# SilkGate App — Subscription System & Monetization Model

This document describes the proposed structure for implementing a subscription system in the SilkGate platform. The purpose is to enable monetization by offering tiered access to features for different user roles, primarily investors.

## 1. User Structure (users.json)

{  
 "user\_id": "440851294",  
 "username": "@angelinvestor\_kz",  
 "email": "investor@example.com",  
 "role": "investor",  
 "subscription": {  
 "plan": "pro",  
 "started\_at": "2025-07-01T12:00:00Z",  
 "expires\_at": "2025-08-01T12:00:00Z",  
 "is\_trial": false,  
 "is\_active": true  
 },  
 "created\_at": "2025-06-25T09:00:00Z"  
}

## 2. Subscription Plans (plans.json)

{  
 "plans": [  
 {  
 "id": "free",  
 "name": "Free",  
 "price\_usd": 0,  
 "limits": {  
 "project\_views\_per\_day": 2,  
 "startup\_contacts\_visible": false,  
 "can\_export": false,  
 "can\_submit\_interest": true  
 }  
 },  
 {  
 "id": "pro",  
 "name": "Pro",  
 "price\_usd": 19,  
 "limits": {  
 "project\_views\_per\_day": -1,  
 "startup\_contacts\_visible": true,  
 "can\_export": true,  
 "can\_submit\_interest": true  
 }  
 },  
 {  
 "id": "partner\_plus",  
 "name": "Partner+",  
 "price\_usd": 49,  
 "limits": {  
 "project\_views\_per\_day": -1,  
 "startup\_contacts\_visible": true,  
 "can\_export": true,  
 "can\_submit\_interest": true,  
 "access\_private\_deals": true  
 }  
 }  
 ]  
}

## 3. Transactions (transactions.json)

{  
 "transactions": [  
 {  
 "txn\_id": "txn\_001239843",  
 "user\_id": "440851294",  
 "plan\_id": "pro",  
 "amount\_usd": 19,  
 "status": "paid",  
 "payment\_provider": "stripe",  
 "created\_at": "2025-07-01T12:00:00Z"  
 }  
 ]  
}

## 4. Feature Access Matrix (features.json)

{  
 "features": {  
 "export\_excel": ["pro", "partner\_plus"],  
 "private\_deals": ["partner\_plus"],  
 "contact\_visibility": ["pro", "partner\_plus"],  
 "limit\_free\_views": {  
 "free": 2,  
 "pro": -1,  
 "partner\_plus": -1  
 }  
 }  
}

## 5. Implementation Notes

- Use the above JSON schema to design your backend database structure (users, plans, transactions).  
- Apply middleware on project viewing & startup contact display based on the active user's plan.  
- Integrate with a payment provider (Stripe, Paddle, etc.) for subscription handling.  
- Store plan metadata and feature access in backend for real-time authorization.  
- Track daily usage (views, contacts, exports) in backend to enforce limits.