Design Document

Overview

Big Five性格モデルを基盤とした行動変容支援AIアシスタントの設計書です。本システムは、科学的な性格測定、日常会話からの性格推定、認知行動療法に基づく対話、実行意図による行動自動化を統合し、個別化された継続的な行動変容支援を提供します。

Architecture

システム全体構成

```mermaid
graph TB
    User[利用者] --> Chat[チャットインターフェース]
    Chat --> N8N[n8nワークフロー]
    N8N --> Symanto[Symanto Big Five API]
    N8N --> MySQL[(MySQL データベース)]
    N8N --> Notification[通知システム]

    subgraph "データ処理フロー"
        N8N --> TextAnalysis[テキスト分析]
        N8N --> CBTDialog[CBT対話処理]
        N8N --> IfThenManager[If-Then管理]
        N8N --> TSL[Trait Pack選定]
        N8N --> KPICalculation[KPI算出]
    end

    subgraph "データストレージ"
        MySQL --> UserProfile[利用者プロファイル]
        MySQL --> PersonalityData[性格測定データ]
        MySQL --> ConversationLog[会話ログ]
        MySQL --> BehaviorTracking[行動追跡]
        MySQL --> TraitPacks[Trait Pack設定]
    end
```

技術スタック

- **ワークフロー管理**: n8n (統合処理エンジン)
- **データベース**: MySQL 8.0+ (長期データ保存・検索)
- **外部API**: Symanto Big Five API (テキスト性格推定)
- **設定管理**: YAML/JSON (Trait Pack設定)
- **通知**: n8n内蔵通知機能
- **多言語対応**: 日本語、ベトナム語、英語

##
Components and Interfaces

1. 性格測定コンポーネント

PersonalityAssessmentService
```typescript
interface PersonalityAssessmentService {
  //初期測定実行
  conductInitialAssessment(userId: string, assessmentType: 'BFI-2' | 'IPIP-NEO-120'):
Promise<AssessmentResult>

  //月次短縮測定
  conductMonthlyReassessment(userId: string): Promise<BFI2SResult>

  //結果保存
  saveAssessmentResult(userId: string, result: AssessmentResult): Promise<void>

  //15ファセット算出
  calculateFacets(rawScores: number[]): FacetScores
}
interface AssessmentResult {
  userId: string
  assessmentDate: Date
  assessmentType: string
  oceanScores: OCEANScores
  facetScores: FacetScores
  reliability: ReliabilityMetrics
}
```

2. テキスト分析コンポーネント

TextAnalysisService
```typescript
interface TextAnalysisService {
  // Symanto API呼び出し
  analyzeText(text: string, language: string): Promise<SymantoResult>

  //信頼度フィルタリング
  filterByConfidence(result: SymantoResult, threshold: number): FilteredResult

  //段階的反映
  updatePersonalityEstimate(userId: string, newEstimate: FilteredResult): Promise<void>

  //再試行処理
  retryWithBackoff(apiCall: () => Promise<any>, maxRetries: number): Promise<any>
}
interface SymantoResult {
  oceanScores: OCEANScores
  confidence: ConfidenceScores
```

  analysisMetadata: AnalysisMetadata
}
```

### 3. CBT対話コンポーネント

**3-**
```typescript
interface CBTDialogService {
  // ABCDE対話進行
  processABCDEDialog(userId: string, userInput: string, currentStage: ABCDEStage):
Promise<DialogResponse>

  // 5分行動生成
  generateFiveMinuteActions(context: DialogContext, targetTrait: string):
Promise<ActionSuggestion[]>

  // BCT技法適用
  applyBCTTechniques(actionType: string, userProfile: UserProfile): BCTApplication

  // 対話記録
  logConversation(userId: string, dialog: DialogSession): Promise<void>
}
interface DialogResponse {
  summary: string
  currentStage: ABCDEStage
  suggestedActions: ActionSuggestion[]
  appliedBCTs: string[]
  nextPrompt: string
}
```#

4. If-Then管理コンポーネント

IfThenManagerService
```typescript
interface IfThenManagerService {
  // If-Then設定作成
  createIfThenRule(userId: string, ifCondition: string, thenAction: string, relatedBCTs: string[]):
Promise<IfThenRule>

  // 実行状況記録
  recordExecution(userId: string, ruleId: string, executionStatus: ExecutionStatus): Promise<void>

  // 実行率算出
  calculateExecutionRate(userId: string, period: TimePeriod): Promise<ExecutionMetrics>

  // 改善提案
  suggestImprovements(userId: string, lowPerformingRules: IfThenRule[]):
Promise<ImprovementSuggestion[]>
}
interface IfThenRule {

```
  id: string
  userId: string
  ifCondition: string
  thenAction: string
  relatedBCTs: string[]
  createdDate: Date
  isActive: boolean
}
```

5. Trait Pack選定コンポーネント (TSL)

TraitPackSelectorService
```typescript
interface TraitPackSelectorService {
  // 週次特性選定
  selectWeeklyTargetTrait(userId: string): Promise<TargetTraitSelection>

  // 偏差計算
  calculateTraitDeviations(initialScores: OCEANScores, currentEstimates: OCEANScores):
Promise<DeviationScores>

  // Trait Pack適用
  applyTraitPack(userId: string, targetTrait: string): Promise<TraitPackConfiguration>

  // 効果評価
  evaluateWeeklyEffectiveness(userId: string, targetTrait: string): Promise<EffectivenessMetrics>
}
interface TraitPackConfiguration {
  targetTrait: string
  primaryFacets: string[]
  recommendedBCTs: string[]
  ifThenTemplates: IfThenTemplate[]
  kpiTargets: KPITarget[]
}
```

6. KPI測定コンポーネント

6-
```typescript
interface KPICalculationService {
  // Cohen's d算出
  calculateCohenD(preScores: number[], postScores: number[]): Promise<EffectSizeResult>

  // 週次移動平均
  calculateWeeklyMovingAverage(userId: string, trait: string, confidenceThreshold: number):
Promise<MovingAverageResult>

  // 行動化率算出
  calculateBehavioralizationRate(userId: string, period: TimePeriod):
Promise<BehavioralizationMetrics>
```

```
  // 統合ダッシュボード
  generateDashboard(userId: string): Promise<DashboardData>
}
interface DashboardData {
  monthlyEffectSizes: EffectSizeResult[]
  weeklyMovingAverages: MovingAverageResult[]
  behavioralizationRates: BehavioralizationMetrics
  proximityIndicators: ProximityIndicator[]
}
```## Dat
a Models
利用者プロファイル
```sql
CREATE TABLE users (
    user_id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    department VARCHAR(100),
    language ENUM('ja', 'vi', 'en') DEFAULT 'ja',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
);
```

性格測定データ

```sql
CREATE TABLE personality_assessments (
    assessment_id VARCHAR(50) PRIMARY KEY,
    user_id VARCHAR(50) NOT NULL,
    assessment_date DATE NOT NULL,
    assessment_type ENUM('BFI-2', 'IPIP-NEO-120', 'BFI-2-S') NOT NULL,
    openness_score DECIMAL(4,3),
    conscientiousness_score DECIMAL(4,3),
    extraversion_score DECIMAL(4,3),
    agreeableness_score DECIMAL(4,3),
    neuroticism_score DECIMAL(4,3),
    facet_scores JSON,
    reliability_metrics JSON,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

テキスト推定データ

```sql
CREATE TABLE text_personality_estimates (
    estimate_id VARCHAR(50) PRIMARY KEY,
    user_id VARCHAR(50) NOT NULL,
    estimate_datetime DATETIME NOT NULL,
    text_summary TEXT,
```

```
    openness_estimate DECIMAL(4,3),
    conscientiousness_estimate DECIMAL(4,3),
    extraversion_estimate DECIMAL(4,3),
    agreeableness_estimate DECIMAL(4,3),
    neuroticism_estimate DECIMAL(4,3),
    confidence_scores JSON,
    is_applied BOOLEAN DEFAULT FALSE,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

会話記録

```sql
CREATE TABLE conversation_logs (
    log_id VARCHAR(50) PRIMARY KEY,
    user_id VARCHAR(50) NOT NULL,
    conversation_datetime DATETIME NOT NULL,
    summary_line TEXT,
    abcde_stage ENUM('A', 'B', 'C', 'D', 'E'),
    suggested_actions JSON,
    applied_bcts JSON,
    conversation_context JSON,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

If-Then規則

```sql
CREATE TABLE if_then_rules (
    rule_id VARCHAR(50) PRIMARY KEY,
    user_id VARCHAR(50) NOT NULL,
    if_condition TEXT NOT NULL,
    then_action TEXT NOT NULL,
    related_bcts JSON,
    created_date DATE NOT NULL,
    is_active BOOLEAN DEFAULT TRUE,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

行動実行記録

```sql
CREATE TABLE behavior_executions (
    execution_id VARCHAR(50) PRIMARY KEY,
    user_id VARCHAR(50) NOT NULL,
    rule_id VARCHAR(50) NOT NULL,
    execution_date DATE NOT NULL,
    execution_status ENUM('Yes', 'No', 'Partial') NOT NULL,
```

```
    notes TEXT,
    FOREIGN KEY (user_id) REFERENCES users(user_id),
    FOREIGN KEY (rule_id) REFERENCES if_then_rules(rule_id)
);
```

Trait Pack設定

```sql
CREATE TABLE trait_pack_configurations (
    config_id VARCHAR(50) PRIMARY KEY,
    user_id VARCHAR(50) NOT NULL,
    target_trait ENUM('O', 'C', 'E', 'A', 'N') NOT NULL,
    week_start_date DATE NOT NULL,
    primary_facets JSON,
    recommended_bcts JSON,
    if_then_templates JSON,
    kpi_targets JSON,
    is_active BOOLEAN DEFAULT TRUE,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

```## Error

Handling

API呼び出しエラー処理

- **Symanto API**: 指数バックオフによる自動再試行（最大3回）
- **タイムアウト**: 30秒でタイムアウト、エラーログ記録
- **レート制限**: 429エラー時は適切な待機時間後に再試行

データ整合性エラー処理

- **必須フィールド欠損**: バリデーションエラーとして適切なメッセージを返す
- **外部キー制約違反**: 関連データの存在確認後に処理実行
- **データ型不整合**: 型変換エラー時は詳細なエラーログを記録

ユーザー体験エラー処理

- **測定中断**: 途中保存機能で進捗を保持
- **ネットワーク切断**: オフライン対応とデータ同期機能
- **不適切な入力**: やさしい言葉でのガイダンス提供

Testing Strategy

単体テスト

- **性格測定算出ロジック**: BFI-2、IPIP-NEO-120の得点計算精度
- **Cohen's d計算**: 効果サイズ算出の数学的正確性
- **If-Then実行率算出**: 期間別集計ロジックの検証

統合テスト
- **n8n ↔ Symanto API**: API呼び出しと結果処理の一連フロー
- **n8n ↔ MySQL**: データ保存・取得の整合性
- **TSL自動選定**: 週次特性選定の意思決定ロジック

エンドツーエンドテスト
- **利用者登録 → 初期測定 → 日次対話 → 週次レビュー**: 完全なユーザージャーニー
- **多言語対応**: 日本語、ベトナム語、英語での全機能動作確認
- **長期データ蓄積**: 数ヶ月分のデータでのパフォーマンス検証

パフォーマンステスト
- **同時利用者数**: 100名同時利用時の応答時間
- **データベース負荷**: 大量履歴データでの検索性能
- **API呼び出し頻度**: Symanto APIの制限内での最適化

セキュリティテスト
- **個人データ保護**: 暗号化、アクセス制御の検証
- **SQL インジェクション**: データベースクエリの安全性
- **認証・認可**: 利用者データへの不正アクセス防止