04/06/2016

1. Team Project Topic
    a. Dish Order Service
2. Techniques: NodeJS + AngularJS
    a. Scaling Design Approaches
    b. NoSQL DB/RDBMS + Web Service
        i. DB Design
        ii. DB Access Method: NodeJS
        iii. Web Service: NodeJS
    c. Web UI
        i. AngularJS
3. Other Questions?

Scaling Design Approaches

Business Logic

a. **Sign up/Login/Log out**

user need account name, password, phone number, email address to sign in

user need account name and password to login

b. **Menu Display**

Menu will display in catalog(starter, main course, noodle, rice, beverage, dessert, soup).

Each dish should show picture, name, price, materia

The menu will display in the main page for use scan, but it has order button, once user click on the order button, the user will need to login first and select the location(zipcode), then the user will go to an order page. Display and order page is not the same one.

c. **Shopping Cart/Order related/Payment**

In order page, every dish has “Add to shopping cart” button, once user click on add to shopping cart, the dish will be saved in the shopping cart. After user has selected all the items, he goes to the shopping cart to check his dishes.

In shopping cart page, the user will see all the dishes he selected and the total price, also the recommended dishes. User can select the recommended dishes into the order or delete the unwanted items from the shopping cart.

After user has confirmed that he wants to check out, he will click on check out and user will be asked to enter the address and contact information, after user finish it, user will go to the checkout page.

In the checkout page, the order details(orders, delivery address, contact number) will show up and user can select different ways to check out(credit card, cash).

d. **Recommendation Dishes**

Part 1: Main page should have special dishes for today.

Part 2: In the shopping cart page, user will see the recommendation dishes. If there is no beverage, soup or dessert in the user's order, we will recommend the corresponding items, user can just add them into the order and proceed to checkout.

e. **Comments on dishes?**

User needs to confirm he received the delivery in the order history. After user confirms, he is able to go to his order history and put comment on the dishes.

If we have comments, then in the order page, every dish needs to have a comment button or total score (stars or something) and once click on the comment button, the comment page for the dishes will show up and display all the comments made by users for this dishes.

f. **User Profile**

The user's sign in information, delivery address, order history should be saved.

a. Tracker?
b. Couple/gift cards/gift point?
c. Branch management?

Scaling Design

. Scaling Up

The simplest methodology for scaling a system is to use bigger, faster equipment. A system that runs too slowly can be moved to a machine with a faster CPU, more CPUs, more RAM, faster disks, faster. This is called scaling up because the system is increasing in size.

. **Replicate the entire system (horizontal duplication);** the technique of using many replicas of a web server behind a load balancer is an example of horizontal scaling.

. **Split the system into individual functions, services, or resources (functional or service splits);**

Example: A single machine that was used for a web server, a database, and an application server (dynamic content generation). The three functions all compete for resources such as disk buffer cache, CPU, and the bandwidth to the disk, memory, and memory subsystems. By moving the three major functions to separate machines, each is able to perform better because it has dedicated resources. Example: When displaying a question (and its answers) to a logged-in user, the page is augmented and customized for the particular person. Anonymous users (users who are not logged in) all see the same generic page. The anonymous pages are now handled by a different system that generates the HTML once and caches it for future requests. Requests from web crawlers are sent to a dedicated pool of replicas.

. Split the system into individual chunks (lookup or formulaic splits).

A lookup-oriented split scales a system by splitting the data into identifiable segments, each of which is given dedicated resources. z-axis scaling is similar to y-axis scaling except that it divides the data instead of the processing.
A simple example of this is to divide, or segment, a database by date. If the database is an accumulation of data, such as log data, one can start a new database server every time the current one fills up. There may be a database for 2013 data, 2014 data, and so on.

. Caching

A cache is a small data store using fast/expensive media, intended to improve a slow/cheap bigger data store. For example, recent database queries may be stored in RAM so that if the same query is repeated, the disk access can be avoided. Caching is a distinct

. Data Sharding

Sharding is a way to segment a database (z-axis) that is flexible, scalable, and resilient. It divides the database based on the hash value of the database keys. This pattern is called a distributed hash table (DHT) since it distributes the data over many machines, and uses hashes to determine where the data is stored.

. Threading

In multithreading, a main thread receives new requests. For each request, it creates a new thread, called a worker thread, to do the actual work and send the reply.
Since thread creation is fast, the main thread can keep up with a flood of new requests and none will be dropped.

. Queueing

With queueing, you are less likely to overload the machine since the number of worker threads is fixed and remains constant. This avoids the overhead associated with new thread creation. Queueing is similar to multithreading in that there is a master thread and worker threads. The master thread collects requests and places them in the queue. There is usually a fixed number of worker threads. Each one takes a request from the queue, processes it, sends a reply, and then takes another item from the queue. This workflow is called feeding from a queue.

. Content Delivery Networks

A content delivery network (CDN) is a web-acceleration service that delivers content (web pages,images, video) more efficiently on behalf of your service. CDNs cache content on servers all over the world. Requests for content are serviced from the cache nearest the user. Geolocation techniques are used to identify the network location of the requesting web browser.

Other technologies not mentioned

. Database Connection pool
. Message Queue (Rabbit MQ)

*** Team Project Extra Credit