**User Stories: Specification Management API (C# Backend)**

This document outlines user stories for developing a C# REST API to perform CRUD operations on the Specification\* tables (SpecificationIdentifyingInformation, SpecificationCore, SpecificationExtensionComponents) in the RegistryDatabase.

Assumptions & Decisions:

* The API will be built using C# / .NET.
* An ORM like Entity Framework Core will likely be used for data access.
* Standard REST principles will be followed (HTTP verbs, status codes).
* Deletion: Deleting a SpecificationIdentifyingInformation record is prevented if related SpecificationCore or SpecificationExtensionComponents records exist (User must delete children first).
* Data Retrieval: Fetching a single SpecificationIdentifyingInformation by ID will include its related Core and Extension details, with pagination applied to these child lists. Fetching the list of all SpecificationIdentifyingInformation headers will also be paginated.

**Feature: Specification Identifying Information Management**

User Story 1: Create a New Specification Header

* As a user/system,
* I want to create a new specification header record by providing its identifying information (name, sector, purpose, version, etc.),
* So that I can establish a new specification entry in the system.
* Tasks (Estimate: < 1 Day Total):
  + Task 1.1 (API Design): Define POST /api/specifications endpoint, request body (DTO), and success/error responses (e.g., 201 Created with location header, 400 Bad Request). (Est: < 1 hr)
  + Task 1.2 (Controller): Implement C# API controller action for the POST endpoint, including input validation. (Est: < 2 hrs)
  + Task 1.3 (Service/Logic): Implement C# service layer logic to handle the creation business rules. (Est: < 1 hr)
  + Task 1.4 (Data Access): Implement C# data access logic (e.g., using EF Core) to insert the new record into SpecificationIdentifyingInformation. (Est: < 2 hrs)
  + Task 1.5 (Testing): Write unit/integration tests for the creation endpoint and logic. (Est: < 2 hrs)

User Story 2: Retrieve Specification Header(s)

* As a user/system,
* I want to retrieve specification header information, either a specific one by its ID (including paginated details) or a paginated list of available headers,
* So that I can view existing specifications efficiently.
* Tasks (Estimate: ~1.5 Days Total):
  + Task 2.1 (API Design - Get by ID): Define GET /api/specifications/{id} endpoint. Include query parameters for pagination of child lists (e.g., corePageNumber=1, corePageSize=10, extPageNumber=1, extPageSize=10). Define response body (DTO) including the main header details and *paginated* lists/metadata for related Core/Extension details. (Est: < 2 hrs)
  + Task 2.2 (Controller - Get by ID): Implement controller action for GET /api/specifications/{id}, parsing pagination parameters for child lists. (Est: < 1.5 hrs)
  + Task 2.3 (Service/Data Access - Get by ID): Implement service/data access logic to retrieve the SpecificationIdentifyingInformation record *and* apply pagination when retrieving its related SpecificationCore and SpecificationExtensionComponents records. (Est: < 4 hrs)
  + Task 2.4 (API Design - Get List): Define GET /api/specifications endpoint. Include query parameters for pagination (e.g., pageNumber=1, pageSize=20). Define response body (paginated list of header DTOs - *without* full child details for performance, but including pagination metadata like total count, page size, current page). (Est: < 1 hr)
  + Task 2.5 (Controller - Get List): Implement controller action for GET /api/specifications, parsing pagination parameters. (Est: < 1 hr)
  + Task 2.6 (Service/Data Access - Get List): Implement service/data access logic to retrieve a paginated list of SpecificationIdentifyingInformation records. (Est: < 2 hrs)
  + Task 2.7 (Testing): Write unit/integration tests for both retrieval endpoints, including pagination logic. (Est: < 4 hrs)

User Story 3: Update an Existing Specification Header

* As a user/system,
* I want to update the details of an existing specification header (e.g., change its name, purpose, version),
* So that the specification information remains accurate.
* Tasks (Estimate: < 1 Day Total):
  + Task 3.1 (API Design): Define PUT /api/specifications/{id} endpoint, request body (DTO), and success/error responses (e.g., 200 OK/204 No Content, 400 Bad Request, 404 Not Found). (Est: < 1 hr)
  + Task 3.2 (Controller): Implement controller action for the PUT endpoint, including input validation. (Est: < 2 hrs)
  + Task 3.3 (Service/Logic): Implement service layer logic to handle update rules. (Est: < 1 hr)
  + Task 3.4 (Data Access): Implement data access logic to find and update the record in SpecificationIdentifyingInformation. (Est: < 2 hrs)
  + Task 3.5 (Testing): Write unit/integration tests for the update endpoint. (Est: < 2 hrs)

User Story 4: Delete a Specification Header

* As a user/system,
* I want to delete a specific specification header, but only if it has no associated core or extension elements,
* So that I can remove obsolete specifications without orphaning data.
* Tasks (Estimate: < 1 Day Total):
  + Task 4.1 (API Design): Define DELETE /api/specifications/{id} endpoint and success/error responses (e.g., 204 No Content, 404 Not Found, 409 Conflict/400 Bad Request if children exist). (Est: < 0.5 hr)
  + Task 4.2 (Controller): Implement controller action for the DELETE endpoint. (Est: < 1 hr)
  + Task 4.3 (Service/Logic): Implement service layer logic to first check if any related records exist in SpecificationCore or SpecificationExtensionComponents for the given id. If children exist, return an appropriate error. If not, proceed with deletion. (Est: < 2.5 hrs)
  + Task 4.4 (Data Access): Implement data access logic to perform the check for children and, if the check passes, delete the SpecificationIdentifyingInformation record. (Est: < 2 hrs)
  + Task 4.5 (Testing): Write unit/integration tests for the delete endpoint, covering both successful deletion and prevention scenarios. (Est: < 2 hrs)

**Feature: Specification Core Element Management**

*(These assume operations are relative to a specific Specification identified by specificationId)*

User Story 5: Add a Core Element to a Specification

* As a user/system,
* I want to add a specific Core Invoice Model element (identified by its BusinessTermID) to an existing specification (identified by specificationId), optionally overriding cardinality and adding usage notes,
* So that I can build the core structure of my specification.
* Tasks (Estimate: < 1 Day Total):
  + Task 5.1 (API Design): Define POST /api/specifications/{specificationId}/coreElements endpoint, request body (DTO with BusinessTermID, Cardinality, UsageNote, TypeOfChange), and responses. (Est: < 1 hr)
  + Task 5.2 (Controller): Implement controller action, validating input and ensuring the parent specificationId exists. (Est: < 2 hrs)
  + Task 5.3 (Service/Logic): Implement service logic, including validating the BusinessTermID exists in CoreInvoiceModel. (Est: < 1 hr)
  + Task 5.4 (Data Access): Implement data access to insert a record into SpecificationCore. (Est: < 1.5 hrs)
  + Task 5.5 (Testing): Write tests. (Est: < 2 hrs)

User Story 6: Retrieve Core Elements for a Specification

* As a user/system,
* I want to retrieve (paginated) core elements associated with a specific specification (identified by specificationId),
* So that I can view the core structure of that specification efficiently.
* Tasks (Estimate: < 1 Day Total):
  + Task 6.1 (API Design): Define GET /api/specifications/{specificationId}/coreElements endpoint. Include query parameters for pagination (e.g., pageNumber=1, pageSize=20). Define response (paginated list of core element DTOs with metadata). (Est: < 1 hr)
  + Task 6.2 (Controller): Implement controller action, parsing pagination parameters. (Est: < 1 hr)
  + Task 6.3 (Service/Data Access): Implement logic to retrieve a paginated list of records from SpecificationCore filtered by IdentityID. (Est: < 2 hrs)
  + Task 6.4 (Testing): Write tests for retrieval and pagination. (Est: < 2 hrs)

User Story 7: Update a Core Element within a Specification

* As a user/system,
* I want to update the details (cardinality, usage note, type of change) of a specific core element (identified by coreElementId) within a specification (identified by specificationId),
* So that I can modify how a core element is used in that specification.
* Tasks (Estimate: < 1 Day Total):
  + Task 7.1 (API Design): Define PUT /api/specifications/{specificationId}/coreElements/{coreElementId} endpoint (using EntityID as coreElementId), request body, and responses. (Est: < 1 hr)
  + Task 7.2 (Controller): Implement controller action. (Est: < 1.5 hrs)
  + Task 7.3 (Service/Logic): Implement update logic. (Est: < 1 hr)
  + Task 7.4 (Data Access): Implement data access to find and update the SpecificationCore record. (Est: < 1.5 hrs)
  + Task 7.5 (Testing): Write tests. (Est: < 2 hrs)

User Story 8: Remove a Core Element from a Specification

* As a user/system,
* I want to remove a specific core element (identified by coreElementId) from a specification (identified by specificationId),
* So that I can refine the core structure of the specification.
* Tasks (Estimate: < 0.5 Day Total):
  + Task 8.1 (API Design): Define DELETE /api/specifications/{specificationId}/coreElements/{coreElementId} endpoint and responses. (Est: < 0.5 hr)
  + Task 8.2 (Controller): Implement controller action. (Est: < 1 hr)
  + Task 8.3 (Service/Data Access): Implement logic to find and delete the SpecificationCore record. (Est: < 1.5 hrs)
  + Task 8.4 (Testing): Write tests. (Est: < 1 hr)

**Feature: Specification Extension Component Management**

*(These assume operations are relative to a specific Specification identified by specificationId)*

User Story 9: Add an Extension Element to a Specification

* As a user/system,
* I want to add a specific Extension Component element (identified by its composite key: ExtensionComponentID and BusinessTermID) to an existing specification (identified by specificationId), providing details like cardinality, usage notes, justification, etc.,
* So that I can incorporate extensions into my specification.
* Tasks (Estimate: < 1 Day Total):
  + Task 9.1 (API Design): Define POST /api/specifications/{specificationId}/extensionElements endpoint, request body (DTO with ExtensionComponentID, BusinessTermID, Cardinality, UsageNote, Justification, TypeOfExtension), and responses. (Est: < 1 hr)
  + Task 9.2 (Controller): Implement controller action, validating input and parent specificationId. (Est: < 2 hrs)
  + Task 9.3 (Service/Logic): Implement service logic, including validating the composite key exists in ExtensionComponentModelElements. (Est: < 1.5 hrs)
  + Task 9.4 (Data Access): Implement data access to insert a record into SpecificationExtensionComponents. (Est: < 1.5 hrs)
  + Task 9.5 (Testing): Write tests. (Est: < 2 hrs)

User Story 10: Retrieve Extension Elements for a Specification

* As a user/system,
* I want to retrieve (paginated) extension elements associated with a specific specification (identified by specificationId),
* So that I can view the extension components included in that specification efficiently.
* Tasks (Estimate: < 1 Day Total):
  + Task 10.1 (API Design): Define GET /api/specifications/{specificationId}/extensionElements endpoint. Include query parameters for pagination (e.g., pageNumber=1, pageSize=20). Define response (paginated list of extension element DTOs with metadata). (Est: < 1 hr)
  + Task 10.2 (Controller): Implement controller action, parsing pagination parameters. (Est: < 1 hr)
  + Task 10.3 (Service/Data Access): Implement logic to retrieve a paginated list of records from SpecificationExtensionComponents filtered by IdentityID. (Est: < 2 hrs)
  + Task 10.4 (Testing): Write tests for retrieval and pagination. (Est: < 2 hrs)

User Story 11: Update an Extension Element within a Specification

* As a user/system,
* I want to update the details (cardinality, usage note, justification, type of extension) of a specific extension element (identified by extensionElementId) within a specification (identified by specificationId),
* So that I can modify how an extension element is used in that specification.
* Tasks (Estimate: < 1 Day Total):
  + Task 11.1 (API Design): Define PUT /api/specifications/{specificationId}/extensionElements/{extensionElementId} endpoint (using EntityID as extensionElementId), request body, and responses. (Est: < 1 hr)
  + Task 11.2 (Controller): Implement controller action. (Est: < 1.5 hrs)
  + Task 11.3 (Service/Logic): Implement update logic. (Est: < 1 hr)
  + Task 11.4 (Data Access): Implement data access to find and update the SpecificationExtensionComponents record. (Est: < 1.5 hrs)
  + Task 11.5 (Testing): Write tests. (Est: < 2 hrs)

User Story 12: Remove an Extension Element from a Specification

* As a user/system,
* I want to remove a specific extension element (identified by extensionElementId) from a specification (identified by specificationId),
* So that I can refine the extensions included in the specification.
* Tasks (Estimate: < 0.5 Day Total):
  + Task 12.1 (API Design): Define DELETE /api/specifications/{specificationId}/extensionElements/{extensionElementId} endpoint and responses. (Est: < 0.5 hr)
  + Task 12.2 (Controller): Implement controller action. (Est: < 1 hr)
  + Task 12.3 (Service/Data Access): Implement logic to find and delete the SpecificationExtensionComponents record. (Est: < 1.5 hrs)
  + Task 12.4 (Testing): Write tests. (Est: < 1 hr)