# **SBS System Flow Analysis: Complete User Journey**

## ****Overview of All Workflows****

Based on the analysis of all JSON workflows, this document outlines the complete user flow through the SBS system.

## ****1. USER ONBOARDING FLOW****

### **Workflow: init\_user\_setup.json (Entry Point)**

* **Webhook:** /user-signup (POST)
* **Purpose:** New user registration and initial system setup

**Flow Steps:**

1. **Webhook Trigger** – Receives user signup data
2. **Validate Input** – Checks user\_id and class are provided
3. **Create Character** – Inserts into PostgreSQL characters table
4. **Create Starting Habits** – Generates initial good/bad habits based on class
5. **Create Starting Skills** – Creates skills depending on class type
6. **Create Starting Projects** – Initializes 2–3 starter projects
7. **Create Integrated SBS System** – Calls sbs\_integrated\_system\_builder.json
8. **Create SBS Routines** – Sets up initial routines
9. **Log Creation Event** – Records user setup
10. **Send Welcome Response** – Returns success message with character details

**External Calls:**

* http://localhost:5678/webhook/create-integrated-sbs-system
* http://localhost:5678/webhook/create-sbs-routines

## ****2. SBS SYSTEM BUILDING FLOW****

### **Workflow: sbs\_integrated\_system\_builder.json (Core System Creation)**

* **Webhook:** /create-integrated-sbs-system (POST)
* **Purpose:** Create SBS systems that auto-generate skills and habits

**Flow Steps:**

1. Webhook Trigger → Receives creation request
2. Validate System Data → Check required fields
3. Get Character Info → Fetch by user\_id
4. Create SBS System → Insert into systems table
5. Map Category to Skills → Determine target skills
6. Check Skill Exists → Verify if already created
7. Create Target Skills → Auto-generate if needed
8. Generate System Habits → Create habits based on system type
9. Log System Creation → Record event
10. Return System Details → Response with generated content

**Auto-Generated Content Examples:**

* **Health:** Fitness, Nutrition skills + Exercise, Meal Prep habits
* **Work:** Productivity, Time Management skills + Focus, Planning habits
* **Personal Development:** Learning, Mindfulness skills + Reading, Meditation habits
* **Finance:** Budgeting, Investing skills + Expense Tracking, Saving habits

### **Workflow: sbs\_spawner.json (System Spawning Logic)**

* **Webhook:** /sbs-system-created (POST)
* **Purpose:** Handle new system creation and spawn routines

**Flow Steps:**

1. Webhook Trigger → System creation notification
2. Validate System Data → Ensure system\_id exists
3. Create System Steps → Generate development steps
4. Create Initial Routines → Spawn category-appropriate routines
5. Set System Stage → Initialize to “define” stage
6. Log Spawning Event → Record system spawn
7. Return Spawn Results → Confirm readiness

## ****3. SKILL PROGRESSION FLOW****

### **Workflow: sbs\_skill\_progression.json (Skill Development Engine)**

* **Webhook:** /sbs-skill-progression (POST)
* **Purpose:** Award skill XP based on SBS progression

**Flow Steps:**

1. Webhook Trigger → Receives progression event
2. Validate Input → Check system\_id and progression type
3. Get System Details → Retrieve system info
4. Map Progression Type → Determine XP amounts
   * Routine Completion → 10–25 XP
   * Stage Advancement → 50–100 XP
   * Milestone Achievement → 100–200 XP
5. Calculate Skill XP → Apply bonuses
6. Update Character Skills → Award XP
7. Check Level Advancement → Detect level up
8. Award Bonus XP → For advancement
9. Log Progression Event → Record XP gain
10. Return Progression Results → Confirm outcome

**Called By:**

* sbs\_telegram\_bot.json (routine completion)
* sbs\_telegram\_bot.json (system advancement)
* System milestone triggers

## ****4. DAILY ROUTINE FLOW****

### **Workflow: sbs\_routine\_engine.json (Daily Automation)**

* **Trigger:** Scheduled (Daily at 9 AM)
* **Purpose:** Send daily routine reminders via Telegram

**Flow Steps:**

1. Schedule Trigger → Executes daily
2. Get Active Routines → Query for today’s tasks
3. Validate Routines → Ensure routines exist
4. Split Into Items → Process individually
5. Send Routine Reminder → Telegram message with /complete option
6. Log Reminder → Record event
7. Send Summary → Total reminders sent

### **Workflow: sbs\_telegram\_bot.json (User Interaction Hub)**

* **Trigger:** Telegram Bot Commands
* **Purpose:** Handle user commands and trigger system actions

**Flow Steps:**

1. Telegram Trigger → Receives user messages
2. **If /complete [routine\_id]:**
   * Complete Routine
   * Trigger Skill Progression (sbs\_skill\_progression.json)
   * Send Completion Response
3. **If /skip [routine\_id]:**
   * Skip Routine
   * Log skip event
   * Send Response
4. **If /advance [system\_id]:**
   * Advance System Stage
   * Trigger Skill Progression
   * Send Advancement Response
5. **If /status:**
   * Get Active Systems
   * Format and Send Status Update

**External Calls:**

* http://localhost:5678/webhook/sbs-skill-progression
* sbs-system-update

## ****5. ACHIEVEMENT FLOW****

### **Workflow: achievement\_unlock.json (SBS Achievement System)**

* **Webhook:** /check-sbs-achievements (POST)
* **Purpose:** Evaluate and unlock achievements

**Flow Steps:**

1. Webhook Trigger
2. Fetch Character, Skill, Habit, Wealth, and System Stats
3. Check Existing Achievements
4. Evaluate Achievement Criteria
5. Insert New Achievements
6. Apply Rewards (XP, Coins)
7. Log and Respond with Results

## ****6. HABIT & QUEST FLOWS****

### **Workflow: habit\_checkin.json (Habit Management)**

* **Webhook:** /habit-checkin (POST)
* **Purpose:** Handle habit check-ins and streaks

**Flow Steps:**

1. Trigger → Habit check-in request
2. Validate Habit
3. Update Habit Status
4. Calculate Streak
5. Award XP/Coins
6. Check Achievement Triggers
7. Return Results

### **Workflow: quest\_engine.json (Project Management)**

* **Webhook:** /complete-task (POST)
* **Purpose:** Manage project task completion

**Flow Steps:**

1. Trigger → Task completion
2. Validate Task
3. Update Status
4. Calculate Progress
5. Award Rewards
6. Check Project Completion
7. Return Results

## ****7. SYSTEM ORCHESTRATION****

### **Workflow: sbs\_orchestrator.json (System Lifecycle Manager)**

* **Webhook:** /sbs-system-update (POST)
* **Purpose:** Manage stage transitions and automation

**Flow Steps:**

1. Webhook Trigger
2. Validate System
3. Determine Stage Transition
4. Update Stage
5. Trigger Stage Actions
6. Update Routines
7. Log System Update
8. Return Results

### **Workflow: sbs\_pg\_listener.json (Database Event Listener)**

* **Webhook:** /sbs-pg-notify (POST)
* **Purpose:** Listen for PostgreSQL notifications and trigger workflows

**Flow Routes:**

* **System Created →** Calls sbs\_spawner.json
* **System Updated →** Calls sbs\_orchestrator.json
* **Routine Completed →** Calls sbs\_skill\_progression.json

## ****8. SUPPORTING FLOWS****

### **Workflow: shop\_check\_flow.json (Shop System)**

* **Webhook:** /shop/purchase (POST)
* **Purpose:** Handle purchases and inventory

### **Workflow: prestige\_calc.json (Prestige System)**

* **Webhook:** /prestige-eligibility (POST)
* **Purpose:** Calculate prestige options and rewards

### **Workflow: damage\_calc.json (Bad Habit Battle)**

* **Webhook:** /bad-habit-battle (POST)
* **Purpose:** Manage battles against bad habits

## ****COMPLETE USER JOURNEY EXAMPLE****

### **Day 1: User Onboarding**

1. User signs up → init\_user\_setup.json
2. Character created → Calls sbs\_integrated\_system\_builder.json
3. “Personal Growth System” created with auto-skills and habits
4. System spawning → sbs\_spawner.json creates routines
5. User receives welcome message with first system ready

### **Daily Operation**

1. 9 AM → sbs\_routine\_engine.json sends reminders
2. User completes routine via /complete 123 → sbs\_telegram\_bot.json
3. Triggers sbs\_skill\_progression.json for XP gain
4. Achievement check via achievement\_unlock.json
5. System progression via sbs\_orchestrator.json

### **System Evolution**

1. User advances system via /advance 456
2. Bonus XP awarded
3. System automates more routines
4. Cross-domain skill development
5. Unlocks major achievements (“Life Systems Master”, etc.)

## ****INTEGRATION POINTS****

### **Primary Integration Flows**

* init\_user\_setup.json → sbs\_integrated\_system\_builder.json → sbs\_spawner.json
* sbs\_telegram\_bot.json → sbs\_skill\_progression.json → achievement\_unlock.json
* sbs\_routine\_engine.json → sbs\_telegram\_bot.json → sbs\_skill\_progression.json
* sbs\_pg\_listener.json → sbs\_orchestrator.json / sbs\_spawner.json

### **Data Flow Summary**

* PostgreSQL → Triggers → n8n Workflows → API Calls → Database Updates
* User Actions (via Telegram) → Workflow Triggers → Skill Progression → Achievement Checks
* Scheduled Tasks → Daily Routines → User Interaction → System Advancement

**Conclusion:**  
The SBS system forms an interconnected gamified ecosystem. Each workflow supports an evolving loop of habit-building, system progression, and skill mastery—turning user productivity into an RPG-style growth experience.