CS4911 Sprint 1 Presentation

Alex Bettadapur Charles Wang Andy Hull Jason Provenzano
Kendall Merritt

September 22, 2015

Overview

Brief Introduction

- Team Members
  - Alex Bettadapur
  - Andy Hull
  - Charles Wang
  - Jason Provenzano
  - Kendall Merritt
- Client: Everette Egun

Product Context

Our client wants a product to perform stock analysis to assist in trading. Currently, the client performs the analysis manually, which is a time-consuming and error-prone task.

In order to make the client's task easier and more accurate, a data acquisition and analysis tool is desired.

Deliverable

We will deliver an application that will collect and analyze stock data based on the client's attributes on a weekly (or otherwise scheduled) basis, generate a report, and email this report to the client.

Furthermore, we will create tools for the client to use independently of these scheduled reports to monitor stocks more closely.

Initial Requirements

The client wants to filter stocks by the following (basic) attributes:

- Float: Amount of tradeable shares
- Growth: Revenue Growth
- Movement: Price Movement

and more complicated attributes as time allows.

The client will receive scheduled emails containing reports listing the stocks which meet the client's thresholds. The client also wants tools to monitor other aspects of stocks. These will be developed as time permits.

Design Alternatives

Since our application is intended more as a data analysis tool, the bulk of our work will be implementing these tools and thus our design does not need to be very complex.

We think the main design decision is where the application should do its work. Thus we came up with the following alternatives:

- Native Client
- Web Application
- Web Server with Thin Client

Alternative: Native Client

We would provide the user with an application that would run on their machine.

- All work is performed on the client's own machine
- Persistent data needed by the application would be stored on the system
- The application would have a GUI to allow the user to interact with the software to change the parameters for analysis and presentation.

Native Client: pros and cons

Pros

- Convenient; always accessible even without internet
- The user can be given the code to maintain/develop
- The user does not need to pay for any server/data storage services

Cons

- Requires user's resources
- Platform dependent
- User's system tied up
- Installation

Alternative: Web Application

We would provide a web service run on a server that would host a webpage where the user could interact with the software. This has the benefit of externalizing all the services so the user's own system is not needed for any part of the software.

- All work is performed by a server independent of client's machine
- Persistent data needed by the application would be stored in a database
- A server would host a webpage to act as a UI

Web Application: pros and cons

Pros

- Does not consume user's resources (space or computation time)
- Platform independent
- Can run completely independently of the user's system
- No installation required on the user's end

Cons

- User may need to pay for external services
- Internet access required to interact with application

Alternative: Server with Thin Client

We would provide the user with a lightweight client backed by a server. The data would be stored on the server and any computations would be performed on the server. The user would interact with the application via a small UI on their system.

- Most work is performed on a server, but small tasks may be run on client's machine
- Persistent data needed by the application would be stored in a databse
- A small client is installed on the user's system to interact with the server

Server with Thin Client: pros and cons

Pros

- Consumes very few resources on user's system
- (Mostly) platform independent
- Heavy work can run independently of user's system
- Tasks can be moved to user's system to run without internet access
- Minimal installation required

Cons

- User may need to pay for external services
- Internet access required for most tasks
- Need to develop a client independent of the server

Final Design - Rationale

We decided to implement the project as a web service.

- platform independent
- does not requiring the user's resources
- no installation

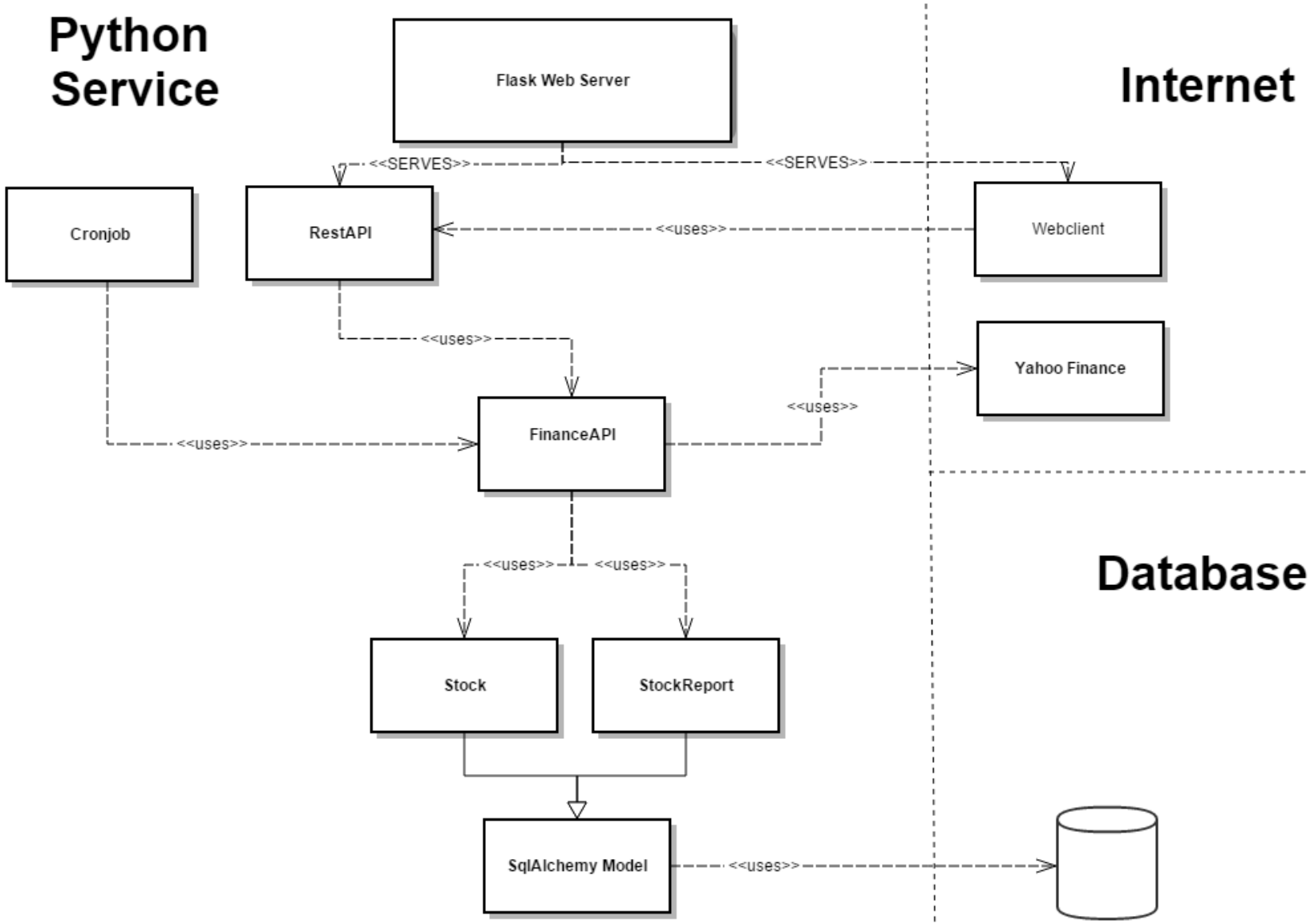
These considerations helped us to eliminate the native client alternative.

The thin client alternative was very similar to the web service alternative. However

- not enough functionality could be moved to the user's system
- extra development cost to implement

so we decided to eliminate the thin client alternative as well

Static Design



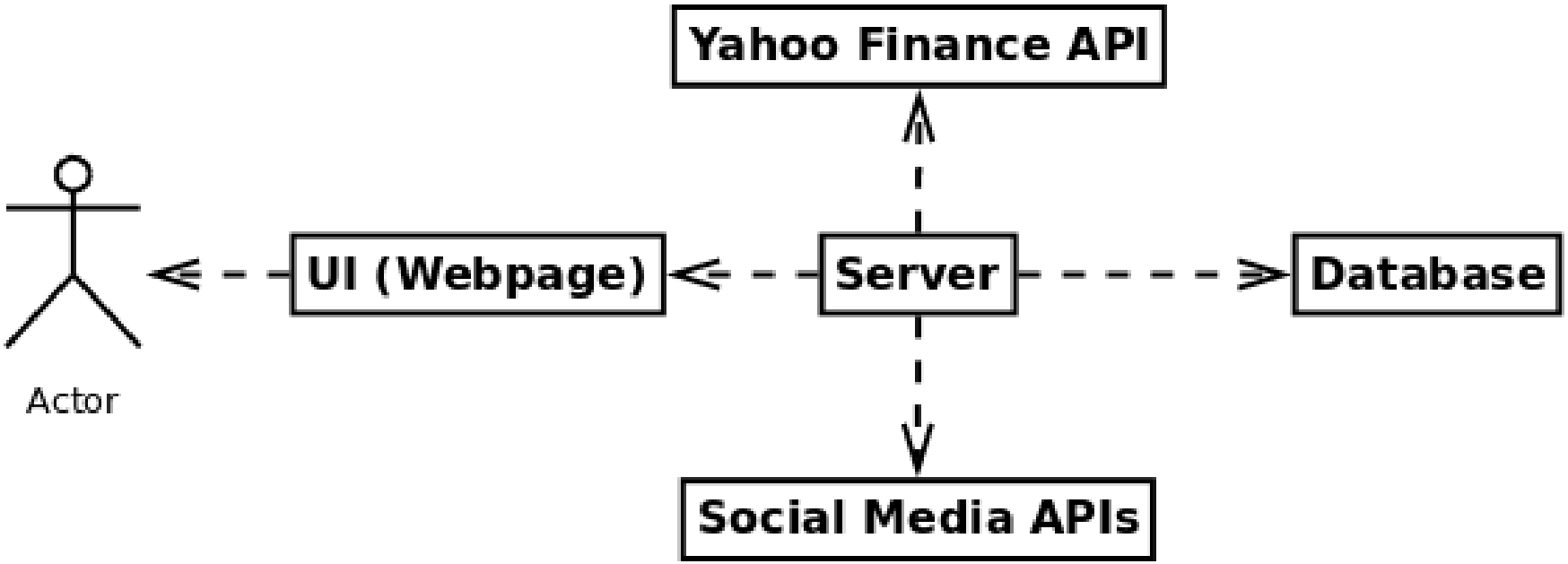
Static Design Components

- webserver: This component knows how to process and fulfill requests from the web client
- cronjob: This component holds information about when to generate the scheduled reports
- restAPI: This component details how to interact with the webclient
- financeAPI: This component details how to fulfill requests by analyzing the data
- yahoo finance API: This component provides stock information necessary to process the user's requests

Static Design Components

- webclient: This component provides a UI for the user to interact with the web server through a layer of abstraction
- stock: This component models the stock object for the system to provide a uniform data retrieval/storage for stocks
- stockreport: This component holds the directions for creating the stock reports that the user requested
- sqlalchemy: This component converts stock information into database entires
- database: This component stores all the persistent information we need about stocks to perform the user's jobs

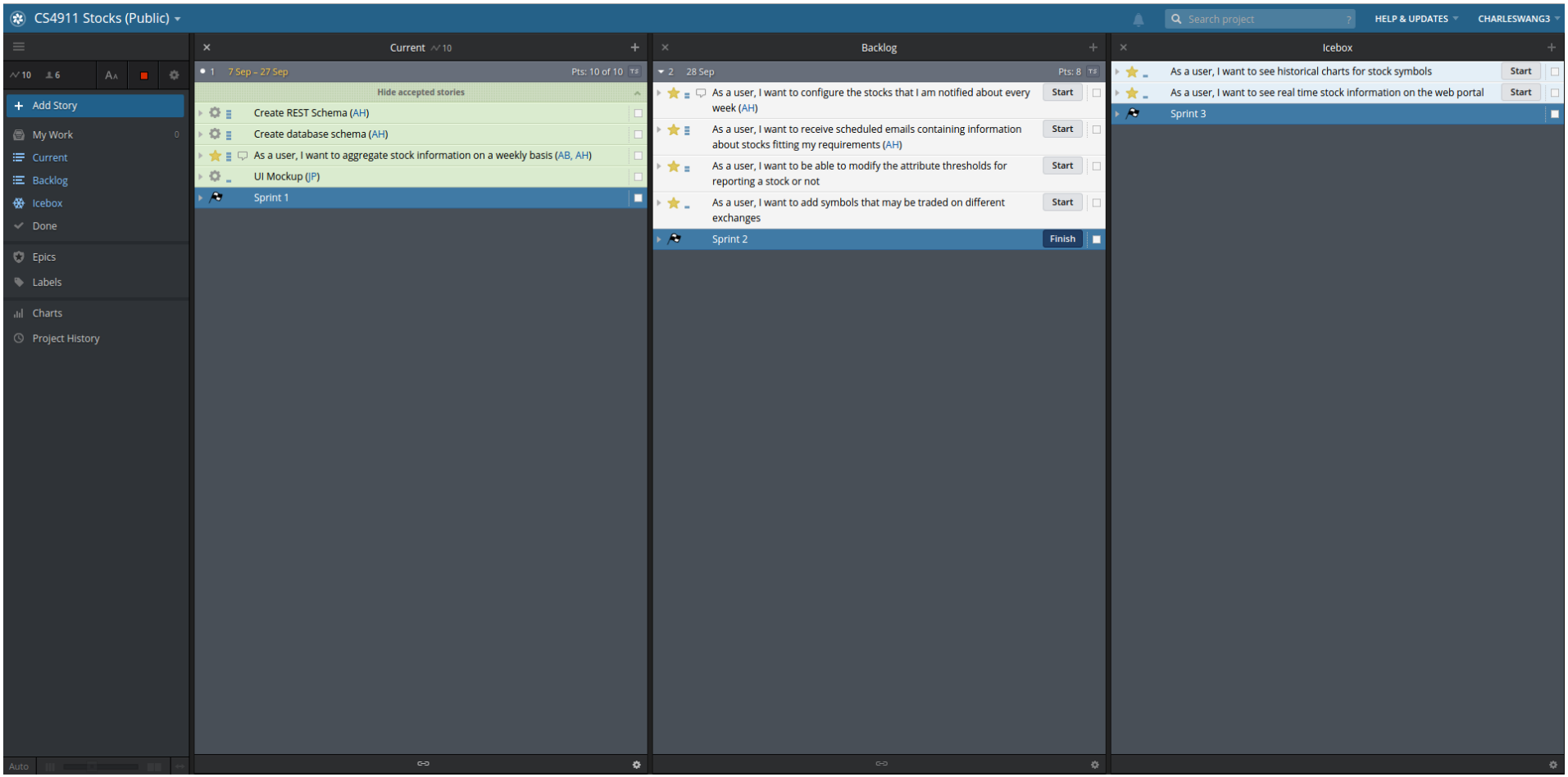
Dynamic Design



Essentially, the dynamic behavior of our application is that the user will interact with the UI (webpage) to request some service. The server will then fulfill this request by gathering the relevant data and reporting the results as appropriate (for example, either delivering a schedule report, or changing the state of a display on the UI).

Sprint 1 Burndown

Pivotal Tracker



Status

Currently, we have finished implementing the client's basic requirements (scheduled report generation based on basic attributes).

At this point, we plan to:

- Work on UI (webpage)
- Work with the client to develop more complex attributes
- Work with client to find what tools they want or need

Demo

Short Demo!