EVzone SDK — Cookie-Based Account Detection & Selection

This note tells you exactly what to implement so the SDK:

- reads **user numbers** from cookies,
- resolves them to **full user profiles** `{ userNo, walletId, owner, email, photo }`,
- and shows the correct modal:
  - **0 users** → *No account / Sign in*
  - **1 user** → *Summary*
  - **2+ users** → *Account Picker*

It also shows you where to **console the full selected profile object**.

What you need to do

1) Implement profile resolution in `src/utils/cookie.js`

You already have helpers like `getUserNoFromCookie`, `getUserNosFromCookie`, etc.

Add/modify **one function** that turns an array of userNos → **full profile objects**.

```
// src/utils/cookie.js

/**
 * Return an array of profiles for the given userNos.
 * Each profile MUST have: { userNo, walletId, owner, email, photo }.
 *
 * Replace the example lookup below with your own cookie/storage logic.
 */
export function resolveUserProfilesFromCookies(userNos = []) {
  const arr = Array.isArray(userNos) ? userNos.filter(Boolean) : [];

  // TODO: Replace this map with your own values read from cookies/local
storage
  const known = {
    'U-000123': {
      walletId: 'W-256-48392018',
      owner: 'John Doe',
      email: 'john@x.com',
      photo: 'https://api.dicebear.com/7.x/initials/svg?seed=John%20Doe',
    },
    'U-000789': {
      walletId: 'W-256-74731323',
      owner: 'Jane Smith',
      email: 'jane@x.com',
```

```
      photo: 'https://api.dicebear.com/7.x/initials/svg?seed=Jane%20Smith',
    },
  };

  return arr.map((u) => {
    const k = String(u);
    const row = known[k];
    // IMPORTANT: always return the full shape
    return row
      ? { userNo: k, ...row }
      : {
          userNo: k,
          walletId: null,
          owner: `User ${k.slice(-3)}`,
          email: `${k.replace(/[^a-z0-
9]/gi,'').toLowerCase()}@example.com`,
          photo: `https://i.pravatar.cc/80?u=${encodeURIComponent(k)}`,
        };
  });
}
```

> When you wire in your real cookie logic: **Don't change the function name or return shape**—just populate the same fields from your data source.

---

2) Make lookup use the cookie resolver (already supported by the SDK)

In `src/sdk/paykitClient.js`, the SDK calls `lookupUsersByNo(userNos)` to get profiles for the Account Picker. Ensure it **delegates to the resolver** you just implemented:

```
// src/sdk/paykitClient.js
import { resolveUserProfilesFromCookies } from '../utils/cookie.js';

async function lookupUsersByNo(userNos) {
  const arr = Array.isArray(userNos) ? userNos.filter(Boolean) : [];
  if (arr.length === 0) return { users: [] };

  // No external calls here. Just resolve from cookies/storage.
  const users = resolveUserProfilesFromCookies(arr);
  return { users };
}
```

> Nothing else needs changing here.

---

3) Log the **full profile object** when an account is chosen

The SDK already provides the `accounts` array to the payment form during the Account Picker step. Add one line to log the full object that was chosen.

```
// src/WalletPaymentForm.js
// ...
else if (view === 'accountPicker') {
  content = (
    <AccountPickerModal
      open
      zIndex={zIndex}
      accounts={accounts || []}
      onSelect={(userNo) => {
        // 🔎 Log the full selected profile:
        const selected = (accounts || []).find(a => a.userNo === userNo);
        try { console.log('[EVZ SDK] selected account:', selected); } catch
{}

        setPasscode('');
        // Continue the flow with the picked account
        selectAccount?.(userNo);
      }}
      onClose={closeAndReset}
    />
  );
}
```

If the user goes through the **Sign In** step instead, log right after a successful sign-in (you get the `userNo` there too):

```
// src/WalletPaymentForm.js
else if (view === 'signin') content = (
  <HasAccountSummary
    open
    onLoginSuccess={(userNo) => {
      // If you need the full object here as well:
      // (Re-)resolve via your cookie resolver
      //   import { resolveUserProfilesFromCookies } from
'./utils/cookie.js';
      // const [profile] = resolveUserProfilesFromCookies([userNo]);
      // console.log('[EVZ SDK] selected account (signin):', profile);

      setPasscode('');
      selectAccount?.(userNo);
    }}
    onClose={closeAndReset}
    zIndex={zIndex}
  />
);
```

> Result: when someone picks an account, the console shows: `{ userNo, walletId, owner, email, photo }`

How the flow decides which modal to show

The hook (`useWalletPaymentFlow`) checks cookies and behaves as follows:

- **No user numbers found** → shows **"No account / Sign in"** modal
- **Exactly 1 user number** → auto-selects and shows **Summary**
- **2 or more user numbers** → calls `lookupUsersByNo` (your resolver) and shows **Account Picker**

You don't need to change the hook logic—just make sure your cookie utilities:

- return userNos via `getUserNosFromCookie()`, and
- return **full profiles** via `resolveUserProfilesFromCookies(userNos)`.

Contract you must keep

- `resolveUserProfilesFromCookies(userNos)` **must** return an array of objects with:

```
[
    {
    userNo: string,
    walletId: string | null,
    owner: string,
    email: string,
    photo: string
  },{
    userNo: string,
    walletId: string | null,
    owner: string,
    email: string,
    photo: string
  },{
    userNo: string,
    walletId: string | null,
    owner: string,
    email: string,
    photo: string
  }
]
```

- `getUserNosFromCookie()` **must** return all available userNos in priority order (active first if you prefer).
- Do not rename these functions or change their return shapes. The UI and flow rely on them.