

Truv - Income and Employment Verification Solution Design Document

Client: Acme Mortgage

Use Case: Income and Employment Verification for Mortgage Eligibility.

Service Integration Partner: Truv

Key Stakeholders:

[Star Mortgage](#) (Website Admin, Frontend Engineering Lead, Security Engineer)

[Truv](#): (Customer Success Engineer, Solutions Architect, Site Reliability Engineers)

Section 1 - Assumptions

1. User Flow

- An Acme mortgage user is applying for a loan via a Website POS System.
- User fills in the basic information. Truv VOIE Widget appears after the user consents to income verification.
- The report is fetched and shared with the mortgage company for underwriting.

2. Authentication

- The User logs in with SSO or User ID / Password (Login Credentials)
- The Bank uses OAuth2 or JWT for secure communication between the frontend and backend systems.

3. General Web Tech Stack

- React frontend and Node.js backend [JavaScript-based web application]
- The App used HTTPS-based REST APIs for data communication.

4. Security/Compliance

- SOC 2, PCI-DSS, Secure Secret Tokens via Idempotency keys on APIs Header, TLS encryption, audit logs, etc.

Section 2 - Integrations

How Truv Link Is Triggered and Used in the POS Flow

1. **Trigger from POS Interface**

After filling the basic user information, the user clicks on continue button during their loan or credit application in the POS .This signals the system to initiate the income verification step.

2. **Generate Truv Link Token**

The POS frontend sends a request to the backend to call `POST /link-tokens`, generating a secure `link_token` for the session tied to VOIE.

3. **Launch Truv Link Widget**

Using the `link_token`, the frontend opens Truv Link as a **modal window or redirects**. This widget securely handles user login with their payroll provider.

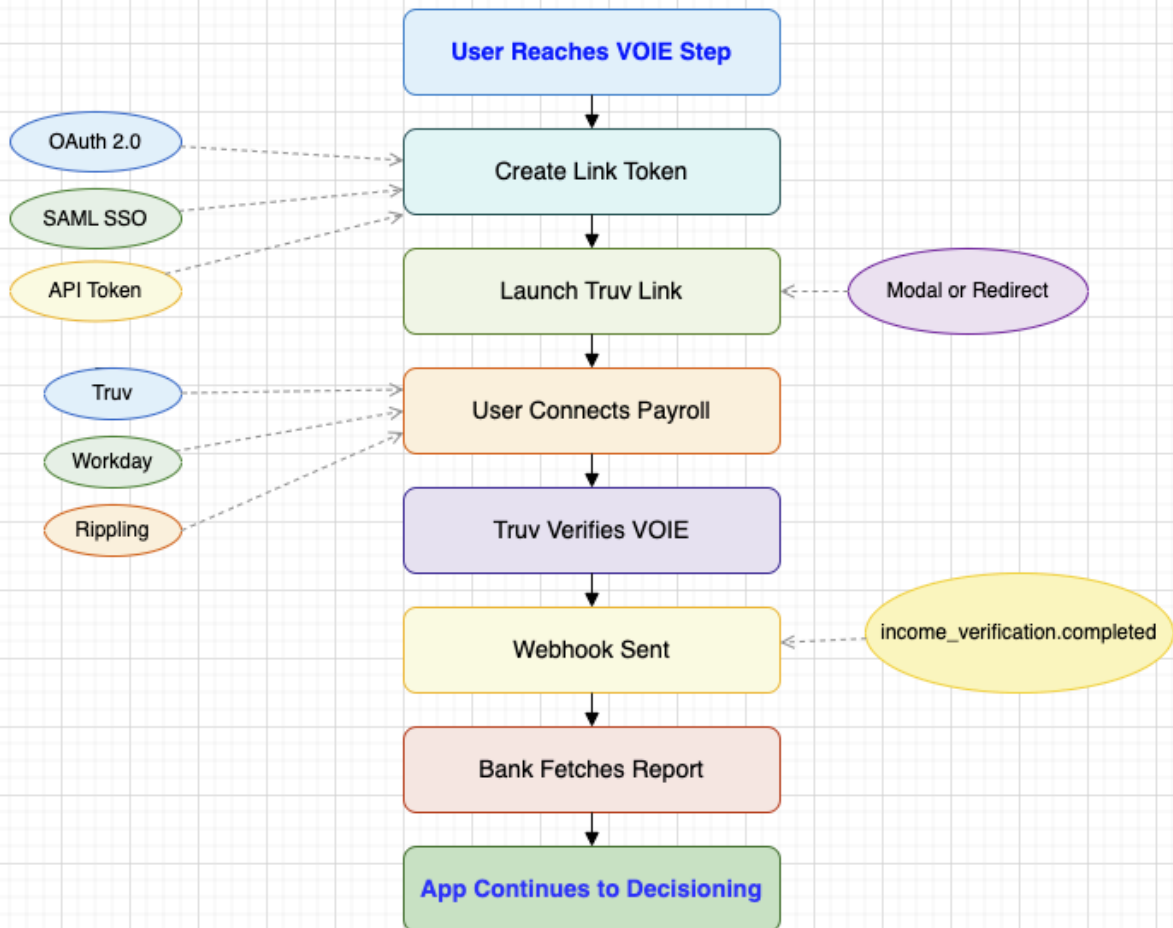
4. **Complete VOIE Verification**

Inside Truv Link, the user selects and logs into their payroll provider. Truv verifies employment and income instantly and displays a success screen.

5. **Return to POS Flow**

After completing verification, the user is returned to the POS screen. Truv sends a **webhook** to the backend, which then fetches the VOIE report to continue decision-making.

Section 3 - High-Level Flow Overview




Section 4 - Truv APIs for VOIE

1. POST /link-tokens – To Initialize Truv Link for VOIE

We will use this POST API call to generate a secure `link_token` that launches the Truv Link widget. We should call this API when the user reaches the VOIE step in the POS workflow. As we hit the API, we will get the `link_token` via API response, and that token is a credential that is embedded in your Truv JS, Flutter, etc, SDK to render the frontend modal or redirect flow, and the user can connect to the Payroll provider.


REQUEST EXAMPLES ▾

```
1 POST /v1/link-access-tokens/ HTTP/1.1
2 Accept: application/json
3 Content-Type: application/json
4 Host: prod.truv.com
5
6
```

 [Try It!](#)

RESPONSE 201 EXAMPLE ▾

```
1 {
2   "access_token": "48427a36d43c4d5aa6324bc06c692456",
3   "link_id": "24d7e80942ce4ad58a93f70ce4115f5c",
4   "link_hash": "bc917458a3da4b2c8cc8282aa1707aaa",
5   "data_source": "payroll"
6 }
```



```
<script src="https://cdn.truv.com/link/truv.link.js"></script>
<script>
  const link = window.Truv.Link.create({
    token: "lk-live-xyzabc123456", // ← link_token from step 1
    onSuccess: function (publicToken, metadata) {
      console.log("Success!", metadata);
    },
    onExit: function (err, metadata) {
      console.log("User exited", err, metadata);
    }
  });

  link.open(); // Launches Truv Link as a modal or redirect
</script>
```

When you create a link token using `POST/link-tokens`, you include your internal user ID / Partner user ID (`user_external_id`). If this ID is new to Truv, the user is automatically created behind the scenes. Therefore, explicitly creating a user with `POST /users` is optional and typically not needed for standard VOIE flows.

2. `POST /users/{user_id}/reports` – To Create Income and Employment Report

This API is called to generate a detailed income and employment report for the specified user. Here we use the `user_id` from the system (`client_user_id`) to trigger the report creation. This report includes paystubs, W2s, bank deposits, and other detailed VOIE data used for deeper underwriting and compliance.

CURL REQUEST ▾

```
1 curl --request POST \  
2   --url https://prod.truv.com/v1/users/user_id/reports/ \  
3   --header 'accept: application/json'
```

[Try It!](#)

RESPONSE

201 EXAMPLE ▾

```
1 {  
2   "report_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  
3   "created_at": "2022-05-04T11:30:00Z",  
4   "completed_at": "2022-05-04T12:00:00Z",  
5   "last_task_at": "2022-05-04T12:00:00Z",  
6   "links": [  
7     {  
8       "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  
9       "link_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  
10      "tracking_info": "string",  
11      "data_source": "payroll",  
12      "provider": "string",  
13      "provider_name": "string",  
14      "employments": [  
15        {  
16          "income": "70000.00",  
17          "income_unit": "YEARLY",  
18          "pay_rate": "6500.00",  
19          "pay_frequency": "M",  
--
```

3. **GET /users/{user_id}/reports/{report_id}** – Retrieve an Income and Employment Report

This endpoint retrieves a previously generated income and employment report for a specific user. Use the `user_id` and `report_id` to access full VOIE report details such as paystubs, W2s, employer info, and deposit records. This API is typically used on the backend for underwriting, audit, or record-keeping purposes.

CURL REQUEST ▾

```
1 curl --request GET \  
2   --url 'https://prod.truv.com/v1/users/user_id/reports/{report_id}' \  
3   --header 'accept: application/json'
```

[Try It!](#)

RESPONSE

200 EXAMPLE ▾

```
1 {  
2   "report_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  
3   "created_at": "2022-05-04T11:30:00Z",  
4   "completed_at": "2022-05-04T12:00:00Z",  
5   "last_task_at": "2022-05-04T12:00:00Z",  
6   "links": [  
7     {  
8       "id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  
9       "link_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",  
10      "tracking_info": "string",  
11      "data_source": "payroll",  
12      "provider": "string",  
13      "provider_name": "string",  
14      "employments": [  
15        {  
16          "income": "70000.00",  
17          "income_unit": "YEARLY",  
18          "pay_rate": "6500.00",  
19          "pay_frequency": "M",  
20          "statements": [  
21            {  
22              "id": "24d7e80942ce4ad58a93f70ce4115f5c",  
23              "check_number": "29205182",  
24              "pay_date": "2018-05-15",
```

Section 5 - Webhooks & Events

1. Link Connected

The sample webhook below is the payload for `link-connected`.

```
JSON
{
  "webhook_id": "ae53152c8c4c4636bda7f22f2cf6b0a5",
  "link_id": "9915c50cc047413bb810767f218390f8",
  "user_id": "88fef4cea64c40b5ad6727cc9b0b9fdc",
  "product": "employment",
  "data_source": "payroll",
  "tracking_info": null,
  "event_type": "link-connected",
  "event_created_at": "2022-08-23T17:32:30.217980Z",
  "user_id": "88fef4cea64c40b5ad6727cc9b0b9fdc",
  "template_id": null
}
```

The `link-connected` event occurs when the `Link` object is created. It displays the `status` of the first related `Task` when set to `done`.

2. Pay Statement Created

This sample webhook payload is for `statements-created`.

```
JSON
{
  "webhook_id": "f82ab0a92ddd4bb7b6117635159b366a",
  "task_id": "c32fb957ec7246828da56be7516da765",
  "link_id": "9915c50cc047413bb810767f218390f8",
  "product": "employment",
  "data_source": "payroll",
  "tracking_info": null,
  "event_type": "statements-created",
  "event_created_at": "2022-08-23T17:32:24.812306Z",
  "objects_count": 9,
  "employment_id": "427abebd8590457e8332fdff77fc412f",
  "user_id": "88fef4cea64c40b5ad6727cc9b0b9fdc",
  "template_id": null
}
```

The `statements-created` events occurs when the initial `Task` is successful and complete with new objects.

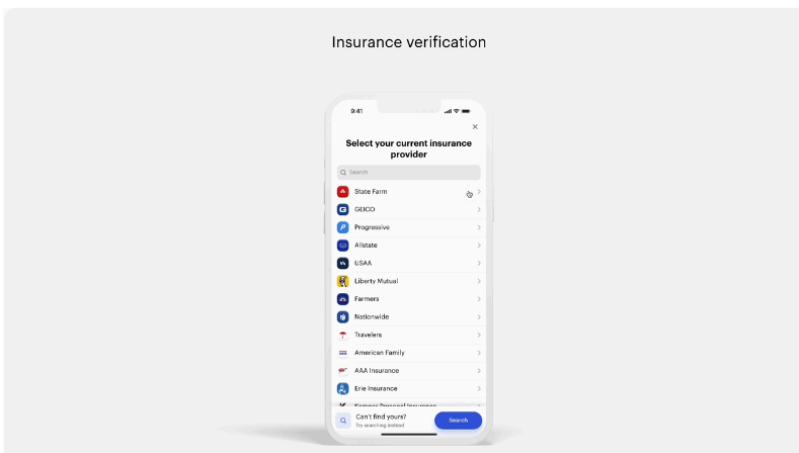
Section 6 - Success Criteria

- ☐ Truv Link widget loads successfully
- ☐ Responsive design on all screen sizes
- ☐ Connection to payroll provider completes smoothly
- ☐ Clear error messages for the user if failure occurs
- ☐ Webhooks received and processed
- ☐ VOIE data successfully fetched from API
- ☐ Secrets securely stored, no credentials in frontend
- ☐ Easy fallback if VOIE fails (e.g., manual upload option)

Section 7 (Optional) - Truv Bridge - Templated Drop In for VOIE

Truv Bridge

This client-side component helps your users connect their account information.



Truv Bridge is a drop-in module for your users to connect their accounts to Truv. This then allows you to access their data using Truv's API.

Truv Bridge handles employer searches, validating credentials, multi-factor authentication, and error handling. The module works across all modern browsers and platforms. This includes web, mobile, iOS, and Android, as well as through React Native, Flutter, and Expo.