Local Entities — Feature Addition

AWS Cost & Delivery Analysis — adding local-entity support *onto the existing, in-production Statewide Agreements Portal*

Complexity: Medium | Feature add — not a new build | Dedicated Step 2 + Admin page | Self-service sign-in for local entities | Reuses existing validation engine & infra | Target: production in 3 months

FORMAL COST ESTIMATE

Local Entities Submission Feature — Statewide Agreements Portal

The following estimate has been prepared in the format of the Office of State Controller cost estimate, and is anchored to the actual build history of the existing Statewide Agreements Portal (one deployed application repository, ~11,500 lines of code across ~135 commits over ~5.5 months with ~3 developers). Unlike a from-scratch clone, this estimate scopes a ***feature addition layered onto the live codebase****.*

PROJECT SUMMARY

Extend the existing Statewide Agreements Portal so that **local entities** (counties, districts, and other local government bodies) can submit their own reportable data through the same portal already serving state agencies. Rather than building a separate product, this work adds a parallel path inside the current application. The primary capabilities required are:

1. A dedicated **Step 2 (Validate Excel) page for local entities** — a local-entity variant of the existing validation step with its own column rules, labels, and example template, reusing the proven in-browser validation engine.
2. A dedicated **admin page for local entities** — a local-entity view/section layered into the existing admin console to manage the local-entity roster, contacts, submissions, and status, separate from the state-agency admin view.
3. A **sign-in path for local-entity users** — self-service authentication for the external local-entity audience, alongside the existing SAML federation already used by internal staff.

4. Supporting plumbing — data-model additions to distinguish local entities from agencies, routing, role/scope gating, and reuse of the existing versioned-batch and lineage machinery for the new entity type.

PROGRAMMING & DEVELOPMENT COST ESTIMATE

The preliminary programming cost estimate assumes 1–2 developers at a $100/hr average on a front-loaded, three-month schedule. Because this is a **feature added to a running application** — the validation engine (`src/lib/excel-validation`), the versioned-batch/lineage logic, the DynamoDB data layer, the admin console (~1,650 LOC already built), the SAML session handling, and the entire serverless infrastructure are **already in place** — the effort is scoped to new pages, a new auth path, and the data-model changes needed to keep local entities distinct from state agencies:

Local-entity Step 2 (Validate Excel) page — new column rules, labels, example template, variant of the existing validation step (~90–150 hrs)	$9,000–$15,000
Local-entity admin page — new view/section in the existing admin console: roster, contacts, submissions, status (~100–170 hrs)	$10,000–$17,000
Local-entity sign-in — self-service auth path (Cognito) alongside existing SAML, onboarding + reset flows, role/scope gating (~110–180 hrs)	$11,000–$18,000
Data model + routing + wiring — distinguish local entities from agencies, scope queries, reuse batch/lineage for the new type (~70–120 hrs)	$7,000–$12,000
Production hardening — QA, accessibility, WAF/login tuning, a cycle of post-release support (~50–90 hrs)	$5,000–$9,000
AI development tooling — per-developer coding assistants / LLM usage across the 3-month build (1–2 devs)	$300–$1,200
Grand Total Feature Addition	**~$42,300–$72,200**

The estimated total development cost includes project management, analysis, program specifications, programming, unit testing, quality assurance activities, and one cycle of post-production implementation support. Because the work extends an already-built, in-production system — reusing the validation engine, batch/lineage logic, data layer, admin shell, and serverless infrastructure wholesale — the risk profile is lower than a greenfield build, and the hours reflect *new surfaces and the new auth path* rather than re-creating the core. AI development tooling is estimated as a per-developer subscription/usage cost (paid AI coding assistants and LLM API usage) spread across the 3-month build for 1–2 developers. These lines can move toward — or beyond — the high end if scope grows (e.g., the local-entity data model diverges substantially from the agency model), developers lean on premium AI tiers, or unforeseen roadblocks extend the build.

HARDWARE/SOFTWARE SUSTAINMENT COST ESTIMATE

This feature ships on the **existing serverless infrastructure** of the Statewide Agreements Portal — the same CloudFront → S3/Lambda → DynamoDB stack already running in both the test and production environments. No new environments are provisioned; the incremental run cost is the marginal traffic, storage, and (if used) identity for local-entity users on top of what is already deployed. Monthly *incremental* cost estimates are as follows:

Production — added local-entity traffic, Lambda invocations, DynamoDB throughput, document storage	**+$10–40/mo**
Test / dev — exercising the new paths, minimal added load	**+$2–10/mo**
Self-service identity for local-entity users (well under 50k MAU free tier)	**~$0/mo**
Maintenance — ongoing support, monitoring, patching, and minor enhancements for the local-entity feature	**+$600/mo**
Total Added Monthly Cost (test/dev + prod, all-in)	**~$612–650/month**

DEVELOPMENT ASSUMPTIONS

1. Estimate **adds a feature to the existing Statewide Agreements Portal** rather than building a new or cloned application — the live repository is the baseline, not a copy.
2. The in-browser Excel validation engine, versioned-batch/lineage logic, DynamoDB single-table data layer, admin console shell, SAML session handling, and the serverless CloudFront → S3/Lambda infrastructure are **reused as-is**.
3. Local entities get a **dedicated Step 2 page** (its own column rules/labels/template) and a **dedicated admin page** (a new section of the existing admin console), kept visually and behaviorally consistent with the agency flow.
4. Authentication is mixed: existing SAML federation continues for internal staff; a **new self-service sign-in** (Cognito, assumed free under 50k MAU) is added for external local-entity users, with onboarding and password-reset flows.
5. The data model is extended to distinguish local entities from state agencies and to scope queries/roles accordingly; no change of database technology (stays on DynamoDB).
6. No new AWS environments are created; the feature deploys to the existing test and production environments via the current Bitbucket Pipelines + AWS OIDC CI/CD.
7. Out of scope: a separate standalone application, migration of agency data, publication templates, advanced cross-entity reporting beyond the existing admin patterns, and automated public publishing.

DELIVERY ESTIMATE

The existing Statewide Agreements Portal was built over roughly **5.5 months** (April–September 2026) by about **3 developers** across ~11,500 lines of application code. Because this work is a **feature addition** that reuses the validation engine, batch/lineage logic, data layer, admin shell, and infrastructure, the preliminary delivery estimate is **3 months** with 1–2 developers on a front-loaded schedule, targeting a production-ready release at the end of month three. Completion is dependent on availability of staff with the required experience/knowledge base, existing and unforeseen project priorities, out-of-scope enhancements, customer acceptance delays, and production processing cycles (including Fiscal and Calendar Year End).

DISCLAIMER

This estimate is valid for 30 days from the creation date of this document. It is an estimate for services only, not legally binding, and not a guarantee that the services will be performed in strict adherence to its terms. Any changes in cost, scope, and/or time required will be clearly communicated to the customer.

CUSTOMER ACCEPTANCE

Name: _____ Signature: _____

Date: _____

~$42.3k–72.2k	~$12–50/mo	3 months
One-time development (labor + AI, feature add)	Added runtime — marginal, on existing environments	To production, 1–2 developers

Executive summary

This project adds **local entities** (counties, districts, and other local bodies) to the Statewide Agreements Portal our office already runs for state agencies — **without building a new system**. Local-entity staff would use the same portal, with a submission page and an admin view tailored to them, and their own way to sign in.

The big advantage is that **the hard parts already exist and are in production**. The spreadsheet validation engine, the versioned submission history, the data store, the admin console, and the

entire cloud setup are already built and running. This work **layers a local-entity path on top** of that live application: a dedicated Step 2 (upload & validate) page, a dedicated admin page, and a self-service sign-in for the external local-entity users.

The main investment is the **one-time cost to add the feature** — roughly **$42,300–$72,200**, using 1–2 developers over about 3 months to a production-ready release. Because it rides on the existing cloud environments, the added ongoing cost to run it is very small — on the order of **a few hundred dollars a year** on top of what we already pay.

- **Who it serves:** local-entity submitters (counties/districts), plus the internal review staff who already use the portal.
- **One-time build:** ~$42.3k–$72.2k (feature added to the proven portal).
- **Added ongoing cost:** ~$144–$600 per year on top of current hosting — covers both test/dev and production.
- **Timeline:** about 3 months to production.
- **Approach:** extend the live Statewide Agreements Portal with a local-entity Step 2 page, admin page, and sign-in — reusing the existing engine and infrastructure.

What this project actually is

This is a **feature addition** to the Statewide Agreements Portal (`statewide-agreements`), not a new or cloned application. The live codebase (~11,500 LOC, ~135 commits over ~5.5 months) already delivers every hard part. The required scope here is to add a **parallel local-entity path** inside that same app, with:

- **A dedicated Step 2 page for local entities** — a local-entity variant of `StepValidateExcel` (~1,100 LOC today) with its own column rules, labels, and example template, reusing the browser-side validation engine (required fields, dates, emails, phones, money, allowed-value dropdowns), inline edit, jump-to-error, and highlighted-error export.
- **A dedicated admin page for local entities** — a new section layered into the existing admin console (`AdminPage`, ~1,650 LOC today) to manage the local-entity roster, contacts, submissions, and status, kept separate from the state-agency admin view.
- **A sign-in path for local-entity users** — self-service authentication (Cognito) for the external audience, added alongside the existing SAML federation (`src/lib/auth`) used by internal staff, with onboarding and password-reset flows.
- **Data-model & routing additions** — distinguish local entities from agencies, scope queries and roles, and reuse the existing versioned-batch and lineage machinery for the new entity type.

Why this is lower-risk than a new build: we are extending an application that is *already running in production*. The validation engine, batch/lineage logic, DynamoDB data layer, admin shell, SAML session handling, and

serverless infrastructure all carry over untouched — the cost is new surfaces (Step 2 page, admin page) and the new sign-in path, not re-creating the core. This is a fraction of the original ~5.5-month, 3-developer effort.

Scope at a glance

Local-entity Step 2 **new page**

- Variant of the existing Validate-Excel step.
- Local-entity column rules, labels, example row.
- Reuses validation, inline edit, error export engine.

Local-entity Admin **new page**

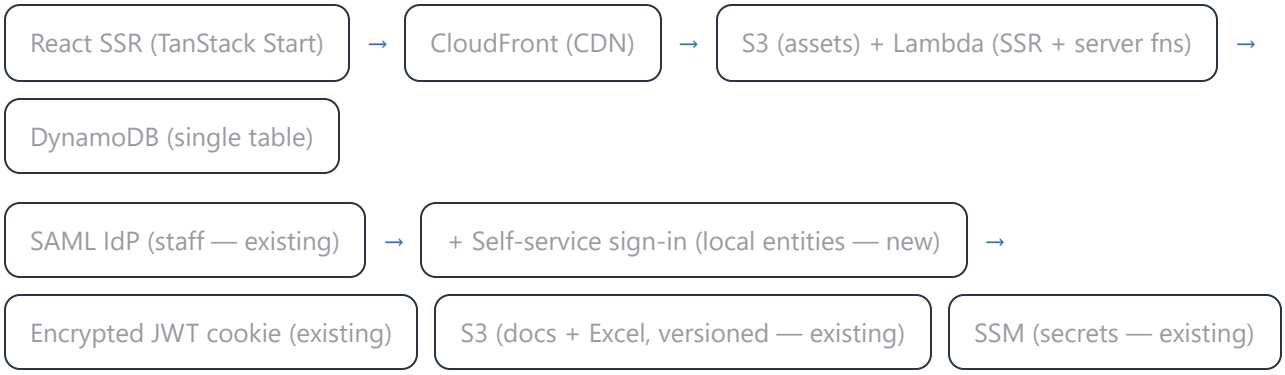
- New section in the existing admin console.
- Roster, contacts, submissions, status.
- Separate from the state-agency admin view.

Local-entity Sign-in **new auth**

- Self-service sign-in for external users.
- Added alongside existing SAML for staff.
- Onboarding + password-reset flows.

Architecture — unchanged, reused

No new infrastructure. The feature ships on the exact serverless stack already running the portal.



What is reused as-is

- In-browser Excel validation engine (`src/lib/excel-validation`).
- Versioned batches + lineage, DynamoDB data layer (`src/lib/db.ts`, `server-fns.ts`).
- Admin console shell, SAML session handling, encrypted JWT cookie.
- CloudFront → S3/Lambda infra + Bitbucket OIDC CI/CD, both environments.

What is net-new

- Local-entity Step 2 page + its column rules/labels/template.
- Local-entity admin section (roster, contacts, submissions, status).
- Self-service sign-in path + onboarding/reset for external users.
- Data-model + routing to keep local entities distinct from agencies.

Development cost (one-time)

Assumes 1–2 developers at a $100/hr average, over 3 months, on a front-loaded (tapering) schedule, plus AI tooling. Hours cover the local-entity Step 2 page, admin page, sign-in path, data-model/routing work, and production hardening. There is *no base-clone line* — the core engine and infrastructure already exist and are reused.

SCENARIO	TEAM & DURATION	DEV-HOURS	LABOR	AI	TOTAL
Low	1 dev × ~3 months	420	$42,000	$300	$42,300
Midpoint	~1.5 devs × ~3 months	570	$57,000	$700	$57,700
High	**2 devs × ~3 months**	**710**	**$71,000**	**$1,200**	**$72,200**

Dev-hours = Step 2 page (90–150) + admin page (100–170) + sign-in path (110–180) + data-model/routing (70–120) + production hardening (50–90). Labor = total dev-hours × $100. AI tooling is scaled to the build: paid AI coding assistants + LLM API usage at roughly $40–$200 per developer per month across 1–2 developers over the 3-month timeline (~$300 low to ~$1,200 high), with headroom to rise if complexity or roadblocks increase.

3-month delivery plan

Front-loaded schedule targeting a production-ready release at the end of month three.

Month 1	**Foundation & sign-in.** Data-model additions to distinguish local entities from agencies; routing and role/scope gating; stand up the self-service sign-in path (Cognito) alongside existing SAML, with onboarding and password-reset flows. Heaviest-hours month.
Month 2	**Core pages.** Build the local-entity Step 2 (Validate Excel) page with its column rules, labels, and example template on top of the existing validation engine; build the local-entity admin page as a new section of the admin console (roster, contacts, submissions, status).
Month 3	**Integration, QA & hardening.** End-to-end testing of the local-entity flow, accessibility, WAF/login tuning, deploy to test then production via the existing CI/CD, and one cycle of post-release support. Tapering hours.

Added monthly cost — marginal, on existing environments

No new environments are provisioned. The figures below are the **incremental** cost of local-entity activity on top of the portal's current bill — not a fresh per-environment total.

SERVICE	INCREMENTAL ASSUMPTION	ADDED $/MO
Lambda (SSR + server fns)	Added submit/review/export invocations for local entities	$3–15
DynamoDB (single table)	On-demand reads/writes for new entities & batches	$2–10
S3 + CloudFront	Added document/Excel storage + transfer	$2–10
Self-service identity	< 50k MAU free tier for local-entity users	~$0
CloudWatch / logs	Added log volume	$1–5
AWS WAF	Already deployed; marginal rule evaluation	$1–5
Total added	**Test/dev + prod, all-in incremental**	**~$12–50/mo**

Tip: because the existing stack is serverless and on-demand, local-entity activity bills only when it happens — heavy at reporting deadlines, near zero otherwise. There is no new fixed floor to pay for an added database or environment.

Total cost of ownership — Year 1

COMPONENT	BASIS	LOW	HIGH
Development (one-time)	Labor + AI, feature add	$42,300	$72,200
Added runtime — test/dev + prod	12 months incremental (on existing environments)	$144	$600
Year 1 total	**Feature build + a full year of added hosting**	**~$42.4k**	**~$72.8k**

Takeaway: Development labor is essentially all of Year 1 cost. The added AWS runtime (~$144–$600/year on top of the existing bill) is a rounding error. Focus estimate scrutiny on developer count, how cleanly the local-entity data model fits alongside the agency model, and the sign-in path — not on infrastructure.

Cost drivers & where it can grow

Reuse is the swing factor

The closer the local-entity data model, column rules, and admin needs track the existing agency flow, the closer this lands to the low end. The engine and infrastructure are proven in production, so the risk is in the new surfaces (Step 2 page, admin page) and the new sign-in path.

Where cost can grow

- Local-entity data model diverging substantially from the agency model (more data-layer work).
- Onboarding/support load for a less-technical external audience.
- Local-entity-specific reporting beyond the existing admin patterns.
- Out-of-scope extras: standalone app, data migration, publication templates, public publishing.

Bottom line

- **Development is the real cost:** ~$42.3k–72.2k one-time (1–2 devs, $100/hr, front-loaded over 3 months, +$0.3k–1.2k AI tooling).
- **Feature add, not a build:** it layers onto an in-production system — the validation engine, batch/lineage, data layer, admin shell, SAML, and serverless infra are reused, so there is no base-clone line.
- **Three new surfaces:** a local-entity Step 2 page, a local-entity admin page, and a self-service sign-in alongside the existing SAML for staff.
- **No new infrastructure:** deploys to the existing test and production environments via the current CI/CD.
- Added runtime for **test/dev + prod**: ~$12–50/mo incremental (~$144–$600/year) on top of the current bill.
- Extend the live Statewide Agreements Portal with local-entity support — the lowest-risk path to delivery, production-ready in 3 months.

Estimates are order-of-magnitude for us-east/us-west, on-demand/serverless pricing, seasonal traffic. Build figures are anchored to the actual git history of the Statewide Agreements application repository (~11,500 LOC, ~135 commits, ~5.5 months, ~3 developers) and scope this work as a feature addition that reuses that codebase and its infrastructure. Free-tier assumptions apply to self-service identity. Validate with the AWS Pricing Calculator and a load estimate for deadline-week peak before quoting.