```
FUNCTION rankProductsAdvanced(userProfile, productDatabase, feedbackLog):
    # Input validation
    IF productDatabase IS empty OR userProfile IS null THEN
        RETURN empty list
    END IF

    # --- DYNAMIC WEIGHTING SYSTEM ---
    # Start with base weights, then adjust based on user learning
    SET weights = getAdaptiveWeights(userProfile, feedbackLog)
    CONSTANT W_TERPENE = weights.terpene        # Default: 0.5
    CONSTANT W_CANNABINOID = weights.cannabinoid # Default: 0.3
    CONSTANT W_FLAVONOID = weights.flavonoid     # Default: 0.2

    SET rankedResults TO empty list
    SET allScores TO empty list  # For cross-product normalization

    # --- FIRST PASS: Calculate raw scores ---
    FOR each product IN productDatabase DO
        SET terpeneScore = 0
        SET cannabinoidScore = 0
        SET modifierScore = 0
        SET penaltyFlags TO empty list

        # --- Terpene Scoring with Condition Severity ---
        IF product.terpenes IS NOT null AND NOT empty THEN
            FOR each terpene IN product.terpenes DO
                SET efficacy = getEfficacyWithSeverity(
                    terpene,
                    userProfile.conditions,
                    "terpene"
                )

                # Sensitivity boost
                IF userProfile.sensitiveTerpenes IS NOT null AND
                   terpene.name IN userProfile.sensitiveTerpenes THEN
                    SET efficacy = efficacy * 1.5
                END IF

                # Normalize concentration (0-100 scale)
                SET normalizedConcentration = MIN(terpene.concentration / 100, 1.0)
                ADD (efficacy * normalizedConcentration) TO terpeneScore
            END FOR
        END IF

        # --- Cannabinoid Scoring with Biphasic/Threshold Handling ---
        IF product.cannabinoids IS NOT null THEN
            SET result = getCannabinoidScoreWithThresholds(
                product,
                userProfile.conditions,
                userProfile.thresholds
```

```
        )
        SET cannabinoidScore = result.score
        IF result.penalties IS NOT empty THEN
            ADD result.penalties TO penaltyFlags
        END IF
    END IF

    # --- EXTENSIBLE Modifier Scoring ---
    # Includes flavonoids, delivery method, grow style, and interactions
    IF product.modifiers IS NOT null OR product.flavonoids IS NOT null THEN
        SET modifierScore = getExtensibleModifierScore(
            product,
            userProfile,
            product.terpenes  # For interaction effects
        )
    END IF

    # --- Apply Personal History Weight ---
    SET H_factor = 0
    IF userProfile.history IS NOT null THEN
        SET H_factor = getUserHistoryFactor(product, userProfile.history)
    END IF

    # Base score calculation
    SET rawScore = ((W_TERPENE * terpeneScore) +
                    (W_CANNABINOID * cannabinoidScore) +
                    (W_FLAVONOID * modifierScore)) * (1 + H_factor)

    # Clamp to prevent negative values
    SET rawScore = MAX(0, rawScore)

    # Store for normalization
    ADD rawScore TO allScores

    ADD {
        "id": product.id,
        "name": product.name,
        "rawScore": rawScore,
        "componentScores": {
            "terpene": terpeneScore,
            "cannabinoid": cannabinoidScore,
            "modifier": modifierScore,
            "historyBoost": H_factor
        },
        "penaltyFlags": penaltyFlags,
        "rationale": null  # Will generate after normalization
    } TO rankedResults
END FOR

# --- CROSS-PRODUCT NORMALIZATION ---
```

```
    SET maxScore = MAX(allScores)
    SET minScore = MIN(allScores)
    SET scoreRange = maxScore - minScore

    IF scoreRange > 0 THEN
        FOR each result IN rankedResults DO
            # Normalize to 0-100 scale
            SET result.matchScore = ((result.rawScore - minScore) / scoreRange) * 100

            # Generate rationale with normalized scores
            SET result.rationale = generateEnhancedRationale(
                result,
                userProfile,
                weights
            )
        END FOR
    ELSE
        # All scores equal - assign uniform score
        FOR each result IN rankedResults DO
            SET result.matchScore = 50
            SET result.rationale = "Similar match across all products"
        END FOR
    END IF

    # Sort by normalized matchScore
    SORT rankedResults BY matchScore DESCENDING

    # --- LOGGING FOR CONTINUOUS LEARNING ---
    logRecommendationEvent(rankedResults, userProfile, weights, feedbackLog)

    RETURN rankedResults
END FUNCTION

# --- DYNAMIC WEIGHTING SYSTEM ---

FUNCTION getAdaptiveWeights(userProfile, feedbackLog):
    # Start with base weights
    SET weights = {
        "terpene": 0.5,
        "cannabinoid": 0.3,
        "flavonoid": 0.2
    }

    IF feedbackLog IS empty OR userProfile.feedbackCount < 5 THEN
        RETURN weights  # Not enough data to adapt
    END IF

    # Analyze user feedback to adjust weights
    SET terpeneEffectiveness = 0
    SET cannabinoidEffectiveness = 0
```

```
    SET flavonoidEffectiveness = 0
    SET totalFeedback = 0

    FOR each feedback IN feedbackLog WHERE feedback.userId == userProfile.id DO
        IF feedback.rating >= 4 THEN  # Positive feedback
            SET terpeneEffectiveness = terpeneEffectiveness + feedback.terpeneScore
            SET cannabinoidEffectiveness = cannabinoidEffectiveness + feedback.cannabinoidScore
            SET flavonoidEffectiveness = flavonoidEffectiveness + feedback.flavonoidScore
            SET totalFeedback = totalFeedback + 1
        END IF
    END FOR

    IF totalFeedback > 0 THEN
        # Calculate average effectiveness
        SET avgTerpene = terpeneEffectiveness / totalFeedback
        SET avgCannabinoid = cannabinoidEffectiveness / totalFeedback
        SET avgFlavonoid = flavonoidEffectiveness / totalFeedback
        SET total = avgTerpene + avgCannabinoid + avgFlavonoid

        IF total > 0 THEN
            # Adjust weights proportionally with dampening
            CONSTANT LEARNING_RATE = 0.1
            SET newTerpene = 0.5 + (avgTerpene / total - 0.5) * LEARNING_RATE
            SET newCannabinoid = 0.3 + (avgCannabinoid / total - 0.3) * LEARNING_RATE
            SET newFlavonoid = 0.2 + (avgFlavonoid / total - 0.2) * LEARNING_RATE

            # Normalize to sum to 1.0
            SET sum = newTerpene + newCannabinoid + newFlavonoid
            SET weights.terpene = newTerpene / sum
            SET weights.cannabinoid = newCannabinoid / sum
            SET weights.flavonoid = newFlavonoid / sum
        END IF
    END IF

    RETURN weights
END FUNCTION

# --- CONDITION SEVERITY SCALING ---

FUNCTION getEfficacyWithSeverity(compound, conditions, compoundType):
    IF conditions IS empty THEN
        RETURN 0.5  # Neutral efficacy
    END IF

    SET weightedEfficacy = 0
    SET totalSeverity = 0

    FOR each condition IN conditions DO
        SET efficacy = lookupEfficacy(compound.name, condition.name, compoundType)
        SET severity = condition.severity  # Scale: 1-10
```

```
      # Weight efficacy by condition severity
      SET weightedEfficacy = weightedEfficacy + (efficacy * severity)
      SET totalSeverity = totalSeverity + severity
    END FOR

    IF totalSeverity > 0 THEN
      RETURN MIN(weightedEfficacy / totalSeverity, 1.0)
    ELSE
      RETURN 0.5
    END IF
END FUNCTION

# --- BIPHASIC/THRESHOLD HANDLING ---

FUNCTION getCannabinoidScoreWithThresholds(product, conditions, thresholds):
    IF product.cannabinoids IS empty THEN
      RETURN {"score": 0, "penalties": []}
    END IF

    SET score = 0
    SET penalties TO empty list

    FOR each cannabinoid IN product.cannabinoids DO
      SET efficacy = getEfficacyWithSeverity(cannabinoid, conditions, "cannabinoid")
      SET normalizedAmount = MIN(cannabinoid.percentage / 100, 1.0)

      # --- THRESHOLD CHECKS ---
      # THC threshold
      IF cannabinoid.name == "THC" THEN
        IF thresholds.maxTHC IS NOT null AND cannabinoid.percentage > thresholds.maxTHC THEN
          SET efficacy = -ABS(efficacy) * 0.5  # Strong penalty
          ADD "THC exceeds user threshold (" + cannabinoid.percentage + "% > " + thresholds.maxTHC + "%)"
TO penalties
        ELSE IF thresholds.minTHC IS NOT null AND cannabinoid.percentage < thresholds.minTHC THEN
          SET efficacy = efficacy * 0.5  # Weak penalty
          ADD "THC below preferred minimum" TO penalties
        END IF
      END IF

      # CBD threshold
      IF cannabinoid.name == "CBD" THEN
        IF thresholds.minCBD IS NOT null AND cannabinoid.percentage < thresholds.minCBD THEN
          SET efficacy = efficacy * 0.7
          ADD "CBD below preferred minimum" TO penalties
        END IF
      END IF

      # Ratio checks (e.g., CBD:THC ratio)
      IF thresholds.preferredRatio IS NOT null THEN
```

```
            SET actualRatio = calculateRatio(product.cannabinoids)
            SET ratioDeviation = ABS(actualRatio - thresholds.preferredRatio)
            IF ratioDeviation > 0.5 THEN
                SET efficacy = efficacy * (1 - MIN(ratioDeviation * 0.2, 0.5))
            END IF
        END IF

        ADD (efficacy * normalizedAmount) TO score
    END FOR

    RETURN {"score": score, "penalties": penalties}
END FUNCTION

# --- EXTENSIBLE MODIFIER LAYER ---

FUNCTION getExtensibleModifierScore(product, userProfile, terpenes):
    SET score = 0

    # Flavonoid scoring
    IF product.flavonoids IS NOT null THEN
        FOR each flavonoid IN product.flavonoids DO
            SET efficacy = getEfficacyWithSeverity(flavonoid, userProfile.conditions, "flavonoid")
            ADD efficacy TO score
        END FOR
    END IF

    # Delivery method preference
    IF product.deliveryMethod IS NOT null AND userProfile.preferredDelivery IS NOT null THEN
        IF product.deliveryMethod IN userProfile.preferredDelivery THEN
            ADD 0.3 TO score  # Boost for preferred delivery
        END IF
    END IF

    # Grow style preference
    IF product.growStyle IS NOT null AND userProfile.preferredGrowStyle IS NOT null THEN
        IF product.growStyle == userProfile.preferredGrowStyle THEN
            ADD 0.2 TO score  # "sun-grown" vs "indoor"
        END IF
    END IF

    # --- INTERACTION EFFECTS ---
    # Example: sun-grown + high pinene might have synergy
    IF product.growStyle == "sun-grown" AND terpenes IS NOT null THEN
        FOR each terpene IN terpenes DO
            IF terpene.name == "pinene" AND terpene.concentration > 50 THEN
                ADD 0.15 TO score  # Synergy bonus
            END IF
        END FOR
    END IF
```

```
    # Terpene-flavonoid interaction (entourage effect)
    IF product.flavonoids IS NOT null AND terpenes IS NOT null THEN
        SET entourageBonus = calculateEntourageEffect(terpenes, product.flavonoids)
        ADD entourageBonus TO score
    END IF

    RETURN MIN(score, 1.5)  # Cap with room for bonuses
END FUNCTION

FUNCTION calculateEntourageEffect(terpenes, flavonoids):
    # Synergistic combinations
    SET synergies = {
        {"pinene", "quercetin"}: 0.1,
        {"limonene", "kaempferol"}: 0.15,
        {"myrcene", "apigenin"}: 0.1
    }

    SET bonus = 0
    FOR each terpene IN terpenes DO
        FOR each flavonoid IN flavonoids DO
            SET pair = {terpene.name, flavonoid.name}
            IF pair IN synergies THEN
                ADD synergies[pair] TO bonus
            END IF
        END FOR
    END FOR

    RETURN MIN(bonus, 0.3)  # Cap entourage bonus
END FUNCTION

# --- ENHANCED HELPER FUNCTIONS ---

FUNCTION getUserHistoryFactor(product, history):
    IF history IS empty THEN
        RETURN 0
    END IF

    SET boostFactor = 0
    SET similarProductBoost = 0

    # Direct product feedback
    FOR each historyItem IN history DO
        IF historyItem.productId == product.id THEN
            IF historyItem.rating >= 4 THEN
                SET boostFactor = boostFactor + 0.4
            ELSE IF historyItem.rating <= 2 THEN
                SET boostFactor = boostFactor - 0.3
            END IF
        END IF
    END FOR
```

```
    # Similar product feedback (same dominant terpene or cannabinoid ratio)
    FOR each historyItem IN history DO
        IF historyItem.dominantTerpene == product.dominantTerpene AND historyItem.rating >= 4 THEN
            SET similarProductBoost = similarProductBoost + 0.15
        END IF
    END FOR

    SET totalBoost = boostFactor + MIN(similarProductBoost, 0.3)
    RETURN MAX(-0.5, MIN(totalBoost, 0.6))
END FUNCTION

FUNCTION generateEnhancedRationale(result, userProfile, weights):
    SET rationale TO empty list
    SET components = result.componentScores

    # Explain top contributor
    SET topComponent = getMaxComponent(components)
    ADD "Primary match: " + topComponent TO rationale

    # Explain weights if adapted
    IF weights.terpene != 0.5 OR weights.cannabinoid != 0.3 OR weights.flavonoid != 0.2 THEN
        ADD "Personalized weighting applied based on your feedback" TO rationale
    END IF

    # Condition severity mentions
    SET criticalConditions = userProfile.conditions WHERE severity >= 8
    IF criticalConditions IS NOT empty THEN
        ADD "Optimized for: " + JOIN(criticalConditions.name, ", ") TO rationale
    END IF

    # Penalties
    IF result.penaltyFlags IS NOT empty THEN
        ADD "Note: " + JOIN(result.penaltyFlags, "; ") TO rationale
    END IF

    # History boost
    IF components.historyBoost > 0.2 THEN
        ADD "You've responded well to similar products" TO rationale
    ELSE IF components.historyBoost < -0.2 THEN
        ADD "Differs from your usual preferences" TO rationale
    END IF

    RETURN JOIN(rationale, " | ")
END FUNCTION

# --- LOGGING FOR CONTINUOUS LEARNING ---

FUNCTION logRecommendationEvent(rankedResults, userProfile, weights, feedbackLog):
    SET event = {
```

```
        "timestamp": CURRENT_TIMESTAMP(),
        "userId": userProfile.id,
        "topRecommendations": rankedResults[0:5],  # Top 5
        "weights": weights,
        "conditions": userProfile.conditions,
        "awaitingFeedback": true
    }

    ADD event TO feedbackLog
    RETURN event.id  # For tracking when user provides feedback
END FUNCTION

# --- UTILITY FUNCTIONS ---

FUNCTION calculateRatio(cannabinoids):
    SET cbd = 0
    SET thc = 0
    FOR each cannabinoid IN cannabinoids DO
        IF cannabinoid.name == "CBD" THEN SET cbd = cannabinoid.percentage
        IF cannabinoid.name == "THC" THEN SET thc = cannabinoid.percentage
    END FOR
    IF thc > 0 THEN RETURN cbd / thc
    RETURN 0
END FUNCTION

FUNCTION getMaxComponent(components):
    SET max = 0
    SET maxName = ""
    IF components.terpene > max THEN
        SET max = components.terpene
        SET maxName = "terpene profile"
    END IF
    IF components.cannabinoid > max THEN
        SET max = components.cannabinoid
        SET maxName = "cannabinoid content"
    END IF
    IF components.modifier > max THEN
        SET max = components.modifier
        SET maxName = "additional compounds & delivery"
    END IF
    RETURN maxName
END FUNCTION

# --- LOOKUP STUB (connects to knowledge base) ---

FUNCTION lookupEfficacy(compoundName, conditionName, compoundType):
    # This would query your knowledge base
    # Returns efficacy score 0-1
    # Could be dynamically updated based on aggregate feedback
    RETURN 0.7  # Placeholder
```

END FUNCTIO