CashTrack
High Level Architecture Specification
By Runtime Terror

Date: 19th March 2021

System Architecture

The deployment of our application, CashTrack, ensures remote application accessibility, scalability and reliability as the application evolves and grows.

The Angular Frontend and the Node Express Server shall be hosted on a Google Cloud Platform's App Engine Instance. GAE is a Platform-as-a-service i.e. PaaS, which ensures automatic scalability and a seamless service experience. The Google OAuth service shall also be utilized, which shall be accessed by the frontend by using APIs hosted and maintained by Google. Moreover, the REST APIs will be accessible via HTTP requests and the User will be able to access the application frontend by using an HTTP URL in a web browser. This URL will be defined later in the production stage. Further, MongoDB will be hosted in a cloud server which is accessible via Mongo drivers for Node by providing the database credentials.

The interactions between the various components of our software development stack, MEAN and Google OAuth API 2.0 (used for User Authentication), are visualised in the diagram below.

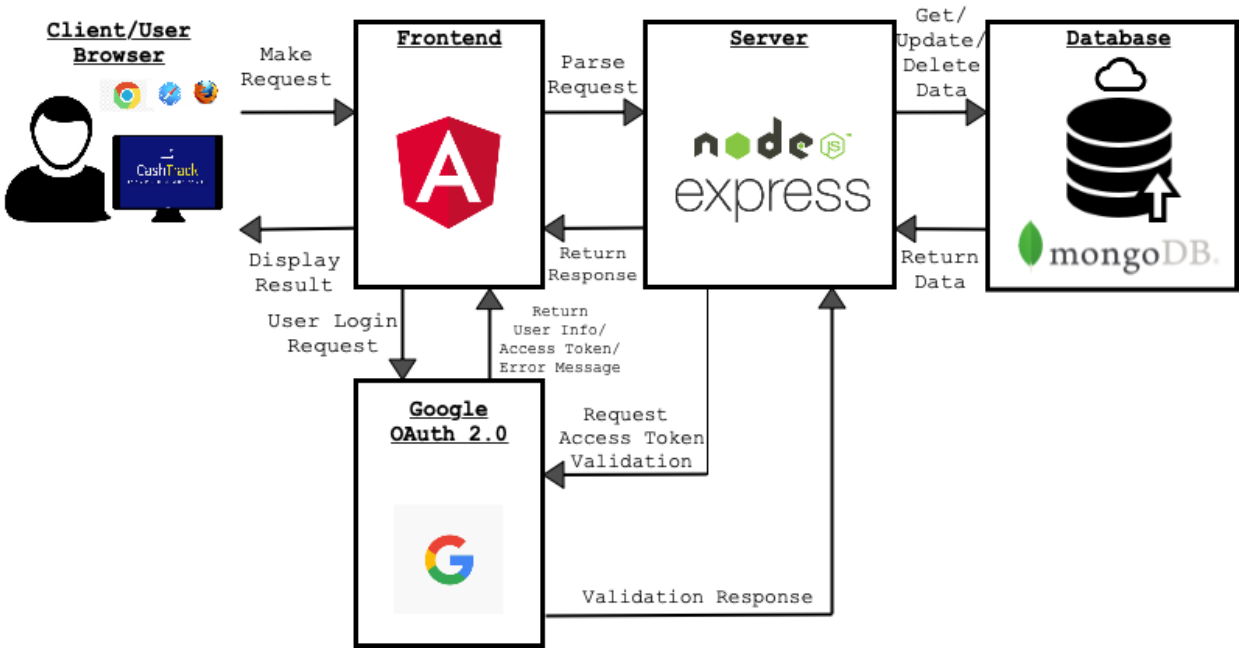

Fig 1. Cashtrack's System Architecture

Database Architecture

Our database has been structured in a way, such that it is maintainable, flexible and extensible. While structuring our database, these 3 factors played an extremely important role as it largely affects the scalability of our application. Cashtrack makes use of a MongoDB database. MongoDB stores data in flexible, JSON-like documents, allowing fields to vary from document to document and allowing data structures to change over time. Our database has the following collections:

1. **Users**

   `<user_id, first_name, last_name, personal_limit, personal_limit_reminder, profile_image, personal_limit_duration>`

2. **User Personal Expenses**

   `<user_id, bill_id, created_at, updated_at, label, tag, expense_amount>`

3. **User Shared Expense**

   `<bill_id, created_at, updated_at, author, label, tag, total_amount, split_by_method, group_id, payer, comments>`

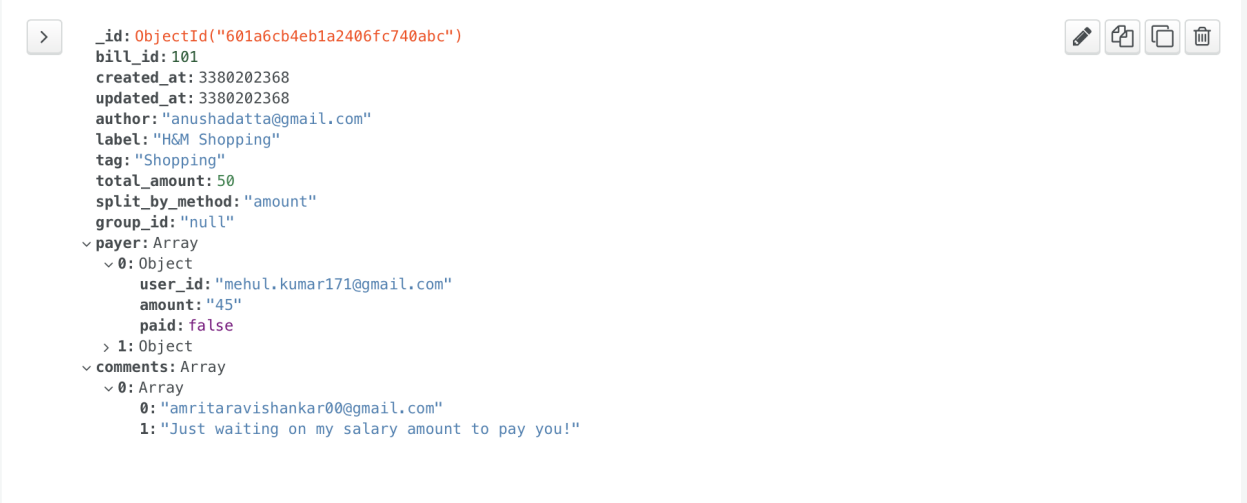

Fig 2. A screenshot of a shared expense record

4. Notification

```
<user_id, type, message,bill_id>
```

5. Reminder

```
<author_id, payer_id, bill_id, due_date,reminder_interval,settled_up>
```


```
_id: ObjectId("601a6fffdd3d6befc9b87fd7")
author_id: "amritaravishankar00@gmail.com"
payer_id: "mehul.kumar171@gmail.com"
bill_id: 100
due_date: 1581379140000
reminder_interval: 5
settled_up: false
```


Fig 3. A screenshot of a reminder record

6. Groups

```
<group_id, group_name, members>
```

7. Chat

```
<user_1, user_2, messages>
```

8. Tag Thumbnail Images

```
<tag, image>
```