**\*|∅ Project Stakeholder Product Review #1**

**Contribution Summary**

**Cody**

Wrote database plugin

Wrote request handle for grouping tickers with same fields

Made sequence diagrams

Made Class diagrams

Helped with CRC cards

Worked on how to make this object oriented

Checked over all code(except the web interface) to make sure it makes sense

**Tony the Simonutti**

Wrote PHP for Odin login page

Worked on Odin login page

Worked on Odin portfolio manager

Wrote Bifrost plugin

Worked on handshake protocol

Reviewed code

Wrote sockets implementation for Bifrost

Integrated Bifrost via network with Yggdrasil

**Chris**

Wrote Yggdrasil

Worked on Odin login page

Worked on Odin portfolio mangager

Worked on handshake protocol

Designed additional protocols for intermodule communication

Worked on CRC cards

Worked on adding object oriented classes to proof of concept methods

Reviewed code

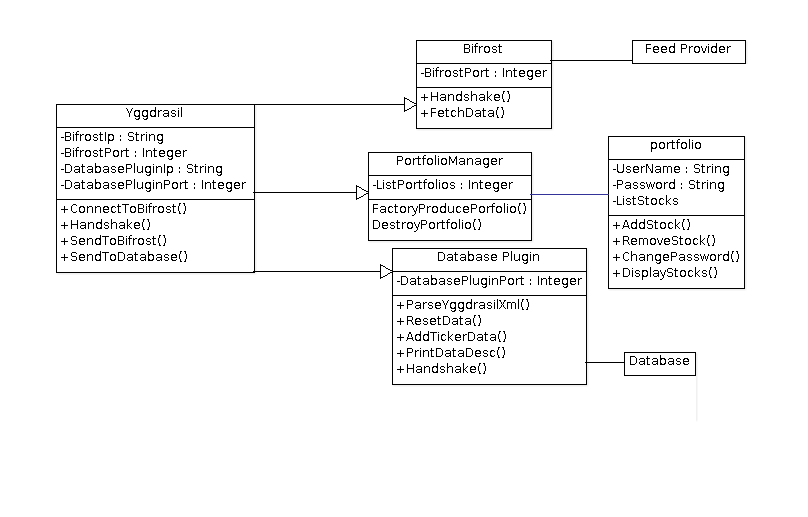
Wrote sockets implementation for Yggdrasil

Integrated Yggdrasil via network with Bifrost

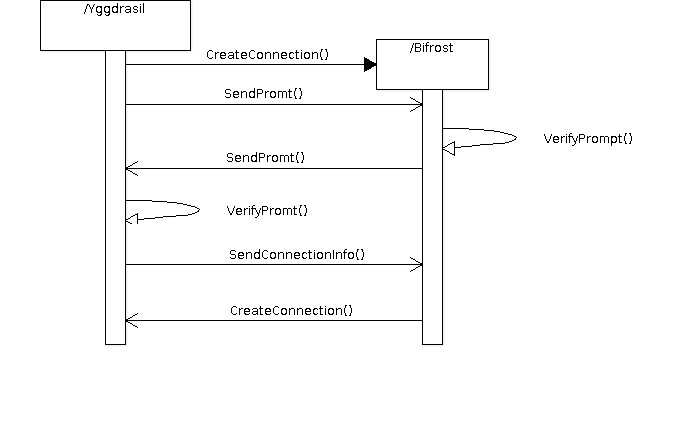
**Tong**

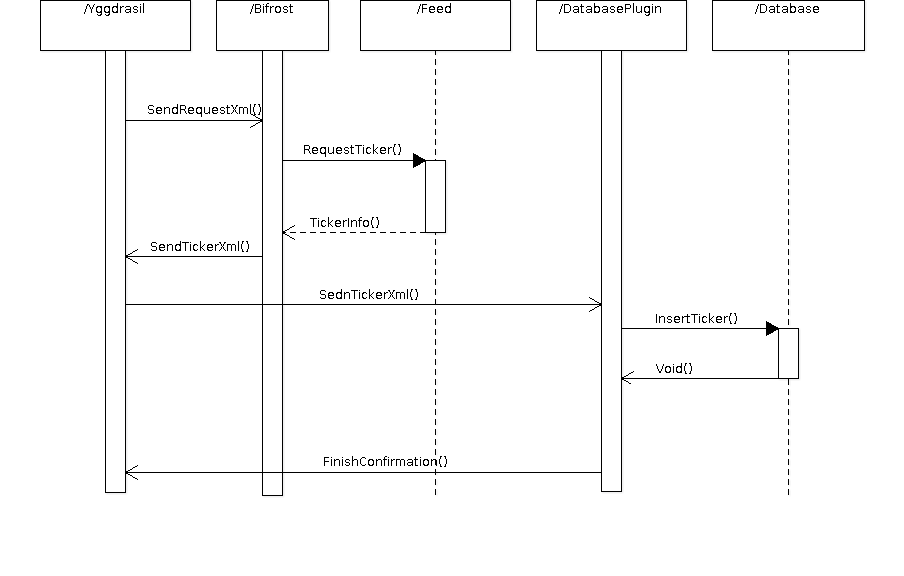
Created Odin web interface for requesting stock information. User can input stock ticker, and desired fields. The form creates an XML file in the standardized format that can be parsed by the feed handler to let the program know what information to pull. Also set up an Apache web server to host this website. Began basic implementations of a CGI script that also receives the form data that can be sent automatically to the Yggdrasil server via sockets.

Class Diagram:



Sequence Diagram, Open Plugin:



Sequence diagram, Request data:

Design Approach:

This is a network-based modular application. Let me explain. Yggdrasil is a main server that connects to all the plugins that you want via sockets. Yggdrasil sends and receives xml via these sockets to communicate with these plugins. Each plugin has a standard protocol for connecting to Yggdrasil and Yggdrasil has a standard protocol for connecting to the plugins. Each of the plugins will be an object from the plugin class from which they will inherit the connection protocols afterwhich they will have their own implementation of what to do with the received data from Yggdrasil. Each plugin will generate its own unique data, and relay the information to the web based interface. The web based interface is constructed using Apache2 web server, bootstrap, html, css, cgi, PHP, javascript, and MySQL. This allows us to have an aesthetically pleasing interface which can push data to the main network using python on the backend.

Status:

At this point we have implemented the core functionalities of Yggdrasil (main server), Bifrost (feed handler), Odin (Web interface), and the database plugin. Each of the plugins works as suggested by the Elaboration schedule. Odin has an interface for requesting stock data and fields, as well as displaying information. Yggdrasil successfully handshakes with Bifrost and can request an information pull from Bifrost over the network. The database plugin correctly stores the stock information with a timestamp and can re-access past information.

The main objective now is to connect the components to function as a modular network-based application. The project is on schedule. The specific plugins that are to be implemented may be revised as the project is modified to an object-oriented standard.

**CRC Cards (pictures)**

