**AnyChange**

Architecture Notebook

There is guidance within this template that appears in a style named InfoBlue. This style has a hidden font attribute that allows you to toggle whether it is visible or hidden in this template. Use the Microsoft® Word® menu **Tools > Options > View > Hidden Text** check box to toggle this setting. There is also an option for printing: **Tools > Options > Print**.

# Purpose

This document describes the philosophy, decisions, constraints, justifications, significant elements, and any other overarching aspects of the system that shape the design and implementation.

[Always address Sections 2 through 6 of this template. Other sections are recommended, depending on the amount of novel architecture, the amount of expected maintenance, the skills of the development team, and the importance of other architectural concerns.]

# Architectural goals and philosophy

The architecture of AnyChange needs to let developers with different area of expertise work together, it has to be adaptable to integrate with many different external product selling platforms and has to be scalable to meet performance criteria of growing customer base.

[Describe the philosophy of the architecture. Identify issues that will drive the philosophy, such as: Will the system be driven by complex deployment concerns, adapting to legacy systems, or performance issues? Does it need to be robust for long-term maintenance?

Formulate a set of goals that the architecture needs to meet in its structure and behavior. Identify critical issues that must be addressed by the architecture, such as: Are there hardware dependencies that should be isolated from the rest of the system? Does the system need to function efficiently under unusual conditions?]

# Assumptions and dependencies

The team is composed developers with varying experience. The project will be built from ground up so it is assumed that there won’t be any legacy interfaces that will be depended on. AnyChange will be a web based software product. It is assumed that there won’t be any publicly available API’s to pull price data from when collecting product data, if there is however, the system should be able to use that as well.

[List the assumptions and dependencies that drive architectural decisions. This could include sensitive or critical areas, dependencies on legacy interfaces, the skill and experience of the team, the availability of important resources, and so forth]

# Architecturally significant requirements

[Insert a reference or link to the requirements that must be implemented to realize the architecture.]

# Decisions, constraints, and justifications

[List the decisions that have been made regarding architectural approaches and the constraints being placed on the way that the developers build the system. These will serve as guidelines for defining architecturally significant parts of the system. Justify each decision or constraint so that developers understand the importance of building the system according to the context created by those decisions and constraints. This may include a list of DOs and DON’Ts to guide the developers in building the system.]

* In order to achieve flexibility required to accommodate several different platforms with the assumption that there is no publicly available API’s to pull price data from, AnyChange will use an external scripting platform that can scrape data from price sources. This will enable us tailor custom solutions for each price source so that we can monitor wide category of products and sources.
* AnyChange will be built using Model-View-Controller architectural pattern. It is a pattern readily supported by Spring Boot framework and allows for separation of data, business logic and presentation of data. This will allow us to develop and create an initial product faster.

# Architectural Mechanisms

## **Scripting Platform**

This mechanism is intended to provide fast to develop, flexible platform to develop web scraping solutions assuming the price sources do not provide free, easy to access API’s to pull price data for products.

[Describe the purpose, attributes, and function of the architectural mechanism.]

# Key abstractions

[List and briefly describe the key abstractions of the system. This should be a relatively short list of the critical concepts that define the system. The key abstractions will usually translate to the initial analysis classes and important patterns.]

System will be composed of products and users. Each user will be authenticated to use the system and users will be able to monitor products that are added to the system. The products can have many different price sources. Each price source is responsible for providing price points at any point of time. A product and price source will have price information, which includes the products current price and price history.

# Layers or architectural framework

[Describe the architectural pattern that you will use or how the architecture will be consistent and uniform. This could be a simple reference to an existing or well-known architectural pattern, such as the Layer framework, a reference to a high-level model of the framework, or a description of how the major system components should be put together.]

When using MVC pattern, the product and users will form the model part of the architecture. The controller part will be provided by the software and it will be responsible for reacting to user events and doing necessary business logic that will result in data that can be presented to user. The view will be made to represent the data produced by the controller.

# Architectural views

