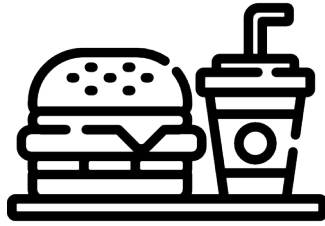


# **Project report.**

**CFS**

*Canteen Food Service.*



*Submitted by: Daksh Sharma*  
*Roll No: IMT2021533*

## PROJECT PLAN.

Major aspect of this project work has been to develop a proper food ordering website which can be used by students to order food from canteen while sitting in their room so as to reduce the waiting time .

I have used node js and mongodb databse to store data of users. The details such as user info , item list, orders placed have been uploaded on mongodb cloud database.

### Modules used:-

```
//shunt esversion.0
require("dotenv").config();
const express = require("express");
const bodyParser = require("body-parser");
const mongoose = require("mongoose");
const { stringify } = require("querystring");
const session = require("express-session");
const passport = require("passport");
const passportLocalMongoose = require("passport-local-mongoose");
const GoogleStrategy = require("passport-google-oauth20").Strategy;
const findOrCreate = require("mongoose-findorcreate");
const dotenv = require("dotenv");
let alert = require('alert');
const app = express();
```

### Database used:-

MongoDB (cloud)

### Collections:-

I have made 4 collectionins in my database.

- **Items:-** To store the items available in canteen at present time. For default purpose i have added only 4 items in my items.

```
const itemsSchema = {  
  name: String,  
  description: String,  
  price: Number,  
};
```

- **User:-** To store the information of the registered users.

```
const userSchema = new mongoose.Schema({  
  email: String,  
  password: String,  
  googleId: String,  
});
```

- **Cart:-** A different cart for each user to store the items user wants to order.

```
const CartSchema = new mongoose.Schema(  
  {  
    userName: String,  
    products: [  
      {  
        quantity: Number,  
        name: String,  
        price: Number,  
      },  
    ],  
    active: {  
      type: Boolean,  
      default: true,  
    },  
    modifiedOn: {  
      type: Date,  
      default: Date.now,  
    },  
  },  
  { timestamps: true }  
);
```

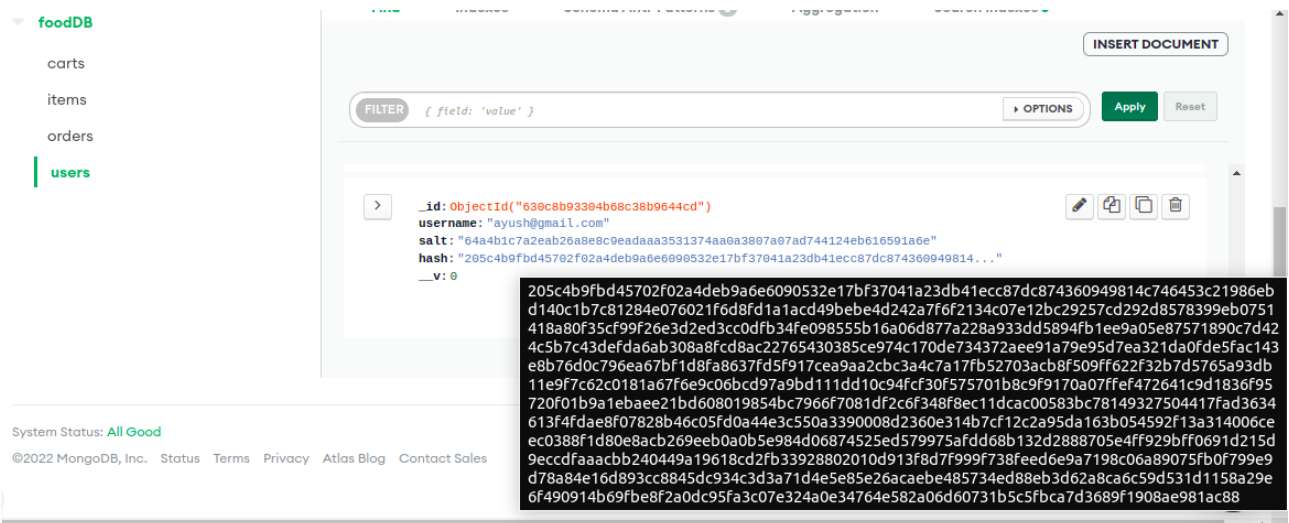
- **Order:-** To store the orders placed by different users this can be used by canteen workers to prepare orders.

```
const orderSchema = new mongoose.Schema({
  name: String,
  price: Number,
  products: [
    {
      quantity: Number,
      name: String,
      price: Number,
    }
  ]
});
```

## Security:-

For security purpose I have used passport module to store the password of user in hashed form. The techniques used are salting and hashing to make the password secure.

I have also used google signin option. This is also a secure login method for my website.



The screenshot shows the MongoDB Atlas web interface. On the left, a sidebar lists collections: foodDB, carts, items, orders, and users (selected). The main area displays a document from the 'users' collection. The document structure is as follows:

```
{
  "_id": ObjectId("639c8b93394b68c38b9644cd"),
  "username": "ayush@gmail.com",
  "salt": "64a4b1c7a2eab26a8e8c9eadaa3531374aa9a3897a67ad744124eb616591a6e",
  "hash": "295c4b9fbd45782f02a4deb9a6e6090532e17bf37041a23db41ecc87dc874360949814...",
  "v": 0
}
```

The 'hash' field is highlighted with a red box. Below the document, a large block of hexadecimal text is visible, which is the raw representation of the hashed password.

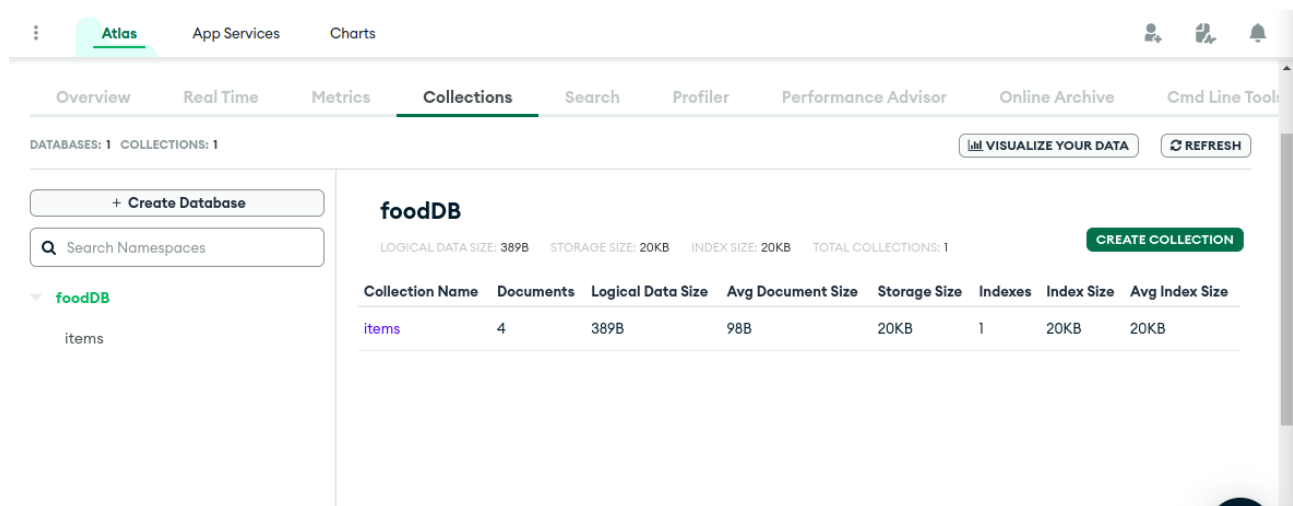
System Status: All Good  
©2022 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

## Express:-

I have made use of express module to display different pages and list of items such as view items in cart or displaying purchased items along with total cost of order, date and time of placing order.

## Working of my website:

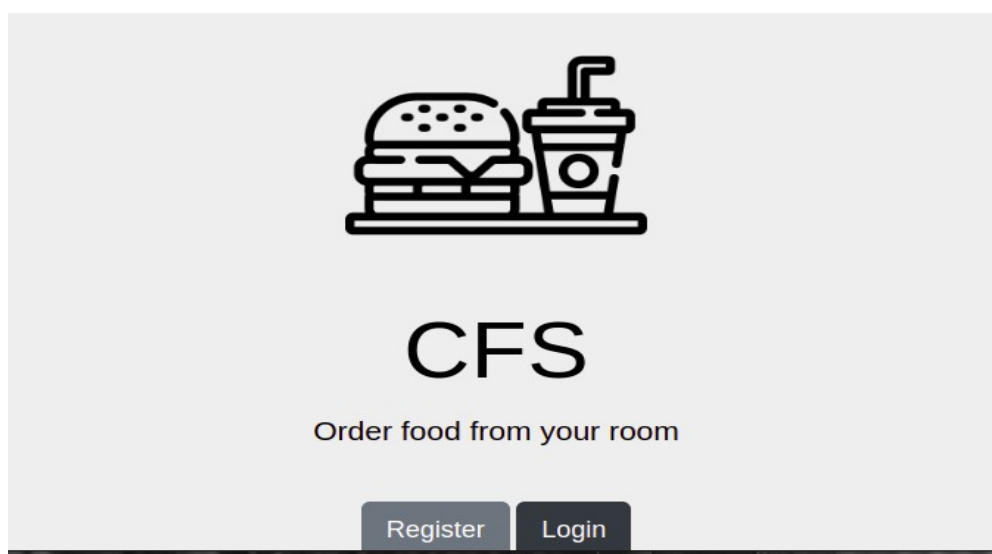
1. Initially my database has 4 default items and no users, no cart and no orders.



The screenshot shows the MongoDB Atlas interface for a database named 'foodDB'. The 'Collections' tab is selected, showing a table with one collection named 'items'. The table has 4 documents, a logical data size of 389B, an average document size of 98B, a storage size of 20KB, 1 index, and an average index size of 20KB. The interface includes navigation tabs like Overview, Real Time, Metrics, Collections, Search, Profiler, Performance Advisor, Online Archive, and Cmd Line Tools. There are also buttons for '+ Create Database', 'Search Namespaces', 'Visualize Your Data', and 'Refresh'.

Collection Name	Documents	Logical Data Size	Avg Document Size	Storage Size	Indexes	Index Size	Avg Index Size
items	4	389B	98B	20KB	1	20KB	20KB

2. From Home page you can register new user or login with previous username and password.



A registration form titled "Register" is shown. It features a light blue input field for email containing "1@2.com" with a green checkmark to its right. Below it is a white password input field with masked characters ".....". A dark grey "REGISTER" button is positioned below the password field. At the bottom, there is a dark grey bar containing a "G Sign Up with Google" button.

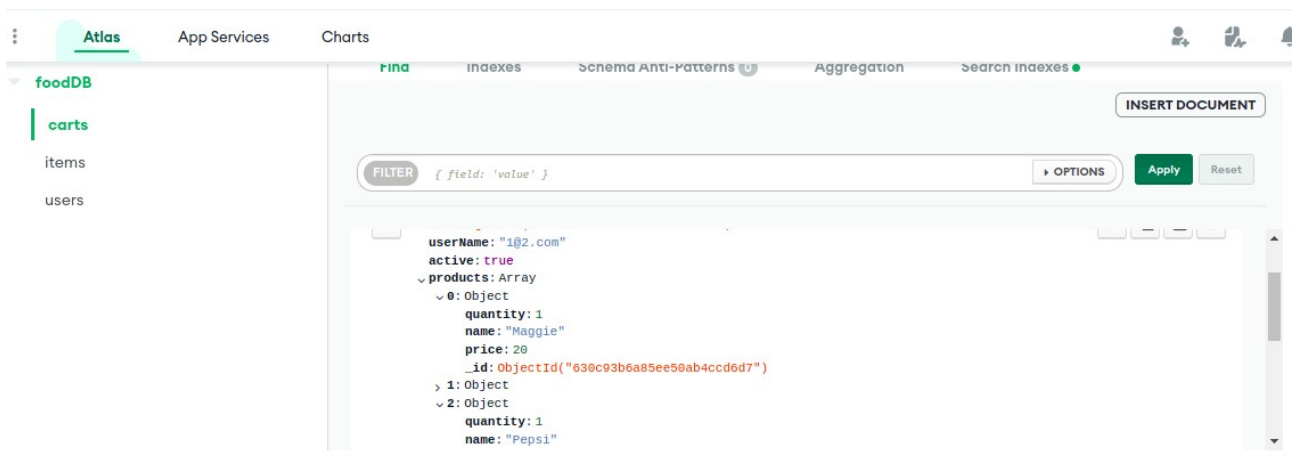
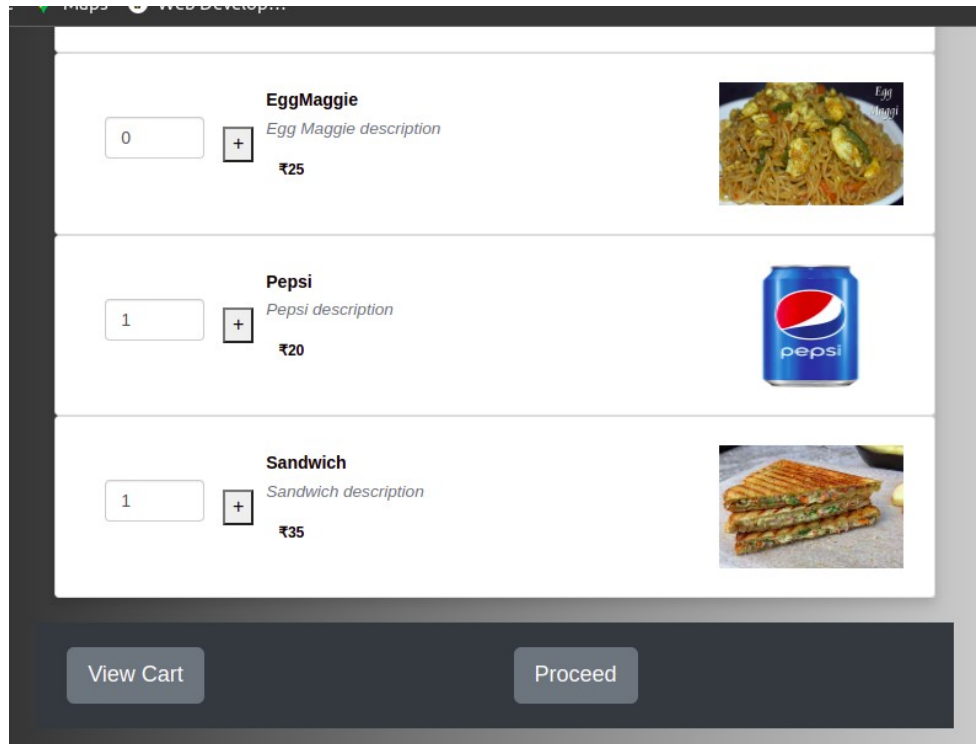
3. After we register new user we can see the details of our new user in our cloud database.

The image shows a database interface with a sidebar on the left containing "items" and "users" (the latter is selected). The main area displays a filter bar with the text "{ field: 'value' }" and buttons for "OPTIONS", "Apply", and "Reset". Below the filter bar, it says "QUERY RESULTS: 1-1 OF 1". The results are shown in a light blue box with the following JSON data:

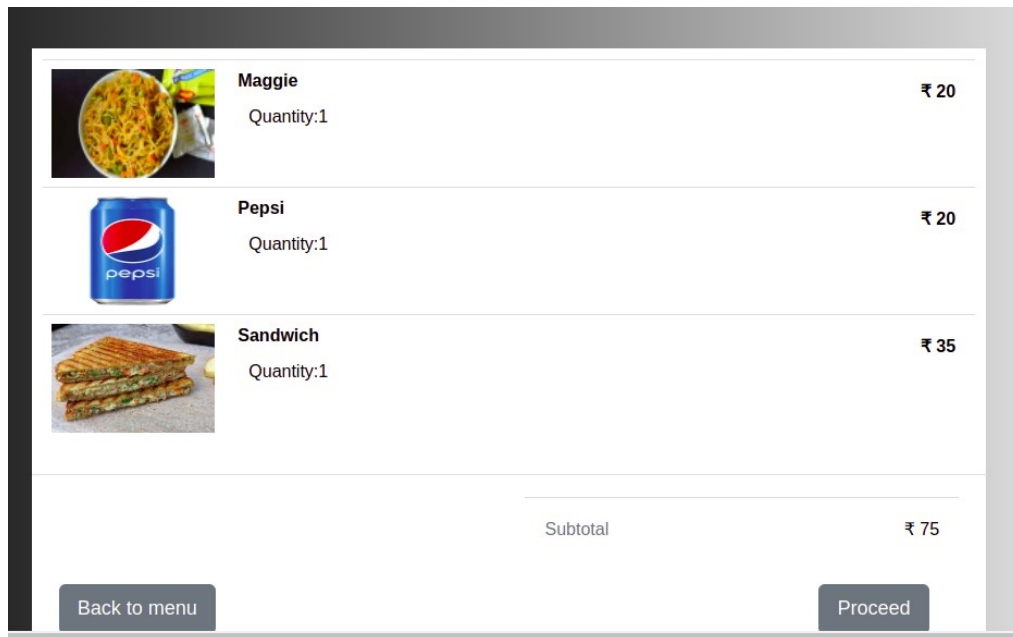
```
{
  "_id": "ObjectId('639c93b5a85ee50ab4ccd6cf')",
  "username": "1@2.com",
  "salt": "7b8374e272f3bf192a3d7ab76bd89ea26499096a2f7721c2ee01c4c891622db1",
  "hash": "c36af1b7c6d792d938aadba7b7b32188c97ef2bc09fe76e7f7dcd1414cd560add55840...",
  "__v": 0
}
```

At the bottom of the page, there is a footer with the text "System Status: All Good" and a copyright notice "©2022 MonooDB, Inc." followed by links for "Status", "Terms", "Privacy", "Atlas Blog", and "Contact Sales".

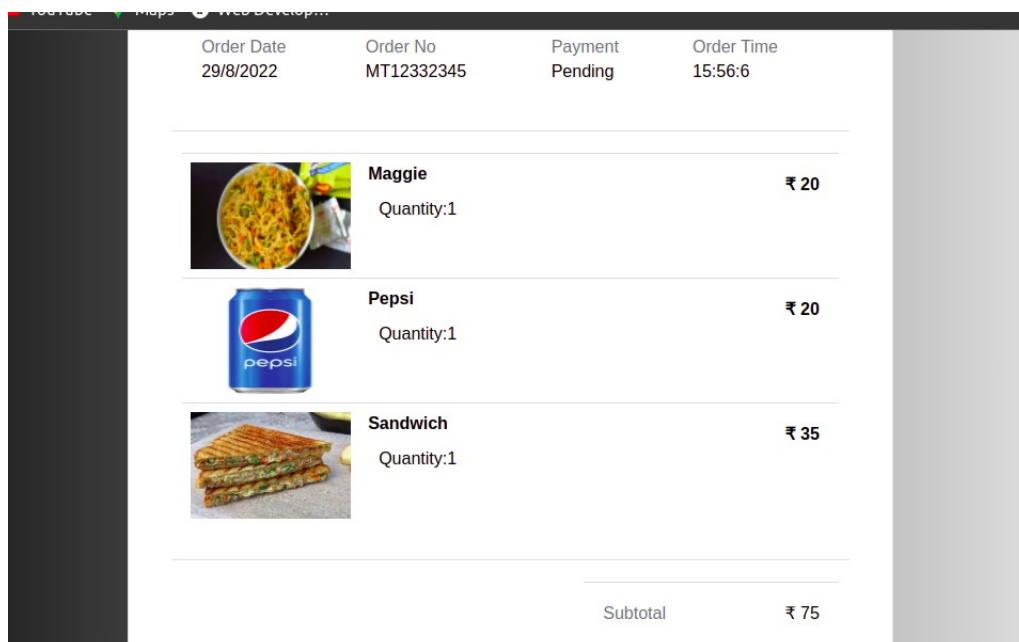
4. We can add items in cart by selecting quantity and clicking on + button to add the items in cart. Every user has a different cart assigned to him after registering on website.



5. User can view cart or place order.



6. By clicking on Proceed button order will be placed for user and new order will be entered in database.





## 7. In Database you can see placed order.

The screenshot displays the MongoDB Compass web interface. On the left sidebar, the 'foodDB' database is selected, and the 'orders' collection is highlighted. The main panel shows a single document from the 'orders' collection. Above the document, statistics are provided: STORAGE SIZE: 4KB, LOGICAL DATA SIZE: 277B, TOTAL DOCUMENTS: 1, and INDEXES TOTAL SIZE: 4KB. Navigation tabs include 'Find', 'Indexes', 'Schema Anti-Patterns 0', 'Aggregation', and 'Search Indexes'. A filter bar is present with the text '{ field: 'value' }'. The document content is as follows:

```
{
  "_id": ObjectId("630c943ea85ee50ab4ccd6f2"),
  "name": "i@2.com",
  "products": Array
    > 0: Object
      > 1: Object
        quantity: 1
        name: "Pepsi"
        price: 20
        _id: ObjectId("630c943ea85ee50ab4ccd6f4")
      > 2: Object
        price: 75
        v: 0
  }
}
```

At the bottom of the interface, the system status is indicated as 'All Good'.