# ☑ PrivGuardian Website: Full User Flow + Data Flow (Step-by-Step)

---

## 👤 Actors

1. **User (Fintech Customer)**
2. **Third-Party Fintech Service**
3. **PrivGuardian (Middleware Platform)**

---

## 🌐 Pages / Components

1. Home / Login
2. Consent Dashboard
3. Share Data Page
4. Access Logs Page
5. Revoke Access Page
6. Admin / Backend APIs (monitoring)

---

## 🔁 FULL INTERACTION FLOW

---

### ◇ STEP 1: User Login/Signup

**Click Flow:**

- Click "Login" → Enters credentials → `POST /api/login`
- Backend verifies using JWT and bcrypt → stores session token in cookie

**Data Flow:**

- User → [Frontend Form] → `/api/login` → Backend → JWT issued → Cookie stored

---

### ◇ STEP 2: Consent Dashboard

**Click Flow:**

- Click "Give Consent"
- Select 3rd-party app (dropdown)
- Choose:
  - Data Fields (checkboxes: income, txn, location)
  - Purpose (dropdown: budget, lending, etc.)
  - Time Limit (slider or input)
  - Access Count (optional)

→ Click "Confirm Consent" → `POST /api/consent`

**Data Flow:**

- User → React form → API `/api/consent`
- Backend stores in MongoDB: `{ userId, fields, expiry, purpose, partner, createdAt }`
- Returns `consentId`

---

### ◇ STEP 3: Data Tokenization & Share

**Click Flow:**

- Click "Share Data" → Select 3rd-party → Confirm

**Data Flow:**

- Frontend → `POST /api/share`
- Backend:

    - Fetches selected data fields
    - Replaces real data with token (`tokenize(value)`)
    - Stores token map in DB (`{tokenId, originalValue, expiresAt}`)
    - Returns tokenized data bundle to frontend

- Frontend → Sends tokenized data via API or downloads JSON

- Tokens expire after policy limits

---

### 🔸 STEP 4: 3rd-Party Tries to Use Token

**Data Flow:**

- 3rd-party → `POST /api/resolve-token` with `tokenId`

- Middleware checks:

    - Is token valid?
    - Is within policy limits? (purpose, count, time)
    - Logs request in Access Logs DB

- If allowed → resolves token → sends real data

- If denied → sends 403 + alert to user

---

### 🔸 STEP 5: Monitor Logs (User View)

**Click Flow:**

- Click "Access Logs"

- Sees list:

    - Who accessed what
    - When
    - From where

**Data Flow:**

- React calls `GET /api/logs`
- Backend queries logs: `{ userId, accessType, partnerId, timestamp, tokenId, purpose }`

---

### 🔸 STEP 6: Revoke Access

**Click Flow:**

- Click "Revoke Access" → Select app → Click "Revoke"

**Data Flow:**

- React → `POST /api/revoke`

- Backend:

    - Deletes active tokens
    - Updates consent status to `revoked`
    - Logs action

---

## ⚙ Behind the Scenes

| Function | What Happens |
|---|---|
| **JWT Authentication** | All pages are protected with JWT tokens in cookies |
| **Token Generation** | Node.js `crypto.randomBytes()` or Hashids to mask values |
| **Policy Enforcement** | Casbin/OPA engine checks token usage rules |
| **Anomaly Alerts (Optional)** | Socket.io can push real-time alerts to user UI if odd behavior is detected |
| **Consent & Logs Storage** | MongoDB collections: `consents`, `accessLogs`, `tokens`, `users` |

---

## 🧱 Example MongoDB Structure

**`consents`**

```
{
  "_id": "xyz",
  "userId": "user123",
  "partnerId": "budgetApp",
  "fields": ["income", "location"],
  "purpose": "budgeting",
  "expiresAt": "2025-06-12T12:00:00Z"
}
```

**`accessLogs`**

```
{
  "userId": "user123",
  "tokenId": "abc123",
  "partnerId": "budgetApp",
  "timestamp": "2025-06-10T13:00:00Z",
  "status": "granted"
}
```

---

## 🧠 How to Represent to Jury (Live or Screenshot Demo)

| Step | What to Show |
|---|---|
| Login | Simple auth with session stored |
| Consent | UI showing granular controls |
| Tokenization | JSON file with masked tokens |
| Token Resolution | 3rd-party mock calling resolve API |
| Logs | Live or dummy entries on dashboard |
| Revocation | Show revoked tokens blocked live |

---