MODERNIZATION SHIELD

Legacy Modernization Risk Report

Subject: SocialGoal

ENGAGEMENT SPONSOR
None (illustrative sample)

ENGAGEMENT TYPE
Public-codebase teardown

ENGAGEMENT PERIOD
June 2026

METHODOLOGY
Modernization Shield

ENGAGEMENT LEAD
Jerry Harlow, Founder, Evincia LLC

DOCUMENT VERSION
1.1

ILLUSTRATIVE SAMPLE · PUBLIC-CODEBASE TEARDOWN

Prepared by Evincia LLC using the Modernization Shield methodology · Atlanta, GA · evincia.co

ENGAGEMENT SUMMARY

What this report is

This Legacy Modernization Risk Report (LMRR) is the primary deliverable of a Modernization Shield engagement. It provides a structured, evidence-based assessment of modernization risk within the subject platform. For this sample, the subject is the public SocialGoal solution and the engagement is a public-codebase teardown rather than a client engagement.

Illustrative sample: public-codebase teardown. SocialGoal is a real open-source ASP.NET MVC reference application. This report is an illustrative Modernization Shield deliverable built from a public-codebase teardown, not a client engagement. There is no client, no sponsor, no live database, and no stakeholders. Investigation was limited to static analysis plus test and coverage inspection (change-safety); business-logic review (Category 11) and stakeholder interviews (Category 12) were not performed. The readiness score and dimension scores are senior architect judgment; the phase trajectory, effort, and cost figures are illustrative.

ENGAGEMENT SPONSOR	None (illustrative public-repo sample)
ENGAGEMENT TYPE	Public-codebase teardown (illustrative)
SUBJECT COMPANY	**SocialGoal** (open-source ASP.NET MVC 5 reference application)
ACQUISITION CLOSE	Not applicable
ENGAGEMENT PERIOD	June 2026
ENGAGEMENT LEAD	Jerry Harlow, Founder, Evincia LLC
METHODOLOGY	Modernization Shield: diagnostic toolchain plus senior architect review
PRIMARY DELIVERABLE	This Legacy Modernization Risk Report (LMRR)
SUPPORTING MATERIALS	Document A: System and Architecture Overview; Document B: Modernization Blockers; Document C: Evidence Appendix

The supporting materials (Documents A, B, and C) are delivered as separate companion artifacts to this report and referenced throughout where applicable. Document C is the diagnostic engine output, referenced throughout as the audit files A1 through A13. The platform was analyzed with the Evincia diagnostic engine across a full nine-category static teardown: 7 projects, all targeting .NET Framework 4.5. Document v1.1 (2026-07-13) resolves R-003 by senior-architect code read; the revision note sits in the Risk Register introduction.

CONTENTS

Contents

SECTION 1

Executive Summary

1.1 Headline Finding

SocialGoal is a small, deliberately layered ASP.NET MVC 5 application that runs adequately but cannot safely absorb modernization or AI-driven change today. Its Modernization Readiness Score of 44 out of 100 places it in the Red zone, just below the Yellow threshold, which means modernization must precede any AI initiative beyond read-only, sandboxed analysis. The dominant risk is not complexity. It is obsolescence.

Every project targets .NET Framework 4.5, and System.Web is woven through the web layer across 98 sites, so there is no in-place upgrade path. There is only a re-platform to modern .NET. The authentication stack compounds it: OWIN identity and Forms authentication have no carry-forward and must be rebuilt on ASP.NET Core Identity. Entity Framework 6 has to move to EF Core, and a layer of deprecated and vulnerable packages sits on top of all of it.

Three near-term findings deserve particular attention. First, the entire solution targets unsupported .NET Framework 4.5 and will require platform migration before any meaningful modernization. Second, the test suite passes (113 of 113) and is real at 44.5 percent line coverage, but it is happy-path-weighted and skewed: the three seams a migration actually rides on, the data layer, the authorization and anti-forgery filters, and any database trigger behavior, are all at zero coverage. Third, one open unknown remains that this teardown could not close from source: whether the SQL Server database carries triggers with hidden behavioral logic.

One thing works in the platform's favor. The abstraction layers are clean and consistent enough to serve as a migration seam, which makes a phased strangler approach realistic rather than a forced big-bang rewrite. The detail follows in the sections and Risk Register below.

1.2 Modernization Readiness Score

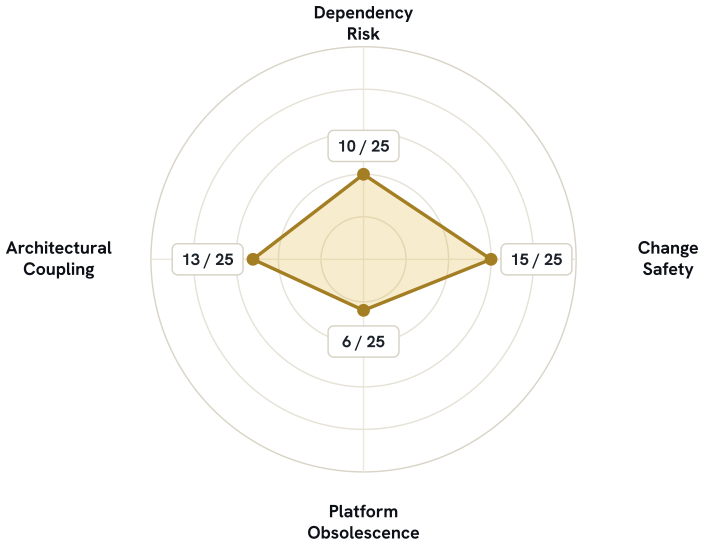
The Modernization Readiness Score synthesizes findings across twelve diagnostic categories: nine produced by the Evincia diagnostic engine and three produced through senior architect review of the code, plus schema review and stakeholder interviews where an engagement includes them (this sample, built from a public repository, had neither). The categories roll up into four composite dimensions that together produce a score on a 0 to 100 scale. The score and the four dimension values are senior architect judgment; the engine does not compute them.

RED ZONE

44/100

Modernization must precede AI

At 44, SocialGoal sits just below the Yellow threshold: close to "functional but not AI-ready," but not yet there. Only read-only or sandboxed analytical AI is defensible before the foundational work is done, and even that is constrained by the change-safety gaps below.



Four composite dimensions, each scored out of 25; their sum is the 44 / 100 readiness score. Zone scale: Red 0-49, Yellow 50-79, Green 80-100.

Platform Obsolescence 6 / 25

Lowest dimension and the heaviest contributor to the Red classification. All 7 projects target .NET Framework 4.5 and System.Web is pervasive across the web layer. There is no in-place upgrade; a re-platform to .NET 10 is required before meaningful modernization can proceed.

Architectural Coupling 13 / 25

The strongest architectural dimension. A real service layer, IRepository abstraction, and DI registration are present and reasonably consistent. The one open question -- the contested Web-to-Data reference (R-003) -- was closed by the v1.1 code read: composition-root wiring, not a bypass. The skepticism was already priced into this read, so the 13 holds. Decomposition is feasible via a phased strangler.

Dependency Risk 10 / 25

A deprecated, vulnerable, and version-conflicted package layer (the OWIN family, Entity Framework, bootstrap, jQuery, Newtonsoft.Json). The OWIN identity stack in particular has no carry-forward to ASP.NET Core.

Change Safety 15 / 25

A real test suite (44.5 percent line) that passes but is happy-path-weighted (23.8 percent branch) and skewed. The data layer, the auth and anti-forgery filters, and any DB trigger behavior are at zero coverage. Adequate for happy-path regressions; not adequate to modernize against.

1.3 Top 5 Risks

Five findings shape the modernization narrative and are referenced throughout the Risk Register and Sequencing Guidance. Each summary below references the full entry in Section 2.

R-001 · Entire Solution Targets .NET Framework 4.5

PLATFORM OBSOLESCENCE · CRITICAL

All 7 projects target .NET Framework 4.5, an out-of-support platform. No in-place path to modern .NET exists; everything else waits on retargeting.

R-002 · System.Web Pervasive Across the Web Layer

PLATFORM OBSOLESCENCE · CRITICAL

System.Web is referenced across 98 sites. It does not exist in modern .NET, so the web layer must be re-implemented on ASP.NET Core, not migrated.

R-004 · Entity Framework 6 Across Three Projects

DATA COMPLEXITY · HIGH

EF6 to EF Core is not a drop-in; query translation differs. The 6.0.2-beta1 pre-release pin in the Web project is an added risk.

R-005 · Deprecated and Abandoned Packages

EXTERNAL DEPENDENCY RISK · HIGH

Fourteen deprecated package firings, including the OWIN identity family, which has no carry-forward to ASP.NET Core. The dependency-risk anchor; couples tightly to the authentication rebuild.

R-007 · Tested, Not Safety-Netted for Change

CHANGE FAILURE RISK · HIGH

A real suite (44.5 percent line) but happy-path-skewed (23.8 percent branch), with the data layer, auth filters, and triggers at zero coverage. The core change-safety concern, and the reason structural work cannot begin without a safety gate first.

> *Curation note: the layer-boundary finding (R-003) carried a higher engine priority (20) than items three through five, but it was held off the top 5 because its Critical severity was contested pending a code read. The v1.1 code read confirmed the contestation and re-rated R-003 to Low (6); it now stays off the top 5 on the merits, not on caution.*

1.4 What Will Break First

The three most likely failure scenarios during modernization, in plain language:

1. **Retarget exposes System.Web everywhere.** The moment the projects leave .NET Framework 4.5, the 98 System.Web sites stop compiling. The web layer cannot build on modern .NET until each is re-implemented on ASP.NET Core. This is the first wall and the largest.

2. **Authentication has no carry-forward.** OWIN identity and Forms authentication do not port. Sign-in, sessions, and the Identity tables break and stay broken until rebuilt on ASP.NET Core Identity. The auth and anti-forgery filters are also at zero test coverage, so breakage will surface late.

3 **The data layer shifts under a net that does not cover it.** EF6 to EF Core changes query translation and the SqlClient provider swaps; the persistence layer is at zero coverage, so data-access regressions surface late and in production-shaped ways. If the database carries triggers (an open unknown), that behavior is invisible to the application tier entirely.

1.5 Modernization Summary

Modernization should proceed in four phases (a safety gate plus three delivery phases). Phase 0 builds the change-safety net that the platform does not yet have. Phase 1 performs the mechanical retarget. Phase 2 executes the two large rebuilds (System.Web removal and EF Core, plus the authentication re-platform) behind that net. Phase 3 clears the dependency and observability layer and prepares the platform for AI use cases.

The platform is in the Red zone, so AI must follow modernization. Read-only and sandboxed analytical AI can begin during the early phases; broad or production-scale AI should be sequenced after Phase 2 completes and the safety net is real.

Estimated effort (illustrative): roughly 6 to 9 months for a team of 1 to 2 engineers, reflecting the small size of the codebase (7 projects, approximately 11,000 lines). Estimated cost: not modeled for this public-repo sample. The engine produces no effort, timeline, or cost signal; in a real engagement these are scoped by the senior architect from engagement context. Detailed phase sequencing and workstream planning are in Section 3.

SECTION 2

Risk Register

16 findings, consolidated from 194 raw engine findings with senior architect rulings and false-positive corrections applied.

Each entry includes scoring fields per the Evincia methodology: Probability and Impact on a 1 to 5 scale, Priority as the product of the two, Priority Rating as the band lookup, Confidence as certainty in the finding (1 to 10), and Automation Level as how much the finding came from engine detection versus senior architect judgment (1 to 10). Confidence and Automation Level are independent. Two findings (R-010, R-011) carry de-rated confidence because they depend on cross-project type resolution, and the coverage assessment logged 26 workspace load failures (unresolved project references) that reduce semantic confidence for exactly those findings; R-003 carried the same de-rate in v1.0 until its code read landed. **Revision note (v1.1, 2026-07-13):** the contested layer-boundary finding (R-003) was resolved by a senior-architect code read at the pinned commit and re-rated from Critical (contested, 4/10 confidence) to Low (9/10). No other entry changed. Refined the same day: the engine's new usage-site classifier corrected the usage detail -- one controller using is a dead import, the other is live for a single leaked paging type (Page). The ruling and scores are unchanged.

Critical Priority PRIORITY SCORE 20 TO 25

Critical-priority risks must be addressed as part of any modernization or AI initiative. They represent blockers that will cause modernization failure if left untreated.

R-001

CRITICAL

Entire Solution Targets .NET Framework 4.5

PLATFORM OBSOLESCENCE

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
5	5	25	10/10	10/10

DESCRIPTION. All 7 projects in the solution target .NET Framework 4.5, a platform whose support ended in January 2016, with no in-place upgrade path to modern .NET. Microsoft ships no updates of any kind for the 4.5 target; even the supported 4.8 endpoint gets security patches only, never new features -- which locks the solution out of modern libraries, tooling, and AI frameworks. Every downstream modernization and AI initiative depends on addressing this first.

EVIDENCE. Engine-detected across all project files (Signal 1.1). See Document C, A4.

RECOMMENDED ACTION. Convert all 7 projects to SDK-style project files in Phase 1 and retarget the platform-agnostic class libraries; the System.Web- and EF6-bound projects complete the move to .NET 10 in Phase 2, when those blockers are removed. This is the prerequisite to every other workstream.

R-002

CRITICAL

System.Web Pervasive Across the Web Layer

PLATFORM OBSOLESCENCE

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
5	5	25	9/10	10/10

DESCRIPTION. System.Web is referenced across 98 sites in the Web, Web.Core, Model, and Tests projects. System.Web does not exist in modern .NET, so every dependent site must be re-implemented on ASP.NET Core. The web tier cannot be migrated incrementally while these references remain. The 98 raw sites are consolidated into this single entry.

EVIDENCE. Engine-detected (Signal 1.2). See Document C, A4.

RECOMMENDED ACTION. Remove System.Web and migrate the web layer to ASP.NET Core (controllers plus middleware pipeline) in Phase 2. This is the largest single mechanical surface in the migration.

Placement note: R-003 sits in its original Critical-tier position so the v1.0 dispute and the v1.1 resolution read together. Its re-rated priority is Low (6).

R-003

RESOLVED · LOW

UI Project References the Data-Access Project Directly (Resolved)

STRUCTURAL COUPLING · WAS CRITICAL · CONTESTED (V1.0) · RESOLVED V1.1 (2026-07-13)

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
2	3	6	9/10	6/10

DESCRIPTION. SocialGoal.Web has a direct project reference to SocialGoal.Data. The engine rated this a Critical layer-boundary violation; that rating assumes active bypass of the service layer, which the engine could not confirm, so the severity shipped contested in v1.0 at 4/10 confidence rather than anchor the register on it. The v1.1 code read resolves it: no bypass -- every controller is service-mediated. The reference exists for the composition root: Autofac registration of the repository and unit-of-work implementations (Bootstrapper.cs), the database seed initializer (Global.asax.cs), and direct construction of the Identity DbContext for its UserStore. Of the two controller-level SocialGoal.Data usings, one is a dead import (AccountController) and one is live for a single leaked type: Page, a paging class from the Data project that the controller constructs and passes into a service call. The residual risk is why the entry stays in the register: the service-layer boundary is convention-only -- every repository is DI-resolvable from any controller, and the service signatures already leak a Data-layer type (Page) -- and Identity runs on a second DbContext outside the unit of work, which the EF Core and Identity rebuilds must unify.

EVIDENCE. Engine-detected (Signal 2.3); flagged in v1.0 as a suspected false positive and de-rated pending a code read; resolved in v1.1 by a senior-architect code read of the four referencing files (Bootstrapper.cs, Global.asax.cs, and the two controller usings -- one dead, one live for the leaked Page type) at pinned commit 42cfdb4; the usage detail was refined the same day by the engine's new usage-site classifier. See Document C, A5.

RECOMMENDED ACTION. Resolved; the guardrail carries into Phase 2. Delete the dead using, and move the leaked paging type behind the service boundary during the rebuild. Keep controllers service-mediated through the rebuild -- add an architecture test or analyzer rule once the solution is on modern .NET -- and unify the Identity DbContext into the unit of work when authentication is re-platformed. The reference itself is legitimate composition-root wiring and moves to the new host's Program.cs.

High Priority PRIORITY SCORE 15 TO 19

High-priority risks materially increase modernization difficulty or post-modernization operational exposure. They should be addressed before or during Phase 2 of the modernization program.

R-004

HIGH

Entity Framework 6 Across Three Projects

DATA COMPLEXITY

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
4	4	16	9/10	10/10

DESCRIPTION. Entity Framework 6 is present in Web (6.0.2-beta1), Data (6.0.1), and Tests (6.0.0). EF6 to EF Core is not a drop-in; query translation differs and behavior can drift. The pre-release beta1 pin in the Web project is an added risk in a shipping application. EF appears under three lenses in this register: migration here (R-004), deprecation (R-005), and version conflict (R-012). It is one dependency seen three ways, not triple-counted; the EF Core migration resolves all three.

EVIDENCE. Engine-detected (Signal 4.3). See Document C, A7.

RECOMMENDED ACTION. Migrate EF6 to EF Core and retire the 6.0.2-beta1 pin. Parallel workstream in Phase 2, gated by the Phase 0 data-layer characterization tests.

R-005

HIGH

Deprecated and Abandoned Packages

EXTERNAL DEPENDENCY RISK

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
4	4	16	9/10	10/10

DESCRIPTION. Fourteen deprecated package firings, including the OWIN family, Entity Framework 6.0.1/6.0.0, and Microsoft.AspNet.Web.Optimization. The OWIN identity stack has no carry-forward to ASP.NET Core, which makes this the dependency-risk anchor and tightly couples it to the authentication rebuild (R-006).

EVIDENCE. Engine-detected (Signal 6.1). See Document C, A9.

RECOMMENDED ACTION. Replace the OWIN identity stack with ASP.NET Core Identity, and replace Web.Optimization with bundling and minification middleware. Phase 2, coupled to R-006.

R-006

HIGH

Forms Authentication Needs Re-Platforming

PLATFORM OBSOLESCENCE

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
4	4	16	8/10	9/10

DESCRIPTION. Forms authentication is detected in the Web.Core, Web, and Tests projects. It has no direct carry-forward to modern .NET and must be rebuilt. The authentication and anti-forgery filters are also at zero test coverage (see R-007), so breakage during the rebuild will not be caught by the existing suite.

EVIDENCE. Engine-detected (Signals 1.8 to 1.9). See Document C, A4.

RECOMMENDED ACTION. Re-platform authentication to ASP.NET Core Identity with cookie authentication in Phase 2, gated by the Phase 0 auth and anti-forgery characterization tests.

R-007

HIGH

Tested, Not Safety-Netted for Change

CHANGE FAILURE RISK · SENIOR ARCHITECT SEVERITY MODERATE

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
4	4	16	7/10	6/10

DESCRIPTION. The engine's structural test ratio for this suite is 0.17, a comparison of the amount of test code to production code (Signal 7.1), not a coverage measurement. It reads as "barely tested" and is misleading here, because it counts how much test code exists, not how much of the codebase that code exercises. Measured effective line coverage is 44.5 percent, with branch coverage at 23.8 percent, run through real service and model collaborators with only repositories mocked. The suite is real and non-trivial (113 of 113 pass) but shallow and skewed. Concretely: SocialGoal.Data is at 0 percent across the whole assembly (every repository, UnitOfWork, RepositoryBase, the DbContext); the SocialGoalAuthorizeAttribute and AntiForgeryTokenFilterProvider filters are both at 0 percent; AccountController is the weakest controller at 39.4 percent. The Service layer is genuinely exercised (50.9 percent), which is why line coverage reaches 44.5 percent; SocialGoal.Model reads 82.2 percent, but that is POCO property access, not logic. The suite catches controller and service happy-path regressions; it is not adequate to refactor or modernize against. The three paths a migration would silently break are all dark: 0 percent data layer, 0 percent auth and anti-forgery filters, and unknown DB triggers.

EVIDENCE. Engine-detected (Signal 7.1, the structural test-to-production ratio) plus senior architect inspection with coverage instrumentation. The 0.17 ratio and the 44.5 percent line coverage measure different things: the first compares the size of the test code to the production code, the second is how much of that code the suite actually runs. Read the suite by the second. See Document C, A10 and Document B.

RECOMMENDED ACTION. Before any migration, build characterization tests over the three dark zones first: the data layer, DB trigger behavior, and the auth and anti-forgery filters. This is the Phase 0 safety gate.

Medium Priority PRIORITY SCORE 10 TO 14

Medium-priority risks increase modernization effort but do not block progress. They should be addressed during the appropriate phase of the modernization program.

R-008

MEDIUM

Packages With Known Vulnerabilities

EXTERNAL DEPENDENCY RISK

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
3	4	12	7/10	9/10

DESCRIPTION. Eight vulnerable package firings (bootstrap, jQuery, jQuery.Validation, Microsoft.AspNet.Identity.Owin, Owin, Owin.Security.Cookies, Newtonsoft.Json, and Owin again in Tests). The detection is metadata-based and cannot confirm whether fixes require breaking major version bumps.

EVIDENCE. Engine-detected (Signal 6.4). See Document C, A9.

RECOMMENDED ACTION. Patch to fixed versions, and flag which require breaking major bumps via a dedicated vulnerability-bump analysis. Address critical items in Phase 1 as a security action; fold the rest into Phase 3.

R-009

MEDIUM

Config-Defined SQL Connection on a Legacy Provider

DATA COMPLEXITY

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
3	4	12	8/10	9/10

DESCRIPTION. The Web project uses System.Data.SqlClient. Microsoft.Data.SqlClient is the supported provider on modern .NET. This is a clean, mechanical swap with low behavioral risk.

EVIDENCE. Engine-detected (Signal 4.7). See Document C, A7.

RECOMMENDED ACTION. Swap System.Data.SqlClient for Microsoft.Data.SqlClient during the Phase 1 retarget.

R-010

MEDIUM

Third-Party SDK Residue (MvcMailer, PagedList)

EXTERNAL SYSTEM DEPENDENCY RISK

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
3	4	12	4/10	7/10

DESCRIPTION. After folding the cross-category re-counts (EntityFramework, AutoMapper, and Owin belong to their primary categories) and dropping the design-time-only T4Scaffolding.Core, the genuine Category-9 residue is MvcMailer and PagedList. The original engine finding listed six packages; corrected to two after review.

EVIDENCE. Engine-detected (Signal 9.3); corrected as a partial false positive and de-rated. See Document C, A12.

RECOMMENDED ACTION. Replace MvcMailer (unmaintained) with MailKit plus Razor templating, and replace PagedList with X.PagedList or built-in paging. Phase 3.

R-011

MEDIUM

High Fan-Out in Web and Tests

STRUCTURAL COUPLING

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
3	4	12	5/10	9/10

DESCRIPTION. The engine detected a fan-out of 5 in the Web project and 5 in Tests. This is modest for a web project and expected in a test project, and it is subject to the cross-project coverage caveat. Low weight relative to the other Medium findings.

EVIDENCE. Engine-detected (Signal 2.1). See Document C, A5.

RECOMMENDED ACTION. Monitor during the Phase 2 refactor; no dedicated workstream required.

Low Priority PRIORITY SCORE BELOW 10

Low-priority risks are noted for completeness and tracking. They may not require dedicated remediation workstreams depending on the chosen modernization target state.

R-012

LOW

Entity Framework Version Conflict

EXTERNAL DEPENDENCY RISK

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
3	3	9	8/10	9/10

DESCRIPTION. Entity Framework is pinned at 6.0.2-beta1, 6.0.1, and 6.0.0 across projects, a version skew that includes a pre-release build. This is the third lens on the EF dependency (see R-004) and is resolved by the EF Core migration.

EVIDENCE. Engine-detected (Signal 6.2, version discrepancies). See Document C, A9.

RECOMMENDED ACTION. Unify the EF version; resolved by the R-004 EF Core migration in Phase 2.

R-013

LOW

AutoMapper Pinned to a CI Pre-Release

PLATFORM OBSOLESCENCE

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
3	2	6	7/10	7/10

DESCRIPTION. AutoMapper is pinned to 3.1.1-ci1000, a continuous-integration pre-release build, in a shipping application. This is an upgrade-path risk: a CI build is not a supported release line.

EVIDENCE. Engine-detected (Signal 1.6). See Document C, A4.

RECOMMENDED ACTION. Move AutoMapper to a stable release during the Phase 3 dependency upgrade.

R-014

LOW

ELMAH Is the Only Observability and Has No Path Into ASP.NET Core

CHANGE FAILURE RISK

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
2	3	6	8/10	7/10

DESCRIPTION. ELMAH is wired and is the only observability present: there is no structured logging, no APM or metrics, and no health checks. It is adequate as Framework-era error capture, but the error paths it feeds are themselves untested, and it has no carry-forward to ASP.NET Core.

EVIDENCE. Engine-detected (Signal 7.4) plus senior architect inspection. See Document C, A10.

RECOMMENDED ACTION. On migration, replace with ILogger plus Application Insights or Serilog, and add the structured logging, metrics, and health checks the application has never had. Phase 3. Minimal-but-present today; not a current defect.

R-015

STRENGTH

Clean Abstraction Layers (Strength)

STRUCTURAL COUPLING · MITIGATING FACTOR

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
1	1	1	6/10	5/10

DESCRIPTION. A service layer, an IRepository abstraction, and dependency-injection registration are present. This is a positive signal: it improves migration feasibility and provides a seam for a strangler approach. It is recorded as a Low entry rather than a risk because it is a mitigating factor, subject to the cross-project coverage caveat; confirm consistent use during the code read.

EVIDENCE. Engine-detected (Signal 2.5). See Document C, A5.

RECOMMENDED ACTION. Preserve the layering as the migration seam; leverage it for a phased strangler rather than a big-bang rewrite.

Confirmed and Deferred COSMETIC · WON'T-FIX-NOW

R-016

COSMETIC

Hardcoded URLs in Test Projects

CHANGE FAILURE RISK · ENGINE PRIORITY 16

PROBABILITY	IMPACT	PRIORITY	CONFIDENCE	AUTOMATION
4	4	16	8/10	9/10

DESCRIPTION. This is a confirmed true positive, not a false positive: http://yoursite/ appears in EmailRequestControllerTest.cs at lines 151 and 238, and a hardcoded http://localhost/ appears in AccountControllerTest. It is test-only, has no production impact, and the tests pass against it. The engine Priority of 16 over-rates the business impact of a test-only placeholder, so this entry is placed by disposition, not raw priority.

EVIDENCE. Engine-detected (Signal 7.3) plus senior architect inspection. See Document C, A10.

RECOMMENDED ACTION. Replace the hardcoded test URLs with configuration or constants. Fix opportunistically; not a blocker.

SECTION 3

Modernization Sequencing Guidance

Modernization of the SocialGoal platform should proceed in four phases: a safety gate (Phase 0) followed by three delivery phases. Each phase has a defined set of workstreams, expected outcomes, and a Modernization Readiness Score trajectory. This sequencing reflects the constraint the evidence forces: structural change cannot proceed safely until the change-safety net covers the three migration-critical seams that are dark today.

The trajectory below is a senior-architect-signed-off projection, not an engine output. The baseline (44) is measured; the downstream values are directional targets, read as the shape of the climb rather than precise scores.

STAGE	PLATFORM	COUPLING	DEPENDENCY	CHANGE-SAFETY	TOTAL
Baseline	6	13	10	15	44
After Phase 0	6	13	10	21	~50
After Phase 1	12	13	11	21	~57
After Phase 2	23	19	16	22	~80
After Phase 3	**23**	**19**	**23**	**23**	~88

0 Safety Gate

Before any migration. Purpose: earn the right to refactor. The current suite passes but does not cover the seams a migration rides on. This phase builds that coverage before any platform work begins. It does not deliver modernization outcomes directly; it makes modernization work safe.

- Build characterization tests over the data/persistence layer, currently at 0 percent (addresses R-007)
- Build characterization tests over the authorization and anti-forgery filters (SocialGoalAuthorizeAttribute, AntiForgeryTokenFilterProvider), currently at 0 percent (addresses R-007, supports R-006)
- Inspect a live database instance to resolve the open trigger question (SELECT * FROM sys.triggers); the repo carries EF Code-First config and a seeder but no migration or SQL scripts, so this cannot be answered from source
- Establish baseline observability and deployment telemetry

> **Expected outcome:** Readiness moves from 44 to approximately 50 as Change-Safety rises from 15 to about 21. Still borderline, but structural readiness for Phase 1 is established.

Estimated effort (illustrative): a few weeks, 1 engineer.

1 Foundation Retarget

Mechanical. Purpose: get onto modern .NET without changing application behavior. This is the lowest-risk, highest-leverage block, and it is prerequisite to everything that follows.

- Retarget all 7 projects from .NET Framework 4.5 to .NET 10 with SDK-style project files (addresses R-001)
- Swap System.Data.SqlClient for Microsoft.Data.SqlClient (addresses R-009)
- Unify the Entity Framework version (addresses R-012; completed by the EF Core migration in Phase 2)
- Remediate the critical subset of the package vulnerabilities as a security action (top-priority part of R-008)

> **Expected outcome:** Readiness moves from approximately 50 to approximately 57 as Platform Obsolescence rises from 6 to about 12. System.Web and Forms authentication remain until Phase 2.

Estimated effort (illustrative): 1 to 2 months, 1 to 2 engineers.

2 The Two Big Rebuilds, Behind the Net

Core of the effort. Purpose: remove the framework blockers that have no in-place path. Three workstreams run in parallel on the retargeted foundation, with the Phase 0 net underneath.

- Remove System.Web and re-implement the web layer on ASP.NET Core, the largest single surface at 98 sites (addresses R-002)
- Migrate EF6 to EF Core, retiring the 6.0.2-beta1 pin; query-translation differences surface here, which is where the Phase 0 data-layer coverage pays off (addresses R-004 and R-012)
- Re-platform authentication from OWIN and Forms to ASP.NET Core Identity, gated on the Phase 0 auth tests (addresses R-005 and R-006)
- Carry the resolved layer-boundary finding's guardrail through the rebuild: keep controllers service-mediated, add an architecture test on the new platform, and unify the Identity DbContext into the unit of work (closes R-003's residual)

> **Expected outcome:** Readiness moves from approximately 57 to approximately 80 as Platform Obsolescence rises to about 23 and Architectural Coupling to about 19. The platform crosses into the Green zone.

Estimated effort (illustrative): 3 to 5 months, 1 to 2 engineers.

3 Dependency and Observability Cleanup

Modernize the edges. Purpose: clear the remaining risk layer once the platform is modern, and prepare for AI use cases.

- Patch the vulnerable packages, with the breaking-major-bump analysis done first (completes R-008)
- Replace the unmaintained SDK residue: MvcMailer to MailKit plus Razor, PagedList to X.PagedList or built-in paging (addresses R-010)
- Replace ELMAH with ILogger plus Application Insights or Serilog, and add structured logging, metrics, and health checks (addresses R-014)
- Move AutoMapper off the CI pre-release to a stable release (addresses R-013)
- Fix the cosmetic hardcoded test URLs opportunistically (addresses R-016)

> **Expected outcome:** Readiness moves from approximately 80 to approximately 88 as Dependency Risk rises to about 23. The platform is structurally ready for broader AI deployment.

Estimated effort (illustrative): 1 to 2 months, 1 engineer.

AI Use-Case Sequencing

The platform is in the Red zone, so AI follows modernization. AI readiness tracks change-safety readiness: the same dark seams that gate the migration gate the AI.

Safe today: read-only or sandboxed (deployable during Phase 0 to 1)

- Read-only analysis of the codebase and dependencies (triage, documentation, code explanation)
- AI-assisted characterization-test generation against the dark zones
- AI-assisted retarget mechanics (csproj conversion proposals)

Safe after Phase 2 completion

- Sandboxed AI code transformation for the System.Web-to-ASP.NET-Core rewrite, the EF6-to-EF-Core migration, and the OWIN-to-Identity scaffolding, run behind the Phase 0 net in a non-production environment

Not safe until Phase 3 completion

- AI in the live pipeline (assisted review, observability and anomaly tooling on the new logging stack, ongoing dependency maintenance)
- Any production-scale AI taking operational actions

Recommendations for the Sponsor

Written for a sponsor as it would read in a real engagement, to show the deliverable's shape. This is an illustrative public-repo teardown, so there is no actual sponsor.

SocialGoal is a modernization candidate, not a modernization-in-progress. It is small, cleanly layered, and genuinely modernizable, but it is not safe to start the big rebuilds today, and the readiness score (44 out of 100, Red zone) reflects that honestly: the platform and dependency layers drag, the architecture helps, and the safety net is real but shallow where it matters most.

Fund the work in order: first the Phase 0 safety net and a live-database inspection to close the trigger unknown, before any migration spend, because that is small money that de-risks everything after it; then the mechanical Phase 1 retarget, which is unlocking; then the Phase 2 rebuilds, which are the bulk of the effort and the bulk of the value.

Verify two things before committing the full budget: the open database-trigger question, because hidden behavioral coupling is the classic late-stage surprise and this teardown could not close it from source; and the breaking-bump analysis on the vulnerable packages, so the dependency work is scoped rather than discovered.

Where AI fits: this is a system where AI can compress the rebuild meaningfully, but only behind the net. Funding Phase 0 first is also what unlocks sandboxed AI in Phase 2, so the safety investment and the AI leverage are the same investment.

Honesty caveat for the data room

Written as it would appear in a real data-room handoff, to show the deliverable's shape. This public-repo teardown has no live data room.

Coverage was reported full, but cross-project semantic confidence is reduced (26 workspace load failures during static analysis); the layer-boundary severity (R-003) was contested in v1.0 and resolved by the v1.1 code read (no bypass; composition-root wiring); and the database-trigger question is open. None of these change the thesis; all of them should be visible to the buyer.

End of Report

Prepared by Evincia LLC using the Modernization Shield methodology · Atlanta, GA · evincia.co