Improvement 1: share list with family members

Idea 1: share information via email. In the front-end application, we can design a “send” button which triggers a back-end service(taking SMTP server as example) when clicked on. The information of the grocery list can be stored in the email message or as a file attachment. And the SMTP server is responsible for processing your email, deciding which server to send the message to, and relaying the message to that server.

Idea 2: REST API design. If the data shared between two users is small, we can consider designing a rest api. The application can expose a api to accept data sharing request. The data can be added to the post payload and then access the api. If data sending is successful, then the api will respond with a 200 status number indicating that the sharing operation is completed. The disadvantage is that there is a limit on the size of api payload.

Improvement 2: scale the project to a million users

hardware improvement/content delivery network/distributed server/

Idea 1: hardware improvement. It is easier if we can scale the app by just investing on the hardware equipment without having to change the software architecture. By adding processing power and memory to the physical machine running the server, the app can handle more requests with faster speed.

Idea 2: Content delivery network. The users of an app might come from different parts of the world. In order to decrease the delay, we can utilize CDN to cache website content such as webpages, images and videos to users’ closest location by proxy server. The caching process enables an agile transfer of resources, which could improve users’ experience.

Idea 3: Distributed server. Instead of having just a server host, we can have a group of server computers to deal with different types of request. Taking this project as example, we have two different kinds of request: user-related(login, register) and data-related(add, read, delete, update). We can have one server focusing on handling user-related API and another server handling data-related API. And among the four operations of data-related API, “add” is the most common operation, so we can even assign a serve specializing in this operation. By having a group of distributed server, we could reduce the traffic that might happen millions of users trying to access one server at the same time.

Improvement 3: additional linked account

We could allow a logged-in user to link one or more accounts and can share data to his/her linked account users.

In order to realize the function of adding user, we need a new data schema(call it Friends) which records which two users are linked. When a user clicks on a button to add a user, it will send a request to a back-end API which checks weather this user exists. If that user exists, it will add a new piece of data in Friends indicating these two users are linked.

When a user clicks a button to share his grocery list to his linked users, it will send a request to another back-end API which searches in Friends and find out the information of this user's linked users and send back to front-end as response. Then the user can choose which of his linked users he would like to share his grocery list to. And then accessing another API to retrieve his grocery list information and add it under the chosen users' item list.