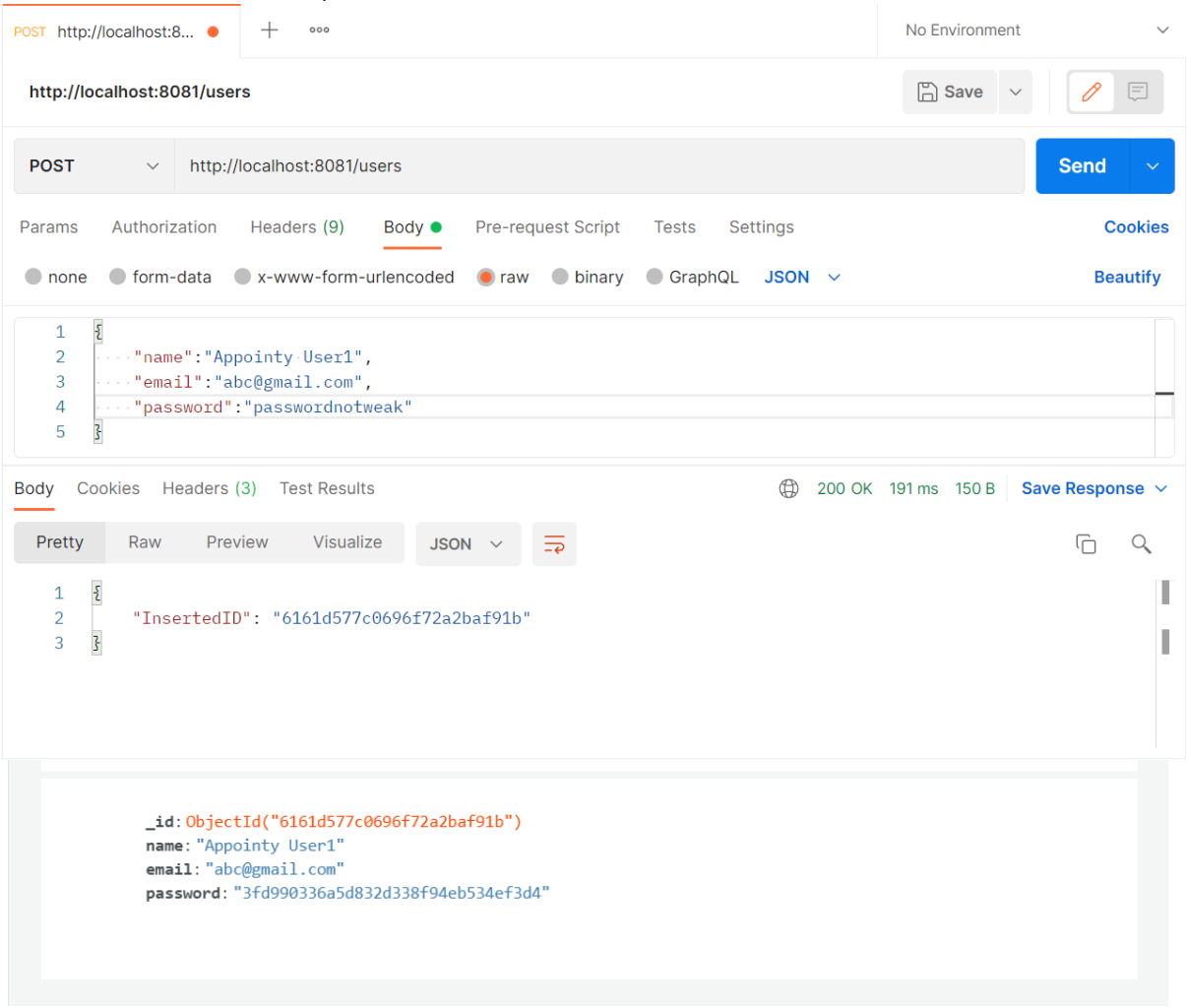
Task | Instagram Backend API

Presented by: Harsh Vardhan(19BCE1661)

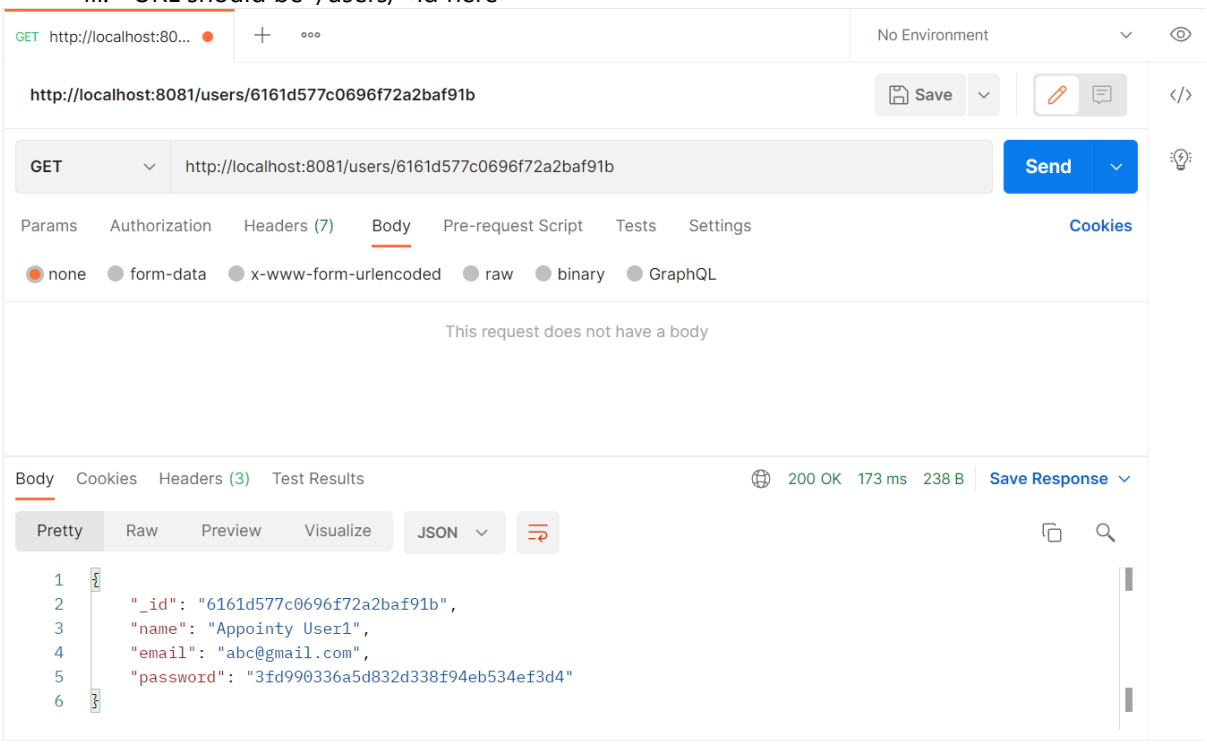
College: VIT,Chennai.

Basic Tasks:

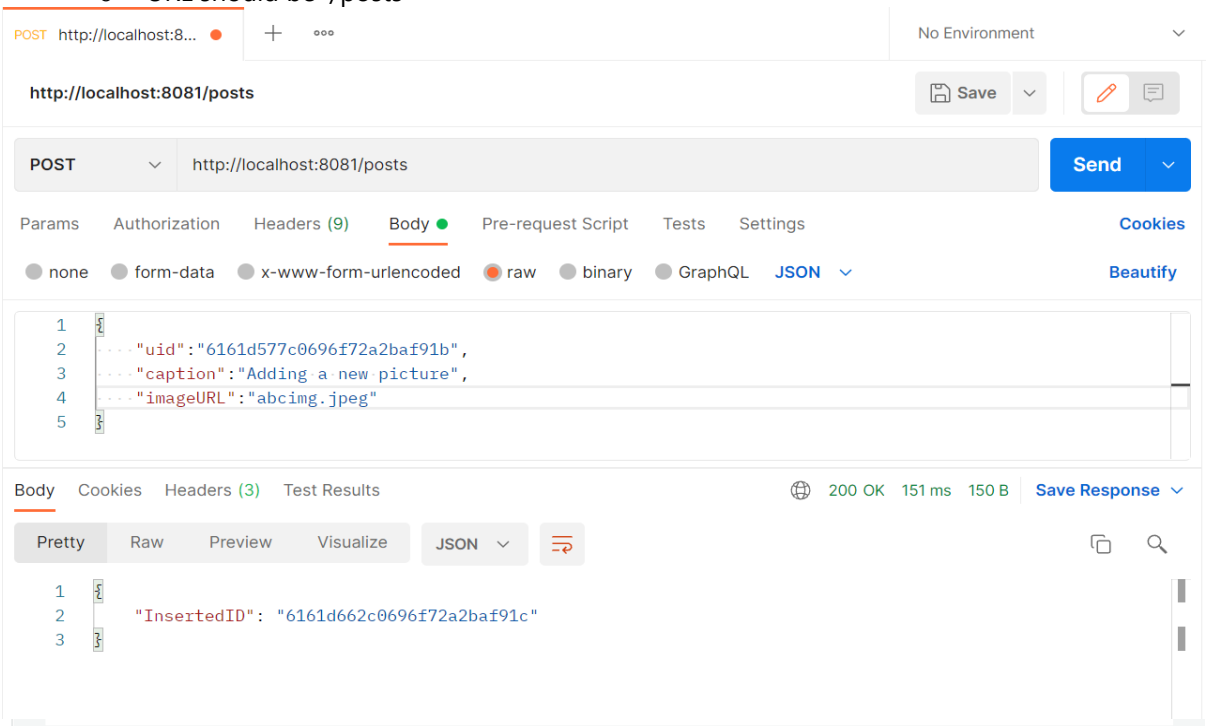
1. You are required to Design and Develop an HTTP JSON API capable of the following operations,
   - Create an User
     a. Should be a POST request
     b. Use JSON request body
     c. URL should be '/users'



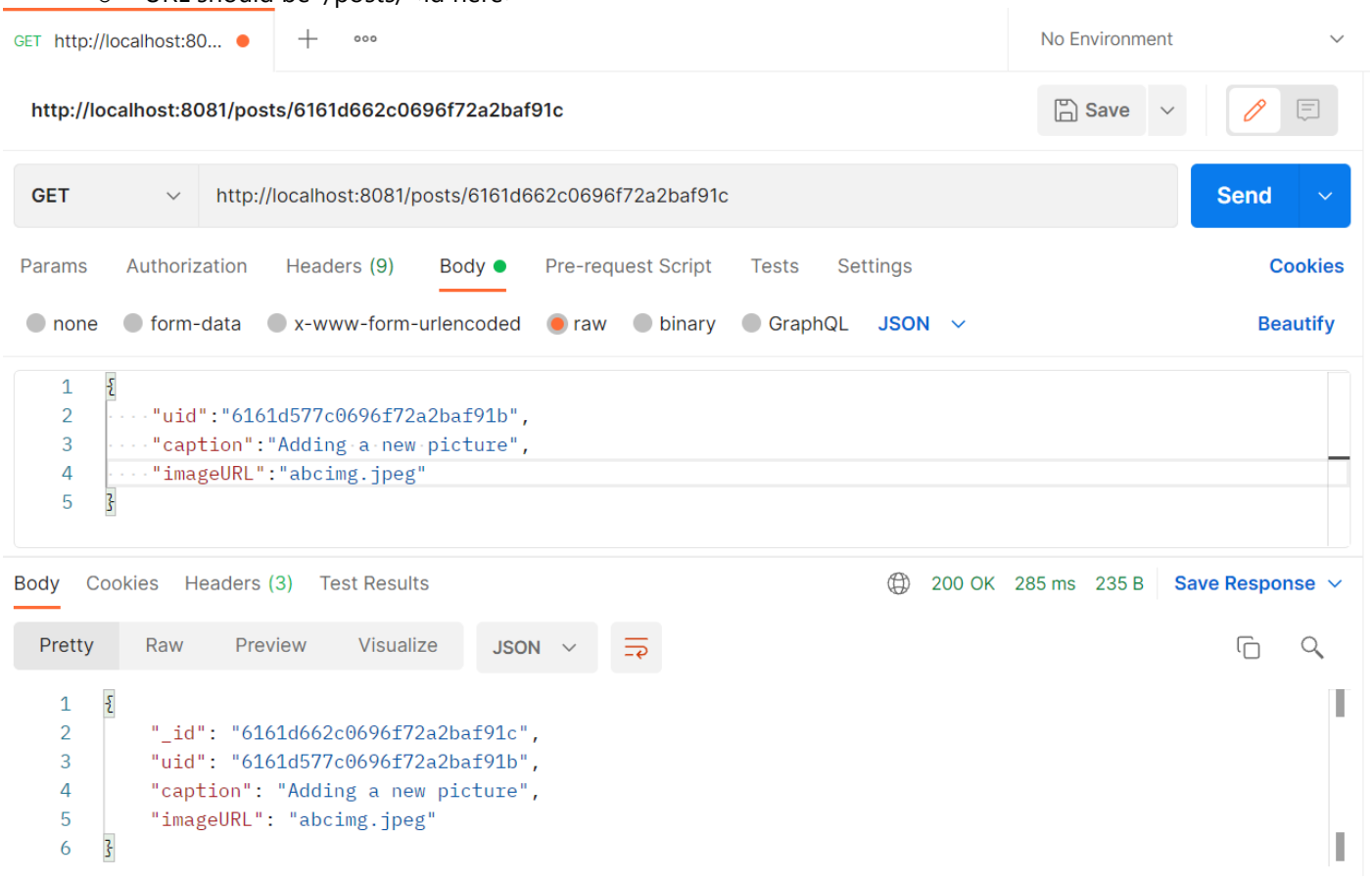
- Get a user using id
  i. Should be a GET request
  ii. Id should be in the url parameter
  iii. URL should be '/users/<id here>



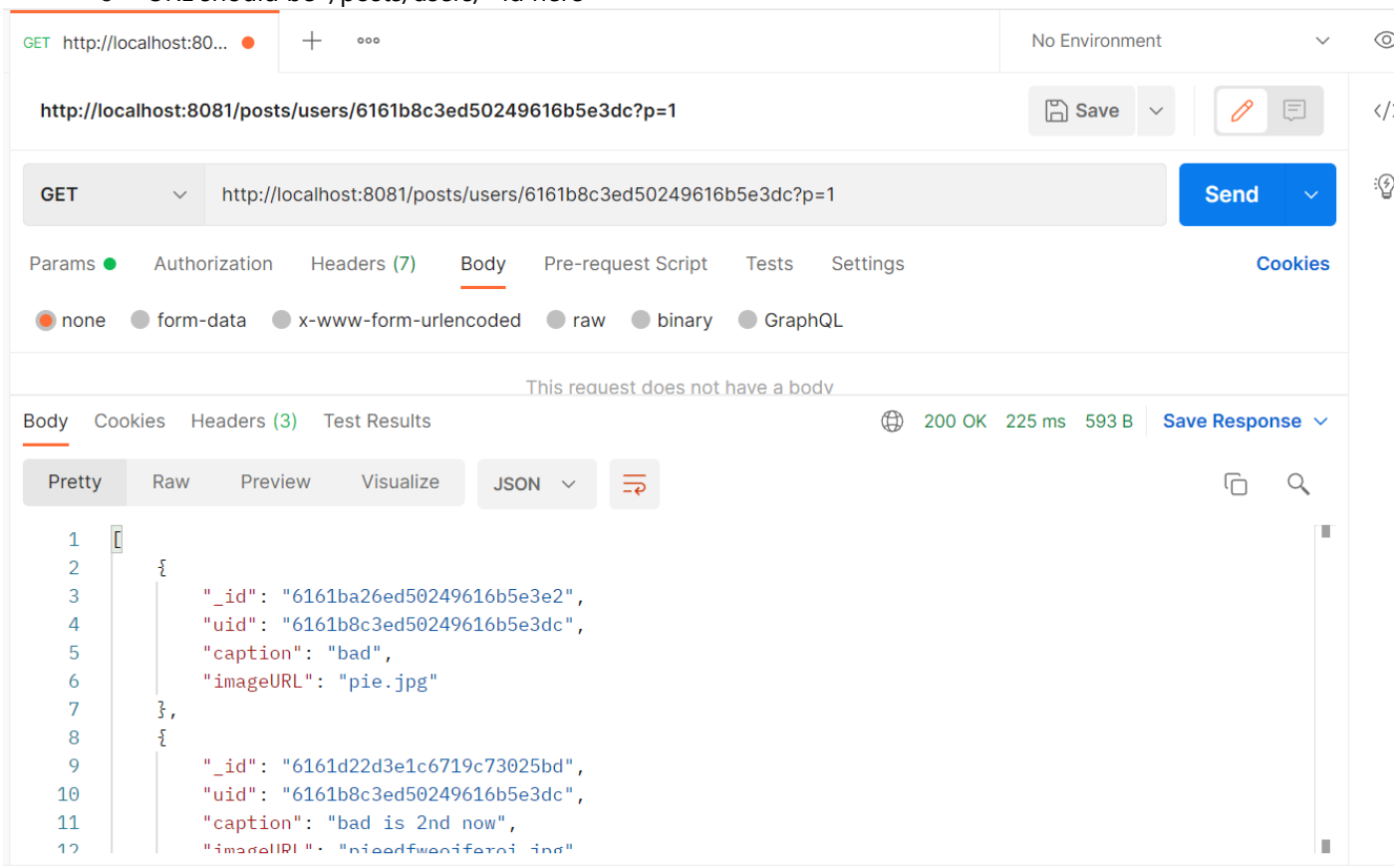
- Create a Post
  - Should be a POST request
  - Use JSON request body
  - URL should be '/posts'



- Get a Post using id
  - Should be a GET request
  - Id should be in the url parameter
  - URL should be '/posts/<id here>'



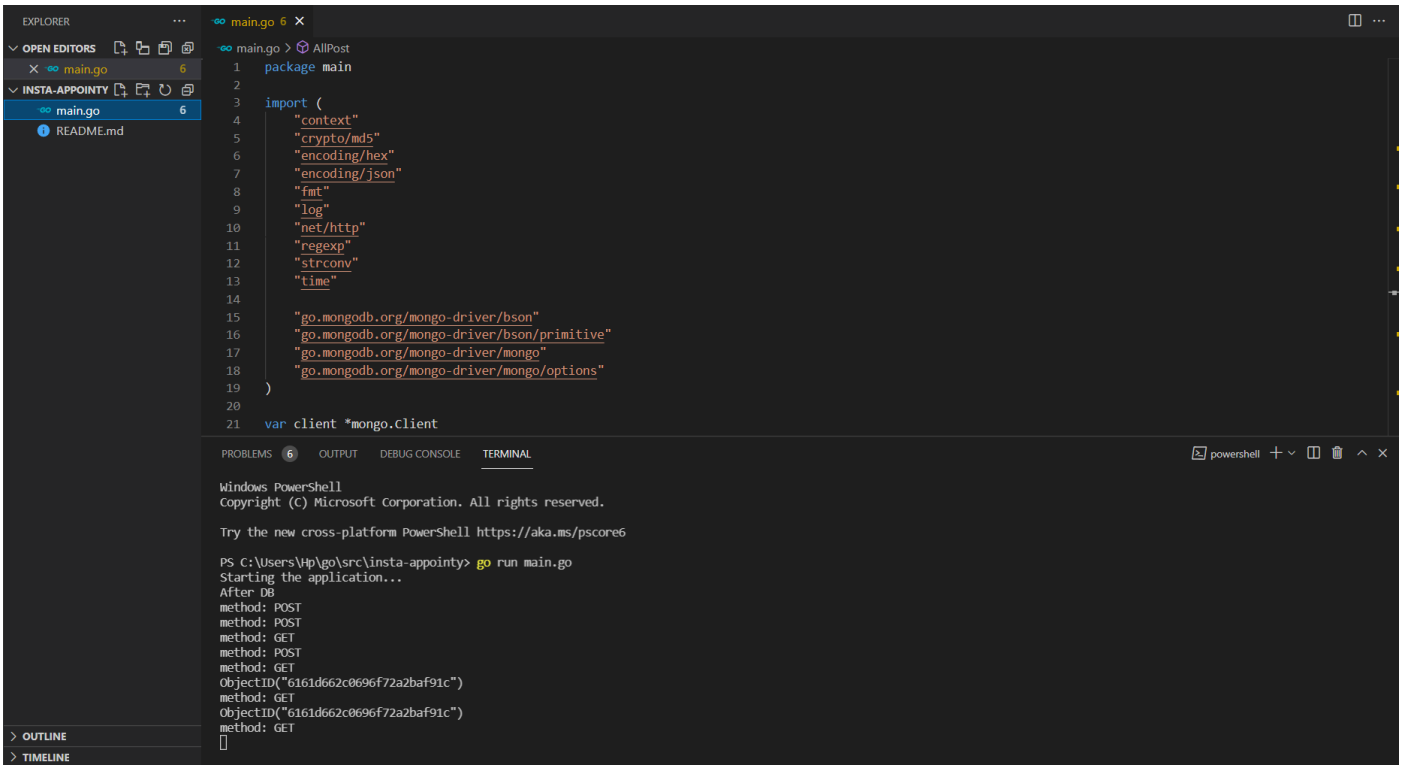
- List all posts of a user (**With Pagination applied**)
  - Should be a GET request
  - URL should be '/posts/users/<Id here>'



Additional Constraints/Requirements:

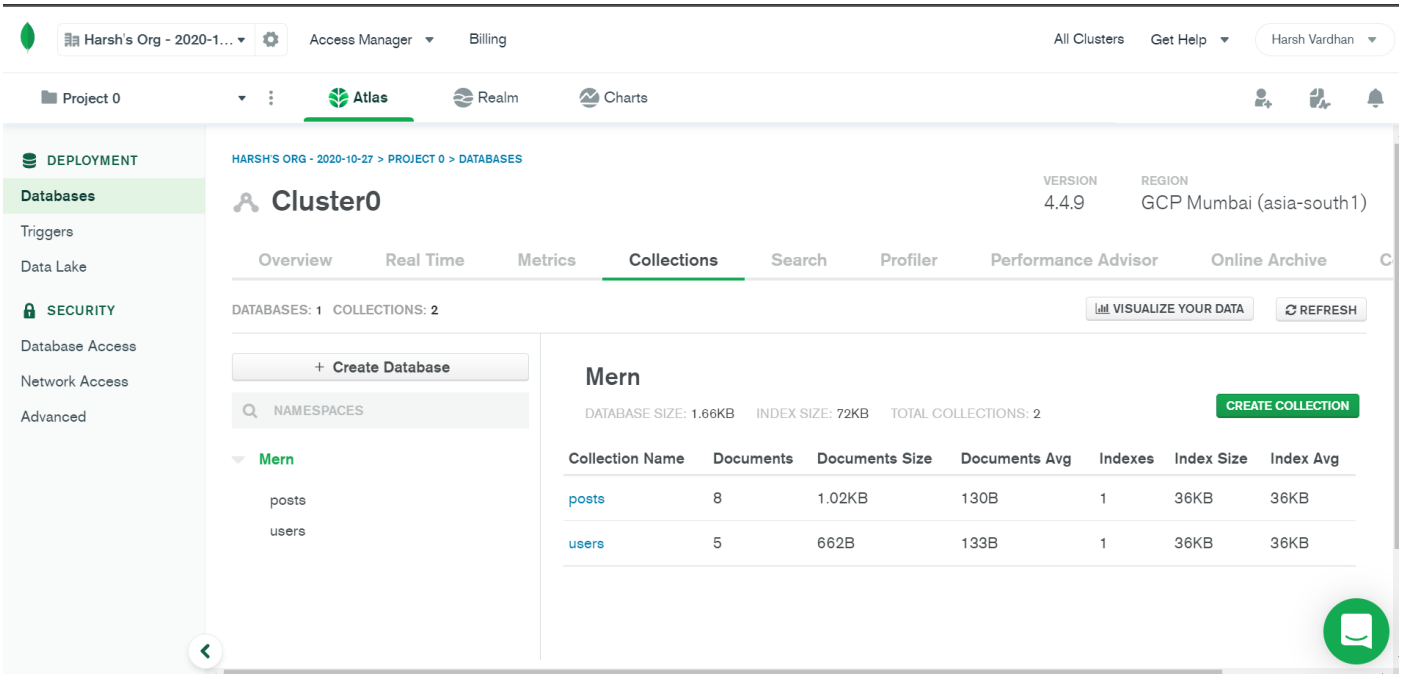
- The API should be developed using Go.

  The link to the whole code of API is attached in form all the packages have been used under the restrictions that was listed.



- MongoDB should be used for storage.

  Mongo DB ATLAS online client was used.



- Only packages/libraries listed here and here can be used.

  The link to the whole code of API is attached in form all the packages have been used under the restrictions that was listed.

  The List of Packages used can be seen Below

```go
package main

import (
    "context"
    "crypto/md5"
    "encoding/hex"
    "encoding/json"
    "fmt"
    "log"
    "net/http"
    "regexp"
    "strconv"
    "time"

    "go.mongodb.org/mongo-driver/bson"
    "go.mongodb.org/mongo-driver/bson/primitive"
    "go.mongodb.org/mongo-driver/mongo"
    "go.mongodb.org/mongo-driver/mongo/options"
)
```

Other Scoring Factors Kept in Mind

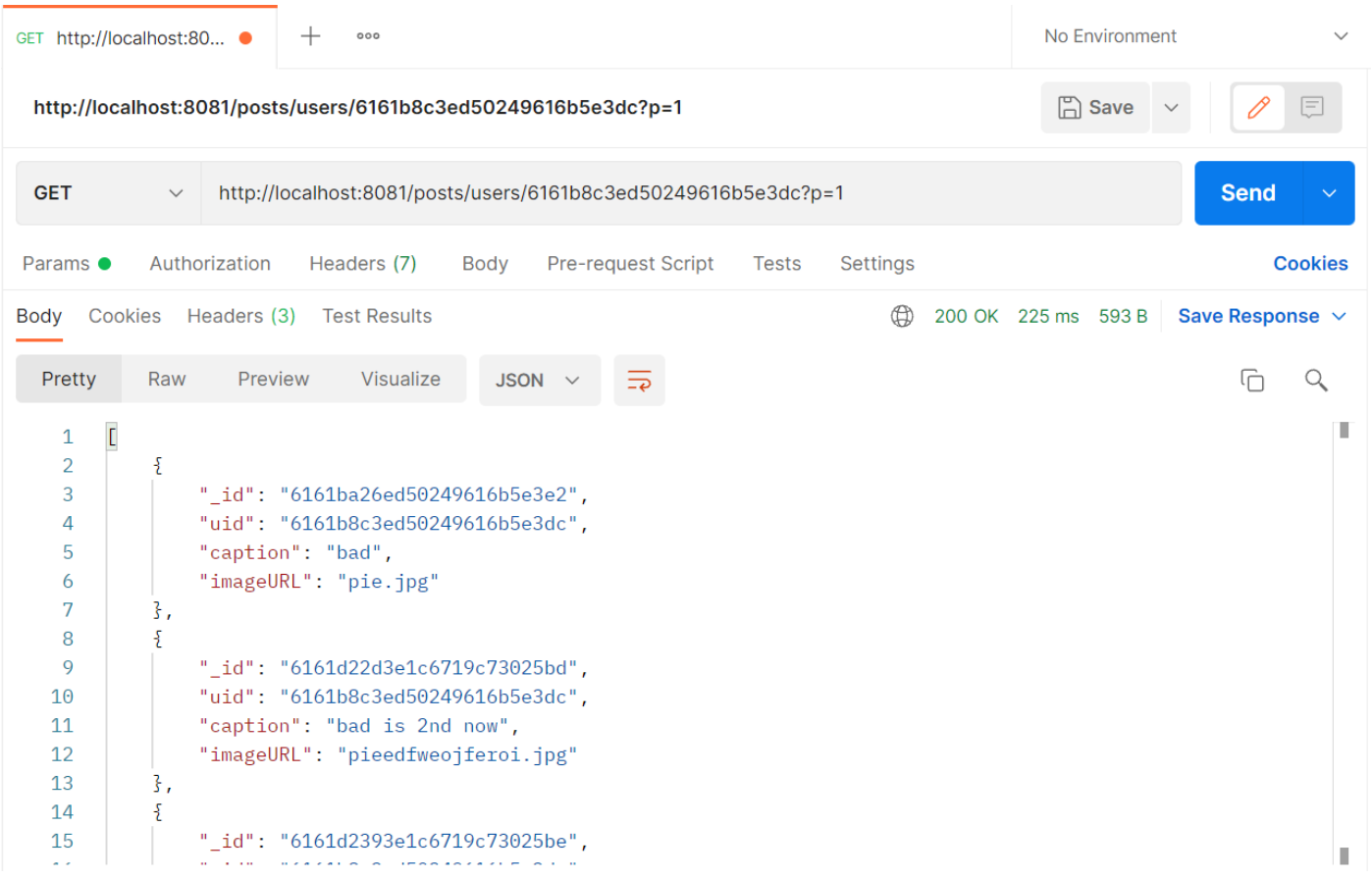
1. **Secure Passwords**

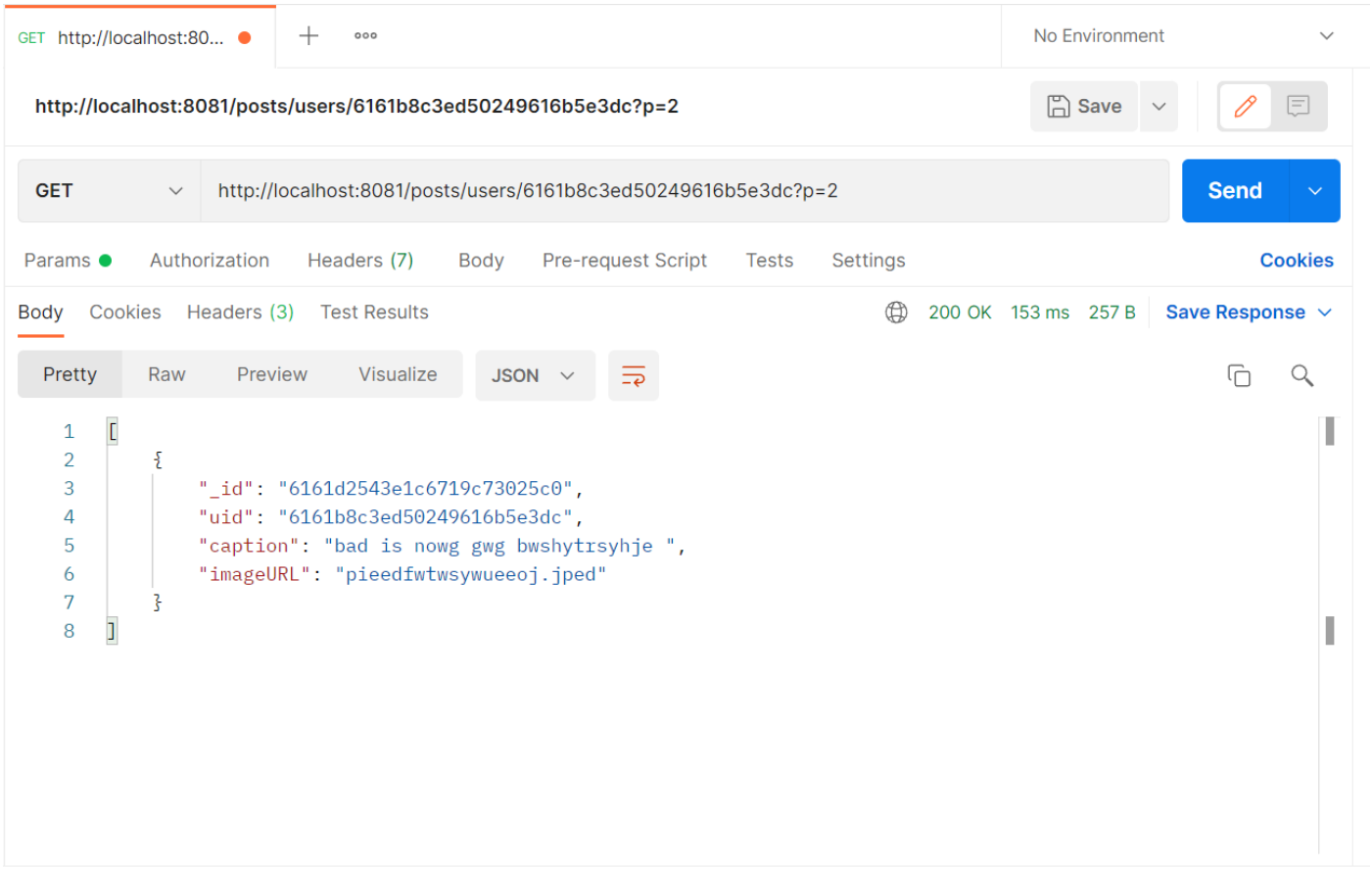
   The plan was to use **bcrypt** for securing and hashing the password but the best Option in the listed package seemed to be **md5.**

```go
func addUsers(w http.ResponseWriter, r *http.Request) {
    fmt.Println("method:", r.Method)
    if r.Method == http.MethodPost {
        if err := r.ParseForm(); err != nil {
            fmt.Fprintf(w, "ParseForm() err: %v", err)
            return
        }
        w.Header().Add("content-type", "application/json")
        var user User
        json.NewDecoder(r.Body).Decode(&user)
        md5HashInBytes := md5.Sum([]byte(user.Password))
        md5HashInString := hex.EncodeToString(md5HashInBytes[:])
        u1 := bson.D{{Key: "name", Value: user.Name},
            {Key: "email", Value: user.Email}, {Key: "password", Value: md5HashInString}}
        collection := client.Database("Mern").Collection("users")
        ctx, _ := context.WithTimeout(context.Background(), 10*time.Second)
        result, _ := collection.InsertOne(ctx, u1)
        json.NewEncoder(w).Encode(result)
    }
}
```

2. **Add pagination to the list endpoint**

   Pagination added for the All posts retrieval API endpoint. With ever page returning 4 data at a time. The page number is accepted as query parameter.





In the above example 5 posts of userid=" 6161b8c3ed50249616b5e3dc" were sent in 2 pages 4 in 1st page and 1 in 2nd page.

3. **Add unit tests**
   This task was incomplete bcs of time Constraints.
4. **Completion Percentage**
   All the Endpoints are UP and working well so the completion is 100%.