**Components**

1. **Lender User**  
   The end-user interacting with the application through a web browser.
2. **PingFederate**  
   The identity provider handling user authentication and reference ID generation.
3. **Consent Application**  
   A central application (frontend and backend combined) that manages user consent.
4. **Datastore Consent API**  
   A service that stores and retrieves user consent records.
5. **TSP Database**  
   A database that validates the lender-TSP relationship.

**Updated Process Flow**

**1. Lender User Login**

**Step 1:** The lender user logs in using their **TSP ID and Password** on the TSP application.  
**Step 2:** TSP application redirects the lender user to **PingFederate** for authentication.

**2. PingFederate Authentication**

**Step 3:** PingFederate authenticates the lender user.  
**Step 4:** PingFederate checks if the lender-TSP relationship exists:

* **If Relationship Exists:** Proceed to **Step 5.**
* **If Relationship Does Not Exist:** Redirect the lender user to an **error message page**.

**Step 5:** PingFederate redirects the lender user to the **Consent Application** with a Reference ID (REF).

**3. Retrieve User Details**

**Step 6:** The **Consent Application** performs the **Attribute Pickup** process by calling PingFederate to retrieve user details (e.g., User ID and attributes) using the REF.  
**Step 7:** PingFederate responds with the User ID and other attributes.

**4. Relationship Validation in TSP Database**

**Step 8:** The Consent Application validates the lender-TSP relationship by querying the **TSP Database**:

* **If Relationship Exists:** Proceed to **Step 9.**
* **If Relationship Does Not Exist:** Redirect the lender user to an **error message page**.

**5. Check Consent Status**

**Step 9:** The Consent Application calls the **Datastore Consent API** to read the consent record:

* **If Consent Exists:** Proceed to **Attribute Dropoff** (Step 15).
* **If Consent Does Not Exist:** Proceed to **Fetch System IDs** (Step 10).

**6. Fetch System IDs**

**Step 10:** The Consent Application retrieves system IDs from a **pre-configured file**.

**7. Display Consent Page**

**Step 11:** The Consent Application displays a **Consent Page** showing:

* Consent text.
* Dropdown for selecting system IDs.

**Step 12:** The lender user interacts with the UI to:

* View the consent text.
* Select desired system IDs.
* Submit their response (Agree/Disagree).

**8. User Response Handling**

**User Agrees to Consent**  
**Step 13:** If the lender user agrees:

* The Consent Application calls the **Datastore Consent API** to create a consent record.
* The Datastore Consent API responds with a success message.
* Proceed to **Attribute Dropoff** (Step 15).

**User Denies Consent**  
**Step 14:** If the lender user denies:

* The Consent Application redirects the lender user to a **Deny Message Page**.

**9. Attribute Dropoff**

**Step 15:** The Consent Application performs the **Attribute Dropoff** process by calling PingFederate.  
**Step 16:** PingFederate responds with a **new REF ID**.  
**Step 17:** The Consent Application redirects the lender user back to PingFederate for further processing.  
**Step 18:** PingFederate redirects the lender user to the TSP application.

**Additional Notes**

1. **Redirects vs API Calls:**
   * **Redirects:** Lender User → PingFederate → Consent Application.
   * **API Calls:** Consent Application → PingFederate (Attribute Pickup), Consent Application → Datastore Consent API, Consent Application → TSP Database.
2. **Error Handling:**
   * Missing relationships or invalid data results in redirects to an **error message page**.
   * A separate **Deny Message Page** is displayed if the lender user denies consent.
3. **Simplified System ID Management:**
   * System IDs are retrieved from a **pre-configured file** instead of external API calls (CIAM API).
4. **Consent Application Focus:**
   * Handles all heavy lifting (relationship validation, consent status checks, and data handling).
   * Ensures smooth redirection back to PingFederate and TSP application.