The Navigator's Digital Toolkit: A Granular Implementation Blueprint

Introduction: From Concept to Concrete Implementation

To fully leverage the AI-powered digital toolkit as a core component of the "No Wrong Door" program, it is essential to move beyond high-level strategy to a granular, operational blueprint. This document provides a deep-dive analysis of the toolkit's architecture and functionality, translating its ambitious vision into a concrete implementation plan. It details the specific processes, technologies, and user experiences that will empower Navigators, streamline service delivery, and ensure the data generated is used to continuously improve the program.
This blueprint is broken down into three key sections: the "Policy ETL" pipeline that creates a reliable data foundation from the official Maryland SNAP Program Manual; a practical example of "Rules as Code" (RaC) in action with specific code; and a detailed plan for ensuring the final product adheres to the highest standards of accessibility, plain language, and user-centered design as mandated for modern government services.

Section 1: The Policy ETL (Extract, Transform, Load) Pipeline in Practice

The platform's trustworthiness hinges on its ability to solve the "first-mile" data problem: the poor quality and inaccessibility of source policy documents. The Policy ETL pipeline is the engine that addresses this challenge by ingesting, structuring, and validating policy information before it ever reaches an AI model.
Objective: To convert the Maryland Department of Human Services (DHS) SNAP Program Manual, a series of complex documents, into a structured, queryable, and machine-readable dataset.
Granular Workflow:

1. **Extraction:**
   - **Source:** The latest version of the Maryland SNAP Program Manual, specifically "Section 212 - Deductions," is uploaded to the system's ingestion endpoint.
   - **Process:** An OCR (Optical Character Recognition) and document parsing service, such as Amazon Textract or Google Cloud Vision AI, analyzes the document. It identifies structural elements like headings, lists, and paragraphs.
   - **Example:** The parser identifies the section on the "Standard Deduction" and extracts the core rule and its associated values.
2. **Transformation:**
   - **Cleaning & Normalization:** The extracted text is cleaned of artifacts like page numbers and headers. A Natural Language Processing (NLP) model standardizes terminology. For instance, it recognizes that "Assistance Unit," "AU," and "household" are used interchangeably and maps them to a single canonical term: household.

- **Structuring:** The system converts the extracted content into a structured JSON format. The rule for the standard deduction is transformed from prose into a logical object.
  - **Input (from Manual Section 212):** "A standard deduction is subtracted from the assistance unit's net income. The standard deduction is based on the household size... For FFY 2024, the standard deduction is $198 for household sizes of 1 through 4, and $225 for a household size of 5, and $258 for a household size of 6 or more."
  - **Output (JSON):**
    ```
    {
      "policy_id": "MD-SNAP-MANUAL-212.1",
      "policy_name": "Standard Deduction",
      "effective_fiscal_year": 2024,
      "rule_type": "deduction",
      "logic_type": "tiered_lookup",
      "parameters": ["household_size"],
      "tiers": [
        { "min_household_size": 1, "max_household_size": 4,
    "deduction_amount": 198 },
        { "min_household_size": 5, "max_household_size": 5,
    "deduction_amount": 225 },
        { "min_household_size": 6, "max_household_size":
    "infinity", "deduction_amount": 258 }
      ],
      "source_document": "SNAP Manual Section 212",
      "last_updated": "2023-08-14T00:00:00Z"
    }
    ```
- **Enrichment:** NLP models tag entities within the text. The phrase "medical expenses over $35 a month for household members who are age 60 or older" is tagged with entities like deduction_type: medical, minimum_amount: 35, and qualifier_age: >=60.

3. **Loading:**
   - **Storage:** The structured, validated JSON data is loaded into a scalable NoSQL database, such as Google Firestore. Each policy rule becomes a versioned document, allowing for rapid, targeted queries.
   - **Versioning:** When a new SNAP manual is released, the ETL process is re-run, creating a new, versioned dataset. This ensures that eligibility determinations can be audited against the exact policy that was in effect at the time.

Section 2: "Rules as Code" (RaC) - From Regulation to API

The RaC engine translates the structured policy data from the ETL pipeline into executable logic, making it available via an API for other parts of the system to use.
Objective: To create a verifiable, real-time API endpoint that calculates the correct SNAP standard deduction for a given household.

Granular Workflow:

1. **Source Regulation:** The RaC engine references the structured JSON data created by the ETL pipeline for the Standard Deduction.
2. **Transformation to Structured Rule:** The logic is codified into a machine-readable rule. This is a structured representation of the policy that can be reviewed by policy analysts for accuracy.
    - **Example Rule (YAML format):**
      ```
      rule_id: "snap_standard_deduction_md_2024"
      name: "SNAP Standard Deduction Calculation"
      description: "Calculates the standard deduction based on
      household size per MD SNAP Manual Section 212."
      parameters:
        - name: household_size
          type: integer
          required: true
      logic:
        - type: "lookup_tiered"
          source_policy_id: "MD-SNAP-MANUAL-212.1"
          input_variable: "{{household_size}}"
          lookup_key: "household_size"
          return_value: "deduction_amount"
          assign_to: "calculated_deduction"
      output:
        deduction_amount: "{{calculated_deduction}}"
        rule_id: "snap_standard_deduction_md_2024"
      ```
3. **API Endpoint Design:** A secure, versioned API endpoint is created to execute this rule. It follows RESTful principles for clarity and predictability.
    - **Endpoint:** POST /api/v1/rules/snap/calculate_standard_deduction
    - **Authentication:** Requires a token authenticating the user as a certified Navigator.
4. **Code Implementation (Node.js/Express):** The following pseudo-code demonstrates how this API endpoint would be implemented on the server.
   ```
   // Server-side code for the API endpoint (e.g., in Express.js)
   // Assume 'authenticateNav' is middleware that verifies the user's
   token
   // and 'rulesEngine' is a service that executes the YAML rule
   definitions.

   app.post('/api/v1/rules/snap/calculate_standard_deduction',
   authenticateNav, async (req, res) => {
     const { household_size } = req.body;

     // Input validation
     if (!household_size |
   ```

| typeof household_size!== 'number' | | household_size < 1) { return res.status(400).json({ error: 'Invalid household_size parameter.' }); }

~~~
try { const ruleId = 'snap_standard_deduction_md_2024'; const parameters = { household_size:
household_size };
// The rulesEngine executes the logic defined in the YAML file const result = await
rulesEngine.execute(ruleId, parameters);
// Add metadata for traceability result.timestamp = new Date().toISOString(); result.executed_by
= req.user.id; // User ID from authentication token
return res.status(200).json(result);
} catch (error) { console.error(Error executing rule: ${error.message}); return
res.status(500).json({ error: 'Internal server error during rule execution.' }); } });
// Expected JSON Response for a household_size of 5: // { // "deduction_amount": 225, //
"rule_id": "snap_standard_deduction_md_2024", // "timestamp": "2025-09-15T16:30:00Z", //
"executed_by": "navigator_123" // } ```
~~~

Section 3: Adherence to Digital Service Standards

The toolkit must be designed for everyone, requiring an unwavering commitment to simplicity, trust, and universal access. This is achieved by adhering to established state and federal digital service standards.

3.1 Plain Language

Government communications must be clear and understandable to the public. The toolkit will transform complex policy language into simple, actionable instructions.

- **Principle:** Use common, everyday words, short sentences, and the active voice.
- **Implementation:** All user-facing text will be written at an 8th-grade reading level or below. Technical jargon will be replaced with simple explanations.
- **Example:**
    - **Original Policy Text (from Manual Section 211):** "Income that is received too infrequently or irregularly to be reasonably anticipated, not in excess of $30 in a quarter, is excluded."
    - **Toolkit UI Text:** "Did you receive any small, one-time payments this quarter (less than $30 total)? This could be from things like a small side job or a gift. If so, you don't need to count it as income."

3.2 Accessibility (WCAG 2.1 Level AA)

The toolkit must be fully accessible to people with disabilities, meeting the WCAG 2.1 Level AA standard, which is the technical standard for government digital services. This is organized around four core principles (POUR) :

- **Perceivable:** Information must be presentable to users in ways they can perceive.
    - **Implementation:** All images will have descriptive alt text. Videos will have captions and transcripts. Color contrast between text and background will meet a minimum 4.5:1 ratio.
- **Operable:** All interface components must be operable.
    - **Implementation:** The entire application will be navigable using only a keyboard. Users will have options to pause or extend time limits on sessions.
- **Understandable:** Information and the operation of the interface must be understandable.

    - **Implementation:** Navigation will be consistent across all pages. Form fields will have clear labels and instructions, and error messages will be specific and helpful.
- **Robust:** Content must be reliably interpreted by a wide variety of user agents, including assistive technologies.
    - **Implementation:** The application will be built with clean, semantic HTML to ensure screen readers can parse the content correctly.

3.3 U.S. Web Design System (USWDS)

To ensure a consistent, trustworthy, and familiar user experience, the toolkit's front-end will be built using the U.S. Web Design System (USWDS). USWDS provides a library of open-source UI components and a visual style guide that already meets federal accessibility and mobile-friendly requirements.

- **Principle:** Using USWDS saves time, ensures compliance, and builds public trust through a consistent look and feel across government services.
- **Code Implementation (HTML with USWDS classes):** The following code shows how a simple, accessible, and mobile-friendly button would be created using USWDS, rather than custom styling.
  ```
  <button class="usa-button" type="button">Submit
  Application</button>

  <button class="usa-button usa-button--secondary"
  type="button">Save and Continue Later</button>
  ```

3.4 Multilingual Strategy

To serve Maryland's diverse population, the toolkit must provide a culturally and functionally equivalent experience for non-English-speaking users.

- **Principle:** Follow the best practices outlined by Digital.gov, including ensuring language quality through human translation, providing prominent links to switch languages, and managing user expectations by labeling English-only content.
- **Implementation:**
    1. The toolkit will launch with full support for English and Spanish, the most common languages spoken by residents who may need these services.
    2. A prominent language selector will be present in the global header of every page.
    3. All content, including form labels, help text, and error messages, will be translated by qualified human translators to ensure cultural relevance and accuracy.
    4. User support channels (chat, phone) will have clear pathways to connect with interpreters or bilingual staff.

By building the Navigator's Digital Toolkit on this granular, standards-compliant foundation, the "No Wrong Door" initiative can ensure it is not just a powerful tool, but a trustworthy, accessible, and equitable one for all Marylanders.

Works cited

1. Supplemental Nutrition Assistance Program (SNAP) Manual ...,
https://dhs.maryland.gov/supplemental-nutrition-assistance-program/food-supplement-program-

manual/ 2. Eligibility Rules - Maryland Department of Human Services, https://dhs.maryland.gov/supplemental-nutrition-assistance-program/eligibility-rules/ 3. Plain Language Guidelines, Plan, and Report - Federal Maritime Commission, https://www.fmc.gov/about/plain-language-guidelines-plan-and-report/ 4. Plain language - Digital.gov, https://digital.gov/topics/plain-language 5. Plain Language Quick Reference Guide - U.S. Department of Labor, https://www.dol.gov/sites/dolgov/files/general/Plain-Language-Quick-Reference-Guide.pdf 6. Plain Language - OPM, https://www.opm.gov/information-management/plain-language/ 7. Plain Language - Governor's Office of Information Technology, https://oit.colorado.gov/standards-policies-guides/a-guide-to-accessible-web-services/plain-language 8. Maryland's digital revamp aims for an online presence that's cohesive, secure, accessible, https://statescoop.com/maryland-web-design-2025/ 9. WCAG 2.1 Level AA - NC DPI, https://www.dpi.nc.gov/about-dpi/technology-services/digital-accessibility/wcag-21-level-aa 10. The Web Content Accessibility Guidelines (WCAG) 2.1 Level AA - University of North Florida, https://www.unf.edu/adacompliance/title-ii/WCAG-2.1-Level-AA.html 11. Accessible Design Using the U.S. Web Design System (USWDS) | Section508.gov, https://www.section508.gov/develop/accessible-design-using-uswds/ 12. U.S. Web Design System - GovFresh, https://govfresh.com/research/us-web-design-system 13. U.S. Web Design System - Wikipedia, https://en.m.wikipedia.org/wiki/U.S._Web_Design_System 14. The U.S. Web Design System (USWDS) and onrr.gov, https://blog-nrrd.doi.gov/USWDS/ 15. uswds/uswds: The U.S. Web Design System helps the federal government build fast, accessible, mobile-friendly websites. - GitHub, https://github.com/uswds/uswds 16. U.S. Web Design System (USWDS) - YouTube, https://www.youtube.com/playlist?list=PLd9b-GuOJ3nGqDYCNsCMHCQ9MdD5jfB01 17. Apply for SNAP - Maryland Hunger Solutions, https://www.mdhungersolutions.org/federal-nutrition-programs/snap/apply-for-snap/