3D Dungeon Ecosystem Engine - Project Structure

```
dungeon-ecosystem-3d/
├── index.html                    # Main entry point
├── package.json                  # Dependencies and scripts
├── webpack.config.js             # Build configuration
├── README.md                     # Project documentation
├── .gitignore                    # Git ignore rules
│
├── src/                          # Source code
│   ├── main.js                   # Application entry point
│   ├── config/
│   │   ├── constants.js          # Global constants
│   │   ├── settings.js           # Configurable settings
│   │   └── species-config.js     # Species definitions
│   │
│   ├── core/                     # Core engine systems
│   │   ├── Engine.js             # Main engine class
│   │   ├── Time.js               # Time management
│   │   ├── EventSystem.js        # Event dispatcher
│   │   ├── ResourceManager.js    # Asset loading and caching
│   │   └── StateManager.js       # Application state
│   │
│   ├── rendering/                # 3D Rendering systems
│   │   ├── Renderer.js           # Main renderer
│   │   ├── WebGLContext.js       # WebGL setup and management
│   │   ├── ShaderManager.js      # Shader compilation and caching
│   │   ├── Camera.js             # Camera system
│   │   ├── Scene.js              # Scene graph management
│   │   ├── Mesh.js               # Mesh data structure
│   │   ├── Material.js           # Material properties
│   │   ├── Texture.js            # Texture management
│   │   ├── Light.js              # Lighting system
│   │   ├── RenderQueue.js        # Render order optimization
│   │   └── PostProcessing.js     # Post-processing effects
│   │
│   ├── geometry/                 # Geometry generation
│   │   ├── GeometryBuilder.js    # Base geometry utilities
│   │   ├── PrimitiveGenerator.js # Basic shapes (cube, sphere, etc.)
│   │   ├── DungeonGeometry.js    # Dungeon room/corridor generation
│   │   ├── TerrainGenerator.js   # Organic terrain surfaces
│   │   └── MeshOptimizer.js      # Mesh optimization utilities
│   │
│   ├── math/                     # Mathematics utilities
│   │   ├── Vector3.js            # 3D vector operations
```

```
│   │   ├── Matrix4.js              # 4x4 matrix operations
│   │   ├── Quaternion.js           # Rotation quaternions
│   │   ├── Ray.js                  # Ray casting
│   │   ├── AABB.js                 # Axis-aligned bounding boxes
│   │   ├── Plane.js                # Plane mathematics
│   │   └── MathUtils.js            # General math utilities
│   │
│   ├── spatial/                    # Spatial data structures
│   │   ├── Octree.js               # Octree for 3D spatial partitioning
│   │   ├── SpatialHash.js          # Hash-based spatial indexing
│   │   ├── BVH.js                  # Bounding volume hierarchy
│   │   └── CollisionDetection.js   # Collision detection systems
│   │
│   ├── simulation/                 # Ecosystem simulation
│   │   ├── EcosystemManager.js     # Main ecosystem coordinator
│   │   ├── Species.js              # Species definitions and behavior
│   │   ├── Population.js           # Population dynamics
│   │   ├── Environment.js          # Environmental factors
│   │   ├── Room.js                 # Individual room simulation
│   │   ├── Migration.js            # Inter-room migration
│   │   ├── FoodWeb.js              # Predator-prey relationships
│   │   ├── Disease.js              # Disease simulation
│   │   ├── Evolution.js            # Evolutionary pressures
│   │   └── RandomEvents.js         # Random ecological events
│   │
│   ├── entities/                   # Game entities
│   │   ├── Entity.js               # Base entity class
│   │   ├── Creature.js             # Individual creature entities
│   │   ├── CreatureGroup.js        # Grouped creature management
│   │   ├── EnvironmentalObject.js  # Static environmental objects
│   │   ├── Resource.js             # Resource entities (food, water)
│   │   └── Effect.js               # Visual effects entities
│   │
│   ├── behaviors/                  # AI and behavior systems
│   │   ├── BehaviorTree.js         # Behavior tree implementation
│   │   ├── StateMachine.js         # Finite state machines
│   │   ├── Flocking.js             # Boids flocking behavior
│   │   ├── Pathfinding.js          # A* pathfinding
│   │   ├── ForagingBehavior.js     # Food seeking behavior
│   │   ├── PredatorBehavior.js     # Hunting behavior
│   │   ├── TerritorialBehavior.js  # Territory management
│   │   └── MatingBehavior.js       # Reproduction behavior
│   │
│   ├── animation/                  # Animation systems
```

```
│   │   ├── AnimationManager.js        # Animation coordinator
│   │   ├── Keyframe.js                # Keyframe animation
│   │   ├── ProceduralAnimation.js     # Procedural movement
│   │   ├── InverseKinematics.js       # IK for creature legs
│   │   ├── MorphTargets.js            # Morph target animation
│   │   └── AnimationBlending.js       # Animation state blending
│   │
│   ├── generation/                    # Procedural generation
│   │   ├── DungeonGenerator.js        # Main dungeon layout generator
│   │   ├── RoomGenerator.js           # Individual room generation
│   │   ├── CorridorGenerator.js       # Corridor and connection generation
│   │   ├── EnvironmentalZones.js      # Environmental condition placement
│   │   ├── ResourcePlacement.js       # Resource distribution
│   │   ├── NoiseGenerator.js          # Perlin/Simplex noise
│   │   └── BiomeGenerator.js          # Biome-like zone creation
│   │
│   ├── audio/                         # Audio systems
│   │   ├── AudioManager.js            # Audio system coordinator
│   │   ├── SpatialAudio.js            # 3D positional audio
│   │   ├── AmbientSounds.js           # Environmental ambience
│   │   ├── CreatureSounds.js          # Creature vocalizations
│   │   └── ProceduralAudio.js         # Procedural sound generation
│   │
│   ├── ui/                            # User interface
│   │   ├── UIManager.js               # UI system coordinator
│   │   ├── HUD.js                     # Heads-up display
│   │   ├── DebugPanel.js              # Debug information overlay
│   │   ├── EcosystemViewer.js         # Ecosystem data visualization
│   │   ├── TimeControls.js            # Time manipulation controls
│   │   ├── CameraControls.js          # Camera control interface
│   │   └── SpeciesPanel.js            # Species information display
│   │
│   ├── input/                         # Input handling
│   │   ├── InputManager.js            # Input system coordinator
│   │   ├── MouseHandler.js            # Mouse input processing
│   │   ├── KeyboardHandler.js         # Keyboard input processing
│   │   ├── TouchHandler.js            # Touch input for mobile
│   │   └── GamepadHandler.js          # Gamepad support
│   │
│   ├── workers/                       # Web Workers
│   │   ├── SimulationWorker.js        # Main ecosystem simulation worker
│   │   ├── PathfindingWorker.js       # Pathfinding calculations
│   │   ├── GeometryWorker.js          # Procedural geometry generation
│   │   └── AudioWorker.js             # Audio processing worker
```

```
│   │
│   ├── utils/                    # Utility functions
│   │   ├── Logger.js             # Logging system
│   │   ├── Performance.js        # Performance monitoring
│   │   ├── Serialization.js      # Save/load functionality
│   │   ├── Debugging.js          # Debug utilities
│   │   ├── ColorUtils.js         # Color manipulation
│   │   └── ArrayUtils.js         # Array helper functions
│   │
│   └── visualization/            # Data visualization
│       ├── PopulationGraphs.js   # Population trend visualization
│       ├── EnvironmentalOverlay.js  # Environmental data overlay
│       ├── MigrationTrails.js    # Migration path visualization
│       ├── HeatMaps.js           # Heat map rendering
│       ├── ParticleEffects.js    # Particle system effects
│       └── TemporalVisualization.js  # Time-lapse visualization
│
├── assets/                       # Asset files
│   ├── shaders/                  # GLSL shader files
│   │   ├── vertex/
│   │   │   ├── basic.vert        # Basic vertex shader
│   │   │   ├── creature.vert     # Creature vertex shader
│   │   │   ├── terrain.vert      # Terrain vertex shader
│   │   │   ├── water.vert        # Water vertex shader
│   │   │   └── particle.vert     # Particle vertex shader
│   │   ├── fragment/
│   │   │   ├── basic.frag        # Basic fragment shader
│   │   │   ├── creature.frag     # Creature fragment shader
│   │   │   ├── terrain.frag      # Terrain fragment shader
│   │   │   ├── water.frag        # Water fragment shader
│   │   │   ├── particle.frag     # Particle fragment shader
│   │   │   └── postprocess.frag  # Post-processing effects
│   │   └── compute/
│   │       ├── flocking.comp     # Flocking compute shader
│   │       ├── population.comp   # Population simulation
│   │       └── environment.comp  # Environmental simulation
│   │
│   ├── models/                   # 3D model files
│   │   ├── creatures/
│   │   │   ├── beetle.obj        # Beetle base model
│   │   │   ├── spider.obj        # Spider base model
│   │   │   ├── rat.obj           # Rat base model
│   │   │   └── slime.obj         # Slime base model
│   │   ├── environment/
```

```
│   │   │   ├── rock-wall.obj          # Rock wall segments
│   │   │   ├── moss-patch.obj         # Moss coverage
│   │   │   ├── water-pool.obj         # Water bodies
│   │   │   └── debris.obj             # Organic debris
│   │   └── props/
│   │       ├── torch.obj              # Light sources
│   │       ├── bones.obj              # Skeletal remains
│   │       └── crystals.obj           # Mineral formations
│   │
│   ├── textures/                      # Texture files
│   │   ├── creatures/
│   │   │   ├── beetle-diffuse.png     # Beetle texture
│   │   │   ├── spider-diffuse.png     # Spider texture
│   │   │   ├── creature-normal.png    # Normal maps
│   │   │   └── creature-specular.png  # Specular maps
│   │   ├── environment/
│   │   │   ├── stone-diffuse.png      # Stone wall texture
│   │   │   ├── moss-diffuse.png       # Moss texture
│   │   │   ├── water-normal.png       # Water normal map
│   │   │   └── dirt-diffuse.png       # Dirt/soil texture
│   │   └── effects/
│   │       ├── particle-atlas.png     # Particle texture atlas
│   │       ├── noise-3d.png           # 3D noise texture
│   │       └── gradient-lut.png       # Color grading LUT
│   │
│   ├── audio/                         # Audio files
│   │   ├── ambient/
│   │   │   ├── cave-drip.ogg          # Water dripping sounds
│   │   │   ├── wind-tunnel.ogg        # Air flow sounds
│   │   │   └── cave-ambience.ogg      # General cave atmosphere
│   │   ├── creatures/
│   │   │   ├── beetle-chirp.ogg       # Beetle sounds
│   │   │   ├── spider-skitter.ogg     # Spider movement
│   │   │   ├── rat-squeak.ogg         # Rat vocalizations
│   │   │   └── slime-squelch.ogg      # Slime movement
│   │   └── effects/
│   │       ├── birth.ogg              # Birth sound effect
│   │       ├── death.ogg              # Death sound effect
│   │       └── migration.ogg          # Migration sound
│   │
│   └── data/                          # Data files
│       ├── species-definitions.json   # Species parameter definitions
│       ├── biome-templates.json       # Environmental templates
│       ├── behavior-trees.json        # AI behavior definitions
```

```
│   └── dungeon-presets.json       # Preset dungeon configurations
│
├── tools/                         # Development tools
│   ├── asset-pipeline/
│   │   ├── model-converter.js     # Convert models to engine format
│   │   ├── texture-processor.js   # Optimize and process textures
│   │   └── audio-processor.js     # Process audio files
│   ├── editors/
│   │   ├── species-editor.html    # Species parameter editor
│   │   ├── dungeon-editor.html    # Dungeon layout editor
│   │   └── behavior-editor.html   # AI behavior editor
│   └── debugging/
│       ├── profiler.js            # Performance profiling tools
│       ├── ecosystem-debugger.js  # Ecosystem state inspector
│       └── renderer-debugger.js   # Rendering pipeline debugger
│
├── tests/                         # Test files
│   ├── unit/
│   │   ├── math/                  # Math utility tests
│   │   ├── simulation/            # Simulation logic tests
│   │   └── rendering/             # Rendering tests
│   ├── integration/
│   │   ├── ecosystem-integration.test.js
│   │   └── rendering-integration.test.js
│   └── performance/
│       ├── simulation-benchmark.js
│       └── rendering-benchmark.js
│
├── docs/                          # Documentation
│   ├── api/                       # API documentation
│   ├── tutorials/                 # Development tutorials
│   ├── architecture.md            # System architecture overview
│   ├── species-guide.md           # Species behavior documentation
│   └── performance-guide.md       # Performance optimization guide
│
└── build/                         # Build output (generated)
    ├── dist/                      # Distribution files
    ├── temp/                      # Temporary build files
    └── assets/                    # Processed assets
```

Key Architectural Decisions

Modular Design

Each system is self-contained with clear interfaces. The simulation can run independently of rendering, allowing for headless testing and different visualization modes.

Worker-Based Simulation

Heavy simulation work runs in web workers to maintain 60fps rendering while complex ecosystem calculations happen in background threads.

Asset Pipeline

Separate tools process raw assets into optimized engine formats. This allows for artist-friendly input formats while maintaining performance.

Data-Driven Configuration

Species behaviors, environmental parameters, and dungeon generation rules are externalized to JSON files for easy modification without code changes.

Scalable Rendering

The rendering system supports both high-detail individual creature rendering and efficient instanced rendering for population visualization.

Debug-First Development

Extensive debugging and visualization tools are built into the architecture from the start, making development and ecosystem tuning much easier.

This structure supports both the complex ecological simulation and the 3D visualization requirements while maintaining clean separation of concerns and scalability for future features.