demo-work-item-tracking

Demo Work Item Tracking - demo/coi Branch

Date Created: November 18, 2025
Demo Branch: `demo/coi`
GitHub Issue: #12 - Upgrade Task Repository from In-Memory to Azure Cosmos DB

Overview

This document tracks the parallel work for the internal Centric demo, where we showcase:

- Agent-driven planning and development
- Custom Copilot instructions and conventions
- Parallel development (C# code + Bicep infrastructure)
- Code review automation
- Pull request workflow

Work Division

Code Track (@shawnewallace)

Responsibility: Upgrade C# code to support Cosmos DB

Tasks:

- □ Create `CosmosDbTaskRepository` implementing `ITaskRepository`
- □ Add Cosmos DB configuration (appsettings.json)

- □ Update DI registration in ServiceExtensions
- □ Add NuGet packages (Microsoft.Azure.Cosmos)
- □ Implement proper error handling and logging
- □ Write unit tests with mocked Cosmos client
- □ Write integration tests with Cosmos Emulator/Testcontainers
- □ Update README with configuration instructions

Branch: `feature/cosmos-db-repository` (from `demo/coi`)

Infrastructure Track (@mdrakiburrahman)

Responsibility: Create Bicep templates for Azure infrastructure

Tasks:

- □ Set up `infra/` directory structure
- □ Create `main.bicep` for orchestration
- □ Create `cosmos.bicep` module
- □ Create App Service Bicep module
- □ Add parameter files (dev, staging, prod)
- □ Create `infra/README.md` with deployment instructions
- □ Test deployment to Azure

Branch: `feature/cosmos-db-infrastructure` (from `demo/coi`)

Demo Workflow

1. Planning Phase (Agent Mode)

Step 1: Fetch the GitHub Issue

```
Use GitHub Copilot Agent Mode or Chat:
"Fetch GitHub issue #12 from centricconsulting/ai-coding-workshop and
summarize the work required"
```

Expected AI actions:

- Calls GitHub API to retrieve issue details
- Parses acceptance criteria (code changes vs. infrastructure changes)
- Identifies assignees (@shawnewallace, @mcollier)
- Extracts technical requirements and implementation phases

Step 2: Generate Implementation Plan

```
"Based on issue #12, create a detailed step-by-step implementation plan for
the C# code track.
Follow the Copilot instructions in .github/copilot-instructions.md for Clean
Architecture and TDD."
```

Expected AI output:

- Phased approach (Setup → Repository → Testing → Infrastructure)
- Specific files to create/modify
- Test-first approach for each component
- Configuration and DI changes

Step 3: Review and Refine

- Review generated plan with team
- Adjust for demo timing and scope
- Assign tasks to parallel tracks (code vs. infrastructure)
- Create feature branches

2. Development Phase (Parallel Work)

- Shawn: Create feature branch and implement C# changes
- Michael: Create feature branch and implement Bicep changes
- Both: Leverage Copilot instructions for consistency
- Both: Use `/check` for automated code review

3. Review Phase (Code Review Automation)

- Run `/check` on completed code
- Request Copilot review on PRs
- Demonstrate custom review prompts
- Merge both PRs into `demo/coi`

4. Integration Phase

- Test integrated solution locally
- Deploy infrastructure with Bicep
- Validate API with Cosmos DB backend
- Document lessons learned

Demo Highlights to Showcase

Copilot Customizations

- **Repository-level instructions:** `.github/copilot-instructions.md`
- **File-specific instructions:** Front matter in instruction files
- **Linked instruction files:** C# standards, Cosmos DB best practices, commit conventions

Agent Capabilities

- **Planning mode:** Decompose work items into actionable steps
- **Code generation:** Generate boilerplate with proper conventions
- **Code review:** `/check` command for deep analysis
- **Multi-file editing:** Copilot Edits for cross-cutting changes

Model Selection

- Demonstrate switching between models (GPT-4.1, Claude Sonnet, etc.)
- Show auto-select vs. manual model choice
- Discuss cost-effectiveness and task suitability

Parallel Development

- Two developers working on same feature simultaneously
- Clean separation of concerns (code vs. infrastructure)
- Integration via pull requests
- No merge conflicts due to good architecture

Current Status

Phase: Planning
Next Steps:

1. Review issue #12 with team
2. Use agent planning mode to create detailed implementation plan
3. Create feature branches for both tracks
4. Begin parallel development

Links

- **GitHub Issue:** https://github.com/centricconsulting/ai-coding-workshop/issues/12
- **Demo Branch:** `demo/coi`
- **Meeting Notes:** `./20251119-demo-notes.md`
- **Copilot Instructions:** `../../.github/copilot-instructions.md`

Notes

- This demo uses `demo/coi` as the "main" branch for the internal demonstration
- All PRs will target `demo/coi`, not the actual `main` branch
- Focus on showcasing best practices, not perfect implementation
- Emphasize team collaboration and AI-assisted development workflows