Add icon Add cover

6.05 Research Location Map View

As a UEL management user

I want to see map of our research knowledge base

So that I can see the spread and reach of UEL's research activities

Acceptance Criteria:

1. In the [homepage or dataset page' there should be a tab for map view
2. A world map will be shown, segmented into countries
3. Datasets will have a new 'research country' metadata field
4. When entering countries in new dataset or modify dataset, there will be a country text box with AutoComplete.
5. The number of available datasets will be shown when hovering over a country on the map
6. Clicking on a country will perform a dataset search for all datasets related to that country
7. The map should have zoom and pan controls
8. Zoom level of the map should default to the world view

Overview

Purpose

`ckanext-locations` will provide a dataset-level geographic discovery experience where users can explore datasets on an interactive world map, matching the `locations-wireframe.html` behavior. The same extension will also add dataset metadata controls for selecting a country (with coordinates) in dataset create/edit flows.

Key Goals

- Show all published datasets on a map grouped by country.
- Support hover/click interactions to reveal dataset counts and drill into dataset lists.
- Maintain a master country table (country name, ISO codes, latitude, longitude, status).
- Expose a scheming-driven autocomplete field in dataset create/edit UI for country selection.
- Keep implementation inside `ckanext-locations` while integrating with `ckanext-scheming`.

Scope

- **In Scope**:
  - Locations page map viewer for dataset coverage.
  - Backend aggregation APIs for map markers and country stats.
  - Master country table + migration + seed strategy.
  - Scheming dataset schema updates for country autocomplete.
  - Dataset create/edit UI extension behavior and validation.
- **Out of Scope**:
  - Multi-level administrative boundaries (state/city polygons).
  - Geocoding arbitrary free-text addresses.
  - Full GIS analytics (heatmaps, choropleths)

Architecture Overview

High-Level Architecture

JSON

```
┌──────────────────────────────────────────────────────────────────┐
│                          CKAN Instance                           │
├──────────────────────────────────────────────────────────────────┤
│ CKAN Core                                                        │
│ ├─ Dataset model/package table                                   │
│ ├─ Resource and search index                                     │
│ ├─ Plugin system                                                 │
│ └─ Form & template rendering                                     │
├──────────────────────────────────────────────────────────────────┤
│ ckanext-scheming                                                 │
│ ├─ Dataset schema JSON/YAML                                      │
│ ├─ Field rendering hooks                                         │
│ └─ Custom validators/converters                                  │
├──────────────────────────────────────────────────────────────────┤
│ ckanext-locations (this extension)                               │
│ ├─ Dataset Locations UI                                          │
│ │  ├─ Leaflet map page                                           │
│ │  ├─ Country markers + popups                                   │
│ │  └─ Dataset drilldown links                                    │
│ ├─ Country Master Data                                           │
│ │  ├─ locations_country table                                    │
│ │  ├─ seed loader + sync command                                 │
│ │  └─ lookup/validation services                                 │
│ ├─ API Endpoints                                                 │
│ │  ├─ country autocomplete                                       │
│ │  ├─ map aggregate stats                                        │
│ │  └─ dataset-country validation                                 │
│ └─ Scheming Integration                                          │
│    ├─ autocomplete field assets                                  │
│    ├─ validators                                                 │
│    └─ template snippets                                          │
├──────────────────────────────────────────────────────────────────┤
│ External Services                                                │
│ ├─ Leaflet.js                                                    │
│ ├─ OpenStreetMap tiles                                           │
│ └─ (Optional) Country seed source (ISO dataset)                  │
└──────────────────────────────────────────────────────────────────┘
```

User Experience Design

1. Locations Screen (Public Discovery)

- Page route: `/locations` (or configurable route prefix).
- Map base: Leaflet + OpenStreetMap.
- Marker model: one marker per country with at least one dataset.
- Hover behavior:
  - Tooltip with `Country • N datasets`.
  - Side panel updates selected country summary.
- Click behavior:
  - Popup with country and count.
  - Redirect to dataset search page with pre-applied country filter.
  - Example target: `/dataset?country_code=SG`.

2. Dataset Create/Edit (Publisher Workflow)

- Country metadata field added via scheming on dataset form.
- Field rendered as autocomplete text input with dropdown suggestions.
- User searches by country name/ISO code.
- On selection, hidden fields populated:
  - `country_code` (ISO2/ISO3, final canonical key)
  - `country_name`
  - `country_latitude`
  - `country_longitude`
- If form opened for edit, existing country is preloaded and displayed.

3. Fallback Behavior

- If no country set on a dataset, dataset is excluded from map aggregate markers.
- Optional admin flag can include an `Unknown` bucket.

Data Model Design

1. Master Table: `locations_country`

	Column	Type	Constraints	Description
1	id	SERIAL / INTEGER	PK	Internal key
2	iso2	VARCHAR(2)	UNIQUE, NOT NULL	ISO-3166-1 alpha-2
3	iso3	VARCHAR(3)	UNIQUE, NOT NULL	ISO-3166-1 alpha-3
4	country_name	VARCHAR(128)	NOT NULL, indexed	Canonical display name
5	latitude	NUMERIC(9,6)	NOT NULL	Marker latitude
6	longitude	NUMERIC(9,6)	NOT NULL	Marker longitude
7	region_name	VARCHAR(128)	NULL	Optional region grouping
8	is_active	BOOLEAN	default TRUE	Enable/disable in autocomplete
9	created_at	TIMESTAMP	default now()	Audit
10	updated_at	TIMESTAMP	default now()	Audit

2. Dataset Metadata Fields (stored in extras via scheming)

	Field	Example	Purpose
1	country_code	SG	Canonical country reference for filtering/joins
2	country_name	Singapore	Human-readable display
3	country_latitude	1.3521	Fast map rendering fallback
4	country_longitude	103.8198	Fast map rendering fallback

3. Why both table + dataset extras?

- Table is authoritative source and powers autocomplete.
- Extras provide denormalized snapshot for search/index and compatibility.
- On update, country_code drives resolution of name/lat/lon from master table.

Backend Components

1. Plugin Entry

File: `ckanext/locations/plugin.py`

- Register blueprint routes for locations page and APIs.
- Expose template helpers for field rendering and country lookups.
- Register `IConfigurer`, `IBlueprint`, optional `IPackageController` hooks.

2. Country Repository Service

File: `ckanext/locations/country_repository.py`

- Query countries by prefix (name/ISO).
- Get country by ISO2/ISO3.
- Paginated and case-insensitive search.
- Returns normalized payload for autocomplete.

3. Dataset Location Service

File: `ckanext/locations/dataset_locations_service.py`

- Aggregate dataset counts by country code.
- Merge with `locations_country` table to obtain coordinates.
- Provide map payload: country, count, region, lat/lon, filter URL.

4. Scheming Validator/Converter

File: `ckanext/locations/validators.py`

- `locations_country_code_exists`: validate selected country code.
- `locations_country_populate_coords`: derive name/lat/lon from code.
- `locations_country_normalize_code`: uppercase, iso2 standardization.

5. API Endpoints

File: `ckanext/locations/views.py`

`GET /api/3/action/locations_country_autocomplete`

Query:

- `q`: search string
- `limit`: optional (default 20)

Response:

JSON

```
{
  "success": true,
  "result": [
    {
      "country_code": "SG",
      "country_name": "Singapore",
      "iso3": "SGP",
      "latitude": 1.3521,
      "longitude": 103.8198,
      "region_name": "Asia"
    }
  ]
}
```

`GET /api/3/action/locations_dataset_map_summary`

Response:

JSON

```
{
  "success": true,
  "result": {
    "generated_at": "2026-06-22T10:30:00Z",
    "total_countries": 42,
    "total_datasets": 1280,
    "countries": [
      {
        "country_code": "SG",
        "country_name": "Singapore",
        "dataset_count": 57,
        "latitude": 1.3521,
        "longitude": 103.8198,
        "dataset_url": "/dataset?country_code=SG"
      }
    ]
  }
}
```

Frontend Components

1. Locations Map Page

Files:

- `ckanext/locations/templates/locations/index.html`
- `ckanext/locations/public/js/locations-map-page.js`
- `ckanext/locations/public/css/locations-map-page.css`

Responsibilities:

- Render Leaflet map and side panel.
- Request aggregate endpoint once on load.
- Create country markers with consistent styling.
- Handle hover/click interactions and filter navigation.

2. Scheming Autocomplete Widget

Files:

- `ckanext/locations/public/js/scheming-country-autocomplete.js`
- `ckanext/locations/templates/scheming/snippets/country_autocomplete.html`

Responsibilities:

- Debounced API calls to country autocomplete endpoint.
- Keyboard navigation (up/down/enter/escape).
- Populate hidden dependent fields (`name`, `lat`, `lon`, `code`).
- Preserve selected value when editing datasets.

Scheming Integration Design

1. Dataset Schema Changes

Update scheming dataset schema (e.g., `dataset.yaml` / `dataset.json`) to include:

YAML

```yaml
- field_name: country_code
  label: Country
  preset: select
  form_snippet: country_autocomplete.html
  validators: locations_country_normalize_code locations_country_code_exists
  required: false
  help_text: Start typing country name or ISO code.

- field_name: country_name
  label: Country Name
  form_snippet: hidden.html
  display_snippet: text.html

- field_name: country_latitude
  label: Country Latitude
  form_snippet: hidden.html

- field_name: country_longitude
  label: Country Longitude
  form_snippet: hidden.html

```

Note:

- `country_code` is the only user-facing editable field.

- Other fields are derived and hidden.

2. Autocomplete Interaction Flow

1. User types `sin`.
2. Widget calls `/api/3/action/locations_country_autocomplete?q=sin`.
3. User selects `Singapore (SG)`.
4. Widget writes:
   - `country_code=SG`
   - `country_name=Singapore`
   - `country_latitude=1.3521`
   - `country_longitude=103.8198`
5. On submit, backend validators re-check code and normalize values.

3. CKAN Search Index Impact

- Include `country_code` and `country_name` in indexed fields.
- Enables direct filtering from locations page click-through.

Master Data Management Design

1. Migration Strategy

Files:

- `ckanext/locations/migration/versions/<timestamp>_create_locations_country.py`

Migration creates:

- `locations_country` table
- indexes on `iso2`, `iso3`, `country_name`, `is_active`

2. Seed Strategy

Files:

- `ckanext/locations/cli.py`
- `ckanext/locations/seed/countries.csv`

CLI command examples:

```shell
ckan -c ckan.ini locations countries seed
ckan -c ckan.ini locations countries sync --source countries.csv

```

3. Seed File Format

```csv
iso2,iso3,country_name,latitude,longitude,region_name,is_active
SG,SGP,Singapore,1.3521,103.8198,Asia,true
GB,GBR,United Kingdom,55.3781,-3.4360,Europe,true
```

4. Update Policy

- Master country data managed by admins only.
- Changes logged in activity/audit logs.
- Optional nightly sync from approved canonical source.

Data Flow

1. Dataset Create/Edit

```
Publisher opens dataset form
  ↓
Scheming renders country autocomplete field
  ↓
User selects country
  ↓
Hidden fields auto-populated
  ↓
Submit dataset
  ↓
Validator checks country_code against locations_country
  ↓
Canonical country metadata persisted in extras
```

2. Locations Map Load

```
User opens /locations
  ↓
Frontend calls locations_dataset_map_summary API
  ↓
Backend aggregates package counts by country_code
  ↓
Join with locations_country for lat/lon + display name
  ↓
Return marker payload
  ↓
Leaflet renders markers and interactions
```

3. Map Drilldown

```
User clicks country marker/popup
  ↓
Redirect to /dataset?country_code=XX
  ↓
CKAN search applies country filter
  ↓
User sees country-specific datasets
```

Security and Validation

Input Validation

- Country autocomplete query length/rate limits.
- SQL-safe parameterized queries only.
- Validate and sanitize all API outputs.

Dataset Metadata Integrity

- Reject unknown country codes.
- Prevent manual tampering of hidden lat/lon by deriving server-side.

Access Control

- Public map endpoint returns only datasets visible to requesting user.
- Private datasets excluded unless authorized.

Performance Considerations

- Aggregate query uses indexed `country_code` in package extras/search table.
- API response can be cached (short TTL, e.g., 5 minutes).
- Map payload is compact (one point per country, not per dataset).
- Debounced autocomplete requests (e.g., 200-300ms).

Implementation Plan

1: Foundations

- Create `locations_country` migration and repository.
- Add seed command + initial country data.
- Add validators and helper functions.

2: Scheming Integration

- Create autocomplete snippet and JS.
- Update dataset schema fields.
- Validate create/edit flow + edit prefill.

3: Locations Screen

- Build `/locations` page template and map JS.
- Implement dataset aggregate API.
- Add click-through filter navigation.

High-Level Estimates

Assumptions:

- Existing CKAN 2.10 environment with `ckanext-scheming` already enabled.
- One frontend-leaning engineer and one backend-leaning engineer can work in parallel.

- Estimates include development + basic refactoring, but exclude long UAT cycles and release approvals.

Effort Summary

	Area	Estimate (hours)	Estimate (days)	Notes
1	Frontend	36	4.5	Locations map UI + scheming autocomplete widget
2	Backend	52	6.5	APIs, migration, validators, country master management
3	Integration + QA Support	12	2	End-to-end fixes, regression checks, polish
4	**Total**	**104**	**13**	

Frontend High-Level Task Breakdown

	Task	Estimate (hours)	Deliverables
1	Locations page scaffold + Leaflet integration	8	Map container, tile layer, panel layout
2	Marker rendering + interactions	6	Hover tooltip, popup, click-through filtering
3	Country autocomplete widget for scheming	6	Debounced search, keyboard support, hidden field population
4	Styling + responsive + accessibility pass	6	Mobile layout fixes, focus states, ARIA checks
5	Frontend test/support	4	UI behavior verification and bug fixes
6	Add locations filters in dataset screen	8	

Backend High-Level Task Breakdown

	Task	Estimate (hours)	Deliverables
1	Country master table migration + indexes	6	`locations_country` schema and migration

2	Seed/sync CLI for country data	8	Seed command, CSV import, update flow
3	Country autocomplete API	6	Search endpoint with limits and normalized payload
4	Dataset map aggregate API	8	Country-level counts + visibility-safe response
5	Scheming validators/converters	8	Code validation + server-side lat/lon derivation
6	Search/filter integration + tests	8	Country filter parity + unit/functional coverage
7	Add locations filter in Dataset	8	

Testing Strategy

Unit Tests

- `test_country_repository.py`
- `test_locations_validators.py`
- `test_dataset_locations_service.py`

Functional Tests

- Dataset create/edit saves canonical country metadata.
- Autocomplete endpoint correctness and limits.
- Map summary endpoint visibility rules (public/private).

UI Tests

- Autocomplete keyboard/mouse behavior.
- Map marker rendering and hover/click behavior.
- Redirect filter correctness.

Required Files (Planned)

CSV

```
ckanext/locations/
├── plugin.py
├── views.py
├── validators.py
├── country_repository.py
├── dataset_locations_service.py
├── cli.py
├── migration/
│   └── versions/
│       └── <timestamp>_create_locations_country.py
├── seed/
│   └── countries.csv
```

```
├── templates/
│   ├── locations/
│   │   └── index.html
│   └── scheming/
│       └── snippets/
│           └── country_autocomplete.html
└── public/
    ├── js/
    │   ├── locations-map-page.js
    │   └── scheming-country-autocomplete.js
    └── css/
        └── locations-map-page.css
```

Configuration

Suggested config keys:

INI

```
ckanext.locations.map_enabled = true
ckanext.locations.map_default_zoom = 2
ckanext.locations.map_min_zoom = 2
ckanext.locations.map_max_zoom = 7
ckanext.locations.country_autocomplete_limit = 20
ckanext.locations.country_seed_file = ckanext/locations/seed/countries.csv
ckanext.locations.map_cache_ttl_seconds = 300
```

Risks and Mitigations

	Risk	Impact	Mitigation
1	Inconsistent existing dataset country metadata	Incorrect map counts	One-time backfill script + strict validator enforcement
2	Missing country in master table	Form save failure	Admin tooling to add/update country records quickly
3	High traffic on autocomplete API	Slow form UX	Debounce + limit + lightweight indexed query + caching
4	Private dataset leakage in map counts	Security issue	Apply CKAN auth checks in aggregate query/service layer

Success Metrics

- 100% of new datasets can select country via autocomplete.
- Locations page loads map markers in < 2 seconds for normal traffic.
- Country marker counts match dataset search results for filter parity.
- <1% validation error rate after migration/backfill period.