Software Architecture Document

Store Sales Prediction

Shubham Gantayat

Version 1.0

01/12/2021

Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Description | Responsible Party | Date |
| 1.0 | Initial version | Shubham Gantayat | 01-12-2021 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

This document provides a high-level overview and explains the architecture of the Store Sales Prediction.

The document defines goals of the architecture, the use cases supported by the system, architectural styles and components that have been selected. The document provides a rationale for the architecture and design decisions made from the conceptual idea to its implementation.

## Purpose

The Software Architecture Document (SAD) provides a comprehensive architectural overview of the Store Sales Prediction (SSP). It presents several different architectural views to depict the different aspects of the system.

To depict the software as accurately as possible, the structure of this document is based on Philippe Kruchten’s “4+1” model view of architecture [Kruchten].



## Scope

The scope of this SAD is to explain the architecture of the Store Sales Prediction (SSP).

This document describes the various aspects of the SSP system design that are architecturally significant. These elements and behaviours are fundamental for guiding the construction of the SSP system and for understanding this project.

## Definitions, Acronyms and Abbreviations

* + - WWW – World Wide WEB
    - HTTP – Hyper Text Transfer Protocol
    - SAD – Software Architecture Document
    - SSP – Store Sales Prediction
    - User – Any user who is registered on our database
    - AWS – Amazon Web Services
    - ECR – Elastic Container Registry
    - S3 – S3 buckets
    - EBS – Elastic Beanstalk

# Architectural Representation

This document details the architecture using the views defined in the “4+1” model [Kruchten]. The views used to document the SSP system are:

**Use Case view**

**Audience**: all the stakeholders of the system, including the end-users.

**Area**: describes the set of scenarios and/or use cases that represent some significant, central functionality of the system. Describes the actors and use cases for the system, this view presents the needs of the user and is elaborated further at the design level to describe discrete flows and constraints in more detail. This domain vocabulary is independent of any processing model or representational syntax (i.e. XML).

**Related Artefacts**: Use-Case Model, Use-Case documents

**Logical view**

**Audience**: Designers.

**Area**: Functional Requirements: describes the design's object model. Also describes the most important use-case realizations and business requirements of the system.

**Related Artefacts**: Design model

**Process view**

**Audience**: Data specialists, Database administrators

**Area**: Persistence: describes the architecturally significant persistent elements in the data model as well as how data flows through the system.

**Related Artefacts**: Data model.

**Deployment view**

**Audience**: Deployment managers.

**Area**: Topology: describes the mapping of the software onto the hardware and shows the system's distributed aspects. Describes potential deployment structures, by including known and anticipated deployment scenarios in the architecture we allow the implementers to make certain assumptions on network performance, system interaction and so forth.

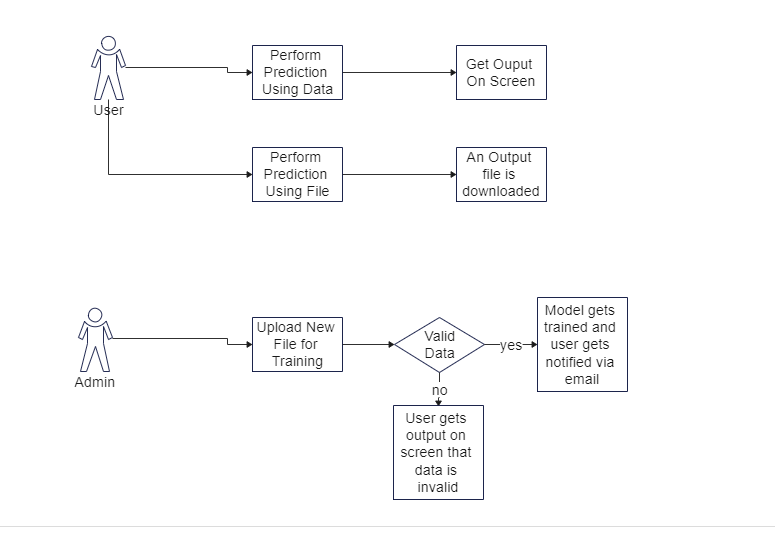
**Related Artefacts**: Deployment model.

# Use-Case View

A description of the use-case view of the software architecture. The Use Case View is important input to the selection of the set of scenarios and/or use cases that are the focus of an iteration. It describes the set of scenarios and/or use cases that represent some significant, central functionality. It also describes the set of scenarios and/or use cases that have a substantial architectural coverage (that exercise many architectural elements) or that stress or illustrate a specific, delicate point of the architecture.

The SSP use-cases are –

* Home – Prediction of new data
* Train data



## Home

First thing a user sees is the home page. This page has two sections – the first section is where user uploads the data and gets a prediction and the second section is which the user uploads a file and a output file gets downloaded at the client side.

## Train data

This page is where administrators can train new models based on the latest data to keep the model up to date. Admin username and password are required to perform this. Admin is also supposed to give an email address to get notified when the model training is completed.

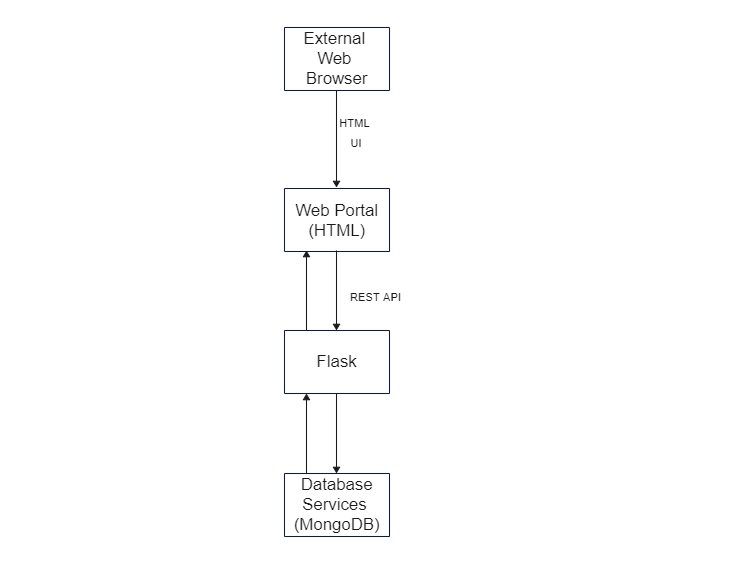
# Logical View

A description of the logical view of the architecture. Describes the most important classes, their organization in service packages and subsystems, and the organization of these subsystems into layers. Also describes the most important use-case realizations, for example, the dynamic aspects of the architecture. Class diagrams may be included to illustrate the relationships between architecturally significant classes, subsystems, packages and layers.

The logical view of the SSP is comprised of the 2 main packages: User Interface and Database Services.

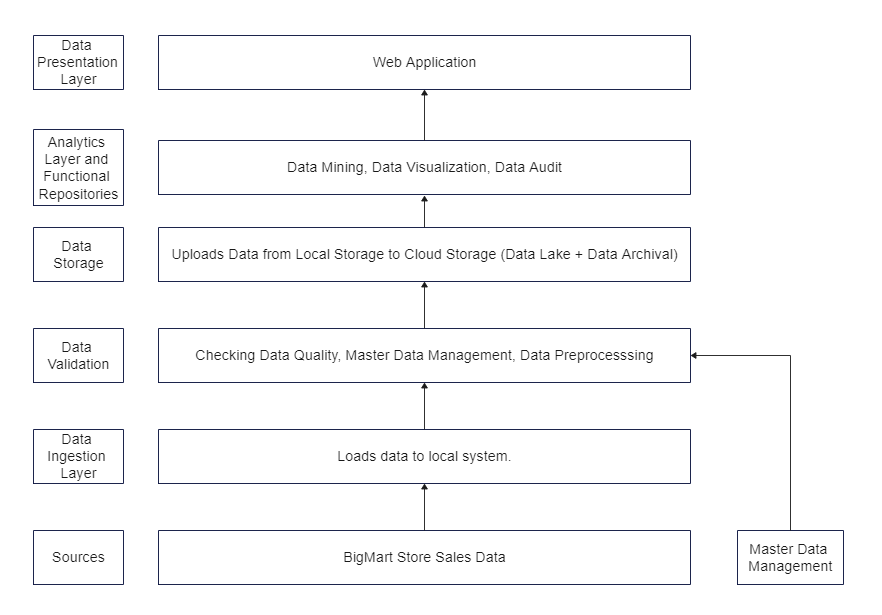
The user interface is used to perform predictions and train new data.

The database services is responsible for storage and retrieval of data.



# Process View

A description of the process-view of the architecture. Describes the tasks (processes and threads) involved in the system's execution, their interactions and configurations. Also describes the allocation of objects and classes to tasks.



# Deployment View

The application is hosted on AWS EBS. AWS ECR is used for containerization. AWS S3 bucket is used for storing “Dockerrun.aws.json” file to run the container. Circle CI is used as CI/CD tool.

