Nabis Compliance Service

Josh Kotrous - Engineering Manager Interview

Compliance Service

Nabis is building a new compliance service to modularize the existing code base. Each US state can have their own regulatory and compliance laws which necessitate runtime injection of logic to map Nabis's data schema to the expectations and schemas of each marketspace. Currently compliance is built as a part of our monolith application and will need to be supported until it can be transisted over to this new service.

Assumptions

Existing Product DB Schema

Assumptions

State compliance configuration schema

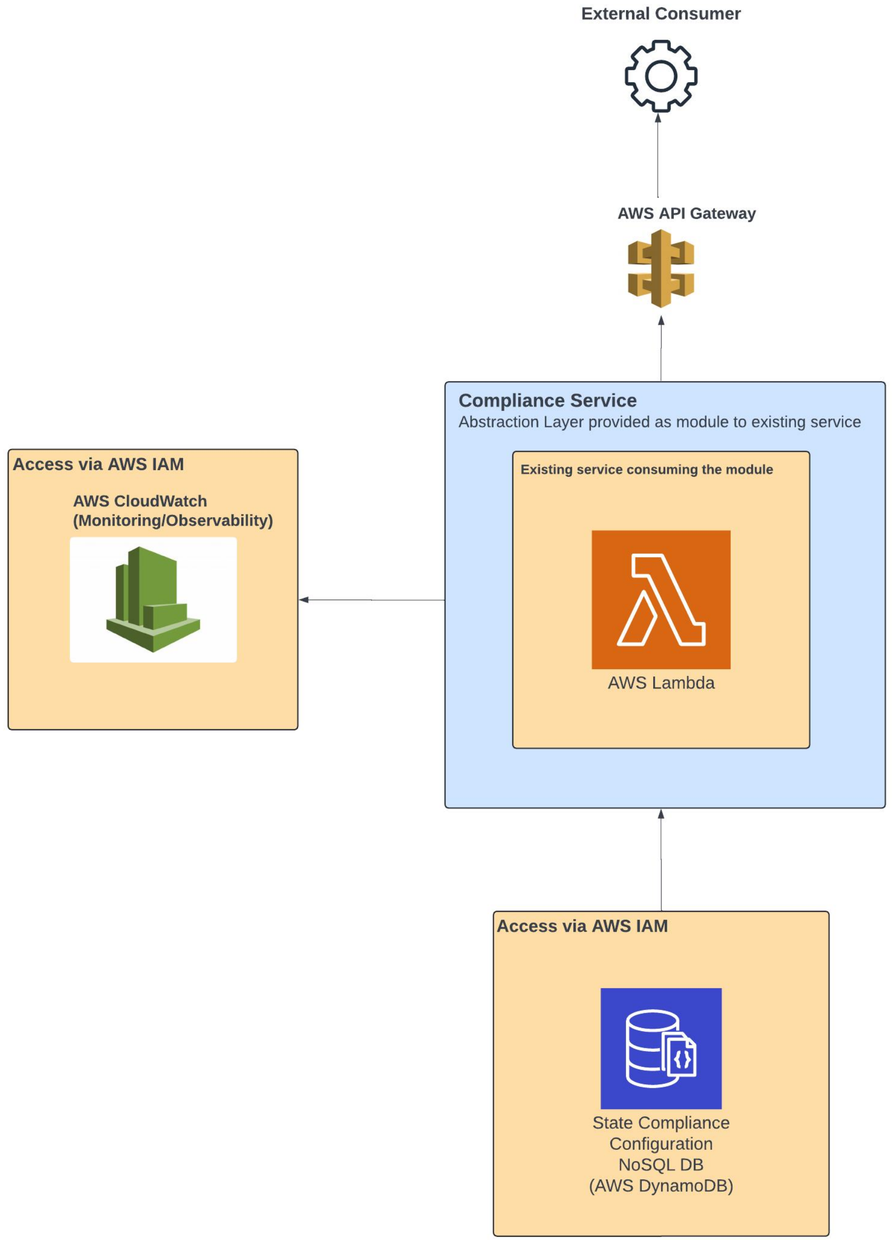
```
{
    "state_code": "",
    "config": {
        "stateSpecificFields":{
            "warningLabel": "",
            "governmentWarning": "",
            ...
        },
        "requiredProductInfo": [
          "productName",
          "netWeight",
          "thcContent",
          "cbdContent",
          "batchNumber",
          "manufacturingDate",
          ...
        ],
        "fieldMapping": {
            "thcContent":"thcPercentage",
            "cbdContent":"cbdPercentage",
            ...
        }
    }
}
```

Assumptions

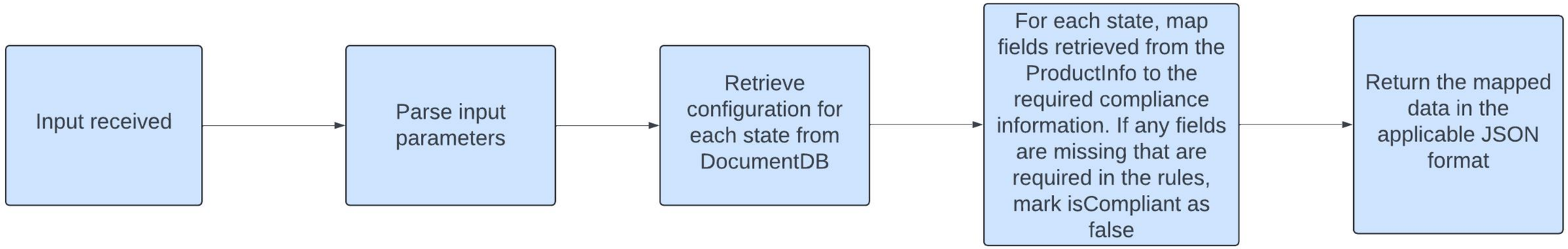
Provided product information schema

```
{
    "productId": "",
    "productName": "",
    "thcContent": "",
    "cbdContent": "",
    "netWeight": "",
    "manufacturer": "",
    ...
}
```

L0 - System


External Consumer
AWS API Gateway
Compliance Service
Abstraction Layer provided as module to existing service
Existing service consuming the module
AWS Lambda
Access via AWS IAM
AWS CloudWatch
(Monitoring/Observability)
Access via AWS IAM
State Compliance
Configuration
NoSQL DB
(AWS DynamoDB)

L1 - Compliance Service



Input

```
{
    "productId": "",
    "productName": "",
    "thcContent": "",
    "cbdContent": "",
    "netWeight": "",
    "manufacturer": ""
}
```

Output

```
{
    "productId": "",
    "states": [{
        "state": "",
        "isCompliant": true,
        "requiredProductInfo": {
            "warningLabel": "",
            "governmentWarning": "",
            "productName": "",
            "netWeight": "",
            "thcContent": "",
            "cbdContent": "",
            "batchNumber": "",
            "manufacturingDate": "",
            "manufacturer": ""
        }
    }]
}
```

Design Discussion Points

State Configuration Data Store

We have the option of storing the State Configuration data in either AWS DynamoDB, DocumentDB, or as a new table in the existing Postgres DB. We could utilize Postgres's JSONB column to store the data. This may be a preferred approach to limit the number of components and account for team skillset, however Dynamo DB was chosen in this scenario.

Runtime Injection

Provide the Compliance Service as a layer to existing services to import and use at runtime. This promotes modularity and reusability.

Lambda Region Availability

We would need to provide this as a layer in any additional regions that Nabis has lambda functions running in.

Transactions Per Second

Assuming that there is a relatively low TPS as the platform is not accessed by the general public.

Project Management Considerations

Timeline and Budget

Are we targeting a specific timeline to deliver for a particular customer or regulation? We would want to incorporate this deadline into our effort analysis and raise any timeline risks ahead of time.

Resource Utilization

Are the developers working on this project also assigned other tasks that may introduce a timeline risk?

Team Skillset

I noticed AWS DynamoDB was not apart of the StackShare. Some developers on the team may not be familiar with its implementation and require time to understand this component.

Limit Scope Creep

We want to limit scope creep by thoroughly documenting existing requirements, touching base with stakeholders, providing updates, and prioritize new requirements according to the existing workload.

Ensuring Successful and Timely Delivery

Implement Agile Methodology

Implement an agile, iterative, sprint based development methodology to break the project up into phases with clear goals. Hold daily stand ups with the team to efficiently clear blockers and provide support.

Regularly engage with Stakeholders and SMEs

Regularly touch base with stakeholders and SMEs to understand the ever changing rules and regulations we need to account for, provide updates on the project status, and align on goals and objectives.
Understand priorities in line with business objectives and account for these during sprint planning.
Understand from developers who will utilize the service their needs in terms of input and output requirements.

Test Driven Development/ Automated Testing

The effort required to build unit and integration tests will be taken into account in the project plan. These tests can then be incorporated into the CI/CD pipeline to automatically run when new PRs are opened, reducing the numbers of bugs and maintenance effort.

We could implement Test Driven Development practices where the code is built around the unit tests. This could help identify misalignment in requirements sooner and streamline the development process.

Document Requirements

Comprehensively document the compliance and technical requirements that allow the application to be effectively maintained and scaled. This could be done in a document such as a Business Requirements Document (BRD).

SDLC

Design

Gather and document all requirements including must haves, should haves, and might haves. Document the requirements in a BRD and turn these requirements into tickets that can be estimated and worked by developers.

As part of the design stage we can build scaffolding for the service which involves creating boilerplates for unit tests, the module, and other components to be handed off to developers.

Development

Developers will work on individual tickets including the accompanying unit tests, opening PRs with their requested changes to be code reviewed by a peer.

Testing

Developers unit tests will be apart of the CI pipeline in addition to manual testing by developers and final sign off from the Engineering Manager. We will collaborate with end-users and stakeholders to perform iterative demos and user acceptance testing to gather and incorporate feedback.

Deployment

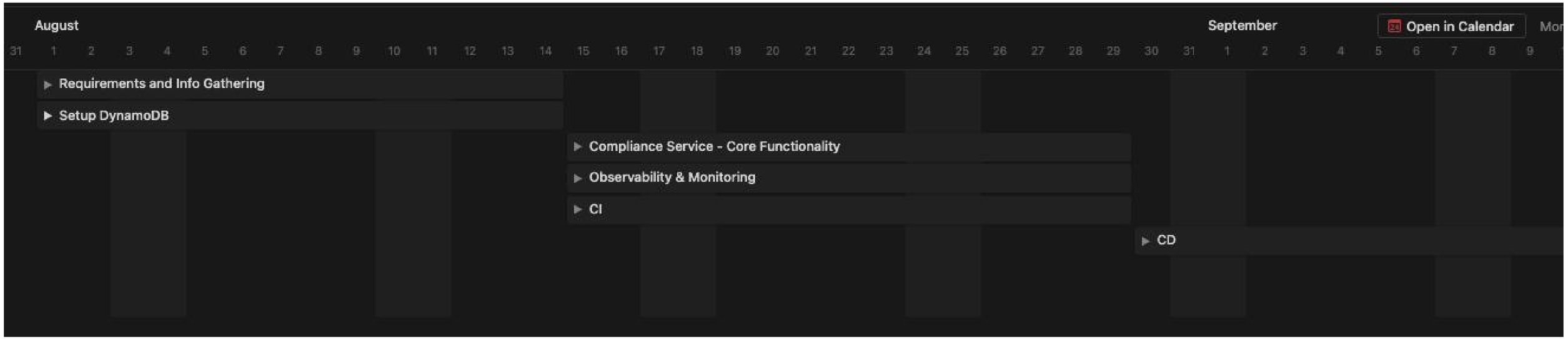
The service will be deployed using a CD pipeline to automate the deployment based on Nabis's Git flow.

Observability & Maintenance

Observability will be built in via CloudWatch so we can receive alerts for issues or events. We will need to continuously maintain the state configurations. A change management process should be put in place to make sure any changes to these configurations are documented and communicated.

Project Timeline

Note: in practice I would take into account stakeholders availability and existing knowledge on team velocity to more accurately plan each sprint and estimate the goals that could be achieved.



Questions & Feedback

Thank you!