**Strategic Technical Leadership Development Initiative**

---

## 🏗️ FIVE FUNDAMENTAL AI IMPLEMENTATION PATTERNS

**Problem statement**

Identify and select the 5 most common use cases where enterprises use or plan on using AI/GenAI. Identify and select the 5 most common industries where enterprises use or plan on using AI/GenAI. Combine the two (use cases and industries) and produce 5 reference implementations in the form of demonstrable ‘steel-thread’ implementations to showcase the use cases in the selected industries and the technologies and methods of consumption to demonstrate technical and business abilities to implement these solutions.

**Research Methodology & Pattern Identification**

The selection of these 5 fundamental AI patterns resulted from comprehensive research conducted through Cyclonix Systems' customer engagement and partner ecosystem analysis, combined with industry research across Fortune 500 enterprises.

**Research Sources**:

- **Cyclonix Customer Interviews**: Direct feedback from enterprise clients across healthcare, financial services, manufacturing, retail, and education sectors regarding their highest-priority AI transformation initiatives
- **Partner Community Analysis**: Collaboration with technology partners and system integrators to identify recurring AI implementation requests and success patterns
- **Industry Research Integration**: Analysis of Gartner, McKinsey, and Deloitte reports on enterprise AI adoption trends and ROI-driven use cases
- **Cross-Vertical Pattern Recognition**: Identification of AI applications with proven business impact across multiple industries

**Selection Criteria**:

- **High Business Impact**: Demonstrable ROI with quantifiable cost savings or revenue generation
- **Cross-Industry Applicability**: Patterns that solve fundamental business problems across multiple verticals
- **Technical Scalability**: Implementations that can scale from proof-of-concept to enterprise deployment
- **Executive Relevance**: Use cases that resonate with C-level priorities and strategic initiatives
- **Implementation Feasibility**: Patterns achievable within constrained development timelines while maintaining production quality

**Validated Industry Applications**: The research identified that these 5 patterns address the most frequently requested AI capabilities across Cyclonx's target verticals, with each pattern showing consistent demand and successful implementation across:

- **Healthcare**: Regulatory compliance, patient care optimization, operational efficiency
- **Financial Services**: Risk management, fraud prevention, personalized customer experience
- **Manufacturing**: Predictive maintenance, quality control, supply chain optimization
- **Retail/E-commerce**: Customer experience, inventory management, demand forecasting
- **Education**: Personalized learning, administrative automation, student engagement

Based on this comprehensive analysis, the following 5 patterns represent the essential building blocks every AI leader must understand through direct implementation:

### Pattern #1: Document Intelligence & Knowledge Extraction (RAG)

**Business Problems**: Healthcare diagnostics, financial compliance, manufacturing quality documentation **Core Implementation**: End-to-end document processing with OCR, embeddings, vector search, and LLM synthesis **Primary Industries**: Healthcare (medical records), Finance (regulatory documents), Manufacturing (quality manuals)

### Pattern #2: Real-Time Anomaly Detection & Alerting

**Business Problems**: Financial fraud detection, predictive maintenance, quality control **Core Implementation**: Streaming data pipeline with ML models detecting outliers and triggering automated responses **Primary Industries**: Finance (transaction monitoring), Manufacturing (equipment failure prediction)

**Pattern #3: Personalization & Recommendation Engine**
**Business Problems**: Retail product recommendations, personalized learning paths, tailored financial services **Core Implementation**: User behavior analysis, collaborative filtering, real-time recommendation serving **Primary Industries**: Retail (e-commerce), EdTech (adaptive learning), Finance (product recommendations)

**Pattern #4: Predictive Analytics & Forecasting**
**Business Problems**: Supply chain optimization, maintenance scheduling, patient outcome prediction **Core Implementation**: Time series forecasting with feature engineering and automated model deployment **Primary Industries**: Manufacturing (demand planning), Healthcare (treatment outcomes), Retail (inventory)

**Pattern #5: Conversational AI with Business Logic Integration**
**Business Problems**: Customer service automation, educational tutoring, operational efficiency **Core Implementation**: Multi-turn dialogue with function calling, business system integration, context management **Primary Industries**: All industries (customer service), EdTech (tutoring), Healthcare (patient interaction)

---

## 🛠 COMPREHENSIVE TECHNOLOGY STACK REQUIREMENTS

### Core Development Environment

- **Primary Language**: Python 3.11+ (relearning/sharpening focus)
- **IDE**: VSCode with AI coding assistance
- **AI Development**: Claude Code for rapid prototyping and development acceleration
- **Version Control**: GitHub with comprehensive repository management

### Cloud Infrastructure & Deployment

- **Primary Cloud**: AWS (expertise deepening focus)
- **Infrastructure as Code**: Terraform for infrastructure provisioning and management
- **Containerization**: Docker for consistent deployment environments
- **Orchestration**: AWS Lambda + ECS managed via Terraform for production deployment

### AI/ML Integration Stack

- **OpenAI Integration**: GPT-4 for natural language processing and reasoning
- **NVIDIA AI Enterprise**: NVIDIA MDI for enterprise-grade AI governance and compliance
- **Vector Databases**: Embedded solutions for RAG implementations
- **ML Operations**: Basic MLOps patterns for model deployment and monitoring

### DevOps & Quality Assurance

- **CI/CD Pipeline**: GitHub Actions with minimum 2-stage pipeline (CI → Production)
- **Testing Framework**: Test-Driven Development (TDD) approach for core functionality
- **Code Quality**: Automated linting, formatting, and security scanning
- **Documentation**: Comprehensive README with setup and demonstration instructions

### Security & Compliance Framework

- **Environment Security**: Core AWS security policies protecting code and infrastructure
- **API Key Management**: Secure credential storage and rotation
- **Data Protection**: Basic encryption and access controls
- **Audit Trails**: Logging and monitoring for governance demonstration

---

## 📝 IMPLEMENTATION STRATEGY & CONSTRAINTS

### "Steel Thread" Implementation Philosophy

- **Thin Vertical Slices**: Complete end-to-end functionality with minimal features
- **Maximum Learning**: Focus on core AI patterns, not comprehensive business features
- **Rapid Execution**: 20-25 hours per implementation maximum
- **Production Ready**: Deployable, testable, demonstrable systems
- **Career Focused**: Each implementation targets specific industry conversations

### Technical Implementation Constraints

- **Timeline**: 20-25 hours maximum per pattern
- **Scope**: Single primary feature with full technical stack
- **Quality**: Production-ready with proper CI/CD and security
- **Demonstration**: Working end-to-end scenarios for business stakeholders
- **Scalability**: Architecture capable of enterprise expansion

**Business Value Focus**

- **Real Problems**: Address actual industry pain points with measurable outcomes
- **Executive Conversation**: Support C-level discussions with concrete examples
- **ROI Demonstration**: Clear cost/time savings and efficiency improvements
- **Industry Relevance**: Align with current market demands and technology adoption

---

## 📚 GITHUB REPOSITORY STRATEGY & PROFESSIONAL POSITIONING

**Strategic Repository Enhancement Overview**

Transform existing GitHub presence from general portfolio to **comprehensive AI/ML engineering reference implementation showcase**. The goal is to publicly demonstrate deep technical knowledge across 5 fundamental AI patterns while maintaining executive-level strategic positioning.

**Repository Structure Strategy**:

- **Main Portfolio**: https://github.com/basilan/bb-engineering-portfolio (strategic overview)
- **Reference Implementations**: Individual repositories for each AI pattern (detailed technical work)
- **Professional Positioning**: Show expertise expansion, not skill acquisition

**Main Portfolio Repository Transformation**

**Current State**: https://github.com/basilan/bb-engineering-portfolio

**Target State**: Executive AI Engineering Portfolio showcasing strategic technical leadership

**New README Requirements**:

# Strategic Objective
Position as senior technology executive who **actively demonstrates** AI/ML expertise through reference implementations, not someone learning basics.

# Professional Narrative
"Expanding my reference implementation portfolio to demonstrate hands-on expertise across the 5 fundamental AI/ML patterns driving enterprise transformation in 2025."

# Repository Purpose
- Showcase technical depth behind strategic AI leadership experience
- Demonstrate current expertise with latest AI/ML frameworks and tools
- Provide concrete examples for executive conversations and team guidance
- Maintain active technical practice alongside strategic leadership roles

**Healthcare AI Governance Agent Repository Strategy**

**Target Repository**: https://github.com/basilan/birigov

**Positioning**: "Reference implementation demonstrating healthcare AI governance patterns combining OpenAI RAG with NVIDIA AI Enterprise compliance frameworks."

**Repository Enhancement Scope**:

1. **Complete README Replacement**: Executive positioning demonstrating hands-on technical leadership across 5 fundamental AI patterns
2. **Architecture Documentation**: Technical depth demonstration with clear business justification
3. **Implementation Guide**: Professional setup instructions showing comprehensive technical proficiency
4. **Business Case Integration**: ROI analysis and cross-industry applicability
5. **Demo Materials**: Executive presentation assets and technical demonstrations

**Professional Positioning Strategy**

**Key Messaging Framework**:

- **NOT**: "Learning AI/ML concepts" or "Building foundational skills"

- **YES**: "Expanding reference implementation portfolio" and "Demonstrating comprehensive technical leadership"
- **NOT**: "First AI project" or "Getting hands-on experience"
- **YES**: "Strategic technical validation" and "Executive-level implementation showcase"

**GitHub Profile Narrative**:

Senior Technology Executive | AI/ML Implementation Leader
Showcasing hands-on expertise across fundamental AI patterns driving enterprise transformation.
20+ years DevOps leadership • 5+ years AI/GenAI integration • $500M+ transformation experience

**Executive Positioning Elements**:

- Strategic technology leadership experience summary
- Awards and recognition (AWS Innovation awards, etc.)
- Enterprise transformation achievements ($500M+ initiatives)
- Current role and industry focus (Field CTO, enterprise AI adoption demonstration)

**Professional Brand Consistency**

**Across All Repository Content**:

- Consistent executive positioning and technical depth balance
- Strategic technology choices with business justification
- Enterprise-grade implementation patterns and security
- Clear business value demonstration with measurable outcomes
- Executive communication style with technical precision

---

## 🎯 PROJECT PURPOSE & STRATEGIC CONTEXT

### Primary Objective

Support a senior technology executive's demonstration of hands-on technical implementation expertise by building **5 fundamental AI/ML/GenAI implementation patterns** that showcase practical, real-world business problem-solving capabilities. This initiative directly addresses the need to "show by doing" - complementing strategic leadership experience with proven technical execution capabilities.

### Executive Background Context

- **Profile**: Digital Transformation Leader with 20+ years DevOps experience, 5+ years AI/GenAI integration
- **Recent Role**: Field CTO at Cyclonix Systems, former AWS Product GM for Security and AI-infused SDLC
- **Achievement Record**: $500M+ enterprise AI transformations, led 250+ global consultant teams, built Code Defender for 62,000+ developers
- **Strategic Initiative**: Expanding technical demonstration portfolio to showcase hands-on expertise alongside strategic leadership experience for executive-level opportunities (Field CTO consulting, Chief AI Officer roles, VP AI positions, board advisory)

### Strategic Career Positioning Requirements

- **Demonstrate Technical Depth**: Showcase personal implementation expertise to complement proven team leadership experience
- **Prove Current Relevance**: Display mastery of latest AI/ML tools and frameworks through working implementations
- **Industry Alignment**: Address top business problems in highest-growth industries through practical solutions
- **Executive Credibility**: Balance strategic perspective with proven hands-on implementation capabilities

---

## 👨‍💼 DEVELOPMENT PERSONA & APPROACH

### Principal AI Engineer Guidance Framework

You are guided by a **Principal AI Engineer** with 30+ years of enterprise architecture and engineering experience who serves as both technical advisor and scope guardian.

**Persona Profile**:

- **Experience**: 30+ years enterprise architecture, Fortune 500 implementations
- **Philosophy**: "Simplicity is the ultimate sophistication" - always favor simple, working solutions

- **Role**: Technical critic focused on preventing unnecessary complexity and scope creep
- **Approach**: Pragmatic engineering with bias toward proven patterns and minimal viable implementations. Relentlessly drives every coding session with an eye for starting with Test-Driven Development(TDD) in combination with Steel-thread or vertical slice implementations. Relentless about building a strong foundation to get the first working slice of functionality working as fast as possible while at the same time ensuring its repeatability is always in tact. Uses test harnesses to instrument a minimum of one happy path fixture and one un-happy path fixture for each core and major entry point. Knows how and where to ensure fixtures are reusable across test harnesses; works to refactor fixtures when duplication is spotted. Once the steel thread for the back end is in place and working, then starts implementing the user interface wiring up the working back end in a surgical manner so as to avoid breaking things that already work and have been proven. Relentlessly works to avoid mocking unless absolutely necessary. Once a previously mocked implentantation has been replaced by the real functionality, they remove and clean unused code and mocks. Relentless about being able to always demo working functionality at any moment. Implements a high degree of security from the early onset. Considers security as job zero. Relentless about keeping the documentation up to date and calls out for visibility any opportunities where non-real code is being used in the steel thread.

**Core Principles**:

- **Scope Discipline**: Ruthlessly eliminate features that don't directly serve the 20-25 hour implementation goal
- **Technical Pragmatism**: Choose boring, reliable technology over cutting-edge complexity
- **Business Focus**: Every technical decision must have clear business justification
- **Quality Simplicity**: High-quality code doesn't mean complex architecture
- **Working Software**: Prefer working demonstrations over perfect architecture

**Scope Guardian Responsibilities**:

- Challenge any feature addition that extends beyond core business value
- Enforce "steel thread" implementation - thin vertical slice, end-to-end functionality
- Prevent over-engineering and premature optimization
- Ensure each component serves a specific, measurable business outcome
- Maintain focus on demonstration value rather than production scale

---

## 📋 PRD STRUCTURE REQUIREMENTS

**PRD Component 1: Portfolio Repository Enhancement**
**Target Repository**: https://github.com/basilan/bb-engineering-portfolio **Scope**: Transform repository to showcase 5 fundamental AI implementation patterns
**Enhancement Requirements**:
## 🏗️ AI/ML Reference Implementation Portfolio

Demonstrating technical leadership across 5 fundamental AI patterns driving enterprise transformation:

### 1. Document Intelligence & Knowledge Extraction (RAG) - Healthcare
**Repository**: [Healthcare AI Governance Agent](https://github.com/basilan/birigov)
**Business Problem**: Medical claims validation automation
**Technology Stack**: OpenAI GPT-4, NVIDIA AI Enterprise, AWS, Python, Terraform
**Industry Impact**: 60% cost reduction, <2 minute processing time
**Status**: 🚧 Under Construction

### 2. Real-Time Anomaly Detection - Financial Services
**Repository**: [Coming Q3 2025]
**Business Problem**: Transaction fraud detection and prevention
**Technology Stack**: AWS Kinesis, SageMaker, Python, React, Terraform

**Industry Impact**: Real-time fraud prevention with <100ms latency
**Status**: 📅 Coming Soon

### 3. Personalization Engine - Retail/E-commerce
**Repository**: [Coming Q3 2025]
**Business Problem**: Dynamic product recommendations and customer experience
**Technology Stack**: TensorFlow, AWS, PostgreSQL, FastAPI, Terraform
**Industry Impact**: 25% revenue increase through personalization
**Status**: 📅 Coming Soon

### 4. Predictive Analytics - Manufacturing
**Repository**: [Coming Q3 2025]
**Business Problem**: Predictive maintenance and operational optimization
**Technology Stack**: Time series forecasting, IoT integration, AWS, Terraform
**Industry Impact**: 30% reduction in unplanned downtime
**Status**: 📅 Coming Soon

### 5. Conversational AI - Multi-Industry
**Repository**: [Coming Q3 2025]
**Business Problem**: Intelligent automation and customer service
**Technology Stack**: LangChain, Vector databases, Streamlit, Terraform
**Industry Impact**: 80% first-call resolution improvement
**Status**: 📅 Coming Soon

**Professional Messaging Framework**:

- Position as **technical leader demonstrating hands-on expertise**
- Emphasize **"I am a builder myself"** narrative alongside proven team leadership
- Show **strategic technology selection** with clear business justification
- Demonstrate **cross-industry applicability** and **enterprise-scale thinking**

**🔧 ENHANCED TECHNICAL SPECIFICATIONS & PROFESSIONAL DEVELOPMENT**

**Enterprise-Grade Development Infrastructure (Comprehensive)**

Based on the attached technical specification document, the implementation must include enterprise-grade professional development practices and comprehensive build systems:

**Infrastructure as Code (Professional Grade)**

**Terraform Implementation**:

```
# Principal Engineer Approach: Simple, reliable, well-documented infrastructure
module "healthcare_ai_governance" {
  source = "./modules/simple-governance"

  # Minimal viable infrastructure - no over-engineering
  environment = var.environment
  aws_region = "us-east-1"

  # Core services only - Lambda, S3, basic networking
  lambda_memory = 512  # Start small, scale if needed
  s3_storage_class = "STANDARD"  # Simple, predictable costs

  # Basic monitoring - CloudWatch only, no complex observability
  log_retention_days = 7  # Demo environment, short retention

  # Cost controls - Principal Engineer discipline
  auto_shutdown = true
  budget_limit = 25  # Half the target budget for safety

  # Simple tagging - clear ownership and purpose
  tags = {
    Project = "Healthcare-Claims-Validator"
    Pattern = "Document-Intelligence-RAG"
    Scope = "Steel-Thread-Demo"
    Owner = "brian-boelsterli"
    Duration = "20-hour-implementation"
  }
}

# Principal Engineer Rule: No unused resources
resource "aws_lambda_function" "claims_validator" {
  # Single function - don't complicate with microservices
  function_name = "healthcare-claims-validator"
  runtime = "python3.11"
  handler = "app.lambda_handler"

  # Right-sized for demo - can scale later if needed
  memory_size = 512
  timeout = 30

  # Simple environment variables - no parameter store complexity
  environment {
    variables = {
      OPENAI_API_KEY = var.openai_api_key
      S3_BUCKET = aws_s3_bucket.claims_storage.bucket
    }
  }
}

# Principal Engineer Discipline: Minimal S3 setup
resource "aws_s3_bucket" "claims_storage" {
  bucket = "healthcare-claims-demo-${random_id.bucket_suffix.hex}"

  # Public access blocked by default - security without complexity
  public_access_block {
    block_public_acls = true
    block_public_policy = true
    ignore_public_acls = true
    restrict_public_buckets = true
  }

  # Simple lifecycle - delete demo data automatically
  lifecycle_rule {
    enabled = true
    expiration {
      days = 7  # Auto-cleanup demo data
    }
  }
}
```

**Enhanced Technology Stack Specifications**

**Core Development Stack (Enterprise Grade)**

- **Language**: Python 3.11+ with enterprise coding standards
- **Framework**: Multi-agent architecture with NVIDIA AI Enterprise integration
- **Testing**: pytest with comprehensive coverage, mocking, and performance validation
- **Code Quality**: black, flake8, pylint, mypy, bandit for enterprise standards
- **AI Integration**: NVIDIA NIM containers with AWS SageMaker coordination
- **Infrastructure**: Terraform (cloud-agnostic with multi-environment support)
- **CI/CD**: GitHub Actions with professional 2-stage pipeline
- **Documentation**: Comprehensive markdown documentation suite
- **Monitoring**: AWS CloudWatch + NVIDIA AI Enterprise Observability
- **Logging**: AWS CloudWatch and CloudTrail, etc.

**Professional Development Tools Integration**

- **Primary IDE**: VS Code with comprehensive Python and AI development extensions
- **AI Assistant**: ClaudeCode for test generation, code review, and optimization
- **Build Automation**: Professional Makefile with 40+ comprehensive targets
- **Version Control**: Git with conventional commits and automated changelog generation
- **Container Platform**: Docker with multi-stage builds and optimization
- **Cloud Platform**: AWS with enterprise security and compliance standards

**Quality Assurance & Professional Standards**

**Professional Quality Metrics (Enforced)**

```
# Enterprise quality requirements
professional_standards = {
    'test_coverage': 85,          # Minimum TDD coverage requirement
    'code_complexity': 10,        # Maximum cyclomatic complexity
    'security_vulnerabilities': 0, # Zero tolerance for security issues
    'documentation_coverage': 80, # Comprehensive docstring requirements
    'performance_sla': '99.9%',   # Uptime and reliability requirements
    'deployment_success': '100%', # Automated deployment reliability
    'cost_efficiency': '<$50',    # Demo environment cost controls
}
```

**Continuous Professional Quality Monitoring**

- **Daily**: Automated test execution with quality gate validation
- **Weekly**: Performance benchmarking and optimization analysis
- **Pre-deployment**: Comprehensive security scanning and compliance validation
- **Post-deployment**: Monitoring, alerting, and SLA compliance verification

This comprehensive technical specification ensures the implementation meets enterprise-grade professional standards while demonstrating executive-level technical leadership capabilities essential for senior technology roles.

---

## 📋 BMAD-METHOD INTEGRATION REQUIREMENTS

**PRD Development Framework**

This prompt serves as the foundational seed for developing a comprehensive Product Requirements Document (PRD) using the BMAD-METHOD framework (https://github.com/bmad-code-org/BMAD-METHOD).

**BMAD-METHOD Alignment**:

- **Business Context**: Healthcare industry AI transformation needs
- **Market Analysis**: Top 5 AI-adoption industries with specific use cases
- **Architecture Definition**: Technical stack and implementation constraints
- **Delivery Strategy**: 20-25 hour implementation with TDD approach

**PRD Development Priorities**

1. **Strategic Alignment**: Connect technical implementation to career objectives

2. **Business Justification**: Clear ROI and industry relevance
3. **Technical Specification**: Detailed architecture with security considerations
4. **Implementation Roadmap**: Day-by-day development plan with milestones
5. **Success Metrics**: Measurable outcomes for both technical and career goals

---

## 🚀 EXPECTED OUTCOMES & SUCCESS METRICS

### Technical Success Criteria

- **Functional System**: Complete healthcare claim validation working end-to-end
- **Production Deployment**: AWS-hosted system accessible via web interface
- **Code Quality**: TDD test coverage with automated CI/CD pipeline
- **Performance**: Sub-2-minute claim processing with confidence scoring
- **Security**: AWS security best practices with secure credential management

### Business Impact Demonstration

- **Cost Analysis**: Clear demonstration of 60% manual process cost reduction
- **Efficiency Gains**: Processing time improvement from days to minutes
- **Quality Improvement**: Consistent decision-making vs human variability
- **Compliance Value**: Audit trail and governance capability demonstration

### Career Positioning Outcomes

- **Technical Credibility**: Hands-on AI implementation expertise proven through working code
- **Industry Expertise**: Healthcare AI domain knowledge with practical application
- **Executive Communication**: Business value demonstration for C-level conversations
- **Portfolio Development**: GitHub repository showcasing modern AI development practices

### Next Phase Foundation

- **Pattern Replication**: Template for implementing remaining 4 AI patterns
- **Technology Stack Mastery**: Python, AWS, OpenAI, NVIDIA integration expertise
- **Development Velocity**: Proven ability to deliver AI solutions in 20-25 hour cycles
- **Business Acumen**: Deep understanding of AI business value across industries

---

## 📞 COLLABORATION & DEVELOPMENT APPROACH

### Development Partnership Requirements

- **Principal Engineer Guidance**: Maintain scope discipline and technical simplicity throughout implementation
- **AI-Assisted Development**: Leverage Claude Code for rapid prototyping while avoiding over-engineering
- **Iterative Refinement**: Continuous feedback focused on working demonstrations rather than perfect architecture
- **Executive Perspective**: Balance technical implementation with clear business value demonstration
- **Industry Alignment**: Ensure technical choices support broader technology leadership conversations
- **Scope Discipline**: Challenge any feature that doesn't directly serve the 20-25 hour steel-thread implementation

### Quality Gates & Professional Milestones

- **Phase 1**: Portfolio repository transformation with 5 AI patterns roadmap (4 "Coming Soon", 1 "Under Construction")
- **Phase 2**: Healthcare AI implementation - core validator with OpenAI integration
- **Phase 3**: AWS deployment via Terraform with simple monitoring
- **Phase 4**: Working demonstration supporting executive technical leadership positioning
- **Success Validation**: End-to-end claims validation in under 2 minutes with clear business value demonstration

### Principal Engineer Success Criteria

- **Simplicity First**: No unnecessary complexity or over-engineering
- **Working Software**: Functional demonstration over perfect architecture
- **Clear Value**: Every component serves measurable business outcome

- **Scope Discipline**: Implementation stays within 20-25 hour constraint
- **Technical Clarity**: Clean, maintainable code that other senior engineers would approve

This comprehensive prompt provides the strategic context, technical requirements, scope discipline, and business justification necessary to transform executive AI vision into hands-on technical implementation, directly supporting career advancement through proven capability demonstration while maintaining rigorous scope discipline and engineering simplicity.

---

USAGE

This repository points to other repositories via read me files, etc and could simply use the github's repository interface. The plan is to reference this material in my personal linkedin profile, but also in Cyclonix Systems website more than likely. (www.cyclonixsystems.com), which we can use. Currently, cyclonix systems website is extremely simple and is more in a stub form currently. I am open to the idea of showcasing these reference implementations perhaps in a theme that shows 'our labs' or something. The reference implementations need to be stored as code in github repositories allowing other to download and install them so they can run them and see how they work on their local hosts. I expect we will want to render examples of output via screen image snapshots or command line screen snapshots or something. I am open to ideas.