2.1.4.2 Input JSON Parameter

S. No.	Parameter	Data Type	Description/Purpose	Required?
HEADER				
1.	lgd_state_code	integer	LGD State Code of depot / godown in which outflow has happened.	Y
2.	lgd_district_code	integer	LGD District code of depot / godown in which outflow has happened.	Y
3.	depot_status	integer	Whether depot is active or inactive? 1. Active 2. Inactive	Y
4.	depot_code	character varying(99)	Depot Code; as maintained in the State storage application.	Y
5.	depot_name	character varying(99)	Depot name.	Y
6.	transaction_reference_id	character varying(12)	Transaction reference number to be tagged with every record pushed by State application (2 digit LGD state code (If LGD	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
			state code is single digit, prefix with '0' + 10 digit running number / random number). Should not be a duplicate.	
7.	data_date (DD-MM-YYYY)	character varying(10)	The date to which the data pertains to.	Y
CHILD DATA				
8.	truck_number	character varying(20)	Truck Number.	Y
9.	truck_chit	Character varying(20)	Truck chit reference or similar reference, generated by the State application to be provided.	Y
10.	net_quantity (in Qtl.Kgs.)	double	Net weight of the commodity issued/dispatched at the weigh bridge i.e. gross weight of the truck minus (tare weight of the truck + tare weight of the bags)	Y
11.	outflow_destination	Integer	Destination to which, the stock is issued or dispatched. 1. Fair Price Shop 2. Miller 3. Inter-depot transfer 4. Other	Y/O
12.	stack_number	character varying(99)	Stack Number or Stack ID from which the stock is withdrawn.	Y
13.	outflow_quantity (in Qtl.Kgs.)	double	Net quantity of the commodity withdrawn from the respective stack / stack in consideration, during every issue/dispatch.	Y
14.	outflow_scheme	Integer	The scheme against which the outflow has happened. 1. Central Scheme 2. State Scheme 3. Other	Y
15.	commodity	Integer	The commodity which is withdrawn from the stack. The code of commodities can be obtained through the master service of commodity. The commodity codes are maintained at the central server. Wheat Rice-Raw-Grade A (RRA) Rice-Raw-Common (RRC)	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
			Rice-Parboiled (PBR) Fortified Rice (FR) Paddy	
16.	marketing_season	Integer	The season to which the commodity issued / dispatched belongs to. 1. KMS (Kharif marketing season) 2. RMS (Rabi marketing season)	Y
17.	crop_year	character varying(9)	2021-2022 (Financial Year)	Y
18.	crop_type_id	integer	1. Kharif crop 2. Rabi crop	Y
19.	bag_type	integer	The type of bag within which the commodity is issued/dispatched. Master available. 1. SBT (580) 2. SBT 3. HDPE	Y
20.	bag_count	Integer	The number of bags withdrawn from the stack.	

2.1.4.3 Request/Response JSON Parameter

Web Service Type: Transaction Service - 4
Web Service Name: OUTFLOW
Method: POST
URL:<domain>/cfsp /outflow/{client_id}
Note: <domain> may be kept as a variable, since domain name may change.

Request JSON:

```
[
 {
   "lgd_state_code": 1,
   "lgd_district_code": 1,
   "depot_status": 1,
   "depot_code": "12345",
   "depot_name": "ABC",
   "data_date": "30-06-2021",
   "truckstackoutflowdetails": [
    {
      "truck_number": "ABC",
      "truck_chit": "ABC",
      "net_quantity": 1,
      "outflow_destination": 1,
      "stack_number": "ABC",
      "outflow_quantity": 1,
      "outflow_scheme": 1,
      "commodity": 1,
      "marketing_season": 1,
      "crop_year": "2021-2022",
      "crop_type_id": 1,
      "bag_type": 1,
      "bag_count": 1
    }
   ],
   "transaction_reference_id": "013212554875"
 }
]
```

<u>Response JSON:</u>

```
{
"data": {
"client_id": "1",
"transaction_id": "111234567",
"service_code": "TRUCK-STACK-OUTFLOW",
"transaction_status": "SUCCESS",
"transaction_remarks": "Truck Stack outflow uploaded",
"data_landing_timestamp": "2022-02-08 09:47:37.895",
"acknowledgement_no": "080220221232125"
},
"message": "Truck Stack inflow data saved successfully.",
"status": 200
}
```

2.1.5 Web Service for State application to push the details of Stack Gain/Loss Summary.

STACK - GAIN - LOSS

This CFSP API pertains only to those stacks which get killed consequent to issues / dispatches. This API will consume the details of Gain Loss details pushed by State storage application to central CFSP Server.

Frequency: **As and when the stacks get killed**

2.1.5.1 Authorization Setup

Header	Value
Client Id	Same client id which is specified by CFSP
Client Password	Same client password which is specified by CFSP

2.1.5.2 Input JSON Parameter

S. No.	Parameter	Data Type	Description/Purpose	Required?
1.	lgd_state_code	integer	LGD State Code of depot / godown	Y
2.	lgd_district_code	integer	LGD District code of depot / godown	Y
3.	depot_status	integer	Whether depot is active or inactive? 1. Active 2. Inactive	Y
4.	depot_code	character varying(99)	Depot Code; as maintained in the State storage application.	Y
5.	depot_name	character varying(99)	Depot name.	Y
6.	stack_number	character varying(99)	Stack Number or Stack ID of the stack which got killed.	Y
7.	stack_created_on (DD-MM-YYYY)	character varying(10)	Date on which the stack was created.	Y
8.	stack_formed_on (DD-MM-YYYY)	character varying(10)	Date on which the stack was fully formed.	Y
9.	stack_formation_quantity (in Qtl.Kgs.)	double	Quantity of stocks stored in the fully formed stack as on stack formation date.	Y
10.	outflow_quantity (in Qtl.Kgs.)	double	Quantity of stocks issued / dispatched from the stack i.e. the cumulative quantity since the formation of the stack, which got killed.	Y
11.	commodity	Integer	The commodity which was stored in the stack which got killed. The code of commodities can be obtained through the master service of commodity. The commodity codes are maintained at the central server. Wheat Rice-Raw-Grade A (RRA) Rice-Raw-Common (RRC) Rice-Parboiled (PBR) Fortified Rice (FR) Paddy	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
12.	marketing_season	Integer	The season to which the commodity which was stored in the stack which got killed. 1. KMS (Kharif marketing season) 2. RMS (Rabi marketing season)	Y
13.	crop_year	character varying(9)	2021-2022 (Financial Year)	Y
14.	crop_type_id	integer	1. Kharif crop 2. Rabi crop	Y
15.	bag_type	integer	The type of bag which was stored in the stack which got killed. Master available. 1. SBT (580) 2. SBT 3. HDPE	Y
16.	bag_count	Integer	Count of bags which was issued / dispatched from the stack which got killed.	Y
17.	stack_killed_on (DD-MM-YYYY)	character varying(10)	Date on which the stack got killed (exhausted)	Y
18.	loss_quantity (in Qtl.Kgs.)	double	Quantity of storage loss in the stack.	Y/O
19.	gain_quantity (in Qtl.Kgs.)	double	Quantity of storage gain in the stack.	Y/O
20.	transaction_reference_id	character varying(12)	Transaction reference number to be tagged with every record pushed by State application (2 digit LGD state code (If LGD state code is single digit, prefix with '0' + 10 digit running number / random number). Should not be a duplicate.	Y
21.	data_date (DD-MM-YYYY)	character varying(10)	The date to which the data pertains to.	Y

2.1.5.3 Request/Response JSON Parameter

Web Service Type: Transaction Service - 5
Web Service Name: STACK GAIN LOSS
Method: POST
URL:<domain>/cfsp /stack_gain_loss/{client_id}
Note: <domain> may be kept as a variable, since domain name may change.

Request JSON:

```
[ {
```

"lgd_state_code":11,
"lgd_district_code":123,
"depot_status":1,
"depot_code":"123",
"depot_name":" Sohan",
"stack_number":"12563",
"stack_created_on":"03-03-2022",
"stack_formed_on":"04-03-2022",
"stack_formation_quantity":526.523,
"outflow_quantity":452.56,
"commodity":2,
"marketing_season":1,
"crop_year":"2022-2023",
"crop_type_id":2,
"bag_type":2,
"bag_count":35,
"stack_killed_on":"14-03-2022",
"loss_quantity":56.255,
"gain_quantity":859.57,
"transaction_reference_id":"111234567891",
"data_date":"14-03-2022"
}]

<u>Response JSON:</u>

{
"data": {
"client_id": "1",
"transaction_id": "111234567891",
"service_code": "GAIN-LOSS",
"transaction_status": "SUCCESS",
"transaction_remarks": "Gain Loss data uploaded",
"data_landing_timestamp": "2022-02-08 09:47:37.895",
"acknowledgement_no": "080220221232125"
},
"message": "Gain Loss data saved successfully.",
"status": 200
}

2.1.6 Web Service for State application to push the details of Infestation details.

Infestation

This CFSP API will consume the details of Infestation related data pushed by State storage application to central CFSP Server.

Frequency: **As and when the stacks are identified as infested**

This CFSP API will consume the details of Infestation related data pushed by State storage application to central CFSP Server.

2.1.6.1 Authorization Setup

Header	Value
Client Id	Same client id which is specified by CFSP
Client Password	Same client password which is specified by CFSP

2.1.6.2 Input JSON Parameter

S. No.	Parameter	Data Type	Description/Purpose	Required?
1.	lgd_state_code	integer	LGD State Code of depot / godown	Y
2.	lgd_district_code	integer	LGD District code of depot / godown	Y
3.	depot_status	integer	Whether depot is active or inactive? 1. Active 2. Inactive	Y
4.	depot_code	character varying(99)	Depot Code; as maintained in the State storage application.	Y
5.	depot_name	character varying(99)	Depot name.	Y
6.	stack_number	character varying(99)	Stack Number or Stack ID of the stack which was treated.	Y
7.	stack_created_on (DD-MM-YYYY)	character varying(10)	Date on which the stack was created.	Y
8.	stack_formed_on (DD-MM-YYYY)	character varying(10)	Date on which the stack was fully formed.	Y
9.	stack_formation_quantity (in Qtl.Kgs.)	double	Quantity of stocks stored in the fully formed stack as on date of formation of stack.	Y
10.	quantity (in Qtl.Kgs.)	double	Quantity of stocks stored in the stack during the treatment.	Y
11.	commodity	Integer	The commodity which was stored in the stack during the treatment. The code of commodities can be obtained through the master service of commodity. The commodity codes are maintained at the central server.	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
			Wheat Rice-Raw-Grade A (RRA) Rice-Raw-Common (RRC) Rice-Parboiled (PBR) Fortified Rice (FR) Paddy	
12.	marketing_season	Integer	The season to which the commodity which was stored in the stack. 1. KMS (Kharif marketing season) 2. RMS (Rabi marketing season)	Y
13.	crop_year	character varying(9)	2021-2022 (Financial Year)	Y
14.	crop_type_id	integer	1. Kharif crop 2. Rabi crop	Y
15.	bag_type	integer	The type of bag which was stored in the stack during treatment. Master available. 1. SBT (580) 2. SBT 3. HDPE	Y
16.	bag_count	Integer	Count of bags stored in the stack at the time of identification of infestation.	Y
17.	infestation_type	Integer	Infestation Type 0 - Clear 1 - Few 2 -Heavy	Y
18.	categorization_type	integer	Category of stocks 1. Issuable – Category A 2. Issuable – Category B 3. Issuable – Category C 4. Issuable – Category D 8. Non-Issuable – Feed 1 9. Non-Issuable – Feed 2 10. Non-Issuable – Feed 3 14. Industrial Use 15. Manure 16. Fit for dumping	Y
19.	transaction_reference_id	character varying(12)	Transaction reference number to be tagged with every record pushed by State application (2 digit LGD state code (If LGD	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
			state code is single digit, prefix with '0' + 10 digit running number / random number). Should not be a duplicate.	
20.	data_date (DD-MM-YYYY)	character varying(10)	The date to which the data pertains to.	Y

2.1.6.3 Request/Response JSON Parameter

Web Service Type: Transaction Service - 6
Web Service Name: INFESTATION
Method: POST
URL:<domain>/cfsp /infestation/{client_id}
Note: <domain> may be kept as a variable, since domain name may change.

Request JSON:

```
[ {
"lgd_state_code":11,
"lgd_district_code":123,
"depot_status":1,
"depot_code":"123",
"depot_name":" Sohan",
"stack_number":"12563",
"stack_created_on":"03-03-2022",
"stack_formed_on":"04-03-2022",
"stack_formation_quantity":526.523,
"quantity":326.523,
"commodity":2,
"marketing_season":1,
"crop_year":"2022-2023",
"crop_type_id":2,
"bag_type":2,
"bag_count":35,
"infestation_type":2,
"categorization_type":8,
"transaction_reference_id":“111234567891”,
"data_date":"14-03-2022"
} ]
```

Response JSON:

```
{
"data": {
"client_id": "1",
"transaction_id": "111234567891",
"service_code": "INFESTATION ",
"transaction_status": "SUCCESS",
"transaction_remarks": "Infestation data uploaded",
"data_landing_timestamp": "2022-02-08 09:47:37.895",
"acknowledgement_no": "080220221232125"
},
"message": "Infestation Details data saved successfully.",
"status": 200
}
```

2.1.7 Web Service for State application to push the details of Treatment details.

Treatment

This CFSP API will consume the details of Treatment related data pushed by State storage application to central CFSP Server.

Frequency: **As and when the stacks are treated**

This CFSP API will consume the details of Treatment related data pushed by State storage application to central CFSP Server.

2.1.7.1 Authorization Setup

Header	**Value**
Client Id	Same client id which is specified by CFSP
Client Password	Same client password which is specified by CFSP

2.1.7.2 Input JSON Parameter

S. No.	**Parameter**	**Data Type**	**Description/Purpose**	**Required?**
1.	lgd_state_code	integer	LGD State Code of depot / godown	Y
2.	lgd_district_code	integer	LGD District code of depot / godown	Y
3.	depot_status	integer	Whether depot is active or inactive? 3. Active 4. Inactive	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
4.	depot_code	character varying(99)	Depot Code; as maintained in the State storage application.	Y
5.	depot_name	character varying(99)	Depot name.	Y
6.	stack_number	character varying(99)	Stack Number or Stack ID of the stack which was treated.	Y
7.	stack_created_on (DD-MM-YYYY)	character varying(10)	Date on which the stack was created.	Y
8.	stack_formed_on (DD-MM-YYYY)	character varying(10)	Date on which the stack was fully formed.	Y
9.	stack_formation_quantity (in Qtl.Kgs.)	double	Quantity of stocks stored in the fully formed stack as on date of formation of stack.	Y
10.	quantity (in Qtl.Kgs.)	double	Quantity of stocks stored in the stack during the treatment.	Y
11.	commodity	Integer	The commodity which was stored in the stack during the treatment. The code of commodities can be obtained through the master service of commodity. The commodity codes are maintained at the central server. Wheat Rice-Raw-Grade A (RRA) Rice-Raw-Common (RRC) Rice-Parboiled (PBR) Fortified Rice (FR) Paddy	Y
12.	marketing_season	Integer	The season to which the commodity which was stored in the stack. 3. KMS (Kharif marketing season) 4. RMS (Rabi marketing season)	Y
13.	crop_year	character varying(9)	2021-2022 (Financial Year)	Y
14.	crop_type_id	integer	3. Kharif crop 4. Rabi crop	Y
15.	bag_type	integer	The type of bag which was stored in the stack during treatment. Master available. 4. SBT (580) 5. SBT	Y

S. No.	Parameter	Data Type	Description/Purpose	Required?
			6. HDPE	
16.	bag_count	Integer	Count of bags stored in the stack at the time of identification of infestation.	Y
17.	treatment_type	Integer	Treatment Type 1 - Prophylactic 2 -Curative	Y
18.	date_of_treatment (DD-MM-YYYY)	character varying (10)	Date on which the treatment was done.	Y
19.	date_of_availability_for_issue (DD-MM-YYYY)	character varying (10)	Date from which stock can be issued / dispatched from the stack.	Y
20.	categorization_type	integer	Category of stocks 1. Issuable – Category A 2. Issuable – Category B 3. Issuable – Category C 4. Issuable – Category D 8. Non-Issuable – Feed 1 9. Non-Issuable – Feed 2 10. Non-Issuable – Feed 3 14. Industrial Use 15. Manure 16. Fit for dumping	Y
21.	transaction_reference_id	character varying(12)	Transaction reference number to be tagged with every record pushed by State application (2 digit LGD state code (If LGD state code is single digit, prefix with '0' + 10 digit running number / random number). Should not be a duplicate.	Y
22.	data_date (DD-MM-YYYY)	character varying(10)	The date to which the data pertains to.	Y

2.1.7.3 Request/Response JSON Parameter

Web Service Type: Transaction Service - 7
Web Service Name: TREATMENT
Method: POST
URL:<domain>/cfsp /treatment/{client_id}

Note: <domain> may be kept as a variable, since domain name may change.

Request JSON:

```
[ {
"lgd_state_code":11,
"lgd_district_code":123,
"depot_status":1,
"depot_code":"123",
"depot_name":" Sohan",
"stack_number":"12563",
"stack_created_on":"03-03-2022",
"stack_formed_on":"04-03-2022",
"stack_formation_quantity":526.523,
"quantity":326.523,
"commodity":2,
"marketing_season":1,
"crop_year":"2022-2023",
"crop_type_id":2,
"bag_type":2,
"bag_count":35,
"treatment_type":"1",
"date_of_treatment":"13-03-2022",
"date_of_availability_for_issue":"14-03-2022",
"categorization_type":8,
"transaction_reference_id":"111234567891",
"data_date":"14-03-2022"
} ]
```

Response JSON:

```
{
"data": {
"client_id": "1",
"transaction_id": "111234567891",
"service_code": "TREATMENT",
"transaction_status": "SUCCESS",
"transaction_remarks": "Treatment data uploaded",
"data_landing_timestamp": "2022-02-08 09:47:37.895",
"acknowledgement_no": "080220221232125"
},
"message": "Treatment Details data saved successfully.",
"status": 200
```

```
        }
```

3. APIs for Master

For the sake of uniformity, relevant Masters like State code, LGD District Code, depot type, LGD Village master, LGD Tehsil Master etc as per CFSP Central Server. This master details and the relevant ID's/Code can be access through URL http://cfsp.nic.in/cfsp/swagger_cfsp

Master 1: Commodity

S. No.	Parameter	Data Type	Description/Purpose	Required
1	commodity_id	Integer		Y
2	commodity_name	character varying(20)	Paddy Wheat Bajra Barley Jowar Ragi Maize Tur Dal Arhar Dal Rice-Raw Common Rice-Raw Grade A Rice Parboiled Common Rice Parboiled Grade A Coarse Grains	Y
3	active	Integer	Active or Inactive 0-Inactive 1-Active	Y

Web Service Type: Master Web Service 1
Web Service Name: Commodity
Method: GET
URL:<domain>/cfsp/commodity
Note: <domain> may be kept as a variable, since domain name may change.
Response JSON:

```
{
"commodity_id": "1",
" commodity_name": "Wheat"
},
{
"commodity_id": "2",
" commodity_name": "Rice"
}
```

Master 2: Bag Type

S. No.	Parameter	Data Type	Description/Purpose	Required
1	bag_type_id	Integer		Y
2	bag_type_name	character varying(20)	The type of bag within which the commodity is stored in the stack. 1. SBT (580) 2. SBT 3. HDPE	Y
3	tare_weight	double	Tare weight of the bag	Y
4	active	Integer	Active or Inactive 0-Inactive 1-Active	Y

Web Service Type: Master Web Service 2
Web Service Name: Bag Type
Method: GET
URL:<domain>/cfsp/bagtype
Note: <domain> may be kept as a variable, since domain name may change.

Response JSON:

```
{
" bag_type_id ": "1",
" bag_type_name ": "SBT(580)"
“tare_weight”:”0.500”
},
{
" bag_type_id ": "2",
" bag_type_name ": "SBT"
“tare_weight”:”0.525”
}
```

Master 3: LGD State Master

S. No.	Parameter	Data Type	Description/Purpose	Required
1	lgd_state_code	integer		Y
2	lgd_state_name_en	character varying(99)		Y

3	state_name_ll	character varying(99)		O
4	Active	integer	Active or Inactive 0-Inactive 1-Active	Y

Web Service Type: Master Web Service 3
Web Service Name: State Master
Method: GET
URL:<domain>/cfsp/states
Note: <domain> may be kept as a variable, since domain name may change.

Response JSON:

```
{
" lgd_state_code": "1",
" lgd_state_name_en": "Jammu & kashmir",
"state_name_ll": "Jammu & kashmir"
}
```

Master 4: LGD District Master

S. No.	Parameter	Data Type	Description/Purpose	Required
1	lgd_state_code	integer		Y
2	lgd_district_code	integer		Y
3	lgd_district_name_en	character varying(99)		Y
4	district_name_ll	character varying(99)		O
5	Active	integer	Active or Inactive 0-Inactive 1-Active	Y

Web Service Type: Master Web Service 4
Web Service Name: District Master
Method: GET
URL:<domain>/cfsp/district/{state_code}
Note: <domain> may be kept as a variable, since domain name may change.

Response JSON:

```
{
" lgd_state_code": "1",
" lgd_district_code": "656",
" lgd_district_name_en": "TEST",
" district_name_ll": "TEST"
}
```

Master 5: LGD Tehsil Master (sub district master/taluk/mandal)

S. No.	Parameter	Data Type	Description/Purpose	Required
1	lgd_state_code	integer		Y
2	lgd_district_code	integer		Y
3	lgd_tehsil_code	integer		Y
4	lgd_tehsil_name_en	character varying(99)		Y
5	tehsil_name_ll	character varying(99)		O
6	active	integer	Active or Inactive 0-Inactive 1-Active	Y

Web Service Type: Master Web Service 5
Web Service Name: Tehsil Master
Method: GET
URL:<domain>/cfsp/tehsil/{state_code}
Note: <domain> may be kept as a variable, since domain name may change.

Response JSON:

```
{
" lgd_state_code": "1",
" lgd_district_code": "123",
" lgd_tehsil_code": "123",
" lgd_tehsil_name_en": "Jammu",
" tehsil_name_ll": "Jammu”
}
```

Master 6: LGD Village Master

S. No.	Parameter	Data Type	Description/Purpose	Required
1	lgd_state_code	integer		Y
2	lgd_district_code	integer		Y
3	lgd_tehsil_code	integer		Y
4	lgd_village_code	integer		Y
5	lgd_village_name_en	character varying(99)		Y
6	village_name_ll	character varying(99)		O
7	Active	integer	Active or Inactive 0-Inactive 1-Active	Y

Web Service Type: Master Web Service 6
Web Service Name: Village Master
Method: GET
URL:<domain>/cfsp/village/{state_code}
Note: <domain> may be kept as a variable, since domain name may change.

Response JSON:

```
{
" lgd_state_code": "1",
" lgd_district_code": "123",
" lgd_tehsil_code": "123",
" lgd_village_code": "123",
" lgd_village_name_en": "Jammu”,
" village_name_ll": "Jammu”
}
```

4. API’s URL and Token Generation

NIC has developed a tool using Swagger, for testing the API's and token generation. For token generation the following path is to be followed by the respective State:

- Visit the URL http://cfsp.nic.in/cfsp/swagger_cfsp
- Reach the service --- Authentication Service -> /login
- Click on 'Try it out"
- In the Schema dialog box, State user shall have to provide the user name and password as provided to them by FCI / NIC.
- After entering the provided username and password, click on "Execute"
- The generated token will be displayed in the "Response Body" dialog box.
- Copy the token which is within double quotes.
- Scroll up to reach the top of the page.
- The Authorize button shall be visible with open lock, which indicates an unauthorized service user.
- Click on the "Authorize" button.
- In the dialog box which shall appear, paste the token, and click on 'Authorize'.
- Now, the user is authorized. User should click on 'Close' and not on "Logout"
- Now, the user is authorized to use any service both for posting and getting data.

Before Pushing data to CFSP server token has to be generated. For this, hit the login service for authentication and this service will generate the token in response.
The generated token has to be passed in the header under Authorization tag.

**Important: Please put Bearer before token.

For example:
Bearer
eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJhbml0YXNoYXJtYSIsImV4cCI6MTYzMjMwNDExOSwiaWF0IjoxNjMyMjg2MTE5fQ.LuMMmJ9dPHyIl8sPUOqoDu09ETCUI8T651gAebusGhV-U_s8F6JxIFGQOQxj09tf4Y00EwlbOWvdugzLTNb-Yw

5. Sample URL's

Sl. No.	Webservice Name	Service URL	Transactional /Master	Method
1.	DEPOT PROFILE	<domain>/cfsp /depot_profile/{client_id}	Transactional	POST
2.	STACK PROFILE	<domain>/cfsp /stack_profile/{client_id}	Transactional	POST
3.	INFLOW	<domain>/cfsp /inflow/{client_id}	Transactional	POST
4.	OUTFLOW	<domain>/cfsp /outflow/{client_id}	Transactional	POST
5.	STACK – GAIN – LOSS	<domain>/cfsp /stack_gain_loss/{client_id}	Transactional	POST
6.	INFESTATION	<domain>/cfsp /infestation/{client_id}	Transactional	POST
7.	TREATMENT	<domain>/cfsp /treatment/{client_id}	Transactional	POST
8.	COMMODITY	<domain>/cfsp/commodity	Master	GET
9.	BAG TYPE	<domain>/cfsp/bagtype	Master	GET
10.	STATE MASTER	<domain>/cfsp/states	Master	GET
11.	DISTRICT MASTER	<domain>/cfsp/district/{state_code}	Master	GET
12.	TEHSIL MASTER	<domain>/cfsp/tehsil/{state_code}	Master	GET
13.	VILLAGE MASTER	<domain>/cfsp/village/{state_code}	Master	GET

¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥¥