Big Data Analysis

Using Amazon Product API, Kafka, and Storm

Removed Names

Georgia Tech ECE 4813 Project Presentation

Project Proposal

The team has created an application that does Big Data analysis of Amazon Products using Storm and Kafka to:

1. Show users the highest discounted products on Amazon.com in certain pre-selected categories (easily configurable to allow users to choose categories in the future).
2. Allow users to receive an SMS notification if a selected product goes on sale for at least a user-defined percentage.

Screenshot of Homepage

Screenshot of Top TV Deals

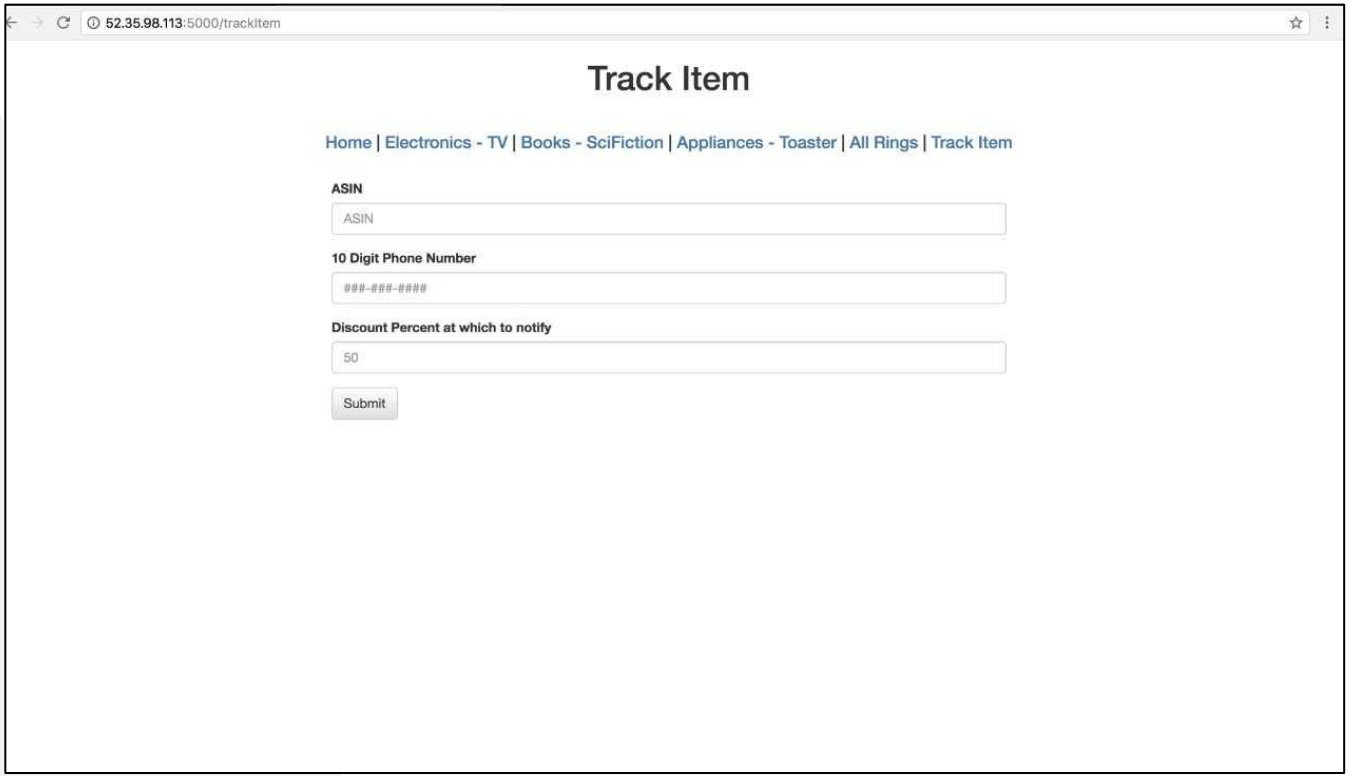
TV

Home | Electronics - TV | Books - SciFiction | Appliances - Toaster | All Rings | Track Item

<< 1 >>

Name	Current Price	Base Price	Discount Percent	Image URL
Sceptre U508CV-UMK 49-Inch Ultra Slim 4K Ultra UHD LED TV, Just Black 2017	$319.99	$549.97	41.8%	
Sceptre X328BV-SR 32-Inch 720p LED TV (2017 Model)	$132.95	$199.99	33.5%	
Samsung Electronics UN50MU6300 50-Inch 4K Ultra HD Smart LED TV (2017 Model)	$527.99	$749.99	29.6%	
TCL 49S405 49-Inch 4K Ultra HD Roku Smart LED TV (2017 Model)	$359.99	$479.99	25.0%	

Screenshot of Track Item Page



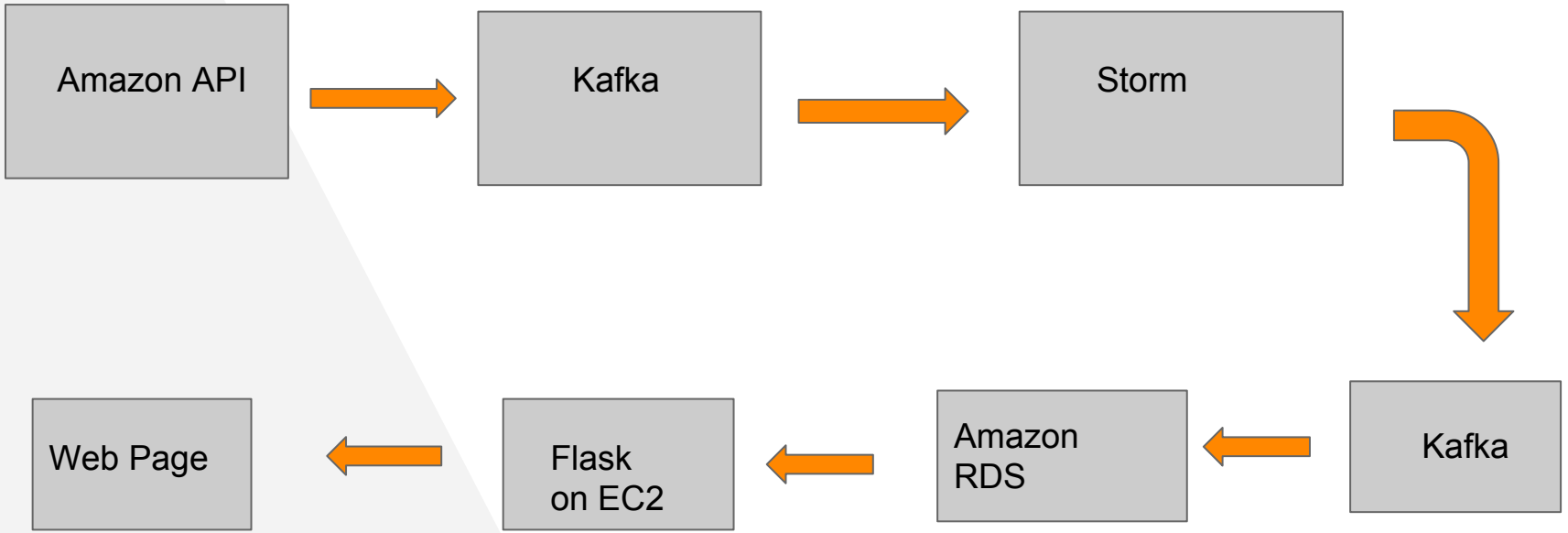
Screenshot of Track Item Confirmation Page

Introduction

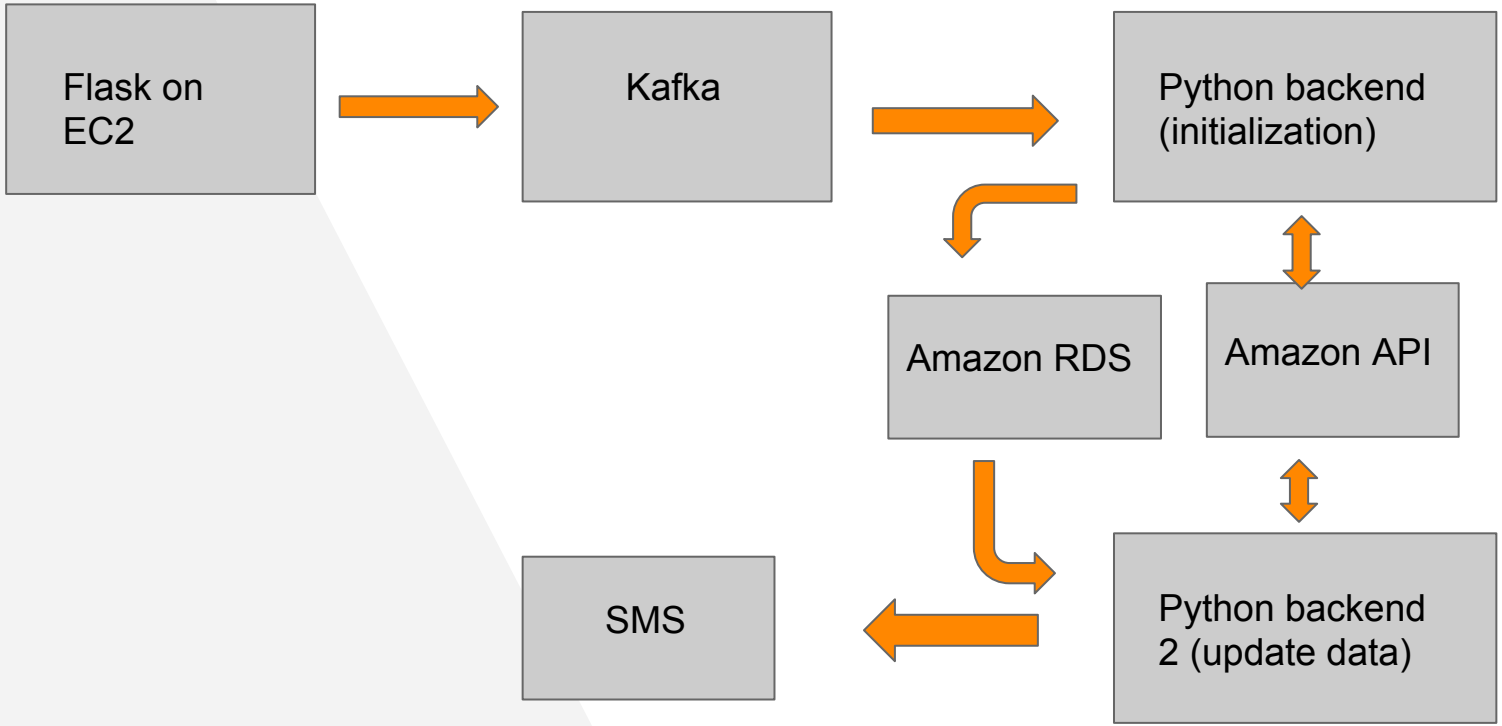
There are four major components of the project:

- Amazon Product API
- Storm Data Analysis
- Kafka Message Broker
- Flask Web App

Overall Architecture of Main Application



Overall Architecture of Tracking Application



Amazon Product API

- An Amazon product API library is used for retrieving prices, product names, and other information directly from Amazon.

Kakfa

- Kakfa is used to publish/subscribe messages. The Amazon API results are published in Kakfa. Kakfa is also used to publish the tracking information from the Flask App.

Storm: Spout

- The Amazon Product API serves as a data storage hub from which the Storm Spout pulls data for pre-selected product categories and sends it to the bolts for processing

Storm: Bolts

- There are two Bolts used in this project:
  - Filtering: The first bolt is used to filter the data that has positive discount.
  - Inserting: The second bolt is used to insert the data into AWS RDS

Flask Web App

- Flask web application framework is used to develop the front-end of the application from where the user will interact with the app.
- Flask reads data from AWS RDS and displays it on the web page.

Availability

- Kakfa, Storm and Flask App are hosted on AWS EC2 Instances.
- The application will be reliably available as long as these AWS services are running.
- Databses will be updated as long as EC2, RDS, and the Amazon Product API are accessible

Scalability

- As everything is hosted on AWS it will be very easy to scale.
- Kakfa and Storm: EC2 instances can be made xlarge
- Front end: Increase the number of EC2 instances and deploy use Load Balancers to handle increased traffic.

Testing

- Manual testing was done by editing the discount values in order to:
  - Ensure changes are updated in the RDS and thus the Flask App quickly
  - Ensure that users are quickly notified when an item they are tracking drops in price

Team Member Contributions

Aliza	Storm and PowerPoint
Nikit	Flask App and PowerPoint
Mao	Flask App and PowerPoint
Julia	Amazon API, Flask App, RDS, PowerPoint
Fei W	Storm, Kafka

THANKS!

Any questions?

You can find us at acharania8@gatech.edu, julian.rosker@gatech.edu, mli399@gatech.edu, ndesai42@gatech.edu, wufei@gatech.edu