Digital Health Competency Framework - Product Requirements Document (PRD)

Executive Summary

The Digital Health Competency Framework is a comprehensive system for organizing, viewing, and managing digital health competencies, roles, and their relationships. This MVP will provide a web-based tool for exploring competency frameworks, mapping roles to competencies, and eventually mapping courses to competencies for training recommendations.

Product Vision

Create an intuitive, flexible platform that enables digital health professionals to:

- Navigate and understand the competency framework structure
- Map roles to required competencies
- Identify training gaps and course recommendations
- Manage and evolve the framework over time

Current Data Inventory

Existing Data

- **9 Domains** (Ethics, Communications, Governance, Health Systems, etc.)
- **41 Subdomains** organized under domains
- **202 Competencies** with detailed performance criteria (+16 added)
- **756 Performance Criteria** providing specific behavioral indicators
- **17 Roles** across 3 categories (Government: 9, Other: 4, World Bank: 4)
- **59 Role-Competency Mappings** for Digital Health Enterprise Architect (example mapping complete)

Missing Data (To Be Created)

- Role-to-competency mappings for remaining 16 roles
- Course catalog
- Course-to-competency mappings

Technical Architecture

Technology Stack

- **Frontend**: React with TypeScript
- **Backend**: Node.js with Express
- **Database**: SQLite (for MVP, PostgreSQL for production)
- **ORM**: Prisma or Drizzle
- **UI Library**: Shadcn/ui or similar
- **State Management**: TanStack Query + React Context

Database Schema

```
sql

-- Core entities (already defined)
domains, subdomains, competencies, performance_criteria, roles

-- New mapping tables needed
role_competencies (role_id, competency_id, is_required, proficiency_level, notes)
courses (course_id, title, description, provider, duration, url, created_at)
course_competencies (course_id, competency_id, coverage_level, notes)
```

Feature Requirements

Phase 1: Core Browsing & Management (MVP)

1.1 Competency Framework Explorer

Priority: High

- **Navigation Tree**: Expandable tree view showing Domains → Subdomains → Competencies
- **Competency Detail View**: Display competency title, statement, and performance criteria
- **Search & Filter**: Search across competencies, filter by domain/subdomain
- **Breadcrumb Navigation**: Clear navigation path

1.2 Role Management

Priority: High

- **Role Catalog**: List all roles organized by type (Government, Other, World Bank)
- **Role Detail View**: Show role description, functions, actions, tasks
- **CRUD Operations**: Add, edit, delete roles
- **Role Search**: Find roles by title, type, or description keywords

1.3 Role-Competency Mapping

Priority: High

- **Mapping Interface**: Drag-and-drop or multi-select interface for assigning competencies to roles
- **Competency Selection**: Browse competencies by domain/subdomain during mapping
- **Mapping Overview**: View all competencies assigned to a specific role (see example: Digital Health Enterprise Architect with 59 mapped competencies)
- **Bulk Operations**: Select multiple competencies for batch assignment
- **Mapping Validation**: Ensure all role mappings are marked as "Required" (MVP scope)
- **Copy Mappings**: Use existing role mappings as templates for similar roles

1.4 Course Management (Foundation)

Priority: Medium

- **Course Catalog**: Basic CRUD for courses (title, description, provider, duration)
- **Course Detail View**: Display course information and metadata
- **Course Search**: Find courses by title, provider, or keywords

Phase 2: Advanced Features

2.1 Proficiency Levels (Future Enhancement)

Priority: Low (Post-MVP)

- **Proficiency Framework**: Add Basic/Intermediate/Advanced levels to competency mappings
- **Role Proficiency Requirements**: Specify required proficiency level per competency for each role
- **Progressive Learning Paths**: Course sequences that build from basic to advanced proficiency

2.2 Course Management & Mapping

Priority: Medium

- **Course Catalog**: Comprehensive CRUD for courses (title, description, provider, duration)
- **Course-Competency Mapping**: Map courses to competencies they address
- **Coverage Levels**: Indicate how well a course covers each competency

2.3 Recommendations Engine

Priority: Medium

- **Role-Based Recommendations**: Suggest courses for a given role based on required competencies
- **Learning Paths**: Sequence courses to build competency progressively
- **Gap Identification**: Show which competencies a role needs that aren't covered by available courses

2.4 Reporting & Analytics

Priority: Low

- **Competency Coverage Reports**: Which competencies are most/least covered by courses
- **Role Analysis**: Competency requirements by role type
- **Training Gap Analysis**: Identify areas needing more course development

User Experience Requirements

Key User Journeys

Journey 1: Explore Competency Framework

1. User lands on homepage with domain overview
2. User clicks on domain to see subdomains
3. User selects subdomain to view competencies
4. User clicks competency to see detailed performance criteria
5. User can search across all competencies

Journey 2: Map Role to Competencies

1. User navigates to Roles section
2. User selects a role or creates new role
3. User opens competency mapping interface
4. User browses/searches competencies and selects relevant ones
5. User can copy mappings from similar roles (e.g., use Digital Health Enterprise Architect as template)
6. User saves mapping and reviews role profile with all required competencies

Journey 3: Find Courses for Role

1. User selects a role
2. System shows required competencies for that role
3. User clicks "Find Courses" for the role
4. System displays courses that address role's competencies
5. User can see coverage gaps and add new courses

Interface Requirements

Navigation

- **Primary Navigation**: Competencies | Roles | Courses | Mappings
- **Secondary Navigation**: Domain-based navigation within competencies
- **Search**: Global search across all content types

Design Principles

- **Clean & Professional**: Healthcare-appropriate design
- **Hierarchical**: Clear information hierarchy (Domain → Subdomain → Competency)
- **Actionable**: Easy access to edit/map functions
- **Responsive**: Works on desktop and tablet

API Requirements

Core Endpoints

```
GET  /api/domains - List all domains
GET  /api/domains/:id/subdomains - Get subdomains for domain
GET  /api/competencies - List/search competencies
GET  /api/competencies/:id - Get competency with performance criteria
GET  /api/roles - List/search roles
POST /api/roles - Create new role
PUT  /api/roles/:id - Update role
GET  /api/roles/:id/competencies - Get competencies for role
POST /api/roles/:id/competencies - Map competencies to role
GET  /api/courses - List/search courses
POST /api/courses - Create new course
GET  /api/roles/:id/recommended-courses - Get course recommendations
```

Data Import/Export Requirements

Import Features

- **Bulk Role Import**: CSV upload for multiple roles
- **Competency Updates**: Import updated competency frameworks
- **Mapping Import**: Bulk import of role-competency mappings

Export Features

- **Framework Export**: Export complete competency framework
- **Role Profiles**: Export individual role with competencies
- **Gap Reports**: Export training gap analysis

Success Metrics

MVP Success Criteria

- All existing data successfully loaded and browsable (202 competencies, 17 roles)
- Role-competency mapping interface functional (building on Digital Health Enterprise Architect example)
- 10+ additional roles successfully mapped to competencies
- All 17 roles have competency mappings completed
- Core user journeys completed without friction
- Template/copy functionality working for efficient role mapping

Usage Metrics

- Time to complete competency mapping for a role
- Search success rate (users finding desired competencies)
- Course recommendation relevance (user feedback)

Implementation Timeline

Phase 1 (4-6 weeks): Foundation

- Week 1-2: Database setup, data migration, basic API
- Week 3-4: React app setup, competency browser, role management
- Week 5-6: Role-competency mapping interface, testing

Phase 2 (3-4 weeks): Courses & Recommendations

- Week 7-8: Course management, course-competency mapping
- Week 9-10: Recommendation engine, gap analysis, polish

Technical Considerations

Performance

- Lazy loading for large competency lists
- Efficient queries with proper indexing
- Client-side caching for static data

Scalability

- SQLite sufficient for MVP (<1000 records per table)
- Easy migration path to PostgreSQL
- API designed for future multi-tenancy

Security

- Input validation and sanitization
- CORS configuration for local development
- Prepared statements for SQL injection prevention

Risk Assessment

Technical Risks

- **Data Complexity**: Complex hierarchical relationships may impact performance
- **Mapping UX**: Role-competency mapping interface needs intuitive design
- **Search Performance**: Full-text search across large datasets

Mitigation Strategies

- Prototype mapping interface early for user feedback
- Implement caching and indexing from start
- Use proven UI patterns for hierarchical data

Next Steps

1. **Review & Approve PRD**: Team alignment on scope and approach
2. **Technical Setup**: Initialize React/Node.js project structure
3. **Database Migration**: Import existing CSV data to SQLite
4. **Core API Development**: Build foundational endpoints
5. **UI Prototyping**: Create key interface mockups
6. **Iterative Development**: Build and test core features

Appendix

Sample Data Structure

- 9 domains spanning ethics to research
- 202 competencies across 41 subdomains (average 20+ competencies per domain)
- 17 roles across government, private, and international sectors
- 756 detailed performance criteria providing behavioral indicators
- Complete role-competency mapping example: Digital Health Enterprise Architect (59 competencies mapped across all domains)

Key Terminology

- **Domain**: High-level competency area (e.g., "Ethics and Equity")
- **Subdomain**: Specific focus within domain (e.g., "Critical Ethical Thinking")
- **Competency**: Specific skill or knowledge area (e.g., "Describe basic ethical concepts")
- **Performance Criteria**: Observable behaviors demonstrating competency
- **Role**: Job function requiring specific competencies
- **Course**: Training that develops specific competencies