

AnyChange	
Architecture Notebook	Date: 14/05/2023

AnyChange Architecture Notebook

1. 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.

2. Architectural goals and philosophy

The architecture of AnyChange 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. It should provide a secure way of communicating between the users and application software.

3. 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.

4. Architecturally significant requirements

- All data, hence the database applications and clients are expected to be secure.
- Functional uptime of the system will be 166 hours per week.
- System shall scale up to support 10000 concurrent transactions.

5. Decisions, constraints, and justifications

- 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. Spring Boot Framework is chosen because the development team is already experienced with it.
- AnyChange will be developed using Java to enable cross-platform operation.
- AnyChange will use Python as is external scripting platform that will be used for web scraping. Python is chosen because it is easy to use and has widely used high quality libraries we can use.
- All communications between users and AnyChange platform will be encrypted using HTTPS. HTTPS is an extension of the HTTP protocol that provides secure communication over internet.
- All critical data such as credit card information, user passwords will be encrypted.
- Any action on user's account will be authenticated.
- User's will be authorized based on the their type(normal user, seller user and admin) to prevent unwanted actions.
- Some actions like changing passwords will require informing the user by a separate channel like e-mail to prevent malicious use of actions.
- To achieve the uptime requirement, an active redundancy method will be used. The system will be built using distributed services where a load balancer is used to keep service operation in case one of the services are down for maintenance or other reasons.
- To meet the required transaction demand, the system will make use of concurrent processing where possible.

AnyChange	
Architecture Notebook	Date: 14/05/2023

6. 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.

Inversion of Control

Inversion of Control is a pattern provided by Spring Framework, it is used to inject dependencies on a generic program to increase modularity and extensibility.

Availability

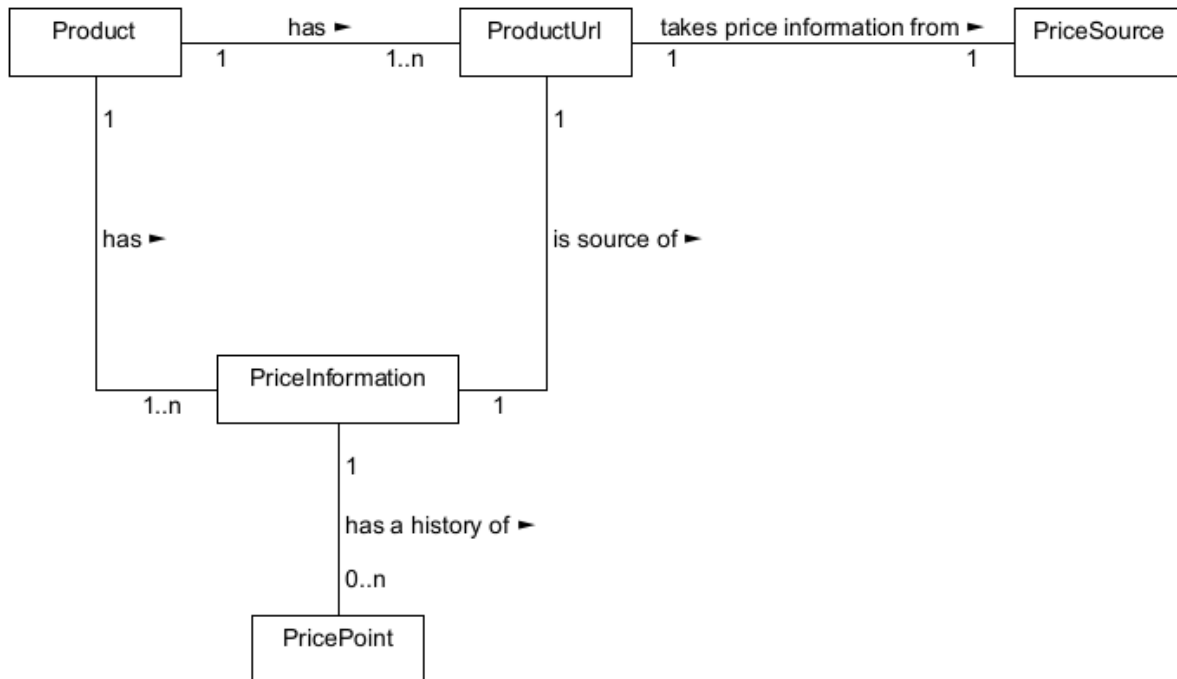
System should be reachable and in working condition to be considered available. To meet the required availability, system should be available during software updates, in an event of crash of a component and it should operate as expected.

Security

The system requires the storage of security critical data such as credit card information and passwords. These should be encrypted against possible attacks against the system. Also since this is a web based application all the communication will be done over public networks. Therefore all communications should be encrypted against eavesdropping and all actions must be authenticated to prevent man in the middle attacks.

7. Key abstractions

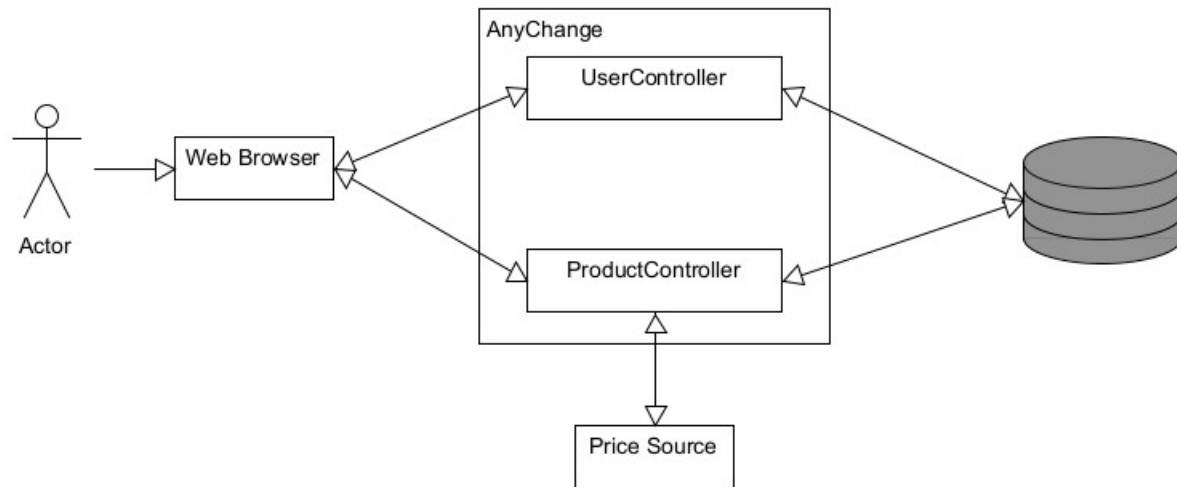
System will be composed of products and users. Each user will be authenticated to use the system and users will be able to monitor products that are added to the system. The products can have many different price sources. Each price source is responsible for providing price points at any point of time.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



AnyChange	
Architecture Notebook	Date: 14/05/2023

8. Layers or architectural framework

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.



9. Architectural views

