**# Mingus Onboarding Progress Tracking Template**

**# Complete system for tracking user onboarding completion status**

**from datetime import datetime, timedelta**

**from typing import Dict, List, Optional, Any**

**import json**

**class OnboardingProgressTracker:**

**"""**

**Comprehensive onboarding progress tracking for Mingus users**

**Tracks completion of all onboarding steps and provides analytics**

**"""**

**def \_\_init\_\_(self, user\_id: str):**

**self.user\_id = user\_id**

**self.onboarding\_steps = self.\_define\_onboarding\_steps()**

**def \_define\_onboarding\_steps(self) -> Dict[str, Dict]:**

**"""Define the complete onboarding flow steps"""**

**return {**

**"welcome": {**

**"step\_number": 1,**

**"title": "Welcome & Introduction",**

**"description": "Introduction to Mingus and onboarding process explanation",**

**"required": True,**

**"estimated\_time": 2, # minutes**

**"completion\_criteria": ["viewed\_welcome\_screen", "clicked\_get\_started"],**

**"weight": 5 # Percentage of total progress**

**},**

**"profile\_setup": {**

**"step\_number": 2,**

**"title": "Basic Profile Setup",**

**"description": "Name, age, location, employment status",**

**"required": True,**

**"estimated\_time": 3,**

**"completion\_criteria": ["name\_provided", "age\_provided", "location\_provided", "employment\_status\_provided"],**

**"weight": 10**

**},**

**"financial\_profile": {**

**"step\_number": 3,**

**"title": "Financial Profile",**

**"description": "Income sources, monthly expenses, current debt, savings",**

**"required": True,**

**"estimated\_time": 5,**

**"completion\_criteria": ["income\_provided", "expenses\_provided", "debt\_info\_provided", "savings\_info\_provided"],**

**"weight": 20**

**},**

**"goals\_setup": {**

**"step\_number": 4,**

**"title": "Financial Goals Setting",**

**"description": "Short-term and long-term financial objectives",**

**"required": True,**

**"estimated\_time": 4,**

**"completion\_criteria": ["short\_term\_goals\_set", "long\_term\_goals\_set", "timeline\_provided"],**

**"weight": 15**

**},**

**"health\_assessment": {**

**"step\_number": 5,**

**"title": "Health & Wellness Assessment",**

**"description": "Initial stress levels, physical activity, mental health baseline",**

**"required": True,**

**"estimated\_time": 3,**

**"completion\_criteria": ["stress\_level\_provided", "activity\_level\_provided", "wellness\_baseline\_set"],**

**"weight": 15**

**},**

**"data\_input": {**

**"step\_number": 6,**

**"title": "Financial Data Integration",**

**"description": "Connect accounts or manual data entry",**

**"required": True,**

**"estimated\_time": 7,**

**"completion\_criteria": ["data\_source\_selected", "accounts\_connected\_or\_manual\_complete", "expense\_categories\_set"],**

**"weight": 20**

**},**

**"preferences": {**

**"step\_number": 7,**

**"title": "Preferences & Settings",**

**"description": "Risk tolerance, notification preferences, privacy settings",**

**"required": False,**

**"estimated\_time": 3,**

**"completion\_criteria": ["risk\_tolerance\_set", "notifications\_configured", "privacy\_settings\_set"],**

**"weight": 10**

**},**

**"dashboard\_intro": {**

**"step\_number": 8,**

**"title": "Dashboard Introduction",**

**"description": "Tour of main features and dashboard walkthrough",**

**"required": True,**

**"estimated\_time": 3,**

**"completion\_criteria": ["dashboard\_tour\_completed", "first\_forecast\_viewed", "help\_resources\_acknowledged"],**

**"weight": 5**

**}**

**}**

**class OnboardingDatabase:**

**"""Database operations for onboarding progress tracking"""**

**@staticmethod**

**def create\_progress\_tables():**

**"""SQL for creating onboarding progress tracking tables"""**

**return """**

**-- Main onboarding progress tracking table**

**CREATE TABLE IF NOT EXISTS onboarding\_progress (**

**id uuid PRIMARY KEY DEFAULT gen\_random\_uuid(),**

**user\_id uuid REFERENCES auth.users(id) NOT NULL,**

**current\_step text NOT NULL,**

**completed\_steps text[] DEFAULT '{}',**

**step\_completion\_data jsonb DEFAULT '{}',**

**progress\_percentage integer DEFAULT 0 CHECK (progress\_percentage BETWEEN 0 AND 100),**

**is\_completed boolean DEFAULT false,**

**completion\_date timestamp,**

**created\_at timestamp DEFAULT now(),**

**updated\_at timestamp DEFAULT now(),**

**-- Index for efficient queries**

**UNIQUE(user\_id)**

**);**

**-- Detailed step completion tracking**

**CREATE TABLE IF NOT EXISTS onboarding\_step\_completions (**

**id uuid PRIMARY KEY DEFAULT gen\_random\_uuid(),**

**user\_id uuid REFERENCES auth.users(id) NOT NULL,**

**step\_name text NOT NULL,**

**step\_number integer NOT NULL,**

**completion\_criteria jsonb NOT NULL,**

**completed\_at timestamp DEFAULT now(),**

**time\_spent\_minutes integer,**

**completion\_data jsonb DEFAULT '{}',**

**-- Prevent duplicate completions**

**UNIQUE(user\_id, step\_name)**

**);**

**-- User onboarding session tracking**

**CREATE TABLE IF NOT EXISTS onboarding\_sessions (**

**id uuid PRIMARY KEY DEFAULT gen\_random\_uuid(),**

**user\_id uuid REFERENCES auth.users(id) NOT NULL,**

**session\_start timestamp DEFAULT now(),**

**session\_end timestamp,**

**steps\_completed\_in\_session text[],**

**total\_time\_minutes integer,**

**abandoned boolean DEFAULT false,**

**completion\_trigger text, -- 'completed', 'abandoned', 'interrupted'**

**device\_info jsonb**

**);**

**-- Create indexes for performance**

**CREATE INDEX IF NOT EXISTS idx\_onboarding\_progress\_user\_id ON onboarding\_progress(user\_id);**

**CREATE INDEX IF NOT EXISTS idx\_onboarding\_step\_completions\_user\_id ON onboarding\_step\_completions(user\_id);**

**CREATE INDEX IF NOT EXISTS idx\_onboarding\_sessions\_user\_id ON onboarding\_sessions(user\_id);**

**"""**

**class OnboardingProgressManager:**

**"""Main class for managing onboarding progress"""**

**def \_\_init\_\_(self, user\_id: str):**

**self.user\_id = user\_id**

**self.tracker = OnboardingProgressTracker(user\_id)**

**def initialize\_onboarding(self) -> Dict[str, Any]:**

**"""Initialize onboarding for a new user"""**

**progress\_data = {**

**'user\_id': self.user\_id,**

**'current\_step': 'welcome',**

**'completed\_steps': [],**

**'progress\_percentage': 0,**

**'is\_completed': False,**

**'step\_completion\_data': {},**

**'estimated\_total\_time': sum(step['estimated\_time'] for step in self.tracker.onboarding\_steps.values()),**

**'started\_at': datetime.now().isoformat()**

**}**

**# Insert into database**

**# INSERT INTO onboarding\_progress (user\_id, current\_step, progress\_percentage, step\_completion\_data)**

**# VALUES (%(user\_id)s, %(current\_step)s, %(progress\_percentage)s, %(step\_completion\_data)s)**

**return progress\_data**

**def complete\_step(self, step\_name: str, completion\_data: Dict[str, Any]) -> Dict[str, Any]:**

**"""Mark a step as completed and update progress"""**

**if step\_name not in self.tracker.onboarding\_steps:**

**raise ValueError(f"Invalid step name: {step\_name}")**

**step\_info = self.tracker.onboarding\_steps[step\_name]**

**# Validate completion criteria**

**validation\_result = self.\_validate\_step\_completion(step\_name, completion\_data)**

**if not validation\_result['is\_valid']:**

**return {**

**'success': False,**

**'error': 'Incomplete step data',**

**'missing\_criteria': validation\_result['missing\_criteria']**

**}**

**# Update step completion**

**completion\_record = {**

**'user\_id': self.user\_id,**

**'step\_name': step\_name,**

**'step\_number': step\_info['step\_number'],**

**'completion\_criteria': completion\_data,**

**'completed\_at': datetime.now().isoformat(),**

**'time\_spent\_minutes': completion\_data.get('time\_spent', 0),**

**'completion\_data': completion\_data**

**}**

**# Calculate new progress**

**current\_progress = self.get\_current\_progress()**

**updated\_progress = self.\_calculate\_updated\_progress(current\_progress, step\_name)**

**return {**

**'success': True,**

**'step\_completed': step\_name,**

**'new\_progress\_percentage': updated\_progress['progress\_percentage'],**

**'next\_step': updated\_progress['next\_step'],**

**'is\_onboarding\_complete': updated\_progress['is\_completed'],**

**'completion\_record': completion\_record**

**}**

**def get\_current\_progress(self) -> Dict[str, Any]:**

**"""Get user's current onboarding progress"""**

**# This would query the database**

**# SELECT \* FROM onboarding\_progress WHERE user\_id = %s**

**# Mock response for template**

**return {**

**'user\_id': self.user\_id,**

**'current\_step': 'financial\_profile',**

**'completed\_steps': ['welcome', 'profile\_setup'],**

**'progress\_percentage': 15,**

**'is\_completed': False,**

**'step\_completion\_data': {**

**'welcome': {'viewed\_welcome\_screen': True, 'clicked\_get\_started': True},**

**'profile\_setup': {'name\_provided': True, 'age\_provided': True, 'location\_provided': True, 'employment\_status\_provided': True}**

**},**

**'estimated\_time\_remaining': 23,**

**'created\_at': (datetime.now() - timedelta(hours=2)).isoformat()**

**}**

**def get\_step\_status(self, step\_name: str) -> Dict[str, Any]:**

**"""Get detailed status of a specific step"""**

**if step\_name not in self.tracker.onboarding\_steps:**

**return {'error': 'Invalid step name'}**

**step\_info = self.tracker.onboarding\_steps[step\_name]**

**current\_progress = self.get\_current\_progress()**

**is\_completed = step\_name in current\_progress['completed\_steps']**

**is\_current = current\_progress['current\_step'] == step\_name**

**is\_accessible = self.\_is\_step\_accessible(step\_name, current\_progress)**

**completion\_data = current\_progress['step\_completion\_data'].get(step\_name, {})**

**completion\_percentage = self.\_calculate\_step\_completion\_percentage(step\_name, completion\_data)**

**return {**

**'step\_name': step\_name,**

**'step\_info': step\_info,**

**'is\_completed': is\_completed,**

**'is\_current': is\_current,**

**'is\_accessible': is\_accessible,**

**'completion\_percentage': completion\_percentage,**

**'completion\_data': completion\_data,**

**'missing\_criteria': self.\_get\_missing\_criteria(step\_name, completion\_data)**

**}**

**def get\_onboarding\_analytics(self) -> Dict[str, Any]:**

**"""Get comprehensive onboarding analytics"""**

**current\_progress = self.get\_current\_progress()**

**# Calculate various metrics**

**total\_steps = len(self.tracker.onboarding\_steps)**

**completed\_steps = len(current\_progress['completed\_steps'])**

**# Time analytics**

**started\_at = datetime.fromisoformat(current\_progress['created\_at'])**

**time\_elapsed = datetime.now() - started\_at**

**estimated\_total\_time = sum(step['estimated\_time'] for step in self.tracker.onboarding\_steps.values())**

**# Completion rate by step type**

**required\_steps = [name for name, info in self.tracker.onboarding\_steps.items() if info['required']]**

**optional\_steps = [name for name, info in self.tracker.onboarding\_steps.items() if not info['required']]**

**required\_completed = len([step for step in current\_progress['completed\_steps'] if step in required\_steps])**

**optional\_completed = len([step for step in current\_progress['completed\_steps'] if step in optional\_steps])**

**return {**

**'overall\_progress': {**

**'total\_steps': total\_steps,**

**'completed\_steps': completed\_steps,**

**'progress\_percentage': current\_progress['progress\_percentage'],**

**'is\_completed': current\_progress['is\_completed']**

**},**

**'time\_analytics': {**

**'time\_elapsed\_minutes': int(time\_elapsed.total\_seconds() / 60),**

**'estimated\_total\_time\_minutes': estimated\_total\_time,**

**'estimated\_time\_remaining': max(0, estimated\_total\_time - int(time\_elapsed.total\_seconds() / 60)),**

**'completion\_pace': 'on\_track' if time\_elapsed.total\_seconds() / 60 <= estimated\_total\_time else 'behind'**

**},**

**'step\_breakdown': {**

**'required\_steps': {**

**'total': len(required\_steps),**

**'completed': required\_completed,**

**'percentage': int((required\_completed / len(required\_steps)) \* 100) if required\_steps else 100**

**},**

**'optional\_steps': {**

**'total': len(optional\_steps),**

**'completed': optional\_completed,**

**'percentage': int((optional\_completed / len(optional\_steps)) \* 100) if optional\_steps else 100**

**}**

**},**

**'next\_actions': self.\_get\_next\_actions(current\_progress),**

**'completion\_forecast': self.\_forecast\_completion\_time(current\_progress)**

**}**

**def \_validate\_step\_completion(self, step\_name: str, completion\_data: Dict[str, Any]) -> Dict[str, Any]:**

**"""Validate that a step meets completion criteria"""**

**step\_info = self.tracker.onboarding\_steps[step\_name]**

**required\_criteria = step\_info['completion\_criteria']**

**missing\_criteria = []**

**for criterion in required\_criteria:**

**if criterion not in completion\_data or not completion\_data[criterion]:**

**missing\_criteria.append(criterion)**

**return {**

**'is\_valid': len(missing\_criteria) == 0,**

**'missing\_criteria': missing\_criteria,**

**'completion\_percentage': ((len(required\_criteria) - len(missing\_criteria)) / len(required\_criteria)) \* 100**

**}**

**def \_calculate\_updated\_progress(self, current\_progress: Dict[str, Any], completed\_step: str) -> Dict[str, Any]:**

**"""Calculate updated progress after step completion"""**

**completed\_steps = current\_progress['completed\_steps'].copy()**

**if completed\_step not in completed\_steps:**

**completed\_steps.append(completed\_step)**

**# Calculate new progress percentage**

**total\_weight = sum(step['weight'] for step in self.tracker.onboarding\_steps.values())**

**completed\_weight = sum(**

**self.tracker.onboarding\_steps[step]['weight']**

**for step in completed\_steps**

**if step in self.tracker.onboarding\_steps**

**)**

**progress\_percentage = int((completed\_weight / total\_weight) \* 100)**

**# Determine next step**

**next\_step = self.\_get\_next\_step(completed\_steps)**

**is\_completed = next\_step is None**

**return {**

**'completed\_steps': completed\_steps,**

**'progress\_percentage': progress\_percentage,**

**'next\_step': next\_step,**

**'current\_step': next\_step or 'completed',**

**'is\_completed': is\_completed**

**}**

**def \_get\_next\_step(self, completed\_steps: List[str]) -> Optional[str]:**

**"""Determine the next step in the onboarding flow"""**

**for step\_name, step\_info in self.tracker.onboarding\_steps.items():**

**if step\_name not in completed\_steps:**

**return step\_name**

**return None**

**def \_is\_step\_accessible(self, step\_name: str, current\_progress: Dict[str, Any]) -> bool:**

**"""Check if a step is accessible based on current progress"""**

**step\_info = self.tracker.onboarding\_steps[step\_name]**

**step\_number = step\_info['step\_number']**

**# Check if all previous required steps are completed**

**for other\_step\_name, other\_step\_info in self.tracker.onboarding\_steps.items():**

**if (other\_step\_info['step\_number'] < step\_number and**

**other\_step\_info['required'] and**

**other\_step\_name not in current\_progress['completed\_steps']):**

**return False**

**return True**

**def \_calculate\_step\_completion\_percentage(self, step\_name: str, completion\_data: Dict[str, Any]) -> int:**

**"""Calculate completion percentage for a specific step"""**

**validation = self.\_validate\_step\_completion(step\_name, completion\_data)**

**return int(validation['completion\_percentage'])**

**def \_get\_missing\_criteria(self, step\_name: str, completion\_data: Dict[str, Any]) -> List[str]:**

**"""Get list of missing completion criteria for a step"""**

**validation = self.\_validate\_step\_completion(step\_name, completion\_data)**

**return validation['missing\_criteria']**

**def \_get\_next\_actions(self, current\_progress: Dict[str, Any]) -> List[Dict[str, Any]]:**

**"""Get recommended next actions for the user"""**

**if current\_progress['is\_completed']:**

**return [{'action': 'explore\_dashboard', 'priority': 'high', 'description': 'Start exploring your personalized dashboard'}]**

**next\_actions = []**

**current\_step = current\_progress['current\_step']**

**if current\_step in self.tracker.onboarding\_steps:**

**step\_info = self.tracker.onboarding\_steps[current\_step]**

**next\_actions.append({**

**'action': f'complete\_{current\_step}',**

**'priority': 'high',**

**'description': f'Complete {step\_info["title"]}',**

**'estimated\_time': step\_info['estimated\_time']**

**})**

**return next\_actions**

**def \_forecast\_completion\_time(self, current\_progress: Dict[str, Any]) -> Dict[str, Any]:**

**"""Forecast when onboarding will be completed"""**

**if current\_progress['is\_completed']:**

**return {'status': 'completed', 'forecast': None}**

**remaining\_steps = [**

**name for name in self.tracker.onboarding\_steps.keys()**

**if name not in current\_progress['completed\_steps']**

**]**

**remaining\_time = sum(**

**self.tracker.onboarding\_steps[step]['estimated\_time']**

**for step in remaining\_steps**

**)**

**started\_at = datetime.fromisoformat(current\_progress['created\_at'])**

**time\_elapsed = datetime.now() - started\_at**

**# Simple forecast based on current pace**

**if len(current\_progress['completed\_steps']) > 0:**

**average\_time\_per\_step = time\_elapsed.total\_seconds() / 60 / len(current\_progress['completed\_steps'])**

**forecasted\_time = remaining\_time \* average\_time\_per\_step**

**else:**

**forecasted\_time = remaining\_time**

**completion\_forecast = datetime.now() + timedelta(minutes=forecasted\_time)**

**return {**

**'status': 'in\_progress',**

**'remaining\_steps': len(remaining\_steps),**

**'estimated\_remaining\_minutes': int(remaining\_time),**

**'forecasted\_completion': completion\_forecast.isoformat(),**

**'confidence': 'medium' if len(current\_progress['completed\_steps']) >= 2 else 'low'**

**}**

**# Usage Examples and API Endpoints**

**class OnboardingAPI:**

**"""API endpoints for onboarding progress tracking"""**

**@staticmethod**

**def get\_user\_progress(user\_id: str) -> Dict[str, Any]:**

**"""GET /api/onboarding/progress/{user\_id}"""**

**manager = OnboardingProgressManager(user\_id)**

**return manager.get\_current\_progress()**

**@staticmethod**

**def complete\_step(user\_id: str, step\_name: str, completion\_data: Dict[str, Any]) -> Dict[str, Any]:**

**"""POST /api/onboarding/complete-step"""**

**manager = OnboardingProgressManager(user\_id)**

**return manager.complete\_step(step\_name, completion\_data)**

**@staticmethod**

**def get\_step\_status(user\_id: str, step\_name: str) -> Dict[str, Any]:**

**"""GET /api/onboarding/step/{step\_name}"""**

**manager = OnboardingProgressManager(user\_id)**

**return manager.get\_step\_status(step\_name)**

**@staticmethod**

**def get\_analytics(user\_id: str) -> Dict[str, Any]:**

**"""GET /api/onboarding/analytics/{user\_id}"""**

**manager = OnboardingProgressManager(user\_id)**

**return manager.get\_onboarding\_analytics()**

**# Example usage**

**if \_\_name\_\_ == "\_\_main\_\_":**

**# Initialize onboarding for new user**

**user\_id = "user\_123"**

**manager = OnboardingProgressManager(user\_id)**

**# Initialize onboarding**

**initial\_progress = manager.initialize\_onboarding()**

**print("Onboarding initialized:", initial\_progress)**

**# Complete welcome step**

**welcome\_completion = {**

**'viewed\_welcome\_screen': True,**

**'clicked\_get\_started': True,**

**'time\_spent': 2**

**}**

**result = manager.complete\_step('welcome', welcome\_completion)**

**print("Welcome step completed:", result)**

**# Get current progress**

**progress = manager.get\_current\_progress()**

**print("Current progress:", progress)**

**# Get analytics**

**analytics = manager.get\_onboarding\_analytics()**

**print("Onboarding analytics:", analytics)**