BMAD Component Mapping & Integration Guide

BioSpark Health AI - Detailed Component Transplantation Strategy

Date: July 24, 2025
Analysis Type: Component-Level Integration Mapping
BMAD Agents: Architect + Developer Coordination
Precision Level: File-by-File Mapping

🗺️ COMPONENT MAPPING OVERVIEW

Integration Strategy

```
OLD SYSTEM (biospark33/lablens)     →     NEW SYSTEM (biospark33/biospark-health-ai)
├── Health Analysis Components      →     Enhanced with Memory Integration
├── Progressive Disclosure UI       →     Direct Port + Zep Enhancement
├── Ray Peat Methodology            →     Preserved + AbacusAI Integration
├── OpenAI Integration              →     Shared (Identical Implementation)
└── Supabase Database               →     Shared (Same Environment)
```

📁 DETAILED FILE MAPPING

1. CORE LIBRARY INTEGRATIONS

OpenAI Integration (MERGE REQUIRED)

```
// SOURCE: ~/correct-lablens/lib/openai.ts
// TARGET: ~/biospark-health-ai/lib/openai.ts
// ACTION: MERGE - Combine Ray Peat methodology with existing OpenAI client

// OLD SYSTEM - Ray Peat Specific Functions
export const healthAI = {
  generateHealthInsights(assessmentData: any)     // PRESERVE
  generateRecommendations(userProfile, results)   // PRESERVE
  explainTerm(term: string, context?: string)     // PRESERVE
  generateCuriosityGap(topic: string, level)      // PRESERVE
}

// NEW SYSTEM - Enhanced OpenAI Client
export const openai = new OpenAI({
  apiKey: process.env.OPENAI_API_KEY,
  // Enhanced configuration
});

// INTEGRATION STRATEGY
export const integratedHealthAI = {
  // Preserve all old system functions
  ...healthAI,

  // Add memory-enhanced versions
  generateMemoryAwareInsights(assessmentData, memoryContext),
  generateContextualRecommendations(userProfile, results, history),
  explainTermWithHistory(term, context, userJourney)
}
```

Supabase Integration (EXTEND EXISTING)

```
// SOURCE: ~/correct-lablens/lib/supabase.ts
// TARGET: ~/biospark-health-ai/lib/supabase.ts
// ACTION: EXTEND - Add health-specific operations to existing client

// OLD SYSTEM - Health Specific Operations
export const healthAI = {
  createAssessment(userId, assessmentData)        // ADD TO NEW SYSTEM
  getAssessment(assessmentId)                     // ADD TO NEW SYSTEM
  getUserAssessments(userId)                      // ADD TO NEW SYSTEM
  saveInsight(assessmentId, insight)              // ADD TO NEW SYSTEM
  trackProgress(userId, metrics)                  // ADD TO NEW SYSTEM
}

// INTEGRATION TARGET
// ~/biospark-health-ai/lib/supabase.ts (EXTEND)
export const supabase = createClient(/* existing config */);

// ADD health operations to existing client
export const healthOperations = {
  ...existingOperations,
  ...healthAI  // Add all health-specific operations
};
```

AbacusAI Integration (NEW ADDITION)

```
// SOURCE: ~/correct-lablens/lib/abacus.ts
// TARGET: ~/biospark-health-ai/lib/abacus.ts
// ACTION: CREATE NEW - Add AbacusAI capabilities to new system

// COMPLETE FILE TRANSPLANT
class AbacusAIClient {
  predictHealthRisk(patientData)                    // TRANSPLANT
  analyzeMetabolicProfile(metabolicData)            // TRANSPLANT
  generatePersonalizedPlan(userProfile, goals)    // TRANSPLANT
  detectAnomalies(healthMetrics)                    // TRANSPLANT
}

export const healthAI = {
  runHealthAssessment(assessmentData)               // TRANSPLANT
  generateInsights(analysisResults, userContext) // TRANSPLANT
  validateModelHealth()                             // TRANSPLANT
}
```

---

2. COMPONENT TRANSPLANTATION MAP

Health Analysis Components (DIRECT PORT + ENHANCE)

```typescript
// SOURCE: ~/correct-lablens/components/health/
// TARGET: ~/biospark-health-ai/components/health/
// ACTION: PORT + MEMORY ENHANCE

// 1. Health Snapshot Component
// SOURCE: components/health/health-snapshot.tsx
// TARGET: components/health/health-snapshot.tsx (ENHANCE EXISTING)
interface HealthSnapshotProps {
  // Existing props (PRESERVE)
  analysisData: AnalysisData;

  // Added from old system
  overallScore: number;
  energyLevel: number;
  metabolicHealth: number;
  stressLevel: number;

  // Memory enhancement
  memoryContext?: ZepMemoryContext;
  historicalTrends?: HealthTrend[];
}

// 2. Detailed Insights Component
// SOURCE: components/health/detailed-insights.tsx
// TARGET: components/health/detailed-insights.tsx (ENHANCE EXISTING)
interface DetailedInsightsProps {
  // Existing props (PRESERVE)
  insights: DetailedInsight[];

  // Added from old system
  thyroidFunction: number;
  mitochondrialHealth: number;
  hormonalBalance: number;
  inflammationLevel: number;
  rayPeatContext: RayPeatAnalysis;

  // Memory enhancement
  personalizedInsights: PersonalizedInsight[];
  contextualRecommendations: ContextualRecommendation[];
}

// 3. Comprehensive Analysis Component
// SOURCE: components/health/comprehensive-analysis.tsx
// TARGET: components/health/comprehensive-analysis.tsx (MAJOR ENHANCEMENT)
interface ComprehensiveAnalysisProps {
  // Existing props (PRESERVE)
  comprehensiveData: ComprehensiveData;

  // Added from old system
  researchReferences: ResearchReference[];
  technicalDetails: TechnicalDetail[];
  biomarkerCorrelations: BiomarkerCorrelation[];

  // Memory enhancement
  historicalAnalysis: HistoricalAnalysis;
  predictiveInsights: PredictiveInsight[];
  personalizedRecommendations: PersonalizedRecommendation[];
}
```

Progressive Disclosure System (CORE TRANSPLANT)

```
// SOURCE: ~/correct-lablens/components/health/progressive-disclosure.tsx
// TARGET: ~/biospark-health-ai/components/health/progressive-disclosure.tsx
// ACTION: CREATE NEW COMPONENT

// COMPLETE TRANSPLANT with Memory Enhancement
export function ProgressiveDisclosure({
  // Layer 1: Health Snapshot (Quick Overview)
  healthSnapshot: {
    overallScore,
    keyFindings,
    urgentActions,
    engagementHooks
  },

  // Layer 2: Detailed Insights (Contextual Analysis)
  detailedInsights: {
    rayPeatAnalysis,
    biomarkerInterpretation,
    contextualExplanations,
    actionableRecommendations
  },

  // Layer 3: Comprehensive Analysis (Professional Grade)
  comprehensiveAnalysis: {
    researchReferences,
    technicalDetails,
    correlationAnalysis,
    predictiveModeling
  },

  // Memory Enhancement
  memoryContext,
  userPreferences,
  explorationHistory
}) {
  // Implementation preserves exact UX patterns from old system
  // Enhanced with Zep memory for personalization
}
```

---

3. DATABASE SCHEMA INTEGRATION

Prisma Schema Extension (ADDITIVE ONLY)

```
// SOURCE: ~/correct-lablens/prisma/schema.prisma
// TARGET: ~/biospark-health-ai/prisma/schema.prisma
// ACTION: EXTEND - Add health models to existing schema

// PRESERVE ALL EXISTING MODELS (User, Analysis, etc.)
// ADD health-specific models from old system

// 1. Health Assessment Model (ADD)
model HealthAssessment {
  id                  String   @id @default(cuid())
  userId              String
  user                User     @relation(fields: [userId], references: [id], onDelete:
Cascade)

  // Assessment metadata
  assessmentType      String   // "bioenergetic", "metabolic", "comprehensive"
  status              String   @default("active")

  // Core health metrics (Ray Peat methodology)
  overallScore        Float
  energyLevel         Float
  metabolicHealth     Float
  stressLevel         Float
  thyroidFunction     Float
  mitochondrialHealth Float
  hormonalBalance     Float
  inflammationLevel   Float

  // Progressive disclosure data
  keyFindings         Json     // Layer 1
  detailedInsights    Json     // Layer 2
  comprehensiveData   Json     // Layer 3

  // Memory integration (NEW)
  zepSessionId        String?  // Link to Zep session
  memoryContext       Json?    // Cached memory context

  // Engagement tracking
  viewCount           Int      @default(0)
  timeSpent           Int      @default(0)
  layerProgress       Json     // Track exploration

  createdAt           DateTime @default(now())
  updatedAt           DateTime @updatedAt

  @@map("health_assessments")
}

// 2. Biomarker Model (ADD)
model Biomarker {
  id                  String   @id @default(cuid())
  userId              String
  user                User     @relation(fields: [userId], references: [id], onDelete:
Cascade)

  // Biomarker details
  name                String
  value               Float
  unit                String
  category            String   // "metabolic", "hormonal", "inflammatory", "nutrition-
al"
```

```
  // Ray Peat optimized reference ranges
  optimalMin          Float?
  optimalMax          Float?
  normalMin           Float?
  normalMax           Float?
  rayPeatContext      String?

  // Analysis
  status              String   // "optimal", "suboptimal", "concerning", "critical"
  trend               String?  // "improving", "stable", "declining"
  significance        String?

  // Memory enhancement
  historicalValues    Json?    // Previous values for trending
  personalizedRanges  Json?    // User-specific optimal ranges

  createdAt           DateTime @default(now())
  updatedAt           DateTime @updatedAt

  @@map("biomarkers")
}

// 3. Extend User Model (MODIFY EXISTING)
model User {
  // Existing fields (PRESERVE ALL)
  id        String   @id @default(cuid())
  email     String   @unique
  name      String?
  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt

  // Existing relationships (PRESERVE ALL)
  analyses          Analysis[]
  userStats         UserStats?
  // ... all other existing relationships

  // ADD health-specific relationships
  healthAssessments HealthAssessment[]
  biomarkers        Biomarker[]
  consultations     Consultation[]
  achievements      Achievement[]
}
```

4. API ENDPOINT INTEGRATION

Health Analysis API (MERGE COMPLEX)

```
// SOURCE: ~/correct-lablens/app/api/health-analysis/
// TARGET: ~/biospark-health-ai/app/api/comprehensive-analysis/route.ts
// ACTION: MERGE - Combine old system Ray Peat analysis with new system capabilities

// EXISTING NEW SYSTEM API (PRESERVE)
export async function POST(request: Request) {
  // Existing comprehensive analysis logic (PRESERVE)
  const existingAnalysis = await comprehensiveAnalysis.analyze(data);

  // ADD Ray Peat methodology from old system
  const rayPeatAnalysis = await healthAI.generateHealthInsights(data);

  // ADD AbacusAI integration from old system
  const abacusInsights = await abacusAI.runHealthAssessment(data);

  // ADD memory enhancement
  const memoryContext = await memoryManager.getHealthContext(sessionId);

  // COMBINE all analyses with progressive disclosure structure
  const integratedResults = {
    // Layer 1: Health Snapshot (Quick insights)
    healthSnapshot: {
      overallScore: rayPeatAnalysis.overallScore,
      keyFindings: rayPeatAnalysis.keyFindings,
      urgentActions: existingAnalysis.criticalFindings,
      memoryEnhanced: memoryContext.personalizedInsights
    },

    // Layer 2: Detailed Insights (Contextual analysis)
    detailedInsights: {
      rayPeatAnalysis: rayPeatAnalysis.detailedInsights,
      biomarkerInterpretation: existingAnalysis.biomarkers,
      contextualExplanations: rayPeatAnalysis.explanations,
      memoryContext: memoryContext.historicalPatterns
    },

    // Layer 3: Comprehensive Analysis (Professional grade)
    comprehensiveAnalysis: {
      researchReferences: rayPeatAnalysis.researchReferences,
      technicalDetails: existingAnalysis.technicalDetails,
      abacusInsights: abacusInsights,
      predictiveModeling: memoryContext.predictiveInsights
    }
  };

  // Store in memory for future context
  await memoryManager.storeHealthAssessment(sessionId, integratedResults);

  return Response.json(integratedResults);
}
```

New Health-Specific Endpoints (CREATE)

```
// CREATE: ~/biospark-health-ai/app/api/health-assessment/route.ts
// SOURCE: Logic from old system health operations

export async function POST(request: Request) {
  // Health assessment creation with memory integration
}

export async function GET(request: Request) {
  // Retrieve user health assessments with memory context
}

// CREATE: ~/biospark-health-ai/app/api/biomarkers/route.ts
export async function POST(request: Request) {
  // Biomarker tracking with Ray Peat interpretation
}

// CREATE: ~/biospark-health-ai/app/api/ray-peat-analysis/route.ts
export async function POST(request: Request) {
  // Dedicated Ray Peat methodology analysis
}
```

---

5. UI COMPONENT INTEGRATION STRATEGY

Page-Level Integration

```
// SOURCE: ~/correct-lablens/app/page.tsx
// TARGET: ~/biospark-health-ai/app/page.tsx
// ACTION: ENHANCE - Add health-focused landing page elements

// PRESERVE existing new system landing page
// ADD health-focused sections from old system

export default function HomePage() {
  return (
    <div className="min-h-screen bg-white">
      {/* PRESERVE existing header */}
      <Header />

      {/* PRESERVE existing hero section */}
      <HeroSection />

      {/* ADD from old system - Health-focused features */}
      <HealthFeaturesSection />

      {/* ADD from old system - Progressive disclosure benefits */}
      <ProgressiveDisclosureBenefits />

      {/* PRESERVE existing sections */}
      <ExistingSections />

      {/* ENHANCE with health-focused CTA */}
      <HealthCTASection />

      {/* PRESERVE existing footer */}
      <Footer />
    </div>
  );
}
```

Dashboard Integration

```
// SOURCE: ~/correct-lablens/app/dashboard/page.tsx
// TARGET: ~/biospark-health-ai/app/dashboard/page.tsx
// ACTION: MAJOR ENHANCEMENT - Integrate progressive disclosure dashboard

// PRESERVE existing dashboard structure
// ADD health assessment dashboard from old system

export default function DashboardPage() {
  return (
    <div className="dashboard-container">
      {/* PRESERVE existing dashboard navigation */}
      <DashboardNav />

      {/* ADD health assessment section */}
      <HealthAssessmentSection>
        <ProgressiveDisclosureInterface />
        <MemoryEnhancedRecommendations />
        <RayPeatAnalysisDisplay />
      </HealthAssessmentSection>

      {/* PRESERVE existing dashboard sections */}
      <ExistingDashboardSections />
    </div>
  );
}
```

---

6. UTILITY AND HELPER INTEGRATION

Type Definitions (MERGE)

```
// SOURCE: ~/correct-lablens/lib/types.ts
// TARGET: ~/biospark-health-ai/lib/types.ts
// ACTION: MERGE - Add health-specific types to existing types

// PRESERVE all existing types
// ADD health-specific types from old system

// Health Assessment Types
export interface HealthAssessment {
  id: string;
  userId: string;
  assessmentType: 'bioenergetic' | 'metabolic' | 'comprehensive';
  overallScore: number;
  energyLevel: number;
  metabolicHealth: number;
  stressLevel: number;
  thyroidFunction: number;
  mitochondrialHealth: number;
  hormonalBalance: number;
  inflammationLevel: number;
  keyFindings: Json;
  detailedInsights: Json;
  comprehensiveData: Json;
  zepSessionId?: string;
  memoryContext?: Json;
}

// Progressive Disclosure Types
export interface ProgressiveDisclosureData {
  layer1: HealthSnapshot;
  layer2: DetailedInsights;
  layer3: ComprehensiveAnalysis;
}

// Ray Peat Methodology Types
export interface RayPeatAnalysis {
  thyroidOptimization: ThyroidAnalysis;
  metabolicEfficiency: MetabolicAnalysis;
  hormonalBalance: HormonalAnalysis;
  inflammationAssessment: InflammationAnalysis;
  recommendations: RayPeatRecommendation[];
}

// Memory Enhancement Types
export interface MemoryEnhancedInsight {
  insight: string;
  confidence: number;
  historicalContext: string;
  personalizedRecommendation: string;
  memorySource: 'short_term' | 'long_term' | 'pattern_recognition';
}
```

Utility Functions (ADD)

```
// CREATE: ~/biospark-health-ai/lib/health-utils.ts
// SOURCE: Utility functions from old system

// Ray Peat calculation utilities
export const rayPeatUtils = {
  calculateMetabolicScore(biomarkers: Biomarker[]): number,
  interpretThyroidFunction(tsh: number, freeT3: number, freeT4: number): ThyroidAnalys-
is,
  generateOptimalRanges(userProfile: UserProfile): OptimalRanges,
  calculateBioenergeticHealth(assessment: HealthAssessment): number
};

// Progressive disclosure utilities
export const progressiveDisclosureUtils = {
  determineUserLayer(engagementHistory: EngagementHistory): number,
  generateCuriosityGaps(topic: string, userLevel: string): string[],
  trackLayerProgression(userId: string, layer: number): void,
  personalizeDisclosure(userPreferences: UserPreferences): DisclosureConfig
};

// Memory integration utilities
export const memoryUtils = {
  formatHealthContextForMemory(assessment: HealthAssessment): MemoryContext,
  extractInsightsFromMemory(memoryData: ZepMemoryData): PersonalizedInsight[],
  generateContextualRecommendations(memoryContext: MemoryContext, currentData: any): Re
commendation[]
};
```

---

7. CONFIGURATION AND ENVIRONMENT

Environment Variables (ADD)

```
# ADD to ~/biospark-health-ai/.env
# Health AI specific variables from old system

# AbacusAI Integration (from old system)
ABACUSAI_API_KEY=your_abacus_api_key
ABACUS_HEALTH_RISK_MODEL_ID=your_model_id
ABACUS_METABOLIC_MODEL_ID=your_metabolic_model_id
ABACUS_PERSONALIZATION_MODEL_ID=your_personalization_model_id
ABACUS_ANOMALY_MODEL_ID=your_anomaly_model_id

# Ray Peat Knowledge Base
RAY_PEAT_KNOWLEDGE_BASE_URL=your_knowledge_base_url

# Progressive Disclosure Configuration
PROGRESSIVE_DISCLOSURE_ENABLED=true
DEFAULT_USER_LAYER=1
CURIOSITY_GAP_GENERATION=true

# Health Data Retention (HIPAA Compliance)
HEALTH_DATA_RETENTION_DAYS=2555  # 7 years
BIOMARKER_HISTORY_RETENTION_DAYS=1095  # 3 years
```

Next.js Configuration (ENHANCE)

```
// ENHANCE: ~/biospark-health-ai/next.config.js
// ADD health-specific optimizations

/** @type {import('next').NextConfig} */
const nextConfig = {
  // PRESERVE existing configuration
  experimental: {
    serverComponentsExternalPackages: ['@prisma/client'],
  },

  // ADD health analysis optimizations
  webpack: (config, { isServer }) => {
    // PRESERVE existing webpack config

    // ADD health analysis bundle optimization
    if (!isServer) {
      config.optimization.splitChunks.cacheGroups.healthAnalysis = {
        test: /[\\/]lib[\\/](openai|abacus|health|ray-peat)[\\/]/,
        name: 'health-analysis',
        chunks: 'all',
      };

      config.optimization.splitChunks.cacheGroups.progressiveDisclosure = {
        test: /[\\/]components[\\/]health[\\/]/,
        name: 'progressive-disclosure',
        chunks: 'all',
      };
    }

    return config;
  },

  // ADD health data security headers
  headers: async () => [
    // PRESERVE existing headers
    ...existingHeaders,

    // ADD health-specific security
    {
      source: '/api/health-analysis/:path*',
      headers: [
        {
          key: 'X-Health-Data-Protection',
          value: 'HIPAA-Compliant',
        },
        {
          key: 'Cache-Control',
          value: 'no-store, no-cache, must-revalidate',
        },
      ],
    },
  ],
};

module.exports = nextConfig;
```

📋 INTEGRATION CHECKLIST

Phase 1: Core Library Integration ✅

- [ ] **OpenAI Integration:** Merge Ray Peat methodology with existing client
- [ ] **Supabase Integration:** Extend with health-specific operations
- [ ] **AbacusAI Integration:** Add complete AbacusAI client to new system
- [ ] **Type Definitions:** Merge health types with existing types
- [ ] **Utility Functions:** Add Ray Peat and progressive disclosure utilities

Phase 2: Component Integration ✅

- [ ] **Health Snapshot:** Port and enhance with memory integration
- [ ] **Detailed Insights:** Port and enhance with contextual analysis
- [ ] **Comprehensive Analysis:** Major enhancement with memory and research
- [ ] **Progressive Disclosure:** Complete transplant with memory enhancement
- [ ] **UI Components:** Port all health-specific UI components

Phase 3: Database Integration ✅

- [ ] **Schema Extension:** Add health models to existing Prisma schema
- [ ] **Migration Scripts:** Create additive-only migration scripts
- [ ] **Data Relationships:** Establish proper foreign key relationships
- [ ] **Indexes:** Add performance indexes for health queries
- [ ] **Constraints:** Add data validation constraints

Phase 4: API Integration ✅

- [ ] **Health Analysis API:** Merge with existing comprehensive analysis
- [ ] **New Endpoints:** Create health-specific API endpoints
- [ ] **Memory Integration:** Add Zep memory to all health APIs
- [ ] **Error Handling:** Implement robust error handling
- [ ] **Rate Limiting:** Add appropriate rate limiting for health APIs

Phase 5: Configuration & Deployment ✅

- [ ] **Environment Variables:** Add all health-specific environment variables
- [ ] **Next.js Configuration:** Enhance with health optimizations
- [ ] **Security Headers:** Add health data protection headers
- [ ] **Performance Optimization:** Implement health-specific optimizations
- [ ] **Monitoring:** Add health-specific monitoring and alerting

---

🎯 SUCCESS VALIDATION

Component Integration Success Criteria

- [ ] All old system health components successfully ported
- [ ] Progressive disclosure system fully operational
- [ ] Ray Peat methodology preserved and enhanced
- [ ] Memory integration provides contextual insights
- [ ] Performance targets met (Layer transitions <200ms)

API Integration Success Criteria

- [ ] All health analysis endpoints operational
- [ ] Memory-enhanced recommendations working
- [ ] AbacusAI integration providing insights
- [ ] Error rates <1% for health APIs
- [ ] Response times <500ms for all health endpoints

Database Integration Success Criteria

- [ ] All health models successfully added
- [ ] No breaking changes to existing functionality
- [ ] Data relationships properly established
- [ ] Query performance optimized
- [ ] HIPAA compliance maintained

This component mapping guide provides file-by-file integration instructions for successfully transplanting biospark33/lablens functionality into biospark33/biospark-health-ai with 95%+ precision and zero data loss.