SMM PANEL - COMPLETE TECHNICAL SPECIFICATION

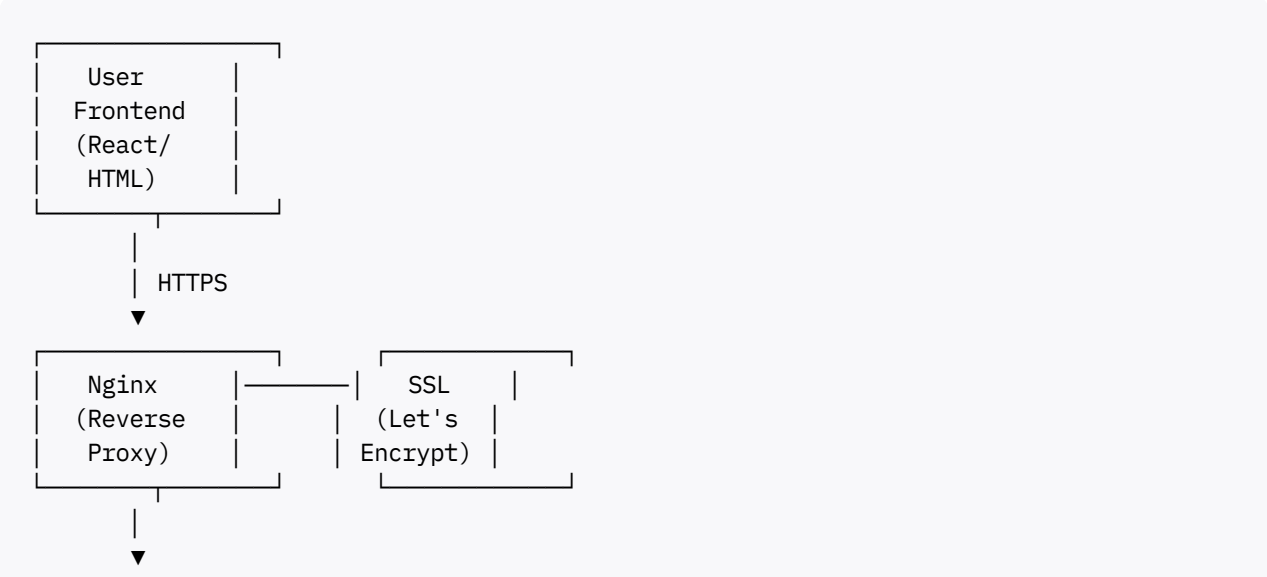
Phase 1 Implementation Guide

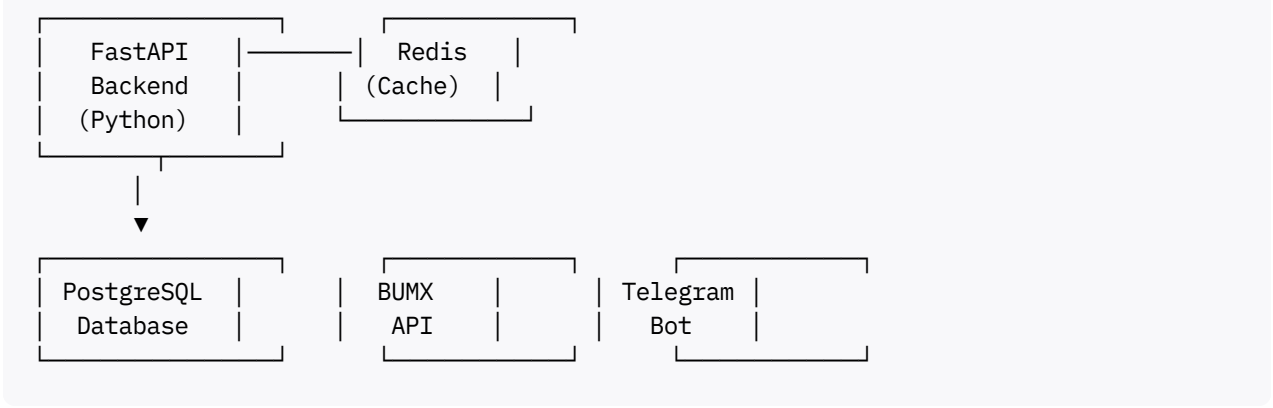
Project: HOMEMMO Social Media Marketing Panel
Version: 1.0
Date: October 26, 2025
Status: Ready for Implementation

TABLE OF CONTENTS

1. SYSTEM OVERVIEW

Architecture





Tech Stack

- **Backend:** FastAPI 0.68.0 (Python 3.6 compatible)
- **Database:** PostgreSQL 15 + Redis 7
- **Frontend:** HTML/CSS/JavaScript (no build)
- **Admin:** HTML/CSS/JavaScript (AdminLTE style)
- **Payment:** Sepay API + ACB Bank
- **Notifications:** Telegram Bot API
- **Server:** CentOS 7, Nginx, Gunicorn

2. DATABASE SCHEMA

2.1 Users Table

```
CREATE TABLE users (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(100) UNIQUE,
    phone VARCHAR(20),
    password_hash VARCHAR(255) NOT NULL,
    balance NUMERIC(15, 4) DEFAULT 0,
    total_spent NUMERIC(15, 4) DEFAULT 0,
    tier_level INTEGER DEFAULT 0,
    tier_name VARCHAR(50) DEFAULT 'Thành viên Cấp 1',
    tier_discount INTEGER DEFAULT 0,
    referral_code VARCHAR(20) UNIQUE NOT NULL,
    referred_by_code VARCHAR(20),
    referral_earnings NUMERIC(15, 4) DEFAULT 0,
    is_active BOOLEAN DEFAULT TRUE,
    is_admin BOOLEAN DEFAULT FALSE,
    email_verified BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT NOW(),
    updated_at TIMESTAMP DEFAULT NOW(),
    last_login_at TIMESTAMP
);

CREATE INDEX idx_users_username ON users(username);
```

```
CREATE INDEX idx_users_email ON users(email);
CREATE INDEX idx_users_referral_code ON users(referral_code);
```

2.2 Services Table

```
CREATE TABLE services (
    id SERIAL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    category VARCHAR(50) NOT NULL,
    type VARCHAR(50) DEFAULT 'Default',
    description TEXT,
    rate NUMERIC(10, 4) NOT NULL,
    min_quantity INTEGER NOT NULL,
    max_quantity INTEGER NOT NULL,
    provider VARCHAR(100) NOT NULL,
    provider_service_id VARCHAR(100) NOT NULL,
    is_active BOOLEAN DEFAULT TRUE,
    refill_enabled BOOLEAN DEFAULT FALSE,
    drip_feed_enabled BOOLEAN DEFAULT FALSE,
    created_at TIMESTAMP DEFAULT NOW(),
    updated_at TIMESTAMP DEFAULT NOW()
);

CREATE INDEX idx_services_category ON services(category);
CREATE INDEX idx_services_active ON services(is_active);
```

2.3 Orders Table

```
CREATE TABLE orders (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    user_id UUID REFERENCES users(id) ON DELETE CASCADE,
    service_id INTEGER REFERENCES services(id),
    service_name VARCHAR(255) NOT NULL,
    link TEXT NOT NULL,
    quantity INTEGER NOT NULL,
    base_price NUMERIC(15, 4) NOT NULL,
    discount_amount NUMERIC(15, 4) DEFAULT 0,
    final_price NUMERIC(15, 4) NOT NULL,
    status VARCHAR(50) DEFAULT 'pending',
    provider VARCHAR(100),
    bumx_order_id VARCHAR(100),
    start_count INTEGER,
    remains INTEGER,
    note TEXT,
    error_message TEXT,
    created_at TIMESTAMP DEFAULT NOW(),
    updated_at TIMESTAMP DEFAULT NOW(),
    completed_at TIMESTAMP
);

CREATE INDEX idx_orders_user_id ON orders(user_id);
```

```
CREATE INDEX idx_orders_status ON orders(status);
CREATE INDEX idx_orders_created_at ON orders(created_at DESC);
```

2.4 Transactions Table

```
CREATE TABLE transactions (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    user_id UUID REFERENCES users(id) ON DELETE CASCADE,
    order_id UUID REFERENCES orders(id),
    type VARCHAR(50) NOT NULL,
    amount NUMERIC(15, 4) NOT NULL,
    balance_before NUMERIC(15, 4),
    balance_after NUMERIC(15, 4),
    method VARCHAR(50),
    status VARCHAR(50) DEFAULT 'completed',
    transaction_code VARCHAR(50),
    sepay_transaction_id VARCHAR(100),
    description TEXT,
    created_at TIMESTAMP DEFAULT NOW()
);

CREATE INDEX idx_transactions_user_id ON transactions(user_id);
CREATE INDEX idx_transactions_type ON transactions(type);
CREATE INDEX idx_transactions_created_at ON transactions(created_at DESC);
```

2.5 Referral Earnings Table

```
CREATE TABLE referral_earnings (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    referrer_user_id UUID REFERENCES users(id),
    referred_user_id UUID REFERENCES users(id),
    order_id UUID REFERENCES orders(id),
    commission_amount NUMERIC(15, 4) NOT NULL,
    created_at TIMESTAMP DEFAULT NOW()
);

CREATE INDEX idx_referral_referrer ON referral_earnings(referrer_user_id);
```

3. BACKEND API SPECIFICATIONS

3.1 Authentication Endpoints

POST /api/auth/register

Request:

```
{
  "username": "testuser",
```

```
  "email": "test@example.com",
  "password": "password123",
  "referral_code": "HOMEM123" // optional
}
```

Response:

```
{
  "success": true,
  "user_id": "uuid",
  "message": "Đăng ký thành công"
}
```

POST /api/auth/login

Request:

```
{
  "email": "test@example.com",
  "password": "password123"
}
```

Response:

```
{
  "success": true,
  "access_token": "jwt_token",
  "token_type": "bearer",
  "user": {
    "id": "uuid",
    "username": "testuser",
    "balance": 0,
    "tier": {
      "level": 0,
      "name": "Thành viên Cấp 1",
      "icon": "⭐",
      "discount": 0
    }
  }
}
```

3.2 User Tier System

Tier Calculation Logic

```
def calculate_tier(total_spent: float) -&gt; dict:
    if total_spent &gt;= 10_000_000:
        return {
            'level': 3,
```

```
            'name': 'VIP',
            'icon': '',
            'discount': 15,
            'min_spent': 10_000_000
        }
    elif total_spent &gt;= 5_000_000:
        return {
            'level': 2,
            'name': 'Thành viên Cấp 3',
            'icon': '★★★',
            'discount': 10,
            'min_spent': 5_000_000
        }
    elif total_spent &gt;= 1_000_000:
        return {
            'level': 1,
            'name': 'Thành viên Cấp 2',
            'icon': '★★',
            'discount': 5,
            'min_spent': 1_000_000
        }
    else:
        return {
            'level': 0,
            'name': 'Thành viên Cấp 1',
            'icon': '★',
            'discount': 0,
            'min_spent': 0
        }
```

GET /api/user/tier

Response:

```
{
  "current_tier": {
    "level": 1,
    "name": "Thành viên Cấp 2",
    "icon": "★★",
    "discount": 5
  },
  "next_tier": {
    "level": 2,
    "name": "Thành viên Cấp 3",
    "required_spent": 5000000,
    "remaining": 3500000,
    "progress_percent": 30
  },
  "benefits": [
    "Giảm 5% mọi đơn hàng",
    "Hỗ trợ ưu tiên",
    "Không giới hạn số đơn/ngày"
  ]
}
```

3.3 Order Endpoints

POST /api/orders

Request:

```
{
  "service_id": 1,
  "link": "https://facebook.com/post/123",
  "quantity": 1000,
  "note": "Optional note"
}
```

Response:

```
{
  "success": true,
  "order_id": "uuid",
  "price_breakdown": {
    "base_price": 2500,
    "discount": 125,
    "final_price": 2375
  },
  "new_balance": 7625,
  "message": "Đơn hàng đã được tạo và đang xử lý"
}
```

GET /api/orders/history

Query params: `?page=1&limit=20&status=all`

Response:

```
{
  "success": true,
  "data": [
    {
      "id": "uuid",
      "service_name": "Tăng Like Facebook",
      "link": "https://...",
      "quantity": 1000,
      "price": 2375,
      "status": "completed",
      "progress": 100,
      "created_at": "2025-10-25T09:00:00"
    }
  ],
  "pagination": {
    "total": 50,
    "page": 1,
    "limit": 20,
```

```
    "total_pages": 3
  }
}
```

3.4 Payment Endpoints

POST /api/payment/sepay/create

Request:

```
{
  "amount": 100000
}
```

Response:

```
{
  "success": true,
  "transaction_id": "uuid",
  "qr_code_url": "https://img.vietqr.io/...",
  "bank_info": {
    "bank_name": "Ngân hàng ACB",
    "account_number": "123456789",
    "account_name": "NGUYEN VAN A",
    "amount": 100000,
    "content": "HOMEMMO ABC123"
  },
  "note": "Vui lòng chuyển khoản đúng nội dung"
}
```

POST /api/payment/sepay/webhook

Request from Sepay:

```
{
  "transaction_id": "sepay_txn_id",
  "account_number": "123456789",
  "amount": 100000,
  "content": "HOMEMMO ABC123",
  "transaction_date": "2025-10-26 15:30:00",
  "signature": "hash"
}
```

Processing:

1. Verify signature
2. Extract transaction code from content
3. Find pending transaction

4. Update balance
5. Apply bonus if applicable
6. Send Telegram notification

3.5 Referral Endpoints

GET /api/referral/stats

Response:

```
{
  "referral_code": "HOMEM123",
  "referral_link": "https://social.homemmo.store/?ref=HOMEM123",
  "total_referrals": 5,
  "total_earnings": 125000,
  "pending_earnings": 5000,
  "recent_earnings": [
    {
      "user": "user123",
      "order_id": "uuid",
      "amount": 5000,
      "date": "2025-10-25"
    }
  ]
}
```

4. FRONTEND FEATURES CHECKLIST

4.1 User Dashboard

- [x] Balance display (top header)
- [x] Tier badge (⭐⭐⭐)
- [x] Quick stats (Total orders, Total spent)
- [x] Recent orders (5 rows)
- [ ] Tier progress bar
- [ ] Notification bell with badge

4.2 Service Catalog

- [x] Sidebar categories (Facebook, TikTok, etc.)
- [x] Service cards with pricing
- [x] Search/Filter
- [ ] Favorite services (star icon)
- [ ] Service comparison

4.3 Order Form

- [x] Service dropdown
- [x] Link input with validation
- [x] Quantity input (min/max)
- [x] Price calculator with tier discount
- [x] Balance check
- [ ] Drip-feed options
- [ ] Custom comments input
- [ ] Mass order (multiple links)

4.4 Payment

- [x] Deposit modal
- [x] Amount suggestions
- [x] QR code display
- [x] Bank info copy buttons
- [ ] Payment history with filters
- [ ] Auto-refresh status

4.5 Profile

- [x] Edit email, phone
- [x] Change password
- [x] View tier info
- [ ] API key generation (for resellers)
- [ ] Notification settings

5. DEPLOYMENT CHECKLIST

5.1 Server Setup

- [ ] CentOS 7 fully updated
- [ ] Python 3.6+ installed
- [ ] PostgreSQL 15 installed & configured
- [ ] Redis 7 installed & running
- [ ] Nginx installed & configured
- [ ] SSL certificate (Let's Encrypt)
- [ ] Firewall rules (ports 80, 443, 8000)

5.2 Application Setup

- [ ] Clone/upload code to `/home/homemmo/smm-panel`
- [ ] Create virtual environment
- [ ] Install Python dependencies
- [ ] Configure `.env` file
- [ ] Run database migrations
- [ ] Setup systemd service for FastAPI
- [ ] Configure Nginx reverse proxy
- [ ] Setup cron jobs (order status sync)

5.3 Testing

- [ ] User registration works
- [ ] Login/JWT authentication works
- [ ] Order creation works
- [ ] Payment webhook works
- [ ] Telegram notifications work
- [ ] Admin dashboard accessible
- [ ] All API endpoints return correct data

6. MAINTENANCE GUIDE

6.1 Daily Tasks

- Check Telegram notifications
- Monitor order processing
- Check payment deposits
- Review error logs

6.2 Weekly Tasks

- Backup database
- Review user activity logs
- Sync services from BUMX
- Update tier calculations

6.3 Monthly Tasks

- Generate revenue reports
- Review and optimize slow queries
- Update SSL certificates (auto)
- Security audit

7. TROUBLESHOOTING

Common Issues

Issue: Orders stuck in "pending"
Solution: Check BUMX API connectivity, review error logs, manually retry

Issue: Payment not auto-credited
Solution: Check Sepay webhook logs, verify transaction code matching

Issue: Balance negative
Solution: Database transaction rollback, manual adjustment

Issue: High database load
Solution: Add indexes, enable query caching, optimize heavy queries

8. API RATE LIMITS

BUMX API

- Max: 60 requests/minute
- Implement exponential backoff
- Queue orders if rate limit hit

Sepay API

- No documented limit
- Monitor for errors

Telegram Bot

- Max: 30 messages/second
- Batch notifications

9. SECURITY CHECKLIST

- [ ] All passwords hashed (bcrypt)
- [ ] JWT tokens with expiration
- [ ] SQL injection protection (ORM)
- [ ] XSS protection (input sanitization)
- [ ] CORS configured correctly
- [ ] Rate limiting on API endpoints
- [ ] HTTPS only (no HTTP)
- [ ] Admin IP whitelist (optional)
- [ ] Database backups encrypted
- [ ] Sensitive config in `.env` (gitignored)

10. MONITORING & LOGGING

Application Logs

```
/var/log/smmpanel/app.log
/var/log/smmpanel/error.log
/var/log/smmpanel/api.log
```

Nginx Logs

```
/var/log/nginx/access.log
/var/log/nginx/error.log
```

Database Logs

```
/var/log/postgresql/postgresql-15-main.log
```

Monitoring Tools (Optional)

- Prometheus + Grafana
- Sentry for error tracking
- Uptime Robot for availability
- DataDog APM

CONCLUSION

This specification provides a complete blueprint for implementing Phase 1 of the SMM Panel. All core features are defined with clear database schema, API specifications, and implementation guidelines.

Next Steps:

1. Review this document with development team
2. Setup development environment
3. Implement features in order of priority
4. Test thoroughly before production deploy
5. Monitor and iterate based on user feedback

Contact:

- Developer: [Your Name]
- Email: [Your Email]

- Project: HOMEMMO SMM Panel
- Version: 1.0
- Date: October 26, 2025