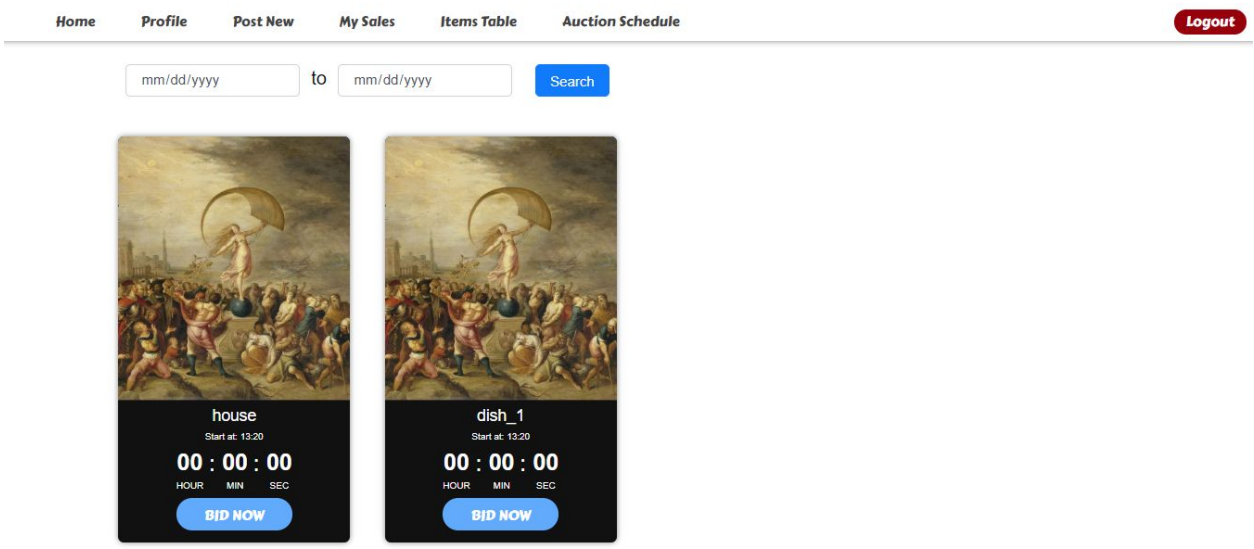
WEB PROGRAMMING LANGUAGES

Marvel Auction Server

Nodejs, Feathersjs, VueJs, Redis, MongoDB



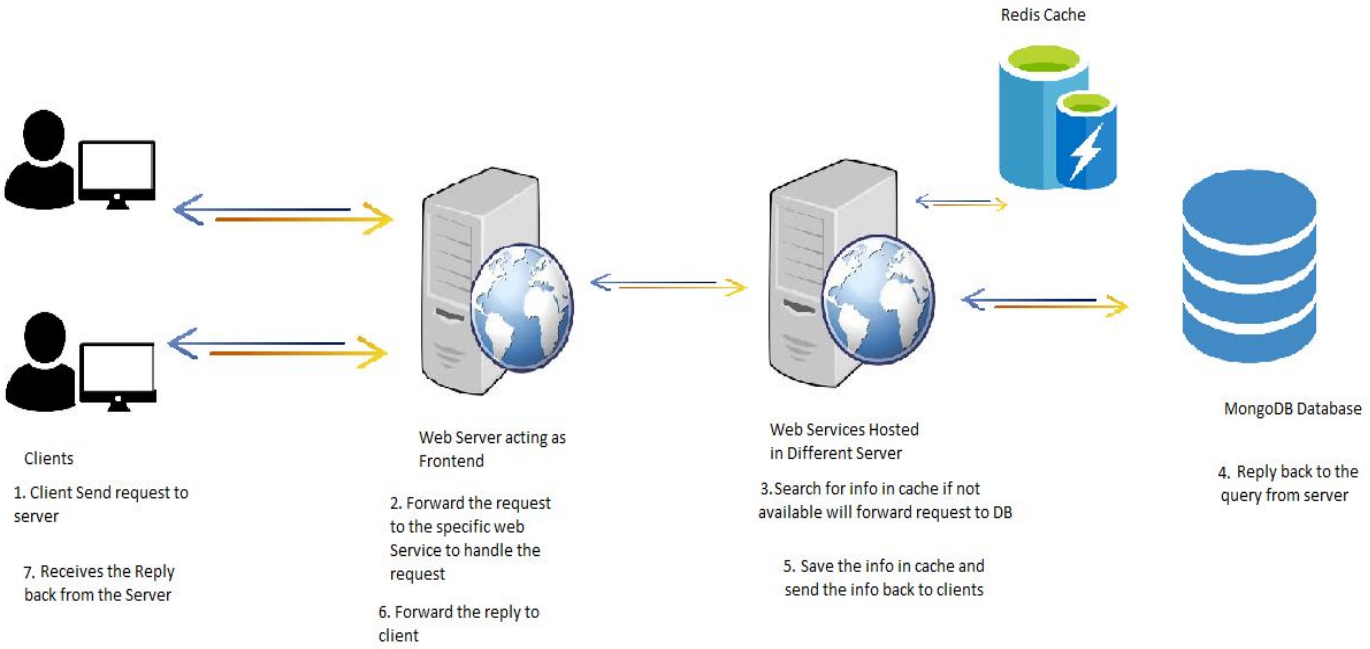
Team name: Avengers
Team Members:

Charles Guo - cxg124130
Ayan Paul - axp170036
Shivdevkumar - sxt160530

Introduction

Marvel Auction Website was designed with Node Js Express framework. With Feathers we can build prototypes in minutes and production ready real-time backends and REST APIs in days. Its lightweight and really powerful for building microservices and web services hooks. MongoDB is a cross-platform document-oriented database program.

Architecture Diagram



Auction Web Server Architecture

Server Backend

FeathersJS

We started with considering MEAN stack for building the Website as using NodeJS as the backend with MongoDB as database. We need a framework for building the backend. We tried building the web services with express framework and had some difficulties in fully

implementing all the requirements of the project. We proceeded with feathersJS which is really powerful and useful in building all the microservices we required with authentication.

Message Queue and Caching

Once we have the Backend setup with feathers JS and MongoDB we need another tool for Message Queue and Caching we consider using rabbitmq but we need to have separate cache tool like memcached so instead we went for Redis which provides both Caching and Message Queue. Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker. It supports data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs, geospatial indexes with radius queries and streams. It's easy to install and use in the web service as feathers had a built-in module to support Redis Caching and Message Queue.Redis is free and open-source.

MongoDB Database

MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time.The document model maps to the objects in your application code, making data easy to work with.Ad hoc queries, indexing, and real time aggregation provide powerful ways to access and analyze your data.MongoDB is free and open-source. We need all the data output in JSON format to display the items in the Frontend so it was easier to use MongoDB to get the database completed.

Server FrontEnd

VueJS

Vue.js is an open-source JavaScript framework for building user interfaces and single-page applications.Vue uses a HTML-based template syntax that allows binding the rendered DOM to the underlying Vue instance's data. All Vue templates are valid HTML that can be parsed by spec-compliant browsers and HTML parsers. Under the hood, Vue compiles the templates into virtual DOM render functions. Combined with the reactivity system, Vue is able to calculate the

minimal number of components to re-render and apply the minimal amount of DOM manipulations when the app state changes.

User Functionality

Registration

User will be able to do registration in from the website. They need to enter username, email and password for registration. Once the registration is completed they need to login to the website to participate in auctions or post any new items for the auctions.

Posting Items for Auction

If a User like to post a new item for auction they need to login. As once the user logs in they will get a authentication token from the server using which will be used for all the request later. For posting the item user need to enter product name, selling price, select the auction time schedule, description and the picture of the item posting for auctions. Once the auction item is posted it will be added to the post items table in the backend database.

Delete items

Users will be able to delete their own items they posted from the website they need to login to delete an item that they posted. They can only delete the item that they posted. They can't delete the other persons items.

Bid for Items

User will be able to bid for items in the auctions items list once they logged in. Then the current price for the item will be updated. Also the bid information will be added in the bid table which will be later used to display the summary of all the bids for an item.

Search for Items

User will be able to search for items. As it will do a get call from the server to get the list of the items posted for auctions. Similarly user will be able to get summary for all the bids for the

items. They will be able to get the items list and bid list without login to the website. As it will just display the information which will be useful for user to review the website before they register and post the items for auctions.

Unavailable Page

If the user access any unavailable pages in the website by mistake he will get a unavailable page with a 404 generic error.

Admin Schedule Auction hours

Admin user will be able to schedule available hours for auctions. Will be used by user when posting the item at that time they will select the available auction time for schedule.

Web Services

We have following web services created for different purposes.

1. Auctions
2. Bids
3. Postitems
4. Users

Auctions Web Service - When admin do schedule for available slots for auction it will use this service and the available slots will be stored in auctions collection on the database

Bids - When a user perform bidding the current price information will be updated in the postitems collection and a new entry will be created in the bids collection.

Post items - when a user post a item this service will be called for creating a new entry in the post items collection.

Users - Where the user registration, profile edit and forgot password all the calls are processed and it updates the users collection in the database.

Summary

Backend Issues : we had issues in setting up and running redis server on windows sometimes it was crashing. We need to restart the application. At first we faced issues in creating authentication for all the individual services which should get the token and that token should be used for all other url to authenticate. When we started using feathers we are able fix that issue. Then on setting up message queue we need to set up workers to listen to the queue and call the service to create the a new entry in the bid collection. Passing the incoming data to the message queue was a difficult part but we were able to complete that part.

Frontend Issues: In the front end we faced multiple issues on uploading the images and in selecting a schedule for the time slots. We have to work with workaround on the backend to fix those issues so that backend handles some of the stuff. Like when you bid for a item it will update the current price in the postitems collection also create a new entry in the bids table. So we handled this in backend by passing the incoming data in message queue to the bid service where there is worker listening to the message and create a new entry in the bid collection by calling the bid service create. Then we also had other issues on getting all the API's integrated with the frontend in VueJS using axioms. We had to follow different online videos to go through the tutorial and complete the project.

We selected these technologies for the main reason to work newly on these technologies. It gave us an opportunity to learn these new technologies and work on it. We learned a lot from working on this project both from the frontend and backend perspective and how the web services are built and how caching, message queue are implemented.