# Quest Feature Schema and API Updates

## Overview

This document outlines the specific schema and API updates required to integrate the refined Quest system with the existing XP system in the Setarcos app.

## Database Schema Updates

### 1. Removed Tables

* ~~user\_wisdom\_xp~~ - Not needed as we'll use the existing XP system

### 2. Modified Tables

#### quests Table

ALTER TABLE quests

ADD COLUMN skill\_tree\_structure JSONB DEFAULT '{}',

ADD COLUMN xp\_threshold INTEGER DEFAULT 0,

ADD COLUMN accessibility\_options JSONB DEFAULT '{}';

* skill\_tree\_structure: JSON structure defining the quest skill tree layout
* xp\_threshold: Standard XP amount required to unlock this quest
* accessibility\_options: JSON object with accessibility settings for this quest

#### quest\_steps Table

ALTER TABLE quest\_steps

ADD COLUMN node\_position JSONB DEFAULT '{}',

ADD COLUMN connected\_nodes JSONB DEFAULT '[]',

ADD COLUMN concept\_links JSONB DEFAULT '[]',

ADD COLUMN is\_challenge\_node BOOLEAN DEFAULT FALSE,

ADD COLUMN bonus\_xp INTEGER DEFAULT 0;

* node\_position: Coordinates for positioning in the skill tree
* connected\_nodes: Array of step IDs this node connects to
* concept\_links: Array of concept IDs linked to this step
* is\_challenge\_node: Whether this is a special challenge node
* bonus\_xp: Additional XP awarded for challenge nodes

### 3. New Tables

#### quest\_badges Table

CREATE TABLE quest\_badges (

id UUID PRIMARY KEY DEFAULT uuid\_generate\_v4(),

name VARCHAR(100) NOT NULL,

description TEXT NOT NULL,

xp\_threshold INTEGER NOT NULL,

quest\_category VARCHAR(50) NOT NULL,

icon\_url VARCHAR(255) NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

* Stores badge definitions for quest achievements
* Uses standard XP thresholds instead of specialized Wisdom XP

#### user\_quest\_badges Table

CREATE TABLE user\_quest\_badges (

id UUID PRIMARY KEY DEFAULT uuid\_generate\_v4(),

user\_id UUID NOT NULL REFERENCES users(id),

badge\_id UUID NOT NULL REFERENCES quest\_badges(id),

earned\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UNIQUE(user\_id, badge\_id)

);

* Tracks which badges users have earned

## API Updates

### 1. Modified Endpoints

#### POST /api/v1/quests/{id}/start

// Response

{

"success": true,

"user\_quest\_id": "750e8400-e29b-41d4-a716-446655440000",

"quest\_id": "550e8400-e29b-41d4-a716-446655440000",

"status": "in\_progress",

"current\_step": 1,

"xp\_earned": 5

}

* Uses standard xp\_earned field instead of Wisdom XP

#### POST /api/v1/quests/{id}/steps/{step\_id}/complete

// Response

{

"success": true,

"step\_id": "650e8400-e29b-41d4-a716-446655440001",

"status": "completed",

"xp\_earned": 20,

"next\_step": {

"id": "650e8400-e29b-41d4-a716-446655440002",

"step\_number": 2,

"title": "The Four Cardinal Virtues",

"content": "...",

"step\_type": "multiple\_choice",

"interaction\_data": {

"question": "Which of the following is NOT one of the four cardinal virtues in Stoicism?",

"options": ["Wisdom", "Justice", "Courage", "Happiness", "Temperance"],

"correct\_answer": 3

},

"xp\_reward": 25

},

"quest\_progress": {

"completed\_steps": 1,

"total\_steps": 5,

"percentage": 20

},

"total\_xp": 1250,

"xp\_to\_next\_level": 750

}

* Added total\_xp and xp\_to\_next\_level fields to integrate with standard XP system

### 2. New Endpoints

#### GET /api/v1/quests/skill-tree

// Response

{

"quests": [

{

"id": "550e8400-e29b-41d4-a716-446655440000",

"title": "Introduction to Stoicism",

"description": "...",

"xp\_threshold": 0,

"user\_status": "completed",

"position": {"x": 100, "y": 200},

"connected\_quests": ["550e8400-e29b-41d4-a716-446655440001"]

},

{

"id": "550e8400-e29b-41d4-a716-446655440001",

"title": "Advanced Stoic Practices",

"description": "...",

"xp\_threshold": 500,

"user\_status": "locked",

"position": {"x": 300, "y": 200},

"connected\_quests": []

}

],

"user\_xp": 350,

"next\_unlock": {

"quest\_id": "550e8400-e29b-41d4-a716-446655440001",

"xp\_required": 500,

"xp\_remaining": 150

}

}

* Uses standard user\_xp field instead of Wisdom XP
* Shows XP thresholds for unlocking advanced quests

## Integration with Existing XP System

### XP Service Integration

# In quest\_service.py

from xp\_service import XPService

class QuestService:

def \_\_init\_\_(self, db):

self.db = db

self.xp\_service = XPService(db)

def start\_quest(self, user\_id, quest\_id):

# Create user\_quest record

user\_quest\_id = self.\_create\_user\_quest(user\_id, quest\_id)

# Award XP for starting quest

xp\_earned = self.xp\_service.award\_xp(

user\_id=user\_id,

action\_type="quest\_start",

context={"quest\_id": quest\_id},

amount=5

)

return {

"success": True,

"user\_quest\_id": user\_quest\_id,

"quest\_id": quest\_id,

"status": "in\_progress",

"current\_step": 1,

"xp\_earned": xp\_earned

}

def complete\_quest\_step(self, user\_id, quest\_id, step\_id, user\_response):

# Update user\_quest\_step record

self.\_update\_user\_quest\_step(user\_id, quest\_id, step\_id, user\_response)

# Get step details

step = self.\_get\_step(step\_id)

# Award XP for completing step

xp\_earned = self.xp\_service.award\_xp(

user\_id=user\_id,

action\_type="quest\_step\_complete",

context={"quest\_id": quest\_id, "step\_id": step\_id},

amount=step["xp\_reward"]

)

# Check for quest completion

if self.\_is\_quest\_completed(user\_id, quest\_id):

quest = self.\_get\_quest(quest\_id)

self.xp\_service.award\_xp(

user\_id=user\_id,

action\_type="quest\_complete",

context={"quest\_id": quest\_id},

amount=quest["xp\_reward"]

)

# Get user's total XP and XP to next level

user\_xp\_info = self.xp\_service.get\_user\_xp\_info(user\_id)

return {

"success": True,

"step\_id": step\_id,

"status": "completed",

"xp\_earned": xp\_earned,

"next\_step": self.\_get\_next\_step(quest\_id, step\_id),

"quest\_progress": self.\_get\_quest\_progress(user\_id, quest\_id),

"total\_xp": user\_xp\_info["total\_xp"],

"xp\_to\_next\_level": user\_xp\_info["xp\_to\_next\_level"]

}

### Celery Task Integration

@celery\_app.task(name="quests.check\_unlockable\_quests")

def check\_unlockable\_quests(user\_id):

"""Check if user has enough XP to unlock new quests"""

db = get\_db\_connection()

quest\_service = QuestService(db)

xp\_service = XPService(db)

# Get user's current XP

user\_xp = xp\_service.get\_user\_total\_xp(user\_id)

# Get quests with XP thresholds

locked\_quests = quest\_service.get\_locked\_quests\_for\_user(user\_id)

# Check for newly unlockable quests

newly\_unlocked = []

for quest in locked\_quests:

if user\_xp >= quest["xp\_threshold"]:

quest\_service.unlock\_quest\_for\_user(user\_id, quest["id"])

newly\_unlocked.append(quest["id"])

# Send notifications for newly unlocked quests

if newly\_unlocked:

notification\_service = NotificationService(db)

for quest\_id in newly\_unlocked:

quest = quest\_service.get\_quest(quest\_id)

notification\_service.send\_notification(

user\_id=user\_id,

notification\_type="quest\_unlocked",

title=f"New Quest Unlocked: {quest['title']}",

body=f"You've unlocked a new quest: {quest['title']}",

data={"quest\_id": quest\_id}

)

return newly\_unlocked