**DPG Document for Rice Fortification**

[1. Introduction 1](#_Toc489315294)

[Definition 1](#_Toc1302105692)

[Purpose 1](#_Toc2083307992)

[2. Background 1](#_Toc1011827818)

[Origin and Development 2](#_Toc1940521331)

[Vocabulary: 3](#_Toc1772065605)

[3. Description of the DPG 5](#_Toc1971771319)

[Functionality 5](#_Toc914977610)

[Detailed Steps for initial setup with DSL 12](#_Toc1650120305)

[Platform Overview 13](#_Toc476591210)

[Technological Considerations 13](#_Toc1986924790)

[Technical Architecture 14](#_Toc1259316296)

[Technology Stack 14](#_Toc1407600704)

[Setup Guide 18](#_Toc1153524932)

[1. Infrastructure Setup 19](#_Toc2137129040)

[2. Integration and Configuration 19](#_Toc1521768571)

[3. Security and Authentication 19](#_Toc1730437717)

[4. Event Handling and Messaging 19](#_Toc1142841786)

[5. Testing and Deployment 19](#_Toc431169715)

[4. Implementation Guide 19](#_Toc470735701)

[Setup Instructions 19](#_Toc259279434)

[Local Installation Guide: 20](#_Toc1901053334)

[5. Operational Documentation 24](#_Toc315473730)

[User Manuals 24](#_Toc1924804103)

[Administrator Guides 24](#_Toc1449702421)

[Maintenance 24](#_Toc509187199)

[6. Sample Implementation 25](#_Toc1731619133)

## **1. Introduction**

### **Definition**

Digital Public Goods (DPGs) are open-source software, open data, open AI models, open standards, and open content that adhere to privacy and other applicable laws and best practices and are designed to be shared and used across different regions and sectors to address critical issues and contribute to sustainable development.

### **Purpose**

This documentation provides a comprehensive guide to understanding, implementing, and contributing to Rice Fortification, a digital public good aimed at ensuring that the quality of fortified rice adheres to the policies and guidelines of a specific geography (country, region, or sub-unit).

**Scope**

This documentation covers the technical specifications, implementation steps, governance, and operational details of the Fortified Food Quality Management System (FFQMS). It is intended for developers, administrators, and end-users.

## **2. Background**

### **Origin and Development**

Fortified Food Quality Management System (FFQMS).

Malnutrition continues to pose a formidable challenge worldwide, particularly in regions grappling with poverty and inadequate access to nutritious food. In the quest to combat this pervasive issue, food fortification emerges as a beacon of hope, offering a practical solution to bolster public health. This article delves into the significance of food fortification, with a spotlight on its application in India, where malnutrition remains a pressing concern.

At its core, food fortification entails enriching staple food items with essential micronutrients to alleviate deficiencies and promote well-being. This evidence-informed intervention, endorsed by the World Health Organization (WHO), stands as a pivotal strategy in the global fight against malnutrition. From adding iodine to salt to fortifying milk with vitamin D, the process of food fortification manifests in various forms, each tailored to address specific nutrient gaps within populations.

In response to the pervasive malnutrition crisis, organizations have collaborated on innovative initiatives, such as rice fortification, to enhance staple food items with essential micronutrients. Rice, being a dietary staple in India, presents a promising canvas for these fortification efforts. By enriching rice with vital micronutrients like iron, zinc, folic acid, and vitamin A, these initiatives aim to improve the nutritional quality of diets without necessitating significant alterations in dietary habits.

The fortification process involves meticulous steps, from sourcing natural rice from mills to manufacturing fortified rice kernels containing the requisite micronutrients. Quality control measures, including rigorous testing in laboratories, ensure compliance with food standards before distribution to the populace. Rice is fortified with iron, folic acid (vitamin B6), and vitamin B12 to address anemia and micronutrient deficiencies.

The process of fortifying rice involves multiple stakeholders. To ensure that the quality of fortified rice aligns with the policies and guidelines of a specific geography (country, region, or sub-unit), a robust quality control and quality assurance infrastructure and processes are required. Establishing a fool-proof process is crucial to producing and supplying fortified rice of the desired quality to consumers, ultimately achieving the goal of curtailing chronic nutritional deficiencies.

In the context of India, the government has decided to scale up the distribution of fortified rice under the Integrated Child Development Scheme (ICDS), PM POSHAN Schemes, and the Public Distribution System (PDS). This expansion will be implemented in a phased manner, leveraging the existing domestic supply chain through the Food Corporation of India (FCI) and/or state agencies. The blending of rice with fortified rice kernels (FRK) will occur at the rice mills.

The rice fortification program currently suffers from a lack of quality due to weak oversight and enforcement mechanisms, primarily caused by poor incentives and a lack of trained personnel in the system. As a result, rice mills, FRK manufacturers, and premix suppliers are not held accountable for meeting government standards, and many may not be fortifying at all, thereby reducing the program's impact.

The Standard Operating Procedure (SOP) for quality assurance primarily covers the following three aspects:

1. **Quality Assurance of the processes in the supply chain**
2. **Quality Control of the inputs and the produce**, primarily premix, fortified rice kernels, and fortified rice
3. **Traceability of material movement within the critical stakeholders** in the supply chain, namely premix manufacturers, FRK manufacturers, and rice mills carrying out the blending of FRK with milled rice to produce fortified rice

It is proposed that a suitable end-to-end digital solution for Quality Assurance (QA), Quality Control (QC), and Traceability (Trace) be implemented across the manufacturing supply chain.

### **Vocabulary:**

**Batch**

* A production unit created from raw materials.
* Ensures consistency and uniformity in the final product.
* Identified by a unique batch number for tracking and quality control.

**Lot**

* A dispatch unit for transferring raw materials or processed products.
* Facilitates logistics and movement through different stages.
* Tracked with a unique identifier for quality assurance and traceability.
* Single lot will have units from a single batch

**Sample**

* A Sample taken for lab test from a batch or lot.
* Used for quality testing in a laboratory.
* Helps ensure the entire batch or lot meets required standards.

**Dispatch**

* The process of sending a lot or sample from one manufacturer/unit to another manufacturer/unit.
* Part of the transition process from one manufacturer/unit.
* Ensures materials move efficiently through different stages.

**Approval**

* The process of accepting a lot or batch at the receiving stage.
* Involves checking compliance with quality and specification requirements.
* Necessary before the material is accepted for further processing.

**Reject**

* Happens when a lot or batch does not meet required standards.
* The material is not accepted for further processing.
* Will be returned to the supplier, reworked, or discarded.

**Categories**

* Different stages or classifications within the production or supply chain.
* Examples include Premix, FRK, FR etc.
* Helps organize and manage the lifecycle of the product.

**States**

* Indicate the current progress of a batch or lot.
* Examples include "Created," "Approved," "Rejected," "SampleInLab," and "Dispatched."
* Tracking states helps monitor the workflow and ensures efficiency.

**Example Use Case:**

A batch is considered a production unit, while a lot is a dispatch unit. For instance, in Manufacturing Unit A, 100 MT of Fortified Rice Kernels (FRK) is produced daily. This 100 MT is tested in the lab and then packed into 20 bags of 50 kg each. This 100 MT constitutes a batch, for which the manufacturer has test results.

The 100 MT can be distributed as follows:

* 70 MT to Miller A
* 30 MT to Miller B

In this scenario:

* The 70 MT dispatched to Miller A is considered one lot.
* The 30 MT dispatched to Miller B is considered another lot.

Once Miller A receives the lot and acknowledges receipt, Miller A can either:

* Send a sample from the 70 MT to the lab for further testing, or
* Approve/reject/return the lot based on the batch test results attached.

Similarly, Miller B can:

* Send a sample from the 30 MT to the lab for testing, or
* Approve/reject/return the lot based on the batch test results attached.

## **3. Description of the DPG**

### **Functionality**

As we are developing rice fortification systems, we have identified the need for a similar QA/QC approach for other commodities across different countries.

To address this need, we have created a customizable and configurable digital public good (DPG) that can be applied to various fortification use cases across countries.

The DPG solution requires a predefined Domain-Specific Language (DSL) for initial configuration, which sets up the default parameters.

**DSL:**

1. The platform aims to track a product through various stages of processing and testing until it becomes a fortified product.
2. Each stage can include:
   1. A set of required input materials
   2. Target categories to which the processed product must be delivered
3. Batches for a category can be created from raw material lots and dispatched to the next category manufacturer or unit.
4. Lots can be created from approved batches.
5. Both lot and batch samples can be sent to the lab for testing.

Considering the above, the platform can be configured to manage the following:

* **Stages:** Define the stages a product goes through from raw materials to a fortified product.
* **Raw Materials and Target Categories:** Specify the raw materials required and the target categories for each stage.
* **Lab Testing:** Include options for lab testing during the dispatch and receipt of products

Sample DSL contains following fields for configuration

* “product” -> name of the product
* “platform\_name” -> display name of platform
* “description” -> description of platform
* “categories” -> list of categories present
  + “name” -> name of category
  + “outside\_platform” -> Boolean whether it is captured outside the platform or not
* “stages” -> list of stages present
* “workflow” -> details of workflow in each category stage
  + “name” -> name of stage
  + “categories” -> list of categories and their workflow
    - “name” -> category name
    - "type" -> it can be either creation/dispatch based upon the stage whether used for batch creation or storing lots.
    - “raw\_materials” -> categories which can be used as raw materials for this stage
    - "target" -> list of categories to which lots can be sent to
      * “name” -> category name of target
      * "receive\_lab\_option" -> whether the lot needs to sent to lab testing before receiving, it can be "OPTIONAL" or “MANDATE” or “NA”
      * "dispatch\_lab\_option": whether the batch needs to sent to lab testing before receiving, it can be "OPTIONAL" or “MANDATE” or “NA”

**Sample DSL Json Schema:**



Json file link (Path team update)

**Sample DSLs with use cases:**

**Use case example DSL for Rice Fortification implementation:**



Json file link (Path team update)

**Workflow**:

**Initial Stage: Premix Manufacturer**

* **Raw Material Lots:** A Premix Manufacturer creates raw material lots sourced from vendors.
* **Premix Batch Creation:** These raw material lots are combined to create a premix batch.
* **Lab Testing:** After creating the premix batch, a sample must be sent to the lab for testing, as specified in the DSL. The lab tests the sample based on various factors and the quantities of raw materials in the premix batch. The batch is either approved or rejected based on whether the test results fall within the required range.
* **Dispatch:** Once the premix batch is tested and approved, it can be dispatched to the manufacturers of the next stage.
* **Lot Dispatch:** Premix manufacturers can dispatch their approved batches as lots only to Fortified Rice Kernel (FRK) manufacturers.
* Here’s a refined version of your description for the second stage at the FRK Manufacturers:

**Second Stage: FRK Manufacturers**

* **Receiving Premix Lot:** The dispatched premix lot is received at the FRK manufacturer and can be either directly approved or sent to the lab for testing.
* **Creating FRK Batch:** Once the lot is approved, it can be used to create an FRK batch. This batch is made by mixing rice mix with the approved premix lot in the required proportions. Rice mix lots are created by the FRK manufacturer, similar to how raw material lots are created for the premix.
* **Lab Testing:** After creating the FRK batch, a sample is tested following the same procedure used for the premix sample test.
* **Dispatch:** Once the FRK batch passes the lab test and is approved, it can be dispatched as lots to the next stage.
* **Target Stages:** The FRK batch can be dispatched to one of two possible target stages:
  + **Miller**
  + **Warehouse**

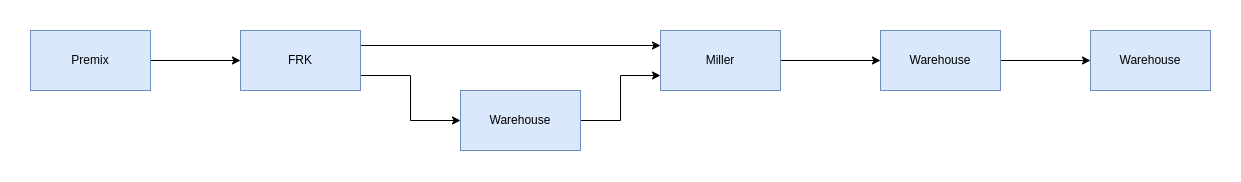
**When the FRK is Dispatched to Miller:**

* **Receiving the FRK Lot:** A miller can either accept the lot directly or send it to the lab for testing.
* **Creating Miller Batch:** If the FRK lot is accepted, the miller can use a calculated mixture of the FRK lot and rice to create a Miller batch. The rice is created by the miller in a manner similar to the creation of raw material lots for the premix.
* **Dispatching Miller Batch:** The miller can dispatch the Miller batch as lots to the next stage, even without testing.
* **Target Stage for Miller Lots:** Miller lots can only be sent to warehouses.
* **Miller Lots at Warehouse:**
* **Receiving Miller Lots:** A warehouse receives Miller lots, with an optional lab test of the lot sample.
* **Storing and Dispatching:** If the Miller lots are accepted at the warehouse, they are stored and can only be dispatched to other warehouses.

**When FRK Dispatches Lots to Warehouse:**

* **Receiving the FRK Lot:** Upon receipt of the FRK lot at the warehouse, it can be either accepted directly or sent to the lab for testing.
* **Storing and Dispatching:** After the FRK lot is accepted, it is stored at the warehouse. The accepted lot can then be:
  + Dispatched to a miller manufacturer for Miller batch creation, or
  + Dispatched to another warehouse.

**Data Flow Diagram:**



**Use case example DSL for premix to warehouse:**



Json file link (Path team update)

**Workflow:**

**Initial Stage: Premix Manufacturer**

1. **Raw Material Lots:** A Premix Manufacturer creates raw material lots from vendors.
2. **Premix Batch Creation:** These raw material lots are combined to create a premix batch.
3. **Lab Testing:** After the premix batch is created, a sample must be sent to the lab for testing as specified in the DSL. The test evaluates multiple factors/material quantities of the raw materials in the premix batch. The batch is approved or rejected based on whether the test results fall within the required range.
4. **Dispatch:** Once approved, the batch can be dispatched to the next stage. Premix manufacturers can dispatch their batches as lots to either FRK manufacturers or warehouses.

**When Premix is Dispatched to Warehouse**

1. **Receiving the Premix Lot:** Upon receipt at the warehouse, the premix lot can be accepted directly or sent to the lab for testing.
2. **Storing and Dispatching:** After acceptance, the premix lot is stored at the warehouse. It can then be dispatched to a FRK manufacturer for FRK batch creation or to another warehouse.

**At FRK Manufacturers**

1. **Receiving the Premix Lot:** The premix lot (received directly from the Premix Manufacturer or via a Warehouse) can be accepted directly or sent to the lab for testing.
2. **Creating FRK Batch:** Once the lot is approved, it is used to create an FRK batch by mixing with rice mix in required proportions. Rice mix lots are created by the FRK manufacturer in a manner similar to creating raw material lots for the premix.
3. **Lab Testing:** A sample from the FRK batch is tested following the same procedure used for premix testing.
4. **Dispatch:** After approval, the FRK batch can be dispatched as lots to the next stage. FRK can be dispatched to either:
   1. **Miller**
   2. **Warehouse**

**When FRK is Dispatched to Miller**

1. **Receiving the FRK Lot:** A miller can accept the FRK lot directly or send it to the lab for testing.
2. **Creating Miller Batch:** If accepted, the miller can use a calculated mixture of FRK lot and rice to create a Miller batch. Rice is created by the miller in a manner similar to the creation of raw material lots for the premix.
3. **Dispatching Miller Batch:** The Miller batch can be dispatched as lots to the next stage, even without additional testing.
4. **Target Stage for Miller Lots:** Miller lots can only be sent to warehouses.

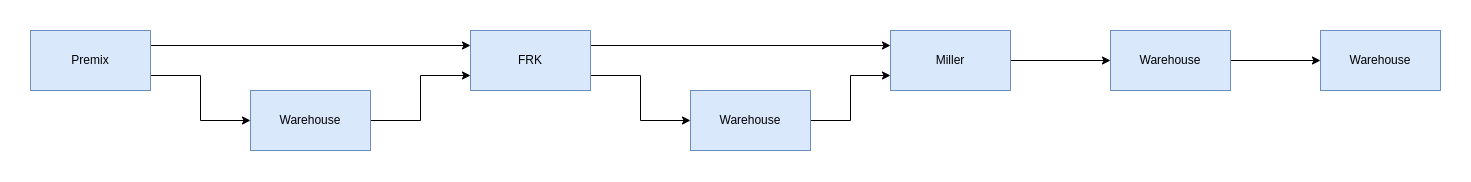
**Miller Lots at Warehouse**

1. **Receiving Miller Lots:** A warehouse receives Miller lots, with an optional lab test of the lot sample.
2. **Storing and Dispatching:** If accepted, the Miller lots are stored and can only be dispatched to other warehouses.

**When FRK Dispatches Lots to Warehouse**

1. **Receiving the FRK Lot:** Upon receipt at the warehouse, the FRK lot can be accepted directly or sent to the lab for testing.
2. **Storing and Dispatching:** After acceptance, the FRK lot is stored and can be dispatched to a miller manufacturer for Miller batch creation or to another warehouse.

**Data Flow Diagram:**



**Use case example DSL for flow without FRK:**



Json file link (Path team update)

**Workflow:**

**Initial Stage: Premix Manufacturer**

* **Raw Material Lots:** A Premix Manufacturer creates raw material lots from vendors.
* **Premix Batch Creation:** These raw material lots are combined to create a premix batch.
* **Lab Testing:** After the premix batch is created, a sample must be sent to the lab for testing as specified in the DSL. The test evaluates multiple factors and the quantities of raw materials in the premix batch. The batch is either approved or rejected based on whether the test results fall within the required range.
* **Dispatch:** Once the batch is tested and approved, it can be dispatched to the next stage. Premix can be dispatched to either:
  + **Miller**
  + **Warehouse**

**When Premix is Dispatched to Warehouse**

* **Receiving the Premix Lot:** At the warehouse, the premix lot can be accepted directly or sent to the lab for testing.
* **Storing and Dispatching:** After acceptance, the premix lot is stored and can be dispatched to:
  + **Miller** for Miller batch creation, or
  + **Another Warehouse**

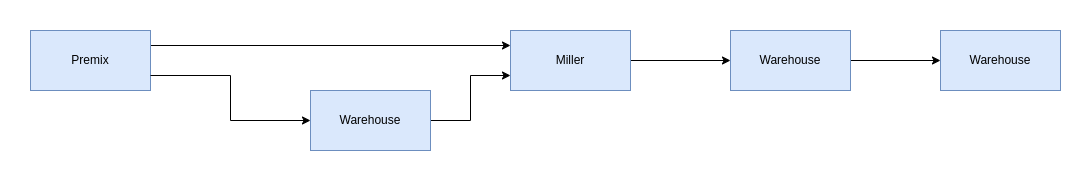
**At Miller**

* **Receiving Premix Lot:** The miller can accept the premix lot (received directly from the Premix Manufacturer or via a Warehouse) or send it to the lab for testing.
* **Creating Miller Batch:** If accepted, the miller uses a calculated mixture of the premix lot and rice to create a Miller batch. The rice is created by the miller similarly to the creation of raw material lots for the premix.
* **Dispatching Miller Batch:** The Miller batch can be dispatched as lots to the next stage, even without additional testing.
* **Target Stage for Miller Lots:** Miller lots can only be sent to warehouses.

**Miller Lots at Warehouse**

* **Receiving Miller Lots:** A warehouse receives Miller lots, with an optional lab test of the lot sample.
* **Storing and Dispatching:** If accepted, the Miller lots are stored and can only be dispatched to other warehouses

**Data Flow Diagram:**



### **Detailed Steps for initial setup with DSL**

1. Product is added with name and description
2. Categories are created from the list of categories
   1. Populate the "outside\_platform” field to indicate if the category is managed outside the fortification platform.
   2. Link each category to the product.
3. Stage and Workflow Entry Creation:
   1. For each stage in the stages list there exists an entry in workflow indicating its actions
4. Workflow Entries
   1. Categories present in raw material source and target are created if not existing
   2. “USER” and “ADMIN” roles are created given MODULE and LAB
   3. If the type is “creation”
      1. Actions are added for creating lots for raw materials with source category as raw materials and base category as category in workflow
      2. Actions are added for creating batches with raw materials and sending them to target categories
      3. Actions are added for sending batches to lab based on dispatch lab option
      4. “APPROVER” role is created for target categories which has actions to approve a batch before being received at the target category module
      5. Based on the dispatch lab option lab actions are also added.
   4. If the type is “dispatch”
      1. Actions are created for creating category lots from the base category to dispatch it to target categories manufacturers/units
      2. Base category module is identified with workflow stage name
      3. Based on dispatch lab option lab actions are added
5. Once the roles and role categories are populated, added roles are created in the keycloak UI
6. Assign the required roles to the users created to get the appropriate actions for categories

### **Platform Overview**

### **Technological Considerations**

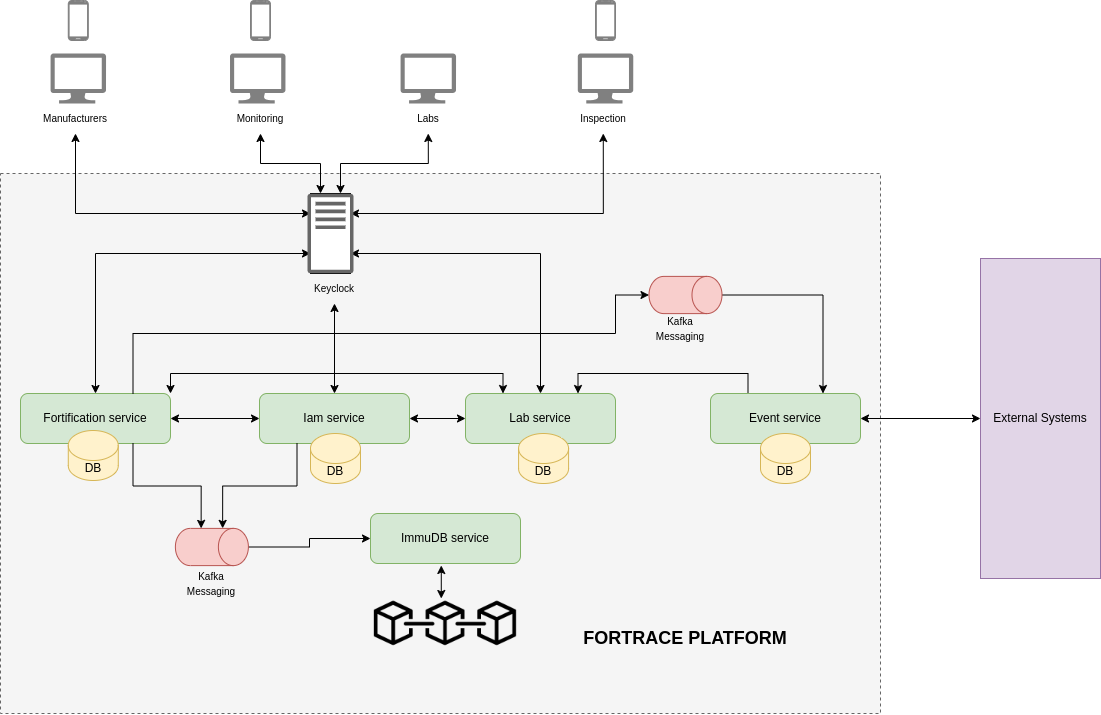
Some of the main considerations for building this software are as follows:

* Traceability — A single barcode stores the trace of all raw materials, their batch, and lot numbers along with quantities utilized for fortification. This ensures that any quality lapses can be traced back to the source.
* Immutability — To maintain traceability and QA/QC, it is crucial to secure and make the database immutable and encrypted. All tracking events go to a database built on top of blockchain principles, ensuring verifiability and immutability of data.
* Configurability — The platform can be configured across regions for workflows of other food fortification processes and to obfuscate manufacturing information in labs.
* Event Broadcast Framework — Since the platform is designed to work with different external systems, a framework allows interested clients to register for geographical state-level events and receive their login details for access.
* SSO - The platform integrates with Single Sign-On (SSO) through Keycloak to ensure seamless access management across different systems and applications. By utilizing SSO, users can securely authenticate once and gain access to all connected services, enhancing user experience while maintaining strong security protocols.
* DPG Compliance — The platform is built following DPG Alliance standards.

### **Technical Architecture**

* Utilizing a microservices architecture, the fortification solution offers flexibility in programming language selection to meet specific requirements. Java is utilized for implementing microservices, ensuring robust functionality. Authentication/Authorization services are managed through Keycloak, providing secure access for users across various roles in fortification and lab functionalities.
* In terms of data management, the application’s database relies on SQL, while ImmuDB is employed for auditing and traceability purposes. ImmuDB enables comprehensive traceability by storing data relationships between various lots and workflow details.
* Workflow management, including approvals, is handled using Spring-Activiti, a lightweight BPMN framework. This allows for easy configuration of multiple workflows tailored to specific roles and stages, ensuring efficient processing of tasks such as batch creation, sample request, dispatch, and receipt.

### **Technology Stack**



Above is a diagram showcasing the technology stack used to build the workflow software. Each tool contributed uniquely to the creation of the software.

* **Backend**: Utilizing Java for microservices presents various benefits compared to other technologies. Java boasts a strong ecosystem with abundant libraries, frameworks, and tools tailored for microservice development. Frameworks such as Spring Boot, drop wizard, and Micronaut offer comprehensive support, including dependency injection, RESTful APIs, and service discovery. Combined with technologies like Kubernetes and Docker, Java ensures scalability for growing workloads. Additionally, Java’s attributes such as high performance, compatibility, strong community support, integration capabilities, enterprise adoption, and security make it a compelling choice.
* **Authentication Network**: Key Cloak provides a robust, secure, and customizable authentication solution for managing user identities and ensuring secure application access. With out-of-the-box support for OpenID Connect, OAuth 2.0, and SAML 2.0, it offers comprehensive security features. Its intuitive admin console simplifies user, role, and permission management. Key Cloak’s extensive features, scalability, and seamless integration make it a preferred choice for application adoption.
* **Frontend**: React.js, a frontend JavaScript library, offers significant advantages over other technologies for UI development. Its component-based architecture promotes code reusability, maintainability, and scalability. Utilizing Virtual DOM enhances rendering performance, while JSX syntax simplifies component creation and readability. With features like unidirectional data flow, a rich ecosystem, cross-platform capabilities, performance optimization, and strong community support, React.js enables the creation of modern and dynamic user interfaces efficiently.
* **Database**: MySQL and ImdmuDB serve distinct data storage needs, each with unique advantages. MySQL, as a mature and reliable relational database management system (RDBMS), offers scalability, a rich feature set, and robust security features. In contrast, ImmuDB provides tamper-evident data storage by design, ensuring immutability and cryptographic proof for every transaction, making data alteration or deletion detectable. ImmuDB is preferred for applications prioritizing data immutability, auditability, and integrity.
* **Event-Driven Architecture (Kafka):** Implementing Kafka for event listening.

**External Integrations**

**Subscription Service for external platforms**

Integration of Existing Portals with Central QA/QC Platform

The fortification platform is planned for implementation across multiple states in India, some of which already have independent working portals. To ensure seamless integration of these existing portals with the central QA/QC platform, we need to address the challenge of integrating multiple systems, which involves customizing APIs and developing new adapters for data formatting, consumption, and publishing.

**Proposed Solution: Webhook Integration**

To facilitate this integration, we offer a solution where stakeholders can:

1. **Register Webhook URLs**: Stakeholders can register their webhook URLs on the central QA/QC platform.
2. **Subscribe to State Events**: Stakeholders can subscribe to events from specific states of interest.
3. **Receive Event Notifications:** Whenever an event is triggered in the central platform, the registered webhook URLs will be called.
4. **Handle POST Data:** Stakeholders will receive POST data containing event information, which they can then save, format, and process according to their needs.

**Possible events:**

These are the possible events from the platform.

1. BatchCreated
2. BatchUpdated
3. BatchSampleSentToLab
4. BatchSampleRejected
5. BatchTestPassed
6. BatchTestFailed
7. BatchRejected
8. BatchApproved
9. LotCreated
10. LotSampleSentToLab
11. LotSampleRejected
12. LotTestPassed
13. LotTestFailed
14. LotRejected
15. LotApproved
16. LotDispatched
17. LotReceived
18. LotSentBack
19. LotRejectedReceived

**Use case Example:**

Integration can occur whether the QA/QC platform is central or if the state portal is central. Here’s an example of how integration works when the QA/QC platform is central and a state portal (e.g., Andhra Pradesh) is central for specific functions:

**Dispatch Event Trigger:**

* When an FRK lot is dispatched from an FRK manufacturer to the FRK Godown, an event is triggered on the QA/QC platform.
* The state portal, having registered with the QA/QC platform, receives this event notification and saves the lot information into its system.

**Lot Acknowledgment:**

* Once the lot arrives at the FRK Godown, the state portal acknowledges receipt of the lot.
* This acknowledgment is pushed to the QA/QC platform using Swagger APIs in the background from the state portal.

**Sample Testing:**

* Samples from the lot can be sent to the lab via the state portal.
* The sample-related data is pushed to the QA/QC platform through Swagger APIs.

**Lab Test Results:**

* After lab testing is completed on the QA/QC platform’s Lab portal, LotTestPassed or LotTestFailed events are triggered based on the test results.
* These events are sent to the state portal through the registered webhook.
* The state portal downloads and saves the lab results from the webhook data and displays the results in its UI.

**Lot Approval or Rejection:**

* Based on the lab results, the lot can be approved or rejected from the state portal.
* The approval or rejection status is then pushed to the QA/QC platform in the background using Swagger APIs.

**Prerequisites for integration:**

The manufacturer data should be unique across all platforms across India. Any new platform/state trying to integrate with QA/QC platform should have the manufacturers synced. Currently fssai license number is considered as unique constraint to avoid duplicates if there are Mutiple platforms are integrated for a single geographical state.

**How Authentication works:**

Upon a client's initial registration within the QA/QC platform, A client id and client secret specific to the client is generated within our authorization server. Clients are expected to store the provided credentials in a secured store and use the credentials for getting access tokens.

**Pushing data to QA/QC platform:**

To submit data to the QA/QC platform, the registered subscriber must first authenticate by invoking the login API with their client ID and client secret obtained during initial registration. This will generate an access token, which is necessary for all subsequent interactions with the system. The platform adheres to the OpenAPI specification and offers Swagger endpoints, allowing the subscriber to effectively use the available APIs for data submission. By referencing the OpenAPI documentation, the subscriber can construct the appropriate request payload to successfully push event data into the QA/QC platform.

### **Setup Guide**

#### **1. Infrastructure Setup**

* **Deploy Microservices:** Deploying microservices on designated hosting environments.
* **MySQL Database:** Setting up MySQL for data management.

#### **2. Integration and Configuration**

* **Spring Boot and MySQL Integration:** Configuring Spring Boot to interact with MySQL.
* **Kafka Integration:** Integrating Kafka with microservices for event handling.
* **Key Cloak Integration:** Setting up Key cloak for authentication and authorization.
* **Immudb Integration:** Configuring microservices to use Immudb for notifications.

#### **3. Security and Authentication**

* **Secure Communication:** Implementing HTTPS for secure data transfer.
* **Token-based Authentication:** Using JWT tokens issued by Keycloak for authentication.

#### **4. Event Handling and Messaging**

* **Event-Driven Communication:** Designing Kafka topics for event-driven architecture.
* **Error Handling:** Implementing error handling and retry mechanisms for Kafka consumers.
* **Monitoring and Scaling:** Monitoring Kafka message processing and scaling consumers as needed.

#### **5. Testing and Deployment**

* **Unit Testing:** Writing unit tests using JUnit and Mockito.

## **4. Implementation Guide**

### **Setup Instructions**

**Pre-Requisites**

* Java version 17
* Mysql db version 8
* Keycloak
* ImmuDB
* Kafka

### **Local Installation Guide:**

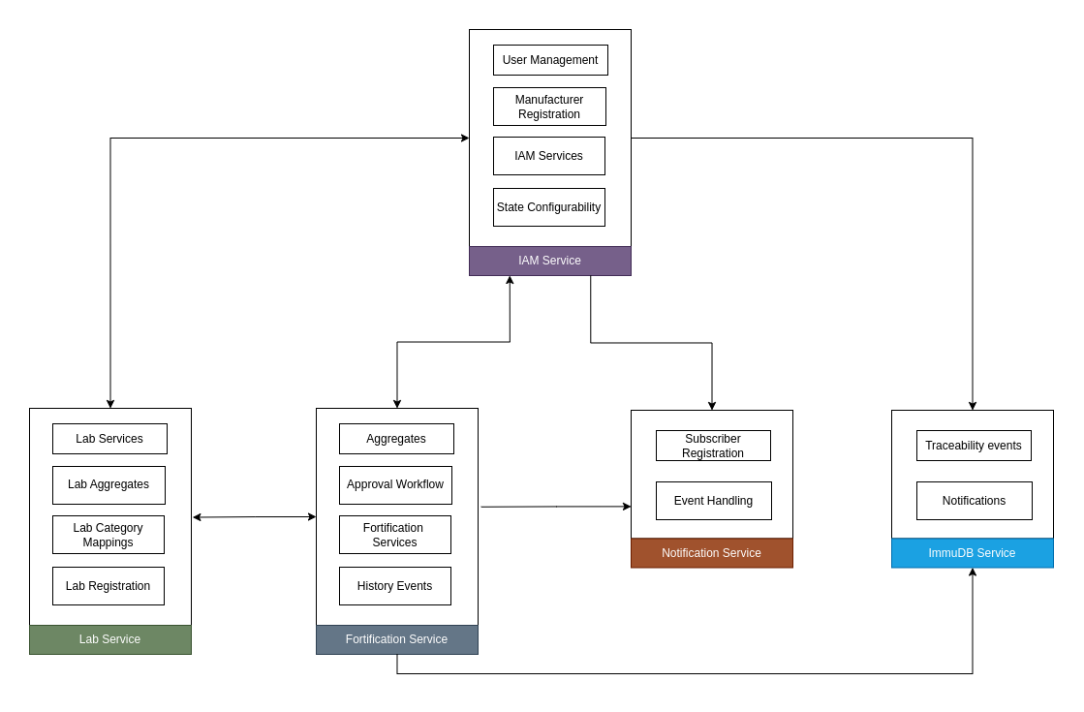
Please refer to the below Readme files running individual services locally

1. [https://github.com/Rice-Fortification-DPG/Rice-Fortification- DPG/blob/main/fortification\_service/README.md](https://github.com/Rice-Fortification-DPG/Rice-Fortification-2. DPG/blob/main/fortification_service/README.md)
2. <https://github.com/Rice-Fortification-DPG/Rice-Fortification-DPG/blob/main/iam_service/README.md>
3. <https://github.com/Rice-Fortification-DPG/Rice-Fortification-DPG/blob/main/immudb/README.md>
4. <https://github.com/Rice-Fortification-DPG/Rice-Fortification-DPG/blob/main/lab_service/README.md>
5. https://github.com/Rice-Fortification-DPG/Rice-Fortification-DPG/blob/main/broadcast\_service/README.md

Please refer to the below file guide for Docker compose single installation

1. https://github.com/Rice-Fortification-DPG/Rice-Fortification-DPG/tree/main/ README.md

**Services**



1. **Fortification service**
   1. **Aggregates**:
      * Responsible for calculating aggregates of batches and lots daily for a manufacturer/unit.
      * An event will be triggered for every action by a user and the quantity for the respective stage will be sent to Kafka and consumed by the service and added to the aggregates.
   2. **Approval workflow**:
      * Access to the stages defined in DSL are controlled through Role Based Access Control (RABC).
      * Users with the required role for a stage can approve the flow and move to the next stage.
   3. **History Events:**
      * For any status change happening in the application due to user performing an action, an event will be triggered for the history of the stage and sent to Kafka.
   4. **Fortification Services**:
   * **Raw Materials Mapped to Given Stage and Inventory**
     1. Tracking and recording the assignment of raw materials to specific stages in the fortification process.
     2. Maintaining an up-to-date inventory of all raw materials.
   * **Creation and Dispatching the Lots for Raw Materials**
     1. Generating and managing lots of raw materials.
     2. Ensuring proper dispatching protocols are followed for each lot.
   * **Creation of Vendors for Fetching Raw Materials from External Sources**
     1. Identifying and creating profiles for vendors.
     2. Ensuring sourcing of raw materials from external vendors is properly documented.
   * **Creation of Batch Using Raw Materials**
     1. Formulating batches utilizing the raw materials.
     2. Maintaining records for each batch created.
   * **Dispatching the Batch to Next Stage or Lab**
     1. Organizing and managing the dispatch of batches to subsequent stages or laboratories for testing.
     2. Ensuring each dispatch is logged and tracked.
   * **Maintaining Flow of Materials Through the Process Flow**
     1. Monitoring and documenting the progression of materials through each stage of the fortification process.
     2. Ensuring smooth and continuous flow of materials from start to finish.
2. **Lab Service** 
   1. **Lab Services**:
      * **List of Batches Sent for Lab Testing**
        + Keeping a record of all batches dispatched for laboratory testing.
      * **Receiving a Product for Testing in Any Stage**
        + Accepting and logging products received for testing at various stages.
      * **Submitting Test Results or Rejecting**
        + Documenting and submitting test results.
        + Recording and processing any rejections based on test outcomes.
   2. **Lab Aggregates:**
      * Count and quantity aggregates for lab samples will be calculated daily for Labs and accumulated through the aggregate tables.
   3. **Lab Registrations:**
      * Responsible for Onboarding of labs by capturing necessary information along with geo details.
   4. **Lab Category Mappings:**
      * Category Mappings of labs are captured in this service, which will be used for testing the batch and lot samples assigned to specific category.
3. **IAM Service**
   1. **User Management:**
      * Responsible for maintaining user information which includes creating user, updating user, resetting user password etc.
      * User role management through Keycloak is maintained in this service.
   2. **Manufacturer Registration**:
      * Responsible for Onboarding and updating of Manufacturers by capturing necessary information along with geo details.
      * Onboarding multiple new manufacturers and updating existing manufacturers through bulk excel upload is done through this service
   3. **Manufacturer Category Mappings:**
      * Category Mappings of manufacturers/units are captured in this service, which will be used for determining roles and stages for manufacturers/units.
   4. **IAM Services:**
      * Responsible for login and accessing APIs.
      * Generates manufacturer and user lists for agency count aggregates in monitor module.
   5. **State Configuration:**

* State level configuration for different features.
* Can edit feature requirements state wise.

1. **Broadcast Service**
   1. **Event Handling**
      * This service contains workflow to connect external service for data exchange between platforms
      * For any approval in the system the subscribers registered to the state of the manufacturer/unit in which the event occurred will receive data related to the said event in encrypted format
   2. **Subscriber Registration:**
      * Responsible for onboarding stakeholders as subscribers to receive callbacks for any events happening in the registered state.
      * Admin onboards subscribers to the system by procuring basic details of the subscriber along with a callback URL for publishing encrypted data.
2. **Immudb Service**
3. **Traceability events:**
   * Stores history of batches and lots. Has methods for accessing history for batch history trees.
   * Responsible for maintaining traceability of the entire application.
4. **Notifications:**
   * Responsible for tracking notifications for users on every action performed.

**Installation**:

* Clone the above services and build with maven
* Run the services on appropriate instances
* {BASE\_URL}/ v3/api-docs hit the given API to get the API’s list on the service

## **5. Operational Documentation**

### **User Manuals**

### **Administrator Guides**

* Once all the services are up and running run the dsl Api in fortification service to setup the application
  + **POST** {FORTIFICATION\_URL}/admin/dsl with JSON generated in the above step
    1. Creates Categories and Actions for Category-Specific Roles
    2. Defines and establishes categories and their respective actions.
    3. Creates USER, APPROVER, ADMIN Roles in key cloak with Appropriate Category Actions
    4. Sets up roles with permissions tailored to the actions relevant to each role.
* Create laboratories and assign them to related categories

### **Maintenance**

* Once the DSL is executed, the platform is configured for the stages with the defined roles and actions.
* Additional configurations can be made to the existing APIs by adding the given service as a dependency and modifying the APIs as required.

**Licensing**

MIT License  
  
Copyright (c) 2024 Rice-Fortification-DPG  
  
Permission is hereby granted, free of charge, to any person obtaining a copy  
of this software and associated documentation files (the "Software"), to deal  
in the Software without restriction, including without limitation the rights  
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  
copies of the Software, and to permit persons to whom the Software is  
furnished to do so, subject to the following conditions:  
  
The above copyright notice and this permission notice shall be included in all  
copies or substantial portions of the Software.  
  
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE  
SOFTWARE.

## **6. Sample Implementation**

* We can use the existing repositories to configure our implementation of DPG
* If we want to override or add any new functionalities, we can make the changes in existing repositories or we can write wrapper applications on top of these repositories to add new functionalities
* Using these services sample UI implementation is added to visualize these functionalities
* Once such implementation example is added here (https://blog.beehyv.com/digitizing-rice-fortification-process-in-india-beehyv-and-path-collaboration-05a24c2e1927)
* Fortified Food Quality Management System (FFQMS) is used for Rice as a product
* Rice fortification stands as a crucial measure in combating anemia and addressing micronutrient deficiencies by infusing iron, folic acid (Vitamin B6), and Vitamin B12 into its composition. This intricate process involves coordination and workflows between various stakeholders. Ensuring that fortified rice adheres to the policies and guidelines of a specific geographic region necessitates the establishment of a robust infrastructure for quality control and assurance. The paramount objective lies in instituting an airtight process to guarantee the production and distribution of fortified rice of desired quality to consumers, thus advancing the mission to alleviate chronic nutritional deficiencies.
* Rice Fortification (FoRTrace) platform contains below categories for the product
* Premix manufacturer (**Premix**) — Prepares the premix of micronutrients (Iron, Folic acid, and Vitamin B12)
  + The premix manufacturer employs the FoRTrace software for key tasks including entering raw material details, batch creation, sending samples to the lab, and dispatching batches to the FRK manufacturer. After inputting raw material information, batches are generated and samples are sent for analysis, with lab reports automatically updating batch statuses. Upon receiving batches, the FRK manufacturer reviews lab reports and either accepts or rejects the material, with rejected batches returned to the premix manufacturer.
* Testing lab — Evaluates rice lot samples for quality and gives approval for lot dispatch.
  + Laboratory users can log in to the portal and access a comprehensive list of samples dispatched from vendors. Upon receipt of these samples, acknowledgements can be promptly issued to the respective vendors. Subsequently, the laboratory user conducts tests based on the sample’s condition and uploads the resulting data. In instances where the sample condition is deemed unsatisfactory, the laboratory user can reject the sample, accompanied by pertinent comments and the date of rejection.
* FRK manufacturer — Procures rice-mix (broken rice powder) from vendor and blends it with micronutrient premix to create fortified rice kernels (**FRK**).
  + The FRK manufacturer procures premix powder and rice flour from external suppliers to craft fortified rice kernels (FRK). Upon receipt of these ingredients, a sample FRK batch undergoes rigorous testing in a laboratory. Upon approval, the batch is dispatched to the rice miller. At the FRK vendor level, two distinct roles, Approver and User, are established, each endowed with administrative privileges. The Approver oversees the reception of premix lots, conducts quality verification, initiates lab tests if necessary, and renders decisions on lot approval. Conversely, the User is tasked with creating rice flour and FRK batches, dispatching approved batches to the miller, and coordinating lab assessments. Upon dispatch from the premix vendor, lots are registered and subject to rigorous scrutiny by the Approver. Approved lots serve as vital inputs in FRK batch creation, where raw materials are automatically selected, initiating batch generation. Following batch creation, samples are forwarded to the lab for testing. Positive lab results prompt batch dispatch to the mill, while unsatisfactory results lead to rejection. Upon receipt, the miller may opt for additional testing, ensuring the highest quality standards are met.
* Rice mill (**Miller**) — Receives FRK from warehouse and blends FRK with natural rice to create fortified rice.
  + The miller’s pivotal role in the rice fortification supply chain involves testing and verifying raw materials from the FRK manufacturer before incorporating them into fortified rice production. Miller manufacturers, utilizing FoRTrace, designate two roles: Approver and User, responsible for quality assurance and batch creation, respectively. Integration of raw materials into the organizational portal ensures seamless batch creation, with approved lots used as input materials. After creation, batches undergo lab testing, with satisfactory results leading to automatic approval for dispatch, while unsatisfactory findings prompt rejection.
* **Warehouse** manager — Receives FRK lot from FRK manufacturer, sends FRK lot to rice mill. Receives fortified rice lot from rice mill, dispatches fortified rice lot to FCI warehouse.
  + The warehouse is the fourth entity in the rice fortification supply chain. Upon receiving lots from both the miller and FRK vendors, the warehouse has the authority to either directly accept or reject the lots. Rejected lots are promptly returned to the miller or FRK vendor. Alternatively, the warehouse may opt to send a sample from the lot to a local lab for self-certification. If the lab report indicates that the results fall within acceptable parameters, the lots are automatically accepted. However, if the lab report reveals deviations from the prescribed standards, the lots are rejected.