別紙２　実装ガイド

■ 本資料の位置づけ
本資料は那須塩原データ連携基盤 (以降「基盤」) を活用して、パーソナルデータを市民サービス間でデータ連携するために 市民サービス事業者にご対応頂きたい実装内容をまとめた実装ガイドです。 下記「必要な実装一覧」全ての項目に対応いただく必要がございます。 ※なお、2023年12月時点での情報となり、以降変更となる場合があります。

■ 必要な実装一覧		
項番	内容	記載シート
1	パーソナルデータ連携基盤IDとユーザIDの連携処理の実装	1.ID連携処理実装
2	データ連携基盤の同意管理画面に遷移するための、ログイン導線の設置	2.データ連携基盤のログイン画面への導線実装
3	データ蓄積処理の実装	3.データ蓄積処理の実装
4	データ取得処理の実装	4.データ取得処理の実装

■ 凡例	
黄色	市民サービス事業者側で値を設定する必要がある項目（例.下図のindexの値　など）
緑色	データ連携基盤と接続する際に、データ連携基盤事業者側で払い出したうえで、別途ご連絡するデータ項目

データモデル：xxxxx

xxxxリクエストボディ例	**設定値説明**

[
{
"app": {
"code": {
"index": "2_3_1",
"value": {
"_value": xxxxxxx,
	
	固定で 2_3_1 を指定
	
	xxxxxコード

・赤文字部分は後続処理で値を利用する項目

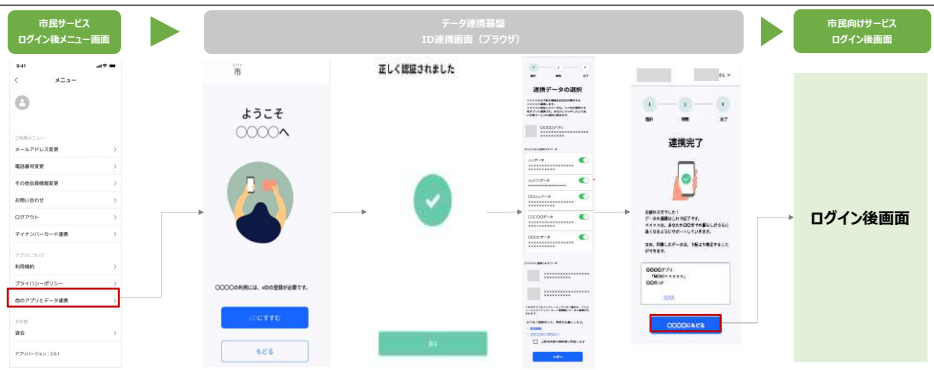
データモデル：xxxxx

xxxxレスポンスボディ例	**設定値説明**

[
{
"app": {
"code": {
"index": "2_3_1",
"value": {
"_value": xxxxxxx,
	
	カタログコード

1.ID連携処理の実装

利用者（市民等）の操作イメージ



処理シーケンス図
（①②③は次に記載の "実装手順" に記載の番号と対応）

実装手順

① リンクやボタン等、連携を開始する導線を準備する。
市民サービスの**「ログイン後の画面に」**、リンクやボタン等のユーザ導線を準備し、押下時の処理を実装してください。導線は、市民サービスの利用者が認知しやすい位置に設置することを推奨しています。

② URL発行APIを実行する（POST /identity-verificate/url/issue）
URL発行API（POST /identity-verificate/url/issueを実行しリダイレクトURLを取得する処理を実装してください。

【URL】
https://personal-app-xxxxxxxxxx.nasushiobara-connect.jp/identity-verificate/url/issue/?from_block={appBlockCode}&from_path=/pxr-block-proxy/&block=$pxr-root-block&path=%2Fidentity-verificate%2Furl%2Fissue
【リクエストパラメータ】
※ 下記の例を参照し、設定してください

▼URL発行APIクエリパラメータ例

クエリパラメータ名	クエリパラメータ値（例）	設定値説明
from_block	1000785	アプリケーションブロックコード
from_path	/pxr-block-proxy/	固定値
block	$pxr-root-block	固定値
path	%2Fidentity-verificate%2Furl%2Fissue	固定値

▼ URL発行APIリクエストボディ例

リクエストボディ例	説明
{	
"login": {	
"type": 2,	固定で 2 を指定
"loginId": "xxxxapp",	ログインID ※市民サービス毎に基盤から払い出されるものを指定。
"hpassword": "pua2xcl6x37nibfahqnmlh5rkhrqv8da25pomd2pwqxksd08odt1pyb	パスワード ※市民サービス毎に基盤から払い出されるものを指定。
},	
"data": {	
"userId": "xxxxxapp_user1",	市民サービス側で払い出されているユーザID
"urlType": 1	固定で 1 を指定
}	
}	

▼ URL発行APIレスポンス例

レスポンスボディ例	説明
{	
"url": "https://personal.nasushiobara-connect.jp/enter-code/?identifyCode=xxxxxxxx	リダイレクトURL + 本人性確認コード
}	

③ブラウザでURLを開く
上記で取得したレスポンスのURLに下記のクエリパラメータを追加付与しブラウザを開いてください。

クエリパラメータ名	クエリパラメータ値（例）	設定値説明
applicationCode	100033	アプリケーションのカタログコード
callbackUrl	https://xxxxxxxx	連携処理が完了後、市民サービスアプリに遷移するためのディープリンク

■ URL例
https://personal.nasushiobara-connect.jp/enter-code/?identifyCode=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx&applicationCode=100033&callbackUrl=xxxxxxxxxxxxxxxx

2.データ連携基盤のログイン画面への導線実装

<table>
<tr><td>利用者の操作イメージ</td><td> 
</td></tr>
<tr><td>処理シーケンス図
（①②は次に記載の "実装手順" に記載の番号と対応）</td><td> 
</td></tr>
<tr><td>実装手順</td><td><u>①ブラウザでURLを開く</u>
市民サービスの『ログイン後の画面に』、リンクやボタン等のユーザ導線を準備し、押下時の処理を実装してください。導線は、市民サービスの利用者が認知しやすい位置に設置することを推奨しています。

【URL】
https://personal.nasushiobara-connect.jp/?callbackUrl=https://xxxxxxxx

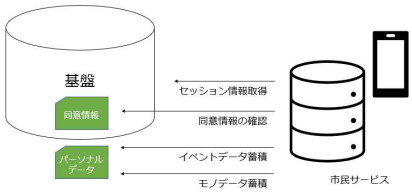
【クエリパラメータ】
※ 下記の例を参照し、設定してください

▼クエリパラメータ例
<table><tr><th>クエリパラメータ名</th><th>クエリパラメータ値（例）</th><th>設定値説明</th></tr><tr><td>callbackUrl</td><td>https://xxxxxxxx</td><td>戻るボタン押下時など、市民サービスアプリに遷移するためのディープリンク</td></tr></table></td></tr>
</table>

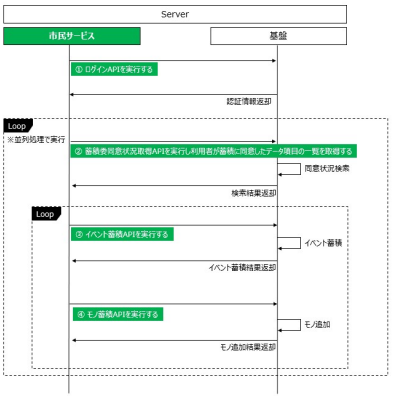
3.データ蓄積処理の実装

データ蓄積処理の実装にて、実現したいこと

利用者のデータ蓄積への同意状況を確認の上、基盤へのデータ蓄積を行います。
※イベント・モノの詳細については「補足_イベント・モノについて」シートを参照して下さい



処理シーケンス図
(①②③は次に記載の "実装手順" に記載の番号と対応)



実行タイミング

本実装はバッチ処理で定期的に実行される想定です。

実装手順

① ログインAPIを実行する (POST /operator/login)

ログインAPIを実行し、後続の処理で利用するセッション情報を取得する処理を実装してください。

【URL】
https://personal-app-xxxxxxxxx.nasushiobara-connect.jp/operator/operator/login
【リクエストパラメータ】
※ 下記の例を参照し、設定してください

▼ ログインAPIリクエストボディ例

リクエストボディ例	説明
{	
"type": 2,	固定で 2 を指定
"loginId": "xxxxapp",	ログインID ※市民サービス毎に基盤から払い出されるものを指定。
"hpassword": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"	パスワード ※市民サービス毎に基盤から払い出されるものを指定。
}	

▼ ログインAPIレスポンスボディ例

レスポンスボディ例	説明
{	
"sessionId": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",	セッションID
"operationId": 2,	
"type": 2,	
"loginId": "xxxxapp",	
"name": "xxxxname",	
"pxrId": null,	
"mobilePhone": null,	
"auth": null,	
"lastLoginAt": "2021-07-01T00:00:00.000+09:00",	
"passwordChangedFlg": true,	
"attributes": null,	
"roles": [	
{	
"_value": xxxxxxx,	
"_ver": 1	
}	
],	
"block": {	
"_value": xxxxxxx,	
"_ver": 1	
},	
"actor": {	
"_value": xxxxxxx,	
"_ver": 1	
}	
}	

②以降の処理については、処理時間短縮のため、並列処理での実装をお願いいたします。

② 蓄積同意状況取得APIを実行する。（POST /book-operate/user/list）

蓄積同意状況取得APIを実行し、利用者が蓄積に同意したデータ項目の一覧を取得する処理を実装して下さい。
また、ステータスが404で返却された場合は、ID連携していないユーザであるため③④の処理をスキップしてください

※ "ヘッダー・Cookie指定方法" シートを参照し、ログインAPIで得られた値をヘッダー・Cookieに指定してください。

【URL】

https://personal-app-xxxxxxxxxx.nasushiobara-connect.jp/pxr-block-proxy/pxr-block-proxy/?path=%2Fbook-operate%2Fuser%2Flist

【リクエストパラメータ】

※ 下記の例を参照し、設定してください

▼ 蓄積同意状況取得APIリクエストボディ例

リクエストボディ例	説明
{	
"userId": [	
"user001xxxxx"	検索対象となるID（利用者ID連携で連携した市民サービス側で管理しているユーザーIDを指定）
]	
}	

▼蓄積同意状況取得APIレスポンスボディ例（赤字で示されるカタログコードは後続のイベント蓄積API、モノ蓄積APIで指定します。）

レスポンスボディ例	説明
[	
{	
"status": 1,	連携状況 (0: 申請中, 1: 連携中)
"app": {	
"_value": xxxxxxx,	
"_ver": 1	
},	
"wf": null,	
"userId": "useTestUser000002",	
"establishAt": "2023-04-28T14:42:26.000+0900",	
"attribute": {},	
"store": [	同意をしている蓄積項目の配列
{	
"document": [],	
"event": [	
{	
"_value": xxxxxxx,	蓄積同意済みイベントのカタログコード
"_ver": 1	カタログのバージョン
}	
],	
"thing": [	蓄積同意済みとなっているモノカタログの配列
{	
"_value": xxxxxxx,	蓄積同意済みモノのカタログコード
"_ver": 1	カタログのバージョン
}	
]	
}	
],	
"userInformation": null	
}	
]	

③イベント蓄積APIを実行する。（POST /book-operate/event/{userId}）

②のレスポンスを確認し、連携状況が「1」かつstore > event、store > thing に項目が含まれているuserIdに対して、イベント蓄積APIを実行しイベントの蓄積を行って下さい。

蓄積対象のイベントは②のレスポンスのstore > event の配列に含まれる各データ項目です。

※ "ヘッダー・Cookie指定方法" シートを参照し、ログインAPIで得られた値をヘッダー・Cookieに指定してください。

【URL】

https://personal-app-xxxxxxxxxx.nasushiobara-connect.jp/pxr-block-proxy/pxr-block-proxy/?path=%2Fbook-operate%2Fevent%2F{userId}

【リクエストパラメータ】

※下記の例を参照し、設定してください

▼イベント蓄積APIパスパラメータ例

パスパラメータ名	パスパラメータ値（例）	設定値説明
userId	xxxxappUser000002	市民サービス側で管理しているユーザーID

▼イベント蓄積APIリクエストボディ例

例）ポータルサービスの利用者情報

リクエストボディ例	説明
{	
"app": {	
"code": {	
"index": "2_3_1",	固定で 2_3_1 を指定
"value": {	
"_value": xxxxxxxx,	アプリケーションプロバイダーのカタログコード
"_ver": 1	カタログバージョン
}	
},	
"app": {	
"index": "2_3_2",	固定で 2_3_2 を指定
"value": {	
"_value": xxxxxx,	アプリケーションのカタログコード
"_ver": 1	カタログバージョン
}	
}	
},	
"code": {	
"index": "3_1_2",	固定で 3_1_2 を指定
"value": {	
"_value": xxxxxxx,	蓄積同意状況取得APIで取得した蓄積同意済みイベントのカタログコード
"_ver": 1	蓄積同意状況取得APIで取得した蓄積同意済みイベントのカタログコードのバージョン
}	
},	
"end": {	
"index": "3_2_2",	固定で 3_2_2 を指定
"value": "2023-10-30T12:00:00.000+0900"	現在時刻を yyyy-MM-ddTHH:mm:ss.SSS+0900 の形式で指定
},	
"id": {	
"index": "3_1_1",	固定で 3_1_1 を指定
"value": null	nullを指定
},	
"sourceId": null,	nullを指定
"start": {	
"index": "3_2_1",	固定で 3_2_1 を指定
"value": "2023-10-30T12:59:59.000+0900"	現在時刻を yyyy-MM-ddTHH:mm:ss.SSS+0900 の形式で指定
},	
"userId": {	
"index": "3_6_1",	固定で 3_6_1 を指定
"value": "testUser1234"	利用者ID連携で連携した市民サービス側で管理しているユーザーIDを指定
}	
}	

▼イベント蓄積APIレスポンス例（赤字で示されるIdは後続のモノ蓄積APIで指定します）

レスポンスボディ例	説明
{	
"app": {	
"code": {	
"index": "2_3_1",	
"value": {	
"_value": xxxxxxx,	
"_ver": 1	
}	
},	
"app": {	
"index": "2_3_2",	
"value": {	
"_value": xxxxxxx,	
"_ver": 1	
}	
}	
},	
"code": {	
"index": "3_1_2",	
"value": {	
"_value": xxxxxxxx,	
"_ver": 1	
}	
},	
"id": {	
"index": "3_1_1",	
"value": "uuuuuuuu-uuuu-uuuu-uuuu-uuuuuuuuuuuu"	パーソナルデータ連携基盤側で生成されたId ※後続のモノ蓄積APIのパスパラメータに指定します。
},	
"sourceId": null,	
"userId": {	
"index": "3_6_1",	
"value": "testUser1234"	
},	
}	

④ モノ蓄積APIを実行する。：POST /book-operate/thing/{userId}/{eventId}

③でイベント蓄積を行ったuserIdに対して、モノ蓄積APIを実行しモノの蓄積を行って下さい。

蓄積対象のモノは②のレスポンスのstore > thing の配列に含まれる各データ項目です。　（データ項目はカタログコードで格納されています。）

※ "ヘッダー・Cookie指定方法" シートを参照し、ログインAPIで得られた値をヘッダー・Cookieに指定してください。

【URL】

https://personal-app-xxxxxxxxxx.nasushiobara-connect.jp/pxr-block-proxy/pxr-block-proxy/?path=%2Fbook-operate%2Fthing%2F{userId}%2F{eventId}

【リクエストラメータ】

※ 下記の例を参照し、設定してください

▼ モノ蓄積APIパスパラメータ例

パスパラメータ名	パスパラメータ値（例）	設定値説明
userId	xxxxappUser000002	市民サービス側で管理しているユーザーID
eventId	uuuuuuuu-uuuu-uuuu-uu	イベント蓄積APIレスポンスに含まれる id > valueを設定 （イベントID）

▼モノ蓄積APIリクエストボディ例

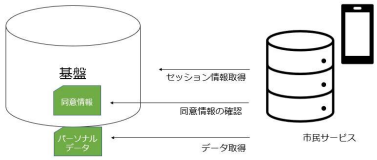
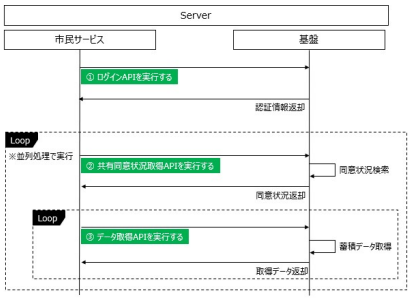
リクエストボディ例	説明
{	
"app": {	
"code": {	
"index": "2_3_1",	固定で 2_3_1 を指定
"value": {	
"_value": xxxxxxx,	アプリケーションプロバイダーのカタログコード ※パーソナルデータ連携基盤構築後に共有
"_ver": 1	カタログバージョン ※パーソナルデータ連携基盤構築後に共有
}	
},	
"app": {	
"index": "2_3_2",	固定で 2_3_2 を指定
"value": {	
"_value": xxxxxxx,	アプリケーションのカタログコード ※パーソナルデータ連携基盤構築後に共有
"_ver": 1	カタログバージョン ※パーソナルデータ連携基盤構築後に共有
}	
}	
},	
"code": {	
"index": "4_1_2",	固定で 4_1_2 を指定
"value": {	
"_value": xxxxxxx,	蓄積同意状況取得APIで取得した蓄積同意済みモノのカタログコード
"_ver": 1	蓄積同意状況取得APIで取得した蓄積同意済みモノのカタログコードのバージョン
}	
},	
"env": [	
{	
"index": "4_3_1",	固定で 4_3_1 を指定
"value": null	nullを指定
}	
],	
"id": {	
"index": "4_1_1",	固定で 4_1_1 を指定
"value": null	nullを指定
},	
"sourceId": null,	nullを指定
"data": [	
{	蓄積したいデータを指定
"FamilyName": "山田",	
"GivenName": "太郎",	
"FamilyNameHiragana": "やまだ",	
"GivenNameHiragana": "たろう",	
"Sex": {	
"Type": "性別",	
"TypeRelatedInformation": "男性"	
}	
}	
]	
}	

▼モノ蓄積APIレスポンス例

レスポンスボディ例	説明
{	
"app": {	
"code": {	
"index": "2_3_1",	
"value": {	
"_value": xxxxxxx,	
"_ver": 1	
}	
},	
"app": {	
"index": "2_3_2",	
"value": {	
"_value": xxxxxxx,	
"_ver": 1	
}	
}	
},	
"code": {	
"index": "4_1_2",	
"value": {	
"_value": xxxxxxx,	
"_ver": 1	
}	
},	
"env": [	
{	
"index": "4_3_1",	
"value": {}	
}	

```
 ],
 "id": {
  "index": "4_1_1",
  "value": "uuuuuuuuu-uuuu-uuuu-uuuu-uuuuuuuuuuuu"
 },
 "sourceId": null,
 "data":[
    {
      "FamilyName": "山田",
      "GivenName": "太郎",
      "FamilyNameHiragana": "やまだ",
      "GivenNameHiragana": "たろう",
      "Sex": {
        "Type": "性別",
        "TypeRelatedInformation": "男性"
      }
    }
]
}
```

4. データ取得処理の実装

データ取得処理の実装にて、実現したいこと	基盤からの蓄積データ取得を行います。 ※イベント・モノの詳細については「補足_イベント・モノについて」シートを参照して下さい 
処理シーケンス図 (①②③は次に記載の "実装手順" に記載の番号と対応)	
実行タイミング	本実装はバッチ処理で定期的に実行される想定です。

実装手順	**① ログインAPIを実行する (POST /operator/login)**
	ログインAPIを実行し、後続の処理で利用するセッション情報を取得する処理を実装してください。
	
	【URL】
	https://personal-app-xxxxxxxxxx.nasushiobara-connect.jp/operator/operator/login
	【リクエストパラメータ】
	※ 下記の例を参照し、設定してください
	
	▼ ログインAPIリクエストボディ例
	リクエストボディ例 / **説明**
	{
	"type": 2, — 固定で 2 を指定
	"loginId": "xxxxapp", — ログインID ※市民サービス事業者毎に基盤から払い出されるものを指定。
	"hpassword": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx" — パスワード ※市民サービス事業者毎に基盤から払い出されるものを指定。
	}
	
	▼ ログインAPIレスポンスボディ例
	レスポンスボディ例 / **説明**
	{
	"sessionId": "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx", — セッションID
	"operationId": 2,
	"type": 2,
	"loginId": "xxxxapp",
	"name": "xxxxname",
	"pxrId": null,
	"mobilePhone": null,
	"auth": null,
	"lastLoginAt": "2021-07-01T00:00:00.000+09:00",
	"passwordChangedFlg": true,
	"attributes": null,
	"roles": [
	{
	"_value": xxxxxxx,
	"_ver": 1
	}
	],
	"block": {
	"_value": xxxxxxx,
	"_ver": 1
	},
	"actor": {
	"_value": xxxxxxx,
	"_ver": 1
	}
	}
	
	②以降の処理については、処理時間短縮のため、並列処理での実装をお願いいたします。

②共有同意状況取得APIを実行する。(GET /book-manage/setting/share)

共有同意状況取得APIを実行し利用者が共有に同意したデータ項目の一覧を取得する処理を実装して下さい。
また、ステータスが400で返却された場合は、ID連携していないユーザであるため③の処理をスキップしてください

※ "ヘッダー・Cookie指定方法" シートを参照し、ログインAPIで得られた値をヘッダー・Cookieに指定してください。

【URL】

https://personal-app-xxxxxxxxxx.nasushiobara-connect.jp/pxr-block-proxy/pxr-block-proxy/?block=$pxr-root-block&path=%2Fbook-manage%2Fsetting%2Fshare%3Fid%3D{userId}%26app%3D{applicationCatalogCode}

【リクエストパラメータ】

※ 下記の例を参照し、設定してください

▼ 共有同意状況取得APIパスパラメータ

パスパラメータ名	パスパラメータ値（例）	設定値説明
id	xxxxappUser000002	市民サービス側で管理しているユーザーID
app	100081	アプリケーションのカタログコード

▼ 共有同意状況取得APIレスポンスボディ例

レスポンスボディ例	説明
[	
{	
"id": 15,	
"actor": {	
"_value": 1000798	
},	
"app": {	
"_value": 1000805	
},	
"share": [	
{	
"code": {	
"_value": "1000806",	共有定義のカタログコード
"_ver": "2"	共有定義のカタログバージョン
}	
}	
]	
},	
…	【以降、上記と同様のデータ構造で共有定義情報が配列で返却されます。】
]	

③ データ取得APIを実行する。（POST /book-operate/share）

データ取得APIを実行し、基盤に蓄積されたデータ項目を取得する処理を実装して下さい。

※ "ヘッダー・Cookie指定方法" シートを参照し、ログインAPIで得られた値をヘッダー・Cookieに指定してください。

【URL】

https://personal-app-xxxxxxxxxx.nasushiobara-connect.jp/pxr-block-proxy/pxr-block-proxy/?&path=%2Fbook-operate%2Fshare

【リクエストパラメータ】

※ 下記の例を参照し、設定してください

▼ データ取得APIリクエストボディ例

リクエストボディ例	説明
{	
"userId": "xxxxappUser000002",	市民サービス側で管理しているユーザーIDを指定してください
"updatedAt": {	データの取得対象期間を指定してください。（データの更新時刻によって取得されます。指定しない場合は全期間のデータが取得されます。）
"start": "2023-10-01T23:00:00.000+09:00",	検索対象時間（開始）※yyyy-MM-ddTHH:mm:ss.SSS+0900で指定
"end": "2023-10-01T23:59:59.000+09:00"	検索対象時間（終了）※yyyy-MM-ddTHH:mm:ss.SSS+0900で指定
},	
"event": [	
{	
"_value": xxxxxxx,	共有同意状況取得APIで取得した共有定義のコードに紐づいているイベントのカタログコードを指定。
"_ver": 1	カタログバージョン
}	
]	
}	

▼ データ取得APIレスポンス例

レスポンスボディ例	説明
[	
[	
{	
"document": null,	
"event": [	
{	
"id": {	
"index": "3_1_1",	
"value": "f348c1b9-cba8-4f8f-b122-b893e05db9e5"	
},	
"code": {	
"index": "3_1_2",	
"value": {	
"_value": 1000793,	
"_ver": 2	
}	
},	
"start": {	
"index": "3_2_1",	
"value": 2023-10-30T12:59:59.000+0900	
},	
"end": {	
"index": "3_2_2",	
"value": 2023-10-30T12:59:59.000+0900	
},	
"location": {	
"index": "3_3_1",	
"value": null	
},	
"sourceId": null,	
"env": null,	
"app": {	
"code": {	
"index": "3_5_1",	
"value": {	
"_value": 1000787,	
"_ver": 3	
}	
},	
"app": {	
"index": "3_5_5",	
"value": {	
"_value": 1000796,	
"_ver": 3	
}	
}	
},	
"wf": null,	
"thing": [	
{	
"app": {	
"code": {	
"index": "2_3_1",	
"value": {	
"_value": 1000787,	
"_ver": 3	
}	
},	
"app": {	
"index": "2_3_2",	
"value": {	
"_value": 1000796,	
"_ver": 3	
}	
}	
},	
"code": {	
"index": "4_1_2",	
"value": {	
"_value": 1000791,	
"_ver": 2	
}	

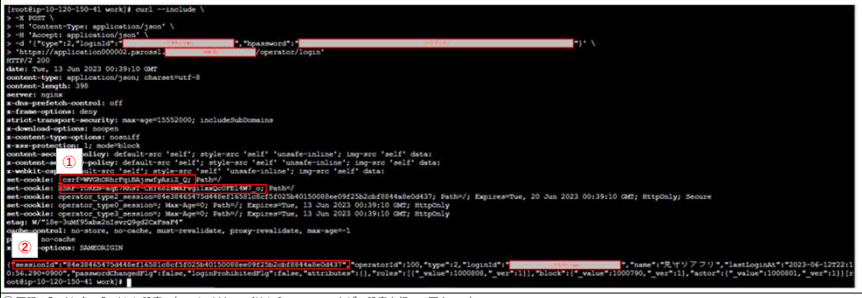
```
                ],
                "data": [
                  {
                    "IdentificationGroup": [
                      {
                        "Identification": "4",
                                    …
                ],
                "env": [
                  {
                    "index": "4_3_1",
                    "value": {}
                  }
                ],
                "envelope": {},
                "id": {
                  "index": "4_1_1",
                  "value": "22746a93-c6e6-4075-8672-9df07f095833"
                },
                "sourceId": null
              }
            ]
          }
…
        ],
        "thing": null
      }
    ]
]
```


モノ蓄積APIで蓄積した値

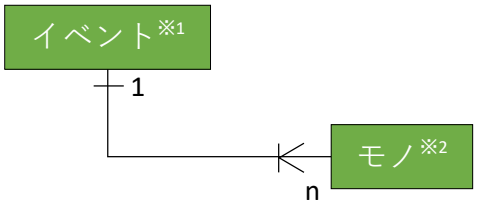
【以降、上記と同様のデータ構造で蓄積された履歴が返却されます】

ヘッダー指定方法

ログインAPIで返却されたレスポンスヘッダの値（ログインAPIレスポンスヘッダ例、画像中の①②）を、実行するAPIのヘッダに設定方法に従って設定して下さい。

ログインAPIレスポンスヘッダ例	
設定方法	① 下記のCookie名でCookieに設定。（sessionIdと_csrfはセミコロンでつなげて設定を行って下さい。） operator_type2_session=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx;_csrf=xxxxxxxxxxxxx ② 下記ヘッダ名でリクエストヘッダーに設定。 x-xsrf-token:xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
設定例（curl）	curl -X POST ¥ -H 'Content-Type: application/json' ¥ -H 'Accept: application/json' ¥ -H 'x-xsrf-token:62Nwc2n4-tBYi3Ai7d2l1Q8QhSn1uWimJ_e8' ¥ -b 'operator_type2_session=9e400f93476d073735a59d6b789bfc11175ebae8fa2e5028e557a9ff5f33f3ea;_csrf=CjGMMBklAbVmqhITqczw1ASs' ¥ -d '{"urlType":1,"userId":"oss-seminar-user-2_app1_exercise"}' ¥ https://personal-app-xxxxxxxxxx.nasushiobara-connect.jp/pxr-block-proxy/pxr-block-proxy/?from_block=1000785&from_path=/pxr-block-proxy/&block=$pxr-root-block&path=%2Fidentity-verificate%2Furl%2Fissue'

補足：イベント･モノについて



※1 イベントはモノの親となるオブジェクト

※2 モノはイベントに対して1対nの関係で格納されるオブジェクト

基盤側で採用している推奨モジュールの仕様上、上記の階層構造でデータ蓄積を行う必要があります。
今回の構成においては「2.データ蓄積処理の実装」に記載されている手順の通り、同名のイベントと
モノが1対1で蓄積されるよう、実装を行って下さい。