Scratch Card Generation Application

Problem Definition:

Create REST API's to allow generation of scratch cards with random amount which can be assigned to valid users in the system.

Entity Definitions:

1. User

- id
- userEmail
- firstName
- lastName
- isActive

2. ScratchCard

- id
- discountAmount
- expiryDate
- isScratched
- isActive

3. Transaction

- id
- dateOfTransaction
- transactionAmount
- userId
- scratchCardId

NOTE: All id fields should be GUID, Email address should be unique.

BE Tasks:

1. ## CRUD USER API
   Create API endpoints to Create/Read/Update/Delete (CRUD) **User**

2. ## GENERATE SCRATCH CARD API
   API end point to generate **N** new scratch cards and set the expiry date to be **5 days** after the day of generation. Where **N** (numberOfScratchCards) is the parameter to be passed to the API endpoint. **discountAmount** should be a random value between 0 to 1000 INR
   If number of unused scratch cards are same or more than the **N**, then do not create any scratch cards. Return a message saying
   "X number of active scratch cards still exists in the DB. Did not create any new scratch cards"

3. ## ADD TRANSCATION API
   Write API endpoint to add transactions and associate an user and unused **Scratch Card** to each transaction.
   Add Server side validations to make sure the data send to the API endpoint is valid data.
   Example: Valid userId, Unused scratchCardId, not expired scratch card.

4. ## GET UN-USED SCRATCH CARDS
   Write API endpoint to get all unused scratch cards

5. ## GET TRANACTIONS API
   Write API endpoint to get **transactions** based on below filters:

   - dateOfTransaction
   - userId
   - transactionAmount

6. ## ADDITIONAL API ENDPOINTS
   Create any additional endpoints you think will be necessary for the Frontend developer to invoke the above API endpoints with proper data.
   In the UI Section you will find the screenshots of the Application UI.

NOTE:

- **Work on Each task individually, don't try to complete all the tasks together.**
- **You are advised to commit your changes once each task is complete and push to GitHub.**
  While committing your changes put proper commit messages.
  Example: If you are working on "task 1. CRUD USER API" and you have completed writing the code for adding user. Then the commit message should be
  "TASK-1::CRUD USER API::Wrote API to Create User"
  Format is shown below:
  "TASK-<<TASK_NUMBER>>-<<TASK-NAME>>::<<COMMIT_MSG>>"

Expected output:

- All the expected API Endpoints should be documented in the Postman Collection.
- Invoking the Postman API endpoint should give the expected results.

Tech Stack:

The following tech stack should be used to implement the task.

- **Framework :** NodeJS
- **Database :** MongoDB
- **API Documentation :** Postman

Software Requirements:

- NodeJS Latest stable version
- MongoDB
- Docker (optional, if you don't want to install mongodb server)

Task Resources:

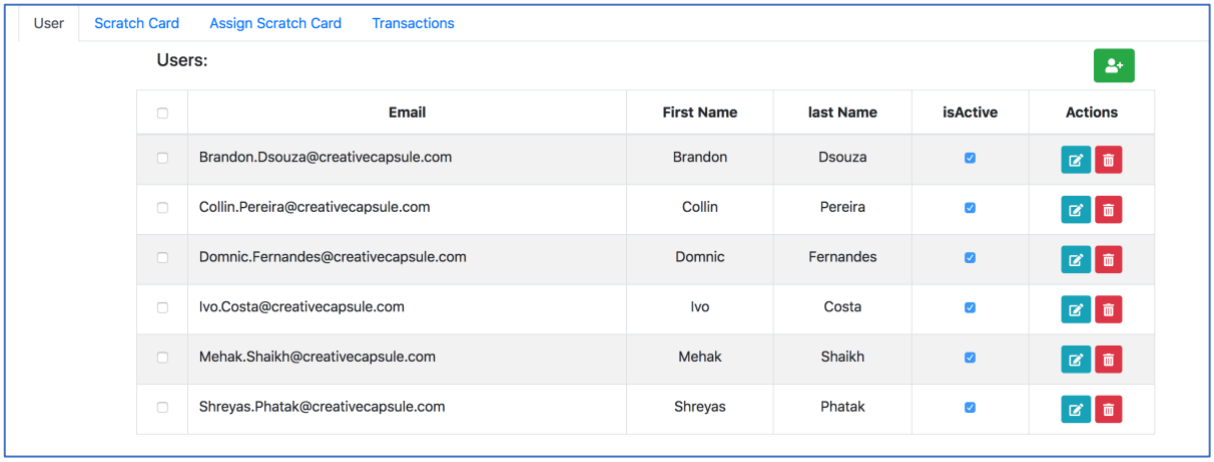
- Backend base application is created and added in the folder **scratchCards-BE.**
  It has basic code to connect to the mongoDb and display a hello world route.
- HTML assets are provided for implementing the UI. So that you have an idea of how the Front end will consume the application. All assets are kept in **ui-assets** folder

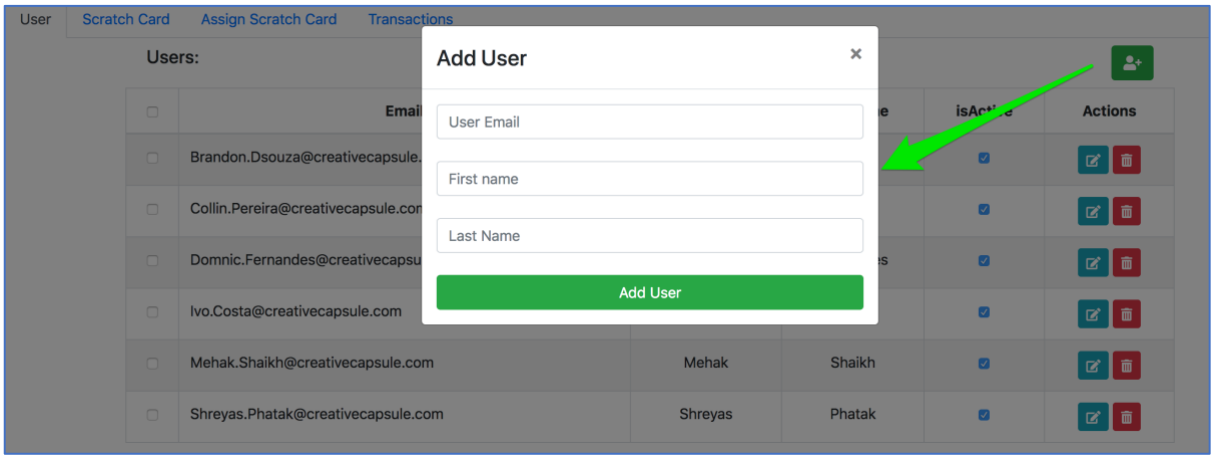
UI Screens:

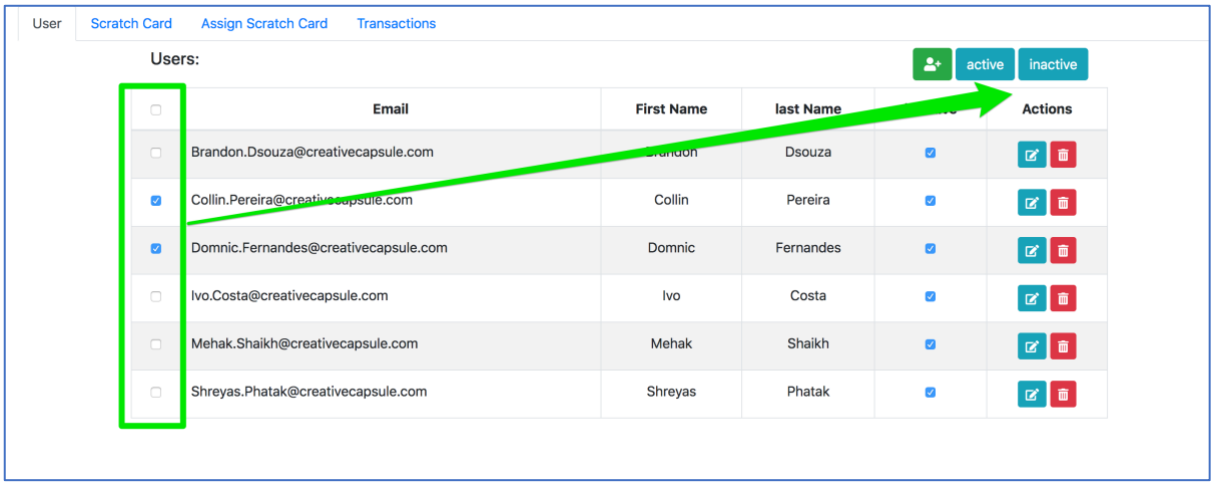
Below are the UI Screens which will consume the API Endpoints.

Note: The below screenshots are your reference only. You don't have to work on the UI Development

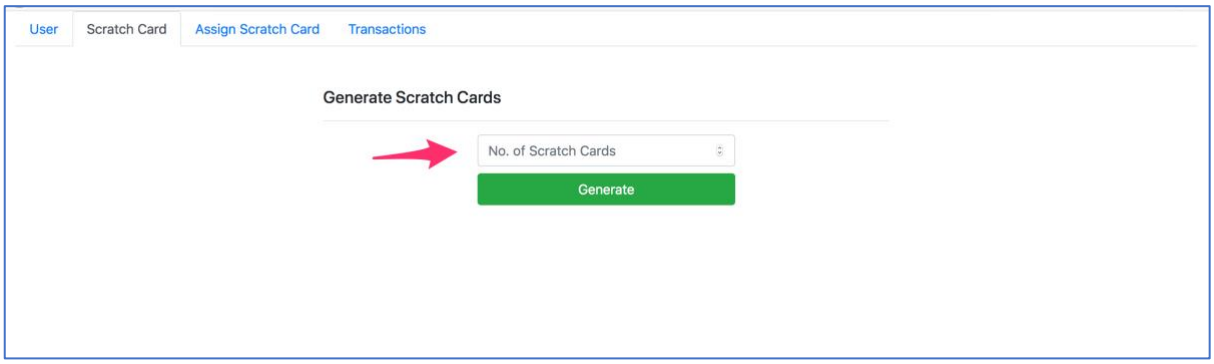
1. *UI Screens for CRUD operations on User Entity*



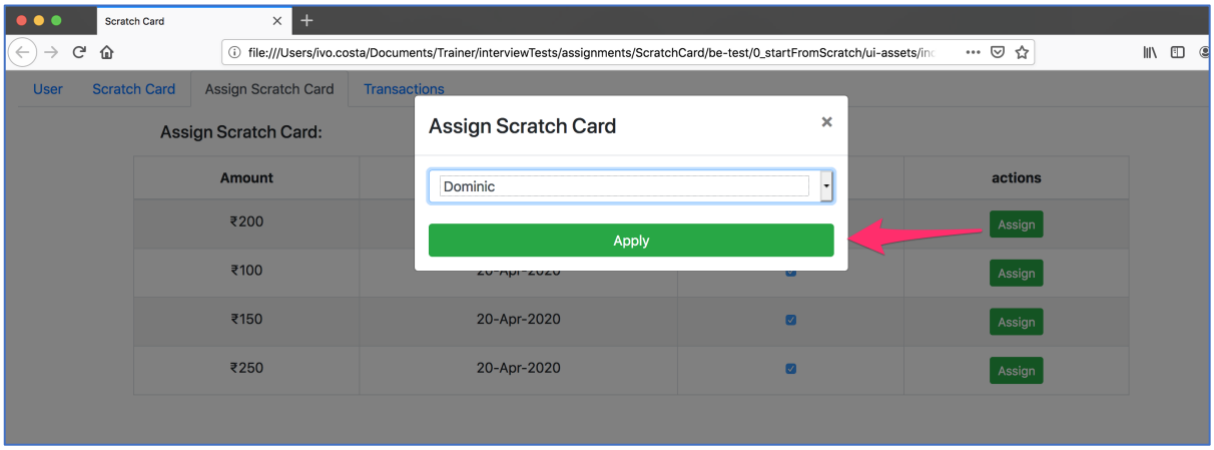




2. *UI Screen to generate scratch cards*



3. *UI screens to assign user to a scratch cards*



4. *UI Screen to display Transaction filters*