**Architecture Design**

**Adult Censes Income Prediction**



**Revision Number - 1.0**

**Last Date of Revision – 05-07-2022**

**SRIPHANI**

# Document Control

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** |  | **Version** | **Description** | **Author** |
| 05-07-2022 | 1.0 |  | Introduction,  Architecture,  Deployment | Sriphani |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**2**

**Contents**

Document Version Control…………………………………………………………………………2

1. Introduction……………………………………………………………………………………….4
   1. What is Architecture Design Document?..............................................................4
   2. Scope……………………………………………………………………………………4
2. Architecture……….……………………………………………………………………….……...5
   1. Components of Machine Learning Architecture……………………………………………….5

# 1. Introduction

## 1.1 What is Architecture Design Document?

Any software needs the architectural design to represent the design of the software. IEEE defines architectural design as “the process of defining a collection of hardware and software components and their interfaces to establish the framework for the development of a computer system.” The software that is built for computer-based systems can exhibit one of these many architectures.

Each style will describe a system category that consists of:

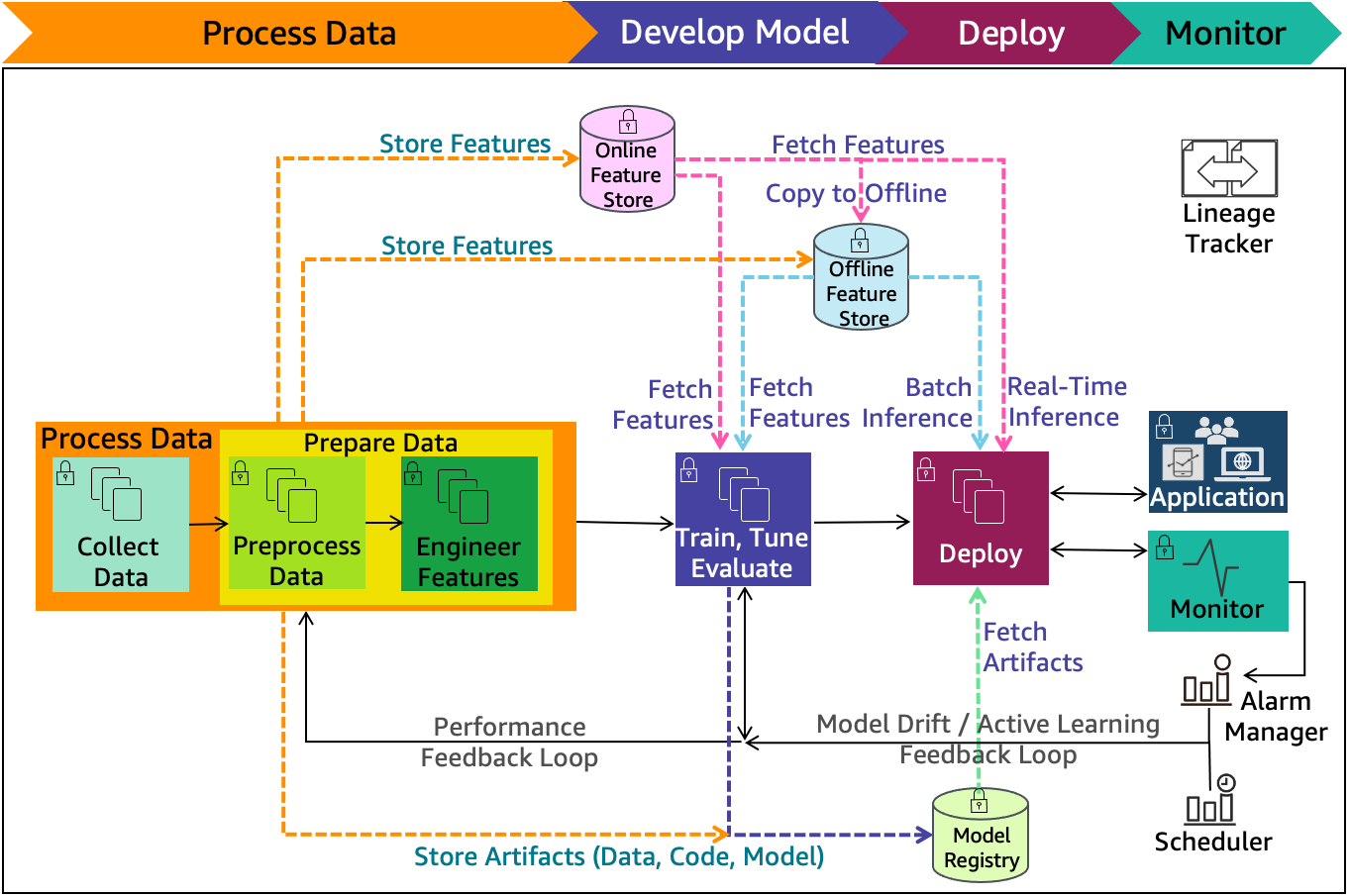
* A set of components (eg: a database, computational modules) that will perform a function required by the system.
* The set of connectors will help in coordination, communication, and cooperation between the components.
* Conditions that how components can be integrated to form the system.
* Semantic models help the designer to understand the overall properties of the system.

**1.2 What is Scope?**

Architecture Design Document (ADD) is an architectural design process that follows a step-by-step refinement process. The process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the design principles may be defined during requirement analysis and then refined during architectural design work.

# 2. Architecture

## 

.

## Components of Machine Learning Architecture:

The cloud agnostic architecture diagrams in this paper provide high-level best practices with the following assumptions:

* All presented concepts are cloud and technology agnostic.
* Solid black lines are indicative of process flow.
* Dashed color lines are indicative of input and output flow.
* Architecture diagram components are color coded for ease of communication across this document.

ML lifecycle as shown in Figure 5 includes the following components:

* **Online/Offline feature store** — Feature store reduces duplication and rerun of feature engineering code across teams and projects. Online store with low-latency retrieval capabilities is ideal for real-time inference. Offline store should maintain a history of feature values and is suited for training and batch scoring.
* **Model registry** — Model registry is a repository for storing ML model artifacts including trained model and related metadata (data, code, model). It enables lineage for ML models as it can act as a version control system.
* **Performance feedback loop** — Automates model performance evaluation tasks initiated from the model development to data processing phase.
* **Model drift feedback loop** — Automates model update re-training tasks initiated from the production deployment to data processing phase.
* **Alarm manager** — Alarm manager receives the alerts from the model monitoring system. It then runs actions by publishing notifications to services that can deliver alerts to target applications to handle them. The model update re-training pipeline is one such target application.
* **Scheduler** — A scheduler can initiate a re-training at business defined intervals.
* **Lineage tracker** — The machine learning lineage tracking enables reproducible machine learning experiences. It enables re-creating the ML environment at a specific point-in-time, reflecting the versions of all resources and environments at that time.

The ML lineage tracker collects references to traceable data, model and infrastructure resource changes. It consists of the following components:

* + System architecture (infrastructure as code to address environment drift)
  + Data (metadata, values, and features)
  + Model (algorithm, features, parameters, and hyperparameters)
  + Code (implementation, modeling, and pipeline)

The lineage tracker collects changed references through alternative iterations of ML lifecycle phases. Alternative algorithms and feature lists are evaluated as experiments for final production deployment.

Figure 6 includes machine learning components and their information that the lineage tracker collects across different releases. The collected information enables going back to a specific point-in-time release and recreate it