**Product Requirements Document (PRD) with Wireframes and Data Model**

**Product Title:**

**xPoints Exchange** – Universal Loyalty Point Exchange Platform

**Purpose:**

To create a digital platform that allows users to **exchange loyalty/reward points** between various sources (e.g., Qantas Points, Guzman y Gomez Loyalty) at real-time market rates. A **base currency/unit** (e.g., xPoint) will serve as the intermediary to ensure accurate and fair conversions between incompatible systems.

**Objectives**

* Enable users to convert points between different loyalty programs.
* Introduce xPoints as a base unit to facilitate conversion and liquidity.
* Provide updated exchange rates for point systems.
* Build a scalable backend for adding future loyalty programs.
* Provide a secure frontend for user operations.

**Target Users**

* Consumers with points in multiple loyalty systems
* Frequent flyers, shoppers, and digital-savvy users
* Deal seekers and value optimizers
* Potential corporate clients (brands with loyalty programs)

**Core Features**

**1. User Accounts**

* User registration (email/social)
* Login and 2FA
* KYC for large conversions

**2. Wallet System**

* Each user has:
  + A wallet for each loyalty program
  + An xPoints wallet as intermediary

**3. Supported Loyalty Programs (Initial Phase)**

* **Qantas Frequent Flyer**
* **Guzman y Gomez (GYG) Loyalty Program**

**4. Conversion Engine**

* Exchange flow: Source Points → xPoints → Target Points
* Example: Qantas → xPoints → GYG
* Rate logic defined in exchange\_rates table
* Fee deductions per conversion (configurable)

**5. Transaction Management**

* Store all point exchanges with timestamps
* Maintain transaction history per user

**6. Admin Controls**

* Update exchange rates
* Monitor suspicious activity
* Manage users and audit logs

**Wireframes (Low-Fidelity)**

**Main Screens:**

**1. Link Account**

* Dropdown: Select loyalty program
* Text: Enter account number
* Button: Link

**2. Conversion Screen**

* From: Select source program and amount
* To: Select target program
* Display: Estimated value
* Button: Convert

**3. Dashboard**

* Balances for each wallet (Qantas, GYG, xPoints)
* Conversion button
* Transaction history preview

**4. Transaction History**

* List of past exchanges
* Program to program, amount, timestamp

🖼️ **Wireframes image file:** A\_wireframe\_digital\_wireframe\_displays\_four\_low-fi.png

**Backend Data Model (Database Schema)**

**Tables:**

**1. users**

| **Field** | **Type** |
| --- | --- |
| id | UUID |
| email | VARCHAR |
| password\_hash | TEXT |
| created\_at | TIMESTAMP |
| kyc\_verified | BOOLEAN |

**2. wallets**

| **Field** | **Type** |
| --- | --- |
| id | UUID |
| user\_id | UUID (FK to users) |
| program | ENUM('QANTAS', 'GYG', 'XPOINTS') |
| balance | FLOAT |
| created\_at | TIMESTAMP |

**3. exchange\_rates**

| **Field** | **Type** |
| --- | --- |
| id | UUID |
| from\_program | ENUM |
| to\_program | ENUM |
| rate | FLOAT (e.g. 1 Qantas = 0.75 xPoints) |
| last\_updated | TIMESTAMP |

**4. transactions**

| **Field** | **Type** |
| --- | --- |
| id | UUID |
| user\_id | UUID |
| from\_program | ENUM |
| to\_program | ENUM |
| amount\_from | FLOAT |
| amount\_to | FLOAT |
| timestamp | TIMESTAMP |
| fee\_applied | FLOAT |

**Frontend Stack (Suggested)**

* React + Tailwind CSS
* Axios for API calls
* Auth0 / Firebase Auth for login

**Backend Stack (Suggested)**

* Node.js with Express OR Django
* PostgreSQL
* REST API structure (or GraphQL for scale)
* Hosted on AWS/GCP

**API Sample (REST)**

**POST /convert**

{

"user\_id": "uuid",

"from\_program": "QANTAS",

"to\_program": "GYG",

"amount": 1000

}

Returns converted value, fee, and transaction ID.

**Timeline**

| **Phase** | **Duration** |
| --- | --- |
| Wireframes | 2 weeks |
| Prototype | 3 weeks |
| Backend setup | 3 weeks |
| MVP build | 4–6 weeks |
| Beta test | 2 weeks |

**KPIs**

* Avg. daily conversions
* Avg. points per transaction
* User retention rate
* Linked accounts per user

**Next Steps**

* Approve schema and endpoints
* Finalize design system (colors, typography)
* Set up initial database and cloud environment
* Begin MVP coding sprint