**SBS System Detailed Flow Map: All Webhook Connections**

**Webhook Endpoint Mapping**

**Incoming Webhooks (Entry Points)**

| **Webhook Endpoint** | **Connected Workflow** |
| --- | --- |
| /user-signup | init\_user\_setup.json |
| /check-sbs-achievements | achievement\_unlock.json |
| /create-integrated-sbs-system | sbs\_integrated\_system\_builder.json |
| /sbs-skill-progression | sbs\_skill\_progression.json |
| /sbs-system-created | sbs\_spawner.json |
| /sbs-system-update | sbs\_orchestrator.json |
| /sbs-pg-notify | sbs\_pg\_listener.json |
| /complete-task | quest\_engine.json |
| /habit-checkin | habit\_checkin.json |
| /shop/purchase | shop\_check\_flow.json |
| /prestige-eligibility | prestige\_calc.json |
| /bad-habit-battle | damage\_calc.json |

**Scheduled Triggers**

| **Schedule / Trigger** | **Connected Workflows** |
| --- | --- |
| Daily at 9:00 AM | sbs\_routine\_engine.json |
| Daily Intervals | cron\_manager.json, event\_seeder.json, ai\_missions.json |
| Telegram Bot (Real-time) | sbs\_telegram\_bot.json |

**Detailed Inter-Workflow Calls**

**1. User Onboarding Chain**

**Flow Description:**

1. **User Signup**  
   → init\_user\_setup.json (Webhook: /user-signup)
   * Creates character in PostgreSQL
   * Generates starting habits, skills, and projects
   * Calls integrated SBS system builder
2. **Integrated System Creation**  
   → sbs\_integrated\_system\_builder.json (Webhook: /create-integrated-sbs-system)
   * Creates SBS system
   * Auto-generates category-based skills and habits
   * Returns system details
3. **Routine Creation**  
   → Routine Creation Handler (implied workflow)
   * Creates initial SBS routines
   * Completes onboarding sequence

**Result:**  
User setup complete, character initialized, first SBS system generated.

**2. Daily Routine Flow**

**Flow Description:**

1. **Daily Schedule Trigger (9:00 AM)**  
   → sbs\_routine\_engine.json
   * Queries active routines
   * Sends Telegram reminders with /complete [routine\_id]
   * Logs reminder events
2. **User Interaction**  
   → sbs\_telegram\_bot.json (Telegram Bot Trigger)
   * Processes /complete [routine\_id]
   * Logs completion
   * Triggers skill progression via:  
     POST /sbs-skill-progression → sbs\_skill\_progression.json
3. **Skill Progression**  
   → sbs\_skill\_progression.json
   * Maps system category to relevant skills
   * Awards XP for routine\_completion
   * Updates skill levels and logs event
4. **Response**
   * Telegram bot confirms: “Routine completed – XP gained!”

**3. System Advancement Flow**

**Flow Description:**

1. **User Command:** /advance [system\_id]  
   → sbs\_telegram\_bot.json
   * Processes system advancement
   * Calls orchestrator via:  
     POST /sbs-system-update → sbs\_orchestrator.json
2. **System Orchestration**  
   → sbs\_orchestrator.json
   * Retrieves pending step
   * Executes stage-specific handlers:
     + Design → Generate Canvas
     + Build → Create Infrastructure
     + Automate → Setup Triggers
     + Review → Schedule Review
   * Marks step complete and updates system stage
3. **Skill Progression Bonus**  
   → POST /sbs-skill-progression → sbs\_skill\_progression.json
   * Awards XP for stage\_advancement
   * Applies higher XP bonus
4. **Response**
   * Telegram bot confirms: “System advanced – XP bonus applied!”

**4. Database Event Flow**

**Flow Description:**

1. **PostgreSQL Notification (Insert/Update on Systems Table)**  
   → sbs\_pg\_listener.json (Webhook: /sbs-pg-notify)
   * Parses notification payload
   * Routes event by type:
     + system\_created → sbs\_spawner.json
     + system\_updated → sbs\_orchestrator.json
2. **System Created**  
   → sbs\_spawner.json (Webhook: /sbs-system-created)
   * Generates system steps
   * Creates initial routines
   * Sets stage to “define”
3. **System Updated**  
   → sbs\_orchestrator.json (Webhook: /sbs-system-update)
   * Manages system lifecycle transitions

**5. Achievement Checking Flow**

**Flow Description:**

1. **Trigger:** Skill progression, system advancement, or habit streak  
   → achievement\_unlock.json (Webhook: /check-sbs-achievements)
   * Fetches character, skill, habit, wealth, and system stats
   * Evaluates SBS-focused criteria:
     + System Building Achievements
     + Generated Skills Achievements
     + Habit Achievements
     + Integrated SBS Achievements
     + Stage Progression Achievements
   * Inserts new achievements
   * Awards XP and coin rewards
   * Returns unlocked achievements

**Complete User Activity Tracking**

**New User – First Day Journey**

**Step-by-Step:**

1. **User Signup** → /user-signup → init\_user\_setup.json
   * Creates character (Level 1)
   * Generates starter habits, skills, and projects
   * Calls /create-integrated-sbs-system → sbs\_integrated\_system\_builder.json
     + Creates “Personal Growth System”
     + Generates skills: *learning*, *mindfulness*
     + Generates habits: *reading*, *meditation*
     + Logs system creation
   * Creates initial routines
   * Returns welcome message
2. **Database Trigger:**  
   → /sbs-pg-notify → sbs\_pg\_listener.json
   * Routes to /sbs-system-created → sbs\_spawner.json
   * Creates system steps and routines
   * Initializes stage to “define”

**Result:**  
User is fully onboarded with an operational SBS system and daily routines.

**Daily User Interaction Cycle**

**Morning (9:00 AM):**

* **Schedule Trigger** → sbs\_routine\_engine.json
  + Sends daily Telegram reminders
  + Example: “Daily Routine: Reading – 30 minutes”

**User Action:**

* /complete 123 → sbs\_telegram\_bot.json
  + Logs routine completion
  + Calls /sbs-skill-progression → sbs\_skill\_progression.json
    - Awards XP (Learning +15 XP, Mindfulness +10 XP)
    - Confirms “Routine Completed – +25 XP!”

**Weekly Action:**

* /advance 456 → sbs\_telegram\_bot.json
  + Calls /sbs-system-update → sbs\_orchestrator.json
  + Advances stage to “Design”
  + Generates system canvas
  + Calls /sbs-skill-progression for stage\_advancement
    - Awards +100 XP total bonus
  + Responds: “System Advanced – +100 XP Bonus!”

**Achievement Unlock Flow**

**Trigger:** Skill progression, system completion, or routine streak

**Process:**

* achievement\_unlock.json retrieves user data and checks:
  + 2+ SBS-generated skills → “SBS Skill Generator”
  + 2+ SBS-generated habits → “SBS Habit Creator”
  + Reached Design stage → “SBS Design Thinker”
* Inserts achievements into PostgreSQL
* Rewards: +450 XP, +300 coins
* Returns confirmation with list of new achievements

**Supporting Game Mechanics**

**Habit System Integration**

* /habit-checkin → habit\_checkin.json
  + Updates streaks
  + Awards XP and coins
  + May trigger achievements
  + Updates character stats

**Quest System Integration**

* /complete-task → quest\_engine.json
  + Marks task complete
  + Updates project progress
  + Awards quest rewards
  + Checks for project completion

**Shop System**

* /shop/purchase → shop\_check\_flow.json
  + Validates purchase eligibility
  + Deducts coins
  + Adds item to inventory
  + Updates character resources

**Complete Integration Summary**

**The SBS System forms a unified feedback loop:**

1. SBS systems auto-generate skills and habits.
2. Daily routines drive consistent skill progression.
3. System advancement grants stage bonuses and XP rewards.
4. Achievements reinforce integrated SBS growth.
5. Database events automate orchestration.
6. Telegram bot ensures interactive engagement.
7. Scheduled tasks maintain ongoing player momentum.

**Key Integration Points:**

* init\_user\_setup.json → sbs\_integrated\_system\_builder.json (Onboarding)
* sbs\_telegram\_bot.json → sbs\_skill\_progression.json (Progression)
* sbs\_pg\_listener.json → sbs\_spawner.json and sbs\_orchestrator.json (Automation)
* All major workflows → achievement\_unlock.json (Gamification Layer)

**Conclusion:**  
The SBS ecosystem functions as an interconnected gamified architecture that transforms user activity into a dynamic, automated life-management RPG.  
Each workflow contributes to habit formation, skill mastery, and achievement progression through a seamless orchestration of systems, data triggers, and player engagement loops.