1. What is e-Signet?

e-Signet is MOSIP’s **OIDC-compliant authentication and authorization gateway** — similar to how “Login with Google” works, but for digital identity (MOSIP ID).
It provides:

- Authentication using **MOSIP ID / Digital ID / VC**
- OIDC-compliant tokens (`ID Token`, `Access Token`)
- Integration for **consent-based login and identity sharing**

2. Requirements for Integration

Requirement	Description
e-Signet Sandbox / Test Environment	Access to MOSIP sandbox or staging environment (https://sandbox.esignet.io).
Client Application (Your Portal)	Your web app / backend service must support **OIDC client flow** (Authorization Code Flow).
Redirect URI	The callback URL in your app where MOSIP redirects after successful login.
Client Credentials	You’ll need a **Client ID** and **Client Secret** from the MOSIP e-Signet admin or API console.
MOSIP Partner Registration	You must register your organization/application with MOSIP to obtain credentials and scopes.

3. Typical OIDC Authorization Flow (with e-Signet)

Step1️⃣— User clicks “Login with MOSIP ID”

Step2️⃣— User Authenticates on e-Signet

- User enters **MOSIP ID** or scans **QR from INJI Wallet**.
- e-Signet performs **identity verification and consent**.
- On success, it redirects to your `redirect_uri` with an **Authorization Code**.

Step3️⃣— Exchange Authorization Code for Token

Step4️⃣— Validate and Retrieve User Info

Step5️⃣— Establish

3. Configuration Requirements in Your App

Setting	Example Value	Notes
Issuer URL	`https://sandbox.esignet.io`	OIDC provider URL
Authorization Endpoint	`/authorize`	For login redirect
Token Endpoint	`/token`	For exchanging code
UserInfo Endpoint	`/userinfo`	For retrieving identity details
JWKS Endpoint	`/.well-known/jwks.json`	For validating tokens
Client ID / Secret	Provided by MOSIP	Used in token exchange
Redirect URI	e.g., `https://myportal.gov/callback`	Must match registered URI

4. Summary Checklist

Task	Description
🔸 Register your app with MOSIP	Get Client ID, Secret, Redirect URI
🔸 Configure OIDC endpoints	`/authorize`, `/token`, `/userinfo`, `/jwks.json`
🔸 Implement Authorization Code Flow	Secure server-side token exchange
🔸 Validate JWT & user claims	Ensure authenticity and integrity

🔸 Integrate into your workflow	Auto-enroll authenticated users for subsidy

Social Grant Distribution using MOSIP ID – Requirements Document

1. Objective

To provide verified and transparent subsidy disbursement to farmers affected by a calamity (e.g., flood, drought, cyclone) by leveraging MOSIP's digital identity framework for authentication and benefit delivery.

2. High-Level Flow

- 1. Farmer Registration / Verification
- 2. Calamity Assessment
- 3. Subsidy Application / Auto-Enrolment
- 4. Verification & Approval
- 5. Disbursement
- 6. Post-Disbursement Audit & Transparency

3. Technical Requirements

Component	Description
MOSIP Integration	Access to MOSIP's e-KYC, e-Signet (OIDC), Credential Issuance, and Verification APIs.
Farmer Registry Database	Database containing farmer land details, linked with their MOSIP ID.

Subsidy Management Module	Workflow to manage calamity-based eligibility, approvals, and payments.
Digital Wallet Integration	Wallet supporting MOSIP-issued verifiable credentials (e.g., Inji or Klefki).
Payment Gateway Integration	Interface to process direct benefit transfers (DBT) using verified details.
Audit Dashboard	Visualization of subsidy distribution, region-wise analytics, and verification tools.

4. Policy / Process Requirements

- Clear eligibility criteria for subsidy (location, crop type, calamity type).
- Legal approval for using MOSIP-based identity for benefit delivery.
- Data protection & consent management in compliance with privacy laws.
- Integration MoU with MOSIP for sandbox & production access.
- Multi-lingual farmer portal / mobile app for accessibility.

5. Optional Enhancements

- Issue Verifiable Presentation Request (VPR) via e-Signet for subsidy authentication.
- Add geo-tagging for affected farmlands using satellite or IoT data.
- Allow offline verification using QR-based Verifiable Credentials.
- Enable reusable credentials for other government benefit programs.

Use Case 1: Social Grant Issuance Using Inji Wallet Integration

Overview

Digital social grant credential system enabling government agencies to issue tamper-proof welfare benefits as verifiable credentials that citizens store in Inji Wallet and present to authorized fund providers (stores/service providers) for redemption, ensuring transparent, fraud-free benefit delivery with real-time transaction tracking

Actors & Roles

Credential Holder (Citizen/Beneficiary)

- Registers on government portal with personal details (Name, Address, Email, DOB)
- Completes identity verification via government-issued ID and facial recognition matching
- Downloads Inji Wallet (mobile app) to receive and store Social Grant credential
- Presents digital grant credential at participating stores/service providers for verification
- Views transaction history and remaining grant balance via wallet dashboard

Issuer (Government/Social Welfare Department)

- Operates government web portal for citizen registration and application processing
- Verifies citizen identity through existing ID systems and biometric authentication
- Issues Social Grant credentials using Inji Certify (OpenID4VCI draft 13 compliant)
- Generates W3C-compliant verifiable credentials in JSON-LD or SD-JWT formats with cryptographic signatures
- Monitors grant usage through dashboard showing aggregate transactions across beneficiaries and providers
- Manages credential lifecycle: issuance, renewal, revocation (if benefits expire/terminate)

Verifier (Fund Provider/Participating Stores)

- Registers as authorized verifier in the Social Grant ecosystem

- Uses mobile app with Inji Verify integration to scan and validate citizen's grant credential via QR code
- Confirms credential authenticity, validity period, and beneficiary identity at point of transaction
- Completes transaction (goods/services) and records transaction on blockchain ledger
- Receives reimbursement from government based on verified transactions

Use Case 2: Klefki + E-Signet Integration - eKYC Wallet Verification MVP

Overview

Enterprise-ready eKYC solution integrating Klefki wallet with MOSIP E-Signet for identity verification, enabling businesses to onboard customers with cryptographically verified government-issued credentials, reducing fraud and compliance costs while providing seamless passwordless authentication

MVP Solution

Foundation & E-Signet Setup

- Deploy E-Signet instance as OpenID Connect (OIDC) provider for identity authentication
- Configure E-Signet integration with MOSIP ID repository (or test environment for demo)
- Setup Klefki wallet backend with DID infrastructure and credential storage
- Define credential schema for eKYC data: Name, DOB, Address, ID Number, Photo, Biometric hash

Credential Issuance & Wallet Integration

- Implement government ID verification flow: User presents physical/digital ID → System verifies against MOSIP registry → Facial recognition liveness check
- Issue eKYC credential to Klefki wallet using OpenID4VCI standard
- Enable DID-based credential binding in Klefki wallet for secure storage
- Implement credential refresh mechanism for updated KYC data

Verifier Integration & Authentication

- Build relying party (RP) integration for businesses/services needing eKYC verification

- Implement E-Signet authentication flow: Service requests eKYC → User scans QR with Klefki wallet → E-Signet authenticates user → Wallet shares consented claims
- Support multiple authentication factors: Biometric, OTP, Cryptographic key (DID-based)
- Enable selective disclosure: User controls which KYC attributes to share (e.g., age verification without full address)