Digital Platform for Organic Farming Certification

Introduction

An agritech company have engaged us to build a technology platform that can help with digitization of process for organic farms certification. Such certifications are growing in importance as the customers of organic products need assurance that the farm products, they are buying, is actually organic and not fraudulent. Customers would also like to use this certification to trace organic produce to the farm, from anywhere in the supply chain. On the other hand, farmers want to use this certification to export their produce as well as sell at a premium in domestic market. Farmers also look at these certifications as a guidance for organic transformation of their farm, which increases the yield long term.

The organic certification process is carried out by certification bodies affiliated to the apex organic produce organization, NPOP (National Program for Organic Production). The entire process of organic certification involves a lot of record keeping and process steps:

- Receipt of application by any accredited organic certification body from farmer(s)
- Provisioning of standard and operational documents to farmer(s) by the certification body
- Agreement of roles and commitments between the farmer(s) and the body
- Demand of fees by accredited body
- Document Audit
- Regular field inspection by internal quality system manager and external inspector and documentation of the same
- Compliance verification through inspection and audits
- Preparation of reports by the field inspector
- Review of reports by a reviewing body
- Decisions on certification

The field inspection step is very critical in the organic certification process and breaks into:

- Visit of external inspector to fields and facilities
- Review of records and accounts
- Calculation of input and output norms and preparation of production estimation from a farm
- Assessment of production system
- Interview with responsible person(s)
- Risk assessment from neighboring farms
- Inspection of use of any genetically modified products
- Inspection of use of off-farm inputs
- Analysis of residue tests by certified laboratories for pesticides, heavy metals if required
- Inspection of sustainable practices
- Inspection and study of entire production unit

The technology platform that we have built is targeted at some 30 certification bodies, responsible for carrying out organic farm certification in India. For these certification bodies the current process is highly time consuming, cumbersome and error prone. Our objective was to use technology to streamline and simplify the entire process workflow for these certification bodies. An additional objective was to leverage this platform to generate a lot of farmer specific data, insights and market analytics that will help farmers improve their yield.
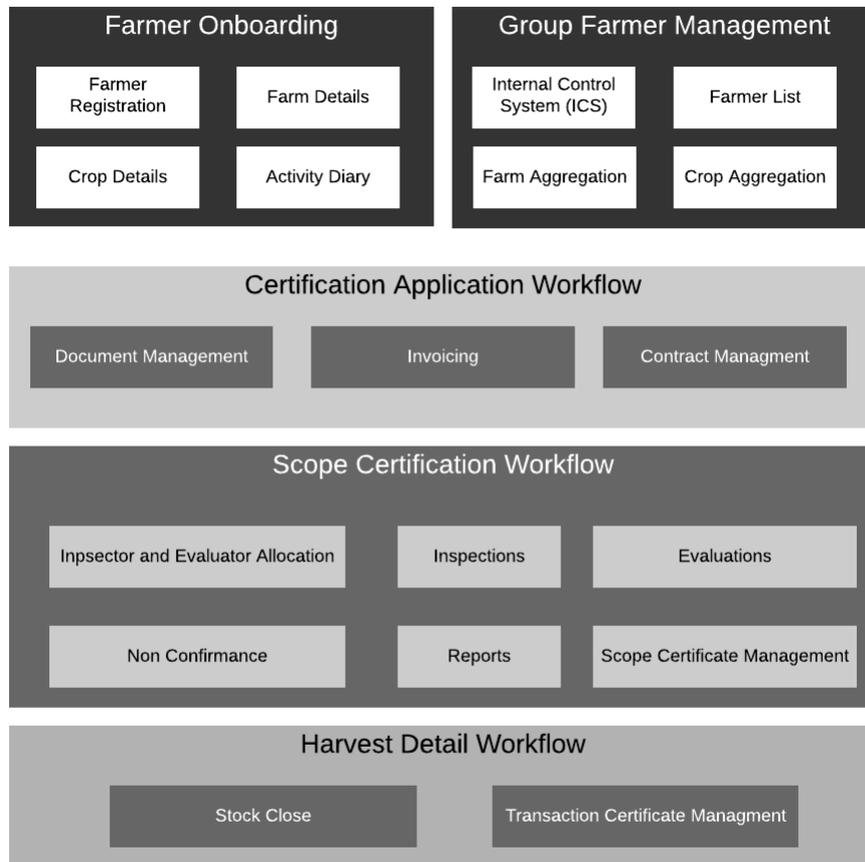
The Need

Originally, our client was looking for an off-shelf platform which, can support complex workflows, can be quickly implemented, is easily customizable, can be extended to different (future) use cases, can scale to approximately 800 M+ users, and has high throughput. In addition to the above there were more specific functional requirements from the platform:

- A user management hierarchy which can support diverse organization structures of different certification bodies
- Ability to support multiple intricate workflows which are sequentially connected with each other
- Ability to have a unified workflow for certification application for – individual farmer as well as farmer group
- Ability to capture performance data and identify bottlenecks in various activities of a workflow
- Ability to support cross device synchronization for users for all the workflows
- Ability to capture productivity statistics for different department of certification bodies
- Ability to capture inventory statistics for individual farmers, farmer groups and different districts

Solution Approach

For addressing the above need, we leveraged our in-house development platform that naturally lends itself to complex solution designing, scalable application building and rapid cycles of development. The platform at the core level is built on event sourcing and CQRS design patterns. Event sourcing means capturing all system state changes, as events, from which the database is created. So, events are principal source of truth and database can be built, changed and rebuilt from events at any point of time. This enables the platform to be highly change absorbent and dynamic. CQRS, on the other hand, ensures that read concerns and write concerns are separately woven into different code bases and databases. Applications usually has 96% read operations but the 4% write operations are more critical for the business. Separating read and write ensures that searching and queries are optimized by appropriate indexing and that it never delays the writes due to index rebuilding and table locking. In effect, resolving the scalability challenges of the application. The platform also uses an actor-based model which helps in concurrent processing and efficient resource utilization that are essential for cleaner implementation for complex business flows.

For the purpose of this solution, we have mostly adopted the **"creation and detour workflow patterns"** for designing the workflows. The workflows are actually created using the combination of – entities, state change requests, validation logic, events generation, and event processing. All the entities have a well-defined set of states and also well-defined set of requests for state change. Each time a state change request is submitted, it is subjected to a validation logic and on successfully passing will create an appropriate event. The event processor reads the events and make state changes to the entities along with any other post processing. The entity states and resource flow are defined at design time. All the detour logics are embedded into the state change requests and generates appropriate events at run time. The details of the various workflows are depicted in the functional and technical architecture below.

Farmer Onboarding

Farmer Registration	Farm Details
Crop Details	Activity Diary

Group Farmer Management

Internal Control System (ICS)	Farmer List
Farm Aggregation	Crop Aggregation

Certification Application Workflow

Document Management	Invoicing	Contract Managment

Scope Certification Workflow

Inpsector and Evaluator Allocation	Inspections	Evaluations
Non Confirmance	Reports	Scope Certificate Management

Harvest Detail Workflow

Stock Close	Transaction Certificate Managment

Functional Architecture

To solve the challenge of user management, a separate notion of application department and organization department was created. All the roles and permissions were mapped to the application departments. Based on the roles and permissions, the user can access the various workflows. The actual users were however added to the organization departments. The platform allowed for mapping of the application department to organization departments. Therefore, by default all the users of an organization department inherited the roles and permission of the application department to which it is mapped. In addition, the direct mapping of user to roles and permission was also allowed. So, using the above combination all the scenarios of user management requirement from different organizations could be addressed.
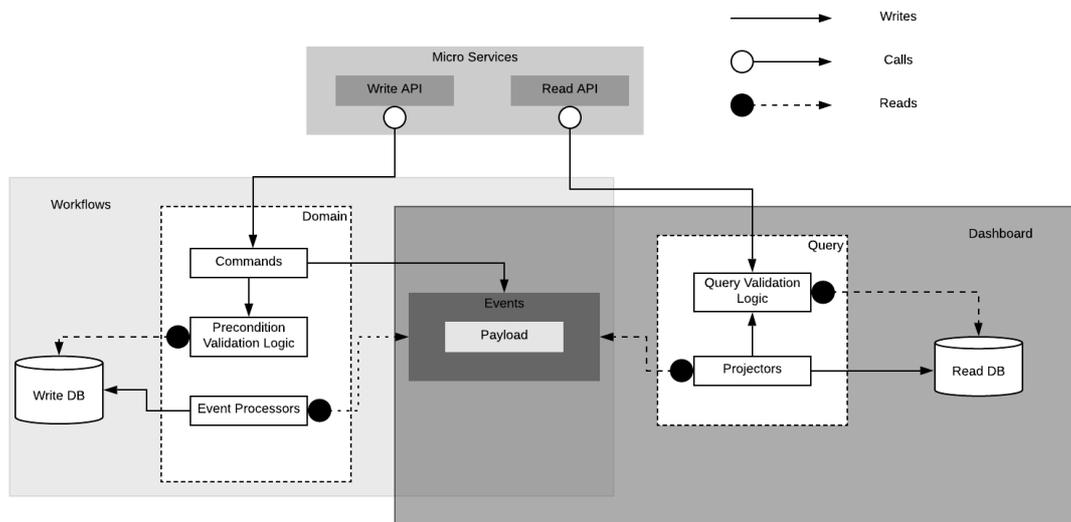
Technical Architecture

The platform also provisions for a projector logic, created around the events, to process the event payload and project the same into the database. We leveraged this capability for the purpose of building statistics and dashboard. Key business events, that contributes to statistics in question, were identified. Such events were broadly of two kinds – activity events and data events. The projector logic around activity events focused on to calculate the productivity metrics and build corresponding dashboard. The data events, on the other hand, were used for building the inventory metrics and dashboard. Multiple projection logic is allowed around the same event. Hence using the same events different types of dashboards could be created, providing a lot of flexibility to analytics and reporting. Also, since the projector logic is part of the code base, the metrics and dashboards were getting built run time along with the events and are actionable and could evolve with usage.

Furthermore, the platform also facilitates system designing by starting with micro events that are of interest to business. The APIs are designed to drop the micro events thereby driving a micro services design pattern and architecture. Microservice architecture is extremely useful in cross device and cross client synchronization and also in creating lighter presentation tier that are bereft of any business logic.

All of the above-mentioned capabilities are built using open source technologies – Scala, mongo, play framework and akka library.

Conclusion

In addition to a steep platform performance, our client's expectation was also to add functionalities to the platform in an evolutionary manner, to attain a balance between speed and functionality. But a greater challenge was to add features to the platform in conjunction with onboarding customers. While developing the platform, we were successfully able to absorb multiple change requests and product pivots, based on end user feedback.

Although, we did not have a workflow product to begin with, we were able to adapt our development platform to quickly build workflow-based application within the stipulated go-to-market timelines. Designing and building this solution was a phenomenal learning experience for us in terms of understanding how to approach complex workflow solutions and modelling for different workflow design patterns. It also helped us gain deeper insights into how to approach building business dashboards, metrics and actionable analytics.

All of this was achievable because of the domain driven design approach enforced by the platform, for application building. This approach emphasizes a lot on the domain (sphere of knowledge and activity around which application logic revolves) and domain logic (the business logic of the application). It also pushes for building complex application design by basing it on models (system of abstractions that describes selected aspects) of domain. Additionally, it encourages constant collaboration between technical and domain experts to iteratively refine a conceptual model that addresses particular domain problem. As a consequence, related pieces of software are connected, with ease, into an ever-evolving model of business concepts.