2022 ISL GOOD TECH SCHOLAR PROGRAM

e-Ration Web Service

Submitted by:

PENDYALA VENKATA SAI VARDHIJA

MUHAMMED NADEEM MOHAMMED BASHEER

Mentors from IBM: Ms. Mamatha J. V., Mr. Sreekanth P.S.

ABSTRACT

Two-thirds of people in India live in poverty: 68.8% of the Indian population lives on less than $2 a day. Over 30% even have less than $1.25 per day available - they are considered extremely poor. It is apparent that their earnings are not enough for their daily existence. This is one of the main reasons that the State Governments have come up with the idea of food supplies for the their needy citizens. These foods are distributed raw to the citizens from centres such as “Ration Shops” or “Fair Price Shops” physically. Thus there are instances of corruptions and inadequate supply of food.

This is where an online system for allocating foods comes into the picture. Here, we present you a simple prototype of a web application where the citizens can directly know their allocated ration items and either accept the needed items or send it back to the government to avoid food wastage.

The app is developed with Atom and VScode IDE using MERN Stack and is compatible on all major updated web browsers; ie,Google Chrome Version 102.0.5005.62.

Development Environment

IDE: Atom, VScode
Language: HTML, CSS, JSX, NodeJS (v14.17.1), ExpressJS, and ReactJS

System: 10th Generation Intel®
Intel® Core™ i7-10510U Processor
1.8GHz quad-core
NVIDIA® GeForce® MX350
8GB 2666MHz DDR4 RAM
512GB SATA 3 SSD
OS: Windows 11 Home

Development Platform:
Google Chrome
Server: Local Host (Node)
Database: Mongo DB - Atlas

DEVELOPER END

USER END

Software Requirements

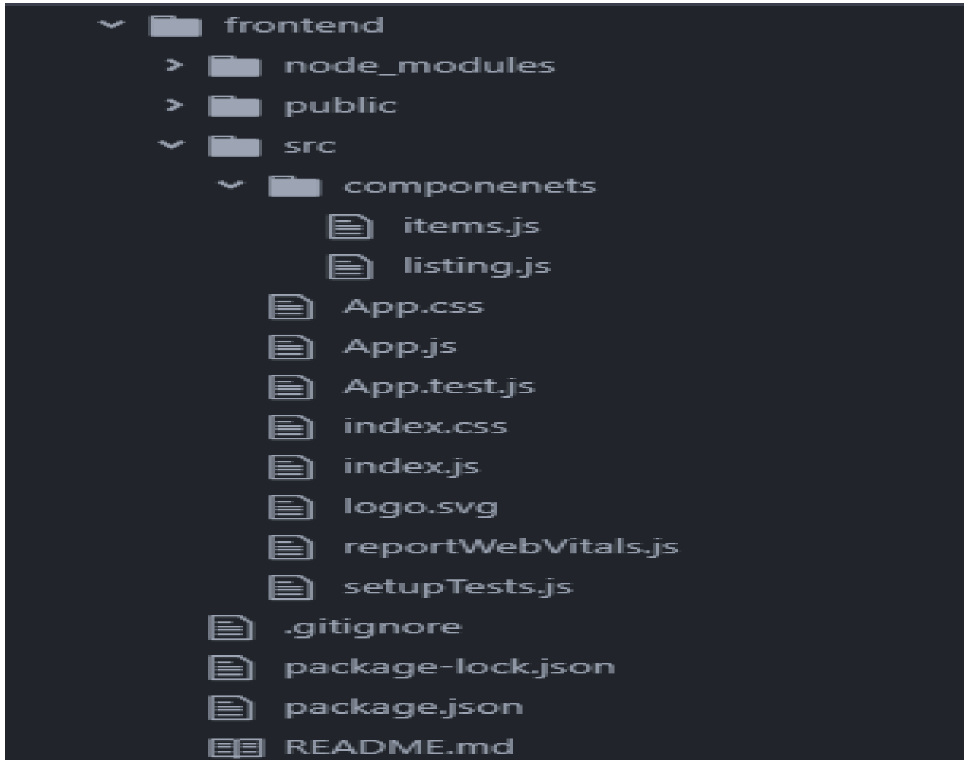
OS: Windows 11 Home

Hardware Requirements

Device: PC/ Smart Phone / Tablet
Processor: Intel Pentium 4 or later, or faster processor
RAM (Avg.): 20 Mb
Storage (Min.): 9 Mb
Connection: Internet Connection equired

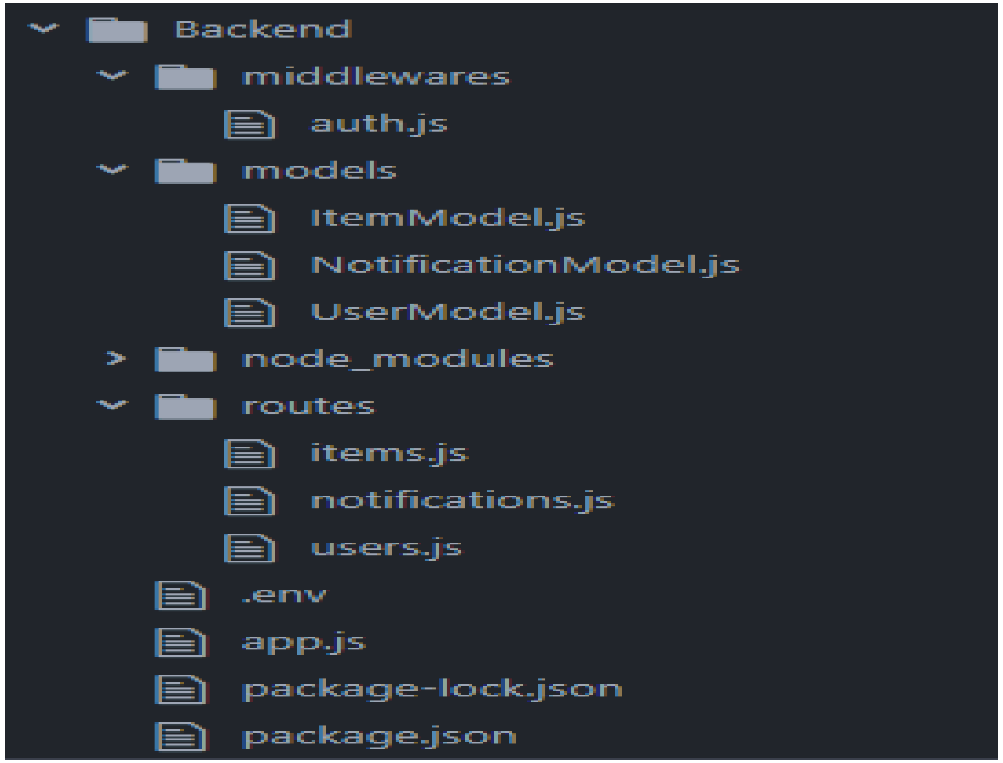
MODULES

Frontend



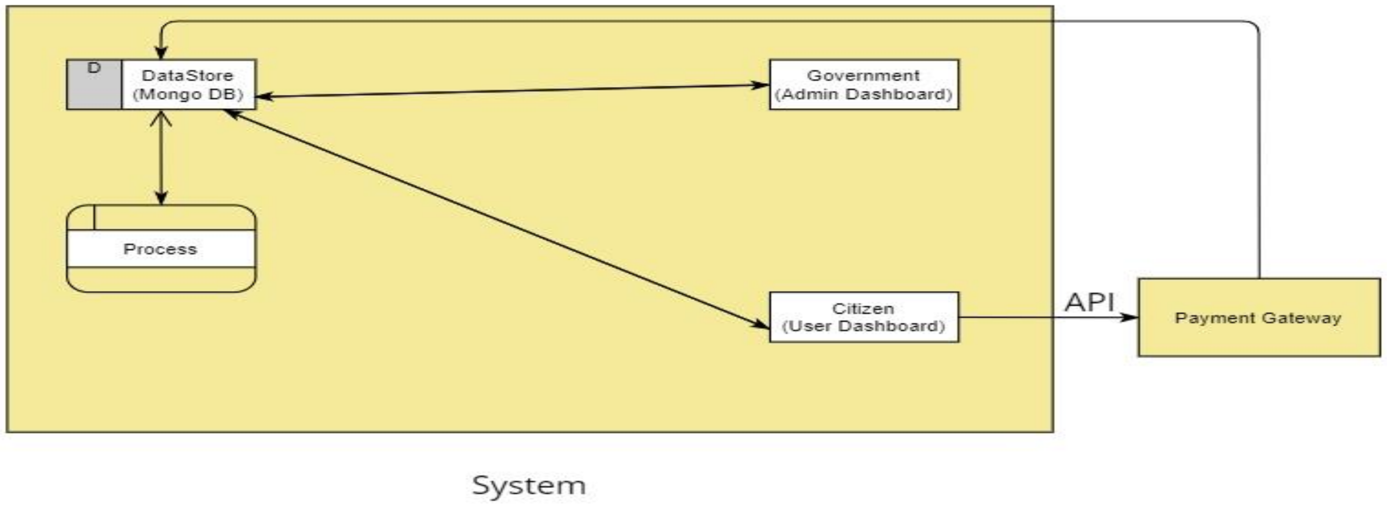
```
⌄ frontend
  › node_modules
  › public
  ⌄ src
    ⌄ componenets
        items.js
        listing.js
      App.css
      App.js
      App.test.js
      index.css
      index.js
      logo.svg
      reportWebVitals.js
      setupTests.js
    .gitignore
    package-lock.json
    package.json
    README.md
```

Backend

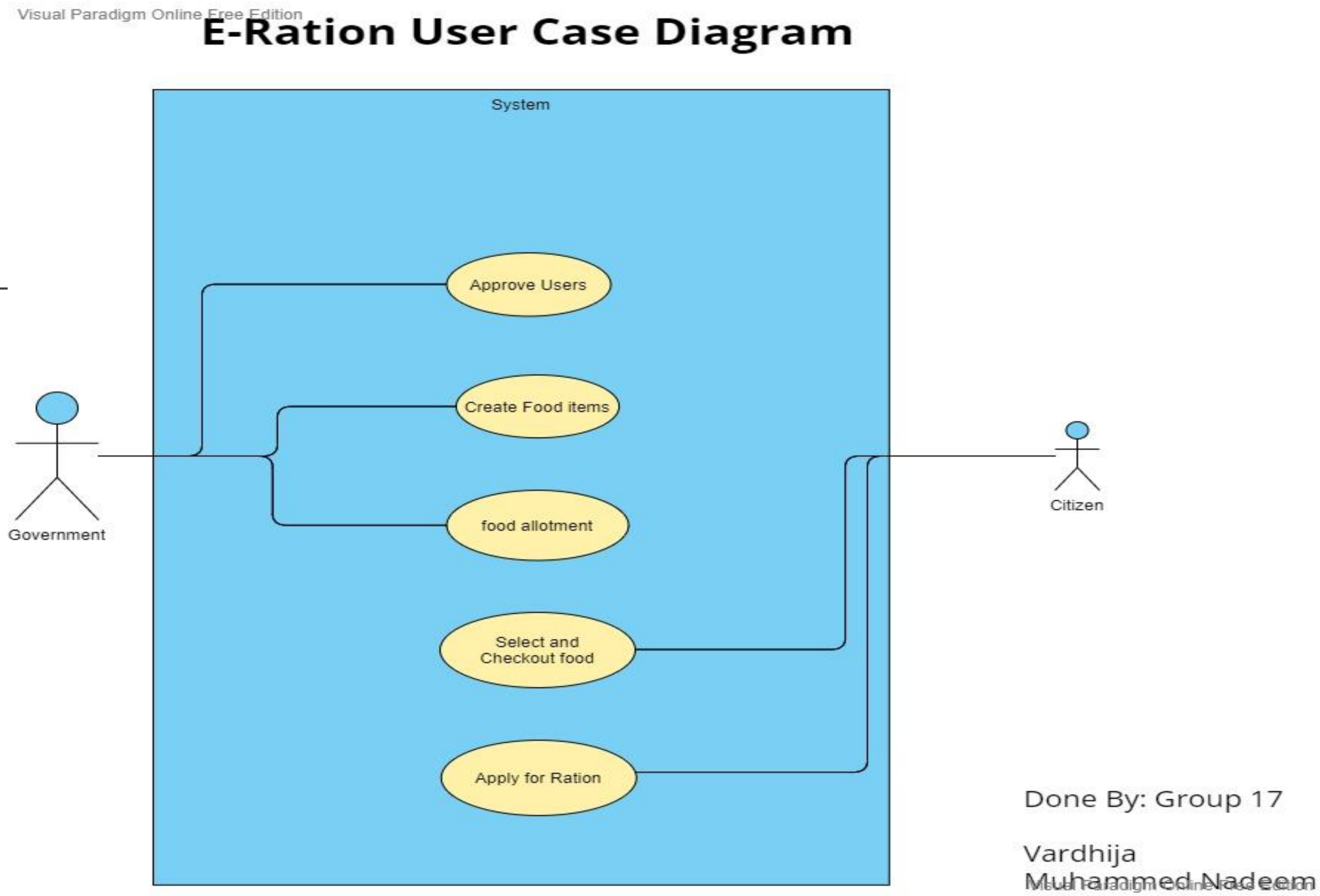


```
⌄ Backend
  ⌄ middlewares
      auth.js
  ⌄ models
      ItemModel.js
      NotificationModel.js
      UserModel.js
  › node_modules
  ⌄ routes
      items.js
      notifications.js
      users.js
    .env
    app.js
    package-lock.json
    package.json
```

e-Ration DFD



Data Flow



E-Ration User Case Diagram

Done By: Group 17

Vardhija
Muhammed Nadeem

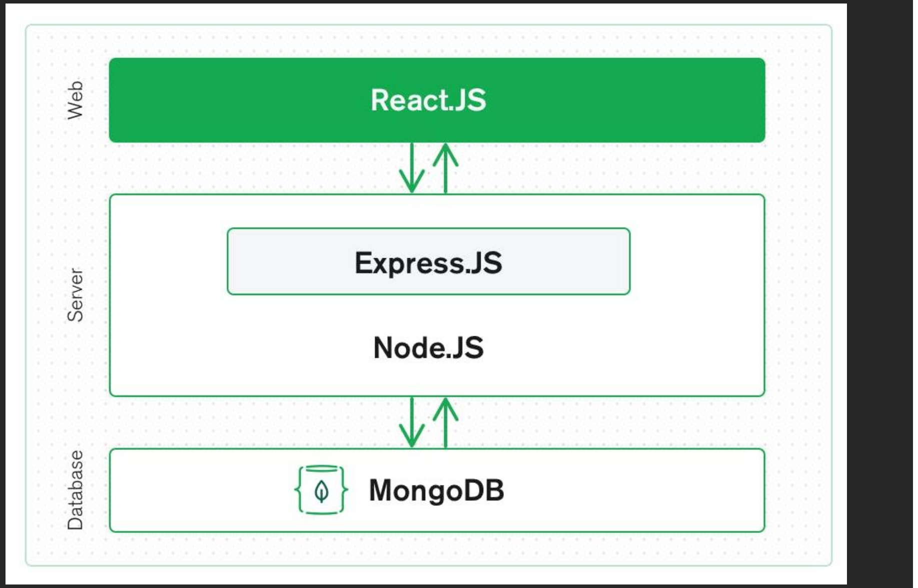
Use Case Diagram

Features

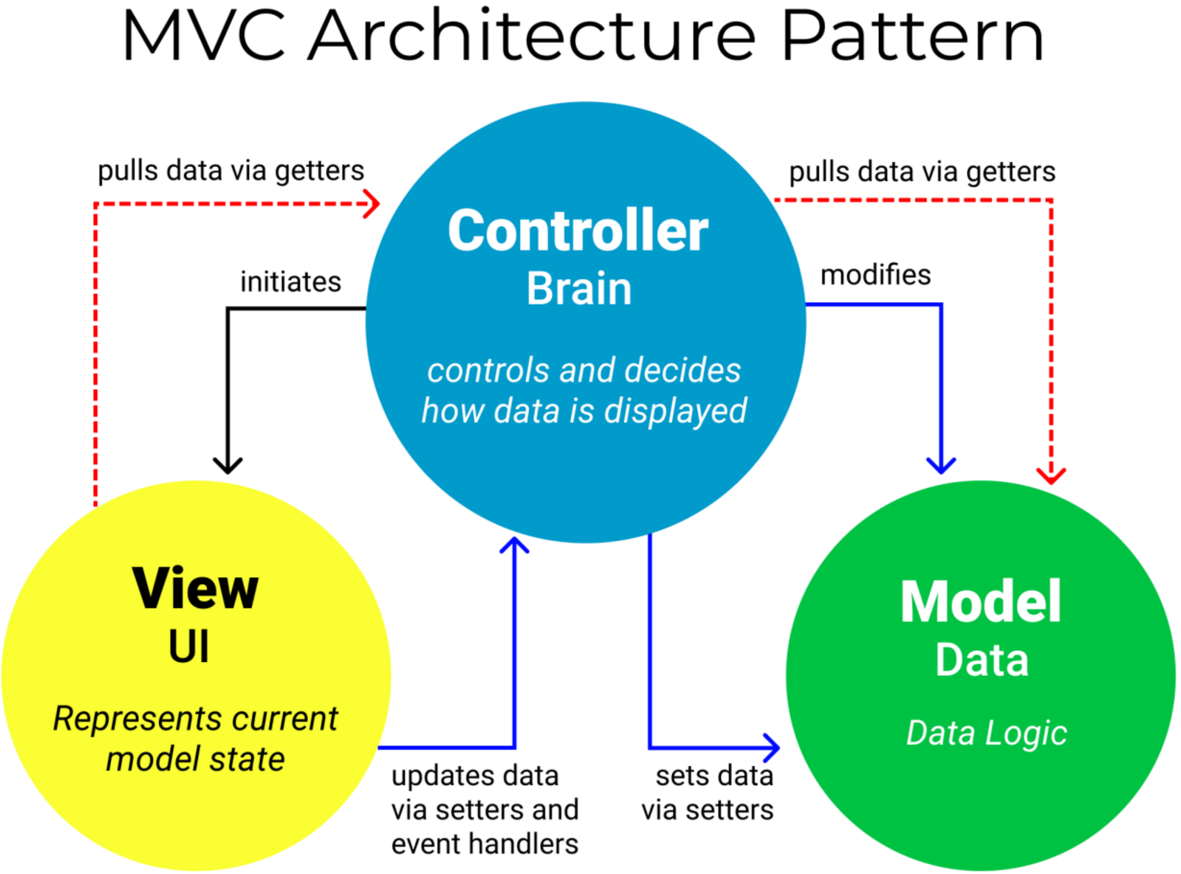
- ✓User authentication
- ✓User Authorization
- ✓Hashed and secured data
- ✓Helps to know the allotment of rations on time.
- ✓Cuts of intermediary costs.
- ✓Transactions can be more transparent as the its done directly.
- ✓Cloud storage is incorporated, so that the data is available anywhere at anytime in all devices.

ARCHITECTURE

MERN ARCHITECTURE



We follow MERN architecture which is an implementation of the very efficient MVC architecture pattern.

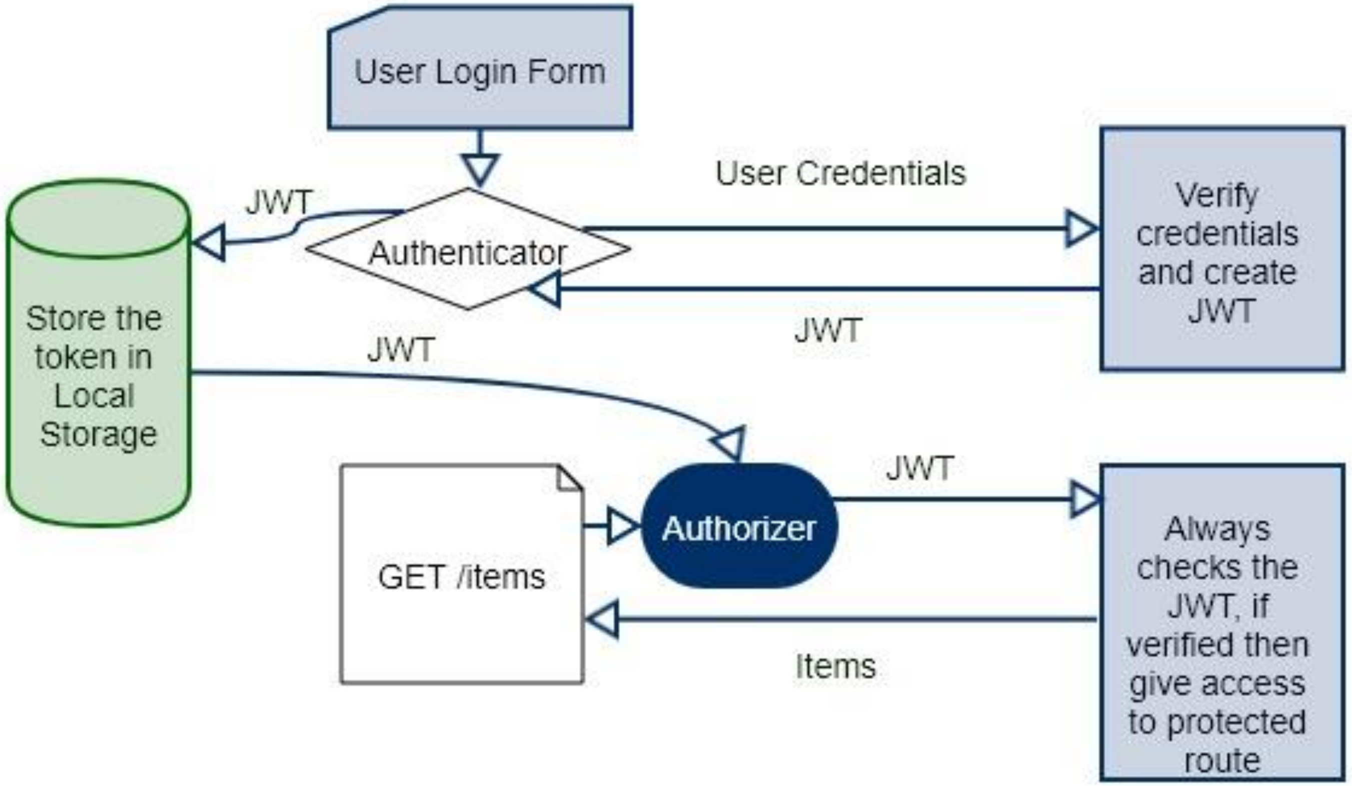


E-RATION WEB SERVICES

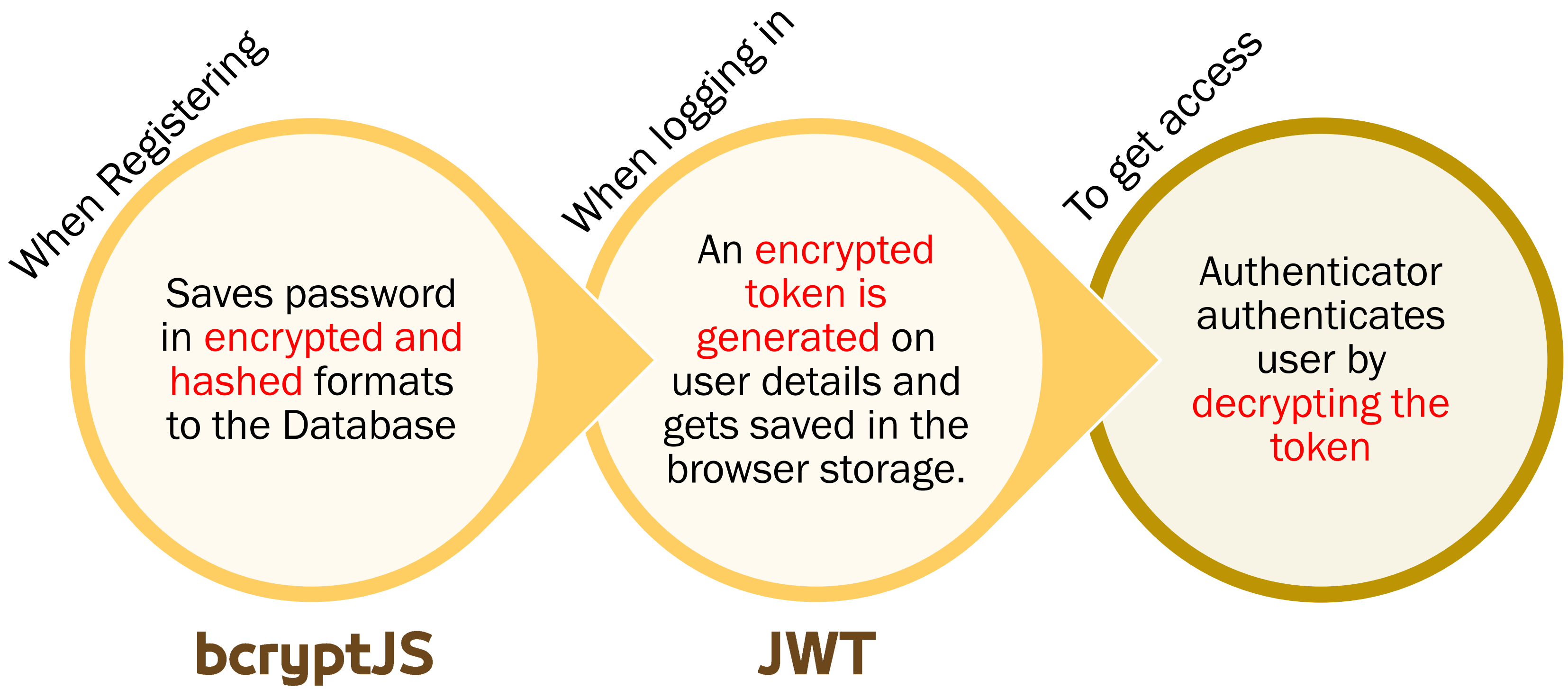
2022 ISL GOOD TECH SCHOLAR PROGRAM

SECURITY

Authorization & Authentication


User Login Form
User Credentials
JWT
Authenticator
Verify credentials and create JWT
JWT
Store the token in Local Storage
JWT
JWT
Authorizer
GET /items
Always checks the JWT, if verified then give access to protected route
Items

How it works…



When Registering

Saves password in encrypted and hashed formats to the Database

bcryptJS

When logging in

An encrypted token is generated on user details and gets saved in the browser storage.

JWT

To get access

Authenticator authenticates user by decrypting the token

USE CASES

Citizen Use Cases

1 SIGN UP & LOGIN

2 CHECK FOR ALLOTMENTS

3 SELECT NEEDED GROCERIES

4 Donate Others

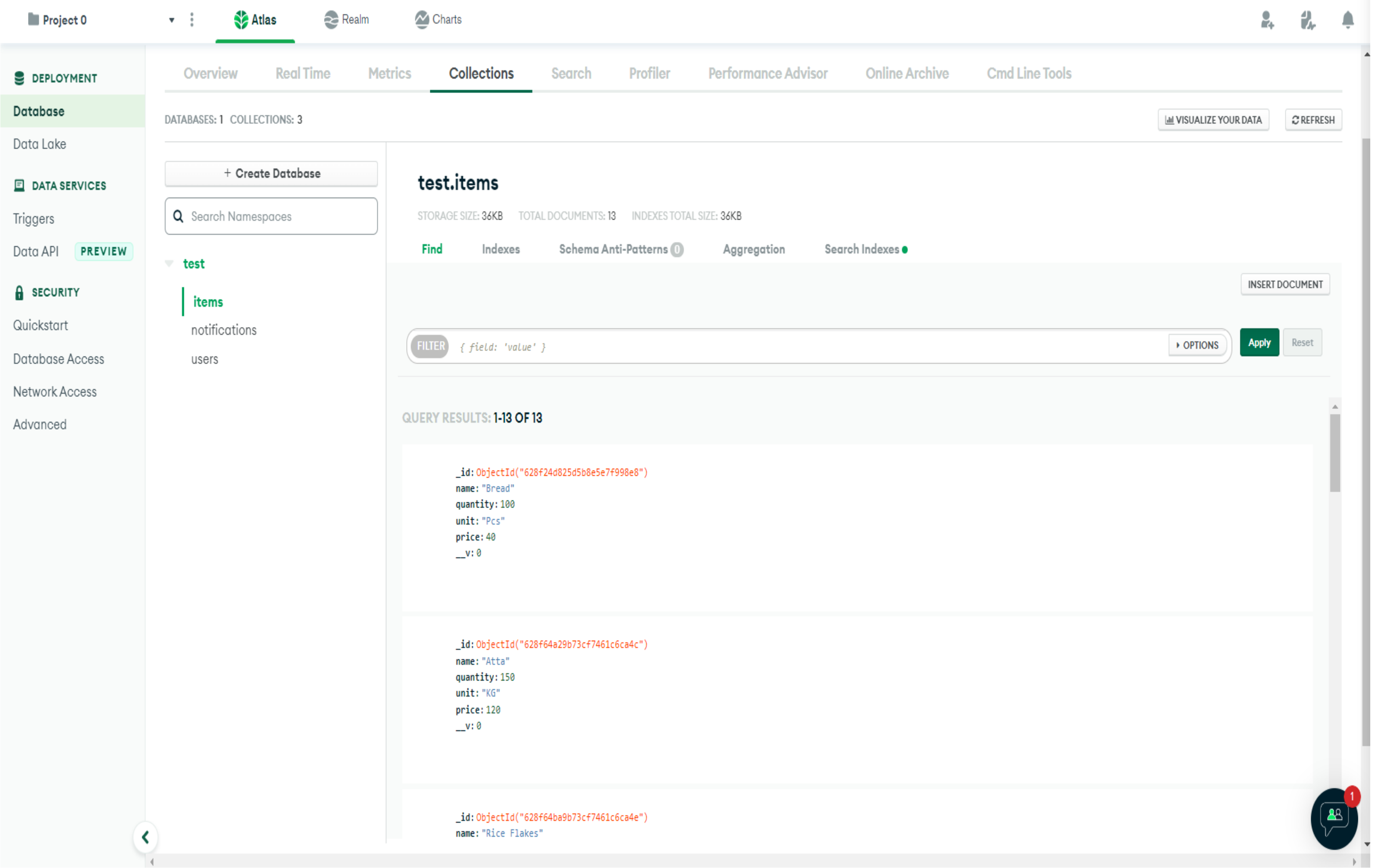
5 CHEKOUT

6 READ NOTIFICATIONS

Govt. Use Cases

1. APPROVE SIGNUP
2. CRUD ITEMS AND ALLOTMENT
3. VIEW ORDERS AND CHECKOUTS
4. SEE DONATIONS
5. USER MANAGEMENT
6. CRUD NOTIFICATIONS

Data Stored in MongoDB-Atlas



DATABASES: 1 COLLECTIONS: 3

test.items

STORAGE SIZE: 36KB TOTAL DOCUMENTS: 13 INDEXES TOTAL SIZE: 36KB

QUERY RESULTS: 1-13 OF 13

```
_id: ObjectId("628f24d825d5b8e5e7f998e8")
name: "Bread"
quantity: 100
unit: "Pcs"
price: 40
__v: 0

_id: ObjectId("628f64a29b73cf7461c6ca4c")
name: "Atta"
quantity: 150
unit: "KG"
price: 120
__v: 0

_id: ObjectId("628f64ba9b73cf7461c6ca4e")
name: "Rice Flakes"
```

References

For concept and security:
Baral, P. (2020). Role-based User Access Control in MERN Stack applications.

For standards and syntax:
MDN - https://developer.mozilla.org/
W3Schools
NodeJS - https://nodejs.org/en/docs/

For Debugging Errors:
Stack overflow - https://stackoverflow.com/questions/72396135/data-is-not-being-saved-in-the-database-when-added-from-the-frontend-mern

For Database Related queries:
MongoDB - https://www.mongodb.com/docs/

Contributors

Supporting And Mentoring:
Ms. Mamatha J. V.
Mr. Sreekanth P.S.
(Mentors from IBM)

Developers:
Ms. Vardhija
Mr. Muhammed Nadeem

Opportunity

2022 ISL Good Tech Scholar Program - IBM

Thank You!