Week 1-2: Foundation & Domain Structure

Project Structure Overview

```
src/
├── domains/          # Business domains (core business logic)
├── shared/           # Cross-cutting concerns
├── app/              # Application shell
├── types/            # Global TypeScript definitions
└── assets/           # Static assets
```

Detailed File Structure

1. Domains Directory

Each domain represents a core business capability and follows consistent internal structure.

```
src/domains/
├── threats/
│   ├── components/
│   │   ├── atoms/
│   │   │   ├── ThreatIcon.tsx
│   │   │   ├── ThreatBadge.tsx
│   │   │   └── index.ts
│   │   ├── molecules/
│   │   │   ├── ThreatCard.tsx
│   │   │   ├── ThreatSearchBar.tsx
│   │   │   └── index.ts
│   │   ├── organisms/
│   │   │   ├── ThreatMatrix.tsx
│   │   │   ├── MitreAttackNavigator.tsx
│   │   │   ├── ThreatIntelligencePanel.tsx
│   │   │   └── index.ts
│   │   └── index.ts
│   ├── context/
│   │   ├── ThreatContext.tsx
│   │   ├── ThreatProvider.tsx
│   │   └── index.ts
│   ├── hooks/
│   │   ├── useThreatData.ts
│   │   ├── useThreatFilter.ts
│   │   ├── useMitreMapping.ts
│   │   └── index.ts
│   ├── services/
│   │   ├── ThreatService.ts
│   │   ├── MitreAttackService.ts
│   │   └── index.ts
│   ├── models/
│   │   ├── Threat.ts
│   │   ├── MitreAttack.ts
│   │   └── index.ts
│   ├── utils/
│   │   ├── threatCalculations.ts
│   │   ├── mitreMapping.ts
│   │   └── index.ts
│   └── index.ts
├── requirements/
│   ├── components/
│   │   ├── atoms/
│   │   │   ├── RequirementStatus.tsx
│   │   │   ├── PriorityBadge.tsx
│   │   │   ├── ComplianceIcon.tsx
│   │   │   └── index.ts
│   │   ├── molecules/
│   │   │   ├── RequirementRow.tsx
│   │   │   ├── RequirementCard.tsx
│   │   │   ├── RequirementFilter.tsx
│   │   │   └── index.ts
│   │   ├── organisms/
│   │   │   ├── RequirementsTable.tsx
│   │   │   ├── RequirementsMatrix.tsx
│   │   │   ├── FrameworkMapper.tsx
│   │   │   └── index.ts
│   │   └── index.ts
│   ├── context/
│   │   ├── RequirementsContext.tsx
│   │   ├── RequirementsProvider.tsx
│   │   └── index.ts
│   ├── hooks/
│   │   ├── useRequirements.ts
│   │   ├── useRequirementFilters.ts
│   │   ├── useFrameworkMapping.ts
│   │   └── index.ts
│   ├── services/
│   │   ├── RequirementService.ts
│   │   ├── FrameworkService.ts
│   │   └── index.ts
│   ├── models/
│   │   ├── Requirement.ts
│   │   ├── Framework.ts
│   │   └── index.ts
│   ├── utils/
│   │   ├── requirementValidation.ts
│   │   ├── frameworkMapping.ts
│   │   └── index.ts
│   └── index.ts
├── capabilities/
│   ├── components/
│   │   ├── atoms/
│   │   │   ├── CapabilityIcon.tsx
│   │   │   ├── MaturityBadge.tsx
│   │   │   ├── ProgressIndicator.tsx
│   │   │   └── index.ts
│   │   ├── molecules/
│   │   │   ├── CapabilityCard.tsx
│   │   │   ├── CapabilityMetrics.tsx
│   │   │   ├── CapabilityFilter.tsx
│   │   │   └── index.ts
│   │   ├── organisms/
│   │   │   ├── CapabilitiesList.tsx
│   │   │   ├── CapabilityMatrix.tsx
│   │   │   ├── CapabilityAssessment.tsx
│   │   │   └── index.ts
│   │   └── index.ts
│   ├── context/
│   │   ├── CapabilitiesContext.tsx
│   │   ├── CapabilitiesProvider.tsx
│   │   └── index.ts
│   ├── hooks/
│   │   ├── useCapabilities.ts
│   │   ├── useCapabilityAssessment.ts
│   │   ├── useMaturityScoring.ts
│   │   └── index.ts
│   ├── services/
│   │   ├── CapabilityService.ts
│   │   ├── AssessmentService.ts
│   │   └── index.ts
│   ├── models/
│   │   ├── Capability.ts
│   │   ├── Assessment.ts
│   │   └── index.ts
│   ├── utils/
│   │   ├── maturityCalculations.ts
│   │   ├── capabilityMapping.ts
│   │   └── index.ts
│   └── index.ts
├── risks/
│   ├── components/
│   │   ├── atoms/
│   │   │   ├── RiskIndicator.tsx
│   │   │   ├── SeverityBadge.tsx
│   │   │   ├── ImpactIcon.tsx
│   │   │   └── index.ts
│   │   ├── molecules/
│   │   │   ├── RiskCard.tsx
│   │   │   ├── RiskMetrics.tsx
│   │   │   ├── RiskFilter.tsx
│   │   │   └── index.ts
│   │   ├── organisms/
│   │   │   ├── RiskMatrix.tsx
│   │   │   ├── RiskHeatmap.tsx
│   │   │   ├── RiskAssessment.tsx
│   │   │   └── index.ts
│   │   └── index.ts
│   ├── context/
│   │   ├── RiskContext.tsx
│   │   ├── RiskProvider.tsx
│   │   └── index.ts
│   ├── hooks/
│   │   ├── useRiskData.ts
│   │   ├── useRiskCalculation.ts
│   │   ├── useRiskAssessment.ts
│   │   └── index.ts
│   ├── services/
│   │   ├── RiskService.ts
│   │   ├── CalculationService.ts
│   │   └── index.ts
│   ├── models/
│   │   ├── Risk.ts
│   │   ├── RiskAssessment.ts
│   │   └── index.ts
│   ├── utils/
│   │   ├── riskCalculations.ts
│   │   ├── probabilityModeling.ts
│   │   └── index.ts
│   └── index.ts
├── evidence/
│   ├── components/
│   │   ├── atoms/
│   │   │   ├── EvidenceIcon.tsx
│   │   │   ├── ValidityBadge.tsx
│   │   │   ├── FreshnessIndicator.tsx
│   │   │   └── index.ts
│   │   ├── molecules/
│   │   │   ├── EvidenceCard.tsx
│   │   │   ├── EvidenceUpload.tsx
│   │   │   ├── EvidenceValidation.tsx
│   │   │   └── index.ts
│   │   ├── organisms/
│   │   │   ├── EvidenceLibrary.tsx
│   │   │   ├── EvidenceChain.tsx
│   │   │   ├── EvidenceAudit.tsx
│   │   │   └── index.ts
│   │   └── index.ts
│   ├── context/
│   │   ├── EvidenceContext.tsx
│   │   ├── EvidenceProvider.tsx
│   │   └── index.ts
│   ├── hooks/
│   │   ├── useEvidence.ts
│   │   ├── useEvidenceValidation.ts
│   │   ├── useEvidenceChain.ts
│   │   └── index.ts
│   ├── services/
│   │   ├── EvidenceService.ts
│   │   ├── ValidationService.ts
│   │   └── index.ts
│   ├── models/
│   │   ├── Evidence.ts
│   │   ├── EvidenceType.ts
│   │   └── index.ts
│   ├── utils/
│   │   ├── evidenceValidation.ts
│   │   ├── freshnessCalculations.ts
│   │   └── index.ts
│   └── index.ts
├── trust/
│   ├── components/
│   │   ├── atoms/
│   │   │   ├── TrustScore.tsx
│   │   │   ├── ConfidenceIndicator.tsx
│   │   │   ├── TrendArrow.tsx
│   │   │   └── index.ts
│   │   ├── molecules/
│   │   │   ├── TrustCard.tsx
│   │   │   ├── TrustMetrics.tsx
│   │   │   ├── TrustBreakdown.tsx
│   │   │   └── index.ts
│   │   ├── organisms/
│   │   │   ├── TrustDashboard.tsx
│   │   │   ├── TrustTimeline.tsx
│   │   │   ├── TrustReporting.tsx
│   │   │   └── index.ts
│   │   └── index.ts
│   ├── context/
│   │   ├── TrustContext.tsx
│   │   ├── TrustProvider.tsx
│   │   └── index.ts
│   ├── hooks/
│   │   ├── useTrustScore.ts
│   │   ├── useTrustCalculation.ts
│   │   ├── useTrustTrends.ts
│   │   └── index.ts
│   ├── services/
│   │   ├── TrustService.ts
│   │   ├── ScoringService.ts
│   │   └── index.ts
│   ├── models/
│   │   ├── TrustScore.ts
│   │   ├── TrustMetrics.ts
│   │   └── index.ts
│   ├── utils/
│   │   ├── trustCalculations.ts
│   │   ├── confidenceModeling.ts
│   │   └── index.ts
│   └── index.ts
└── resources/
    ├── components/
    │   ├── atoms/
    │   │   ├── ResourceIcon.tsx
    │   │   ├── UtilizationBadge.tsx
    │   │   ├── SkillLevel.tsx
    │   │   └── index.ts
    │   ├── molecules/
    │   │   ├── ResourceCard.tsx
    │   │   ├── ResourceAllocation.tsx
    │   │   ├── SkillMatrix.tsx
    │   │   └── index.ts
    │   ├── organisms/
    │   │   ├── ResourceSwimlane.tsx
    │   │   ├── ResourcePlanning.tsx
    │   │   ├── CapacityPlanning.tsx
    │   │   └── index.ts
    │   └── index.ts
    ├── context/
    │   ├── ResourceContext.tsx
    │   ├── ResourceProvider.tsx
    │   └── index.ts
    ├── hooks/
    │   ├── useResources.ts
    │   ├── useResourcePlanning.ts
    │   ├── useCapacityPlanning.ts
    │   └── index.ts
    ├── services/
    │   ├── ResourceService.ts
    │   ├── PlanningService.ts
    │   └── index.ts
    ├── models/
    │   ├── Resource.ts
    │   ├── ResourcePlan.ts
    │   └── index.ts
    ├── utils/
    │   ├── resourceCalculations.ts
    │   ├── capacityModeling.ts
    │   └── index.ts
    └── index.ts
```

2. Shared Directory

Cross-cutting concerns used across domains.

```
src/shared/
├── components/
│   ├── atoms/
│   │   ├── Button/
│   │   │   ├── Button.tsx
│   │   │   ├── Button.stories.tsx
│   │   │   ├── Button.test.tsx
│   │   │   └── index.ts
│   │   ├── Input/
│   │   │   ├── Input.tsx
│   │   │   ├── Input.stories.tsx
│   │   │   ├── Input.test.tsx
│   │   │   └── index.ts
│   │   ├── Icon/
│   │   │   ├── Icon.tsx
│   │   │   ├── IconLibrary.ts
│   │   │   └── index.ts
│   │   ├── Badge/
│   │   │   ├── Badge.tsx
│   │   │   ├── Badge.stories.tsx
│   │   │   └── index.ts
│   │   ├── LoadingSpinner/
│   │   ├── Tooltip/
│   │   ├── Avatar/
│   │   └── index.ts
│   ├── molecules/
│   │   ├── SearchBar/
│   │   │   ├── SearchBar.tsx
│   │   │   ├── SearchBar.stories.tsx
│   │   │   └── index.ts
│   │   ├── FilterDropdown/
│   │   ├── Pagination/
│   │   ├── DatePicker/
│   │   ├── Modal/
│   │   ├── Notification/
│   │   └── index.ts
│   ├── organisms/
│   │   ├── DataTable/
│   │   │   ├── DataTable.tsx
│   │   │   ├── DataTableColumn.tsx
│   │   │   ├── DataTableRow.tsx
│   │   │   └── index.ts
│   │   ├── Navigation/
│   │   │   ├── Sidebar.tsx
│   │   │   ├── TopBar.tsx
│   │   │   ├── Breadcrumbs.tsx
│   │   │   └── index.ts
│   │   ├── Dashboard/
│   │   │   ├── DashboardGrid.tsx
│   │   │   ├── DashboardCard.tsx
│   │   │   └── index.ts
│   │   └── index.ts
│   ├── templates/
│   │   ├── DashboardLayout/
│   │   │   ├── DashboardLayout.tsx
│   │   │   └── index.ts
│   │   ├── AuthLayout/
│   │   ├── ReportLayout/
│   │   └── index.ts
│   └── index.ts
├── contexts/
│   ├── AppContext.tsx
│   ├── ThemeContext.tsx
│   ├── AuthContext.tsx
│   ├── NotificationContext.tsx
│   └── index.ts
├── hooks/
│   ├── useLocalStorage.ts
│   ├── useDebounce.ts
│   ├── useToggle.ts
│   ├── useAsync.ts
│   ├── usePagination.ts
│   ├── useSort.ts
│   ├── useFilter.ts
│   └── index.ts
├── services/
│   ├── api/
│   │   ├── BaseApiService.ts
│   │   ├── ApiClient.ts
│   │   ├── ApiTypes.ts
│   │   └── index.ts
│   ├── storage/
│   │   ├── LocalStorageService.ts
│   │   ├── SessionStorageService.ts
│   │   └── index.ts
│   ├── auth/
│   │   ├── AuthService.ts
│   │   ├── TokenService.ts
│   │   └── index.ts
│   ├── notification/
│   │   ├── NotificationService.ts
│   │   └── index.ts
│   └── index.ts
```

```
 ├── utils/
│   ├── validation/
│   │   ├── validators.ts
│   │   ├── schemas.ts
│   │   └── index.ts
│   ├── formatters/
│   │   ├── dateFormatters.ts
│   │   ├── numberFormatters.ts
│   │   ├── textFormatters.ts
│   │   └── index.ts
│   ├── constants/
│   │   ├── apiEndpoints.ts
│   │   ├── appConstants.ts
│   │   ├── errorMessages.ts
│   │   └── index.ts
│   ├── helpers/
│   │   ├── arrayHelpers.ts
│   │   ├── objectHelpers.ts
│   │   ├── stringHelpers.ts
│   │   └── index.ts
│   └── index.ts
 ├── styles/
│   ├── tokens/
│   │   ├── colors.ts
│   │   ├── typography.ts
│   │   ├── spacing.ts
│   │   ├── shadows.ts
│   │   └── index.ts
│   ├── components/
│   │   ├── globals.css
│   │   ├── reset.css
│   │   └── utilities.css
│   └── index.ts
 └── config/
    ├── theme.ts
    ├── environment.ts
    ├── features.ts
    └── index.ts
```

3. App Directory

Application shell and configuration.

```
src/app/
├── routing/
│   ├── AppRouter.tsx
│   ├── ProtectedRoute.tsx
│   ├── RouteConfig.ts
│   └── index.ts
├── layout/
│   ├── AppLayout.tsx
│   ├── ErrorBoundary.tsx
│   └── index.ts
├── providers/
│   ├── AppProviders.tsx
│   ├── QueryProvider.tsx
│   ├── ErrorProvider.tsx
│   └── index.ts
└── index.ts
```

4. Types Directory

Global TypeScript definitions.

```
src/types/
├── api.ts          # API response/request types
├── common.ts       # Shared utility types
├── entities.ts     # Core business entity types
├── events.ts       # Domain events
├── permissions.ts  # Authorization types
└── index.ts
```

5. Root Level Files

```
src/
├── index.tsx       # Application entry point
├── App.tsx         # Root App component
└── setupTests.ts   # Test configuration
```

Implementation Order

Day 1-2: Core Structure Setup

1. Create the folder structure
2. Set up basic TypeScript configurations
3. Create initial index.ts files with proper exports

Day 3-4: Shared Foundation

1. Implement base atomic components (Button, Input, Icon)
2. Set up theme and styling tokens
3. Create core contexts (App, Theme, Auth)

Day 5-6: Domain Scaffolding

1. Create domain-specific folder structures
2. Define initial TypeScript interfaces for each domain
3. Set up basic service interfaces

Day 7-8: Context Architecture

1. Implement domain-specific contexts
2. Set up data flow patterns
3. Create basic hooks for each domain

Day 9-10: Migration Planning

1. Map existing components to new structure
2. Create migration scripts for moving code
3. Set up build and test configurations

This structure provides:

- **Clear separation of concerns** between domains
- **Atomic design consistency** across all components
- **Scalable architecture** for adding new features
- **Type safety** throughout the application
- **Testability** with isolated, mockable services
- **Maintainability** through consistent patterns

Would you like me to detail the specific implementation of any particular domain or shared component?