

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'

The screenshot displays a REST client interface with the following details:

- Request:**
 - Method: POST
 - URL: http://localhost:8081/users
 - Body (JSON):

```
{  "name": "Appointy User1",  "email": "abc@gmail.com",  "password": "passwordnotweak"}
```
- Response:**
 - Status: 200 OK
 - Time: 191 ms
 - Size: 150 B
 - Body (JSON):

```
{  "_id": ObjectId("6161d577c0696f72a2baf91b"),  "name": "Appointy User1",  "email": "abc@gmail.com",  "password": "3fd990336a5d832d338f94eb534ef3d4"}
```

- 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>'

GET http://localhost:8081/users/6161d577c0696f72a2baf91b

Params Authorization Headers (7) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

Body Cookies Headers (3) Test Results 200 OK 173 ms 238 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "_id": "6161d577c0696f72a2baf91b",
3   "name": "Appointy User1",
4   "email": "abc@gmail.com",
5   "password": "3fd990336a5d832d338f94eb534ef3d4"
6 }
  
```

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

POST http://localhost:8081/posts

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1 {
2   "uid": "6161d577c0696f72a2baf91b",
3   "caption": "Adding a new picture",
4   "imageURL": "abcmimg.jpeg"
5 }
  
```

Body Cookies Headers (3) Test Results 200 OK 151 ms 150 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "InsertedID": "6161d662c0696f72a2baf91c"
3 }
  
```

```

_id: ObjectId("6161d662c0696f72a2baf91c")
uid: "6161d577c0696f72a2baf91b"
caption: "Adding a new picture"
imageURL: "abcmimg.jpeg"
timestamp: 2021-10-09T17:50:26.071+00:00
  
```

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

GET http://localhost:80... + ... No Environment

http://localhost:8081/posts/6161d662c0696f72a2baf91c Save

GET http://localhost:8081/posts/6161d662c0696f72a2baf91c Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```

1 {
2   ... "uid": "6161d577c0696f72a2baf91b",
3   ... "caption": "Adding a new picture",
4   ... "imageURL": "abcimg.jpeg"
5 }

```

Body Cookies Headers (3) Test Results 200 OK 285 ms 235 B Save Response

Pretty Raw Preview Visualize JSON

```

1 {
2   "_id": "6161d662c0696f72a2baf91c",
3   "uid": "6161d577c0696f72a2baf91b",
4   "caption": "Adding a new picture",
5   "imageURL": "abcimg.jpeg"
6 }

```

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

GET http://localhost:80... + ... No Environment

http://localhost:8081/posts/users/6161b8c3ed50249616b5e3dc?p=1 Save

GET http://localhost:8081/posts/users/6161b8c3ed50249616b5e3dc?p=1 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

Body Cookies Headers (3) Test Results 200 OK 225 ms 593 B Save Response

Pretty Raw Preview Visualize JSON

```

1 [
2   {
3     "_id": "6161ba26ed50249616b5e3e2",
4     "uid": "6161b8c3ed50249616b5e3dc",
5     "caption": "bad",
6     "imageURL": "pie.jpg"
7   },
8   {
9     "_id": "6161d22d3e1c6719c73025bd",
10    "uid": "6161b8c3ed50249616b5e3dc",
11    "caption": "bad is 2nd now",
12    "imageURL": "needforseinfeld.jpg"
13  }
14 ]

```

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.

The screenshot shows the VS Code editor interface. On the left, the Explorer sidebar shows a project named 'INSTA-APPOINTY' with a file named 'main.go' selected. The main editor area displays the content of 'main.go', which is a Go program for a REST API. The code includes imports for 'context', 'crypto/md5', 'encoding/hex', 'encoding/json', 'fmt', 'log', 'net/http', 'regexp', 'strconv', 'time', and MongoDB drivers. The main function defines a 'client' of type 'mongo.Client' and a 'main' function that sets up a MongoDB connection and a simple REST API with endpoints for creating, updating, and deleting appointments. The terminal at the bottom shows the command 'go run main.go' being executed, and the output indicates that the application started successfully and is listening on port 8080.

```
1 package main
2
3 import (
4     "context"
5     "crypto/md5"
6     "encoding/hex"
7     "encoding/json"
8     "fmt"
9     "log"
10    "net/http"
11    "regexp"
12    "strconv"
13    "time"
14
15    "go.mongodb.org/mongo-driver/bson"
16    "go.mongodb.org/mongo-driver/bson/primitive"
17    "go.mongodb.org/mongo-driver/mongo"
18    "go.mongodb.org/mongo-driver/mongo/options"
19 )
20
21 var client *mongo.Client
```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/powershell>

PS C:\Users\Hp\go\src\insta-appointy> go run main.go
Starting the application...
After DB
method: POST
method: POST
method: GET
method: POST
method: GET
ObjectID("6161d662c0696f72a2baf91c")
method: GET
ObjectID("6161d662c0696f72a2baf91c")
method: GET

- MongoDB should be used for storage.

Mongo DB ATLAS online client was used.

Harsh's Org - 2020-1...

Access Manager

Billing

All Clusters

Get Help

Harsh Vardhan

Project 0

Atlas

Realm

Charts

DEPLOYMENT

Databases

Triggers

Data Lake

SECURITY

Database Access

Network Access

Advanced

HARSH'S ORG - 2020-10-27 > PROJECT 0 > DATABASES

Cluster0

VERSION

4.4.9

REGION

GCP Mumbai (asia-south1)

Overview

Real Time

Metrics

Collections

Search

Profiler

Performance Advisor

Online Archive

DATABASES: 1

COLLECTIONS: 2

VISUALIZE YOUR DATA

REFRESH

+ Create Database

NAMESPACES

Mern

posts

users

Mern

DATABASE SIZE: 1.66KB

INDEX SIZE: 72KB

TOTAL COLLECTIONS: 2

CREATE COLLECTION

Collection Name	Documents	Documents Size	Documents Avg	Indexes	Index Size	Index Avg
posts	8	1.02KB	130B	1	36KB	36KB
users	5	662B	133B	1	36KB	36KB

- 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

```
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**.

```
> {
  "_id": ObjectId("6161b8c3ed50249616b5e3dc"),
  "name": "Ayush Sawarn",
  "email": "ayush.sawarn2019@vitstudent.ac.in",
  "password": "a080a3f2c6fc16f60cc7655e3f2f2151"
}
```

```
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 every page returning 4 data at a time. The page number is accepted as query parameter.

GET http://localhost:8081/posts/users/6161b8c3ed50249616b5e3dc?p=1

Save

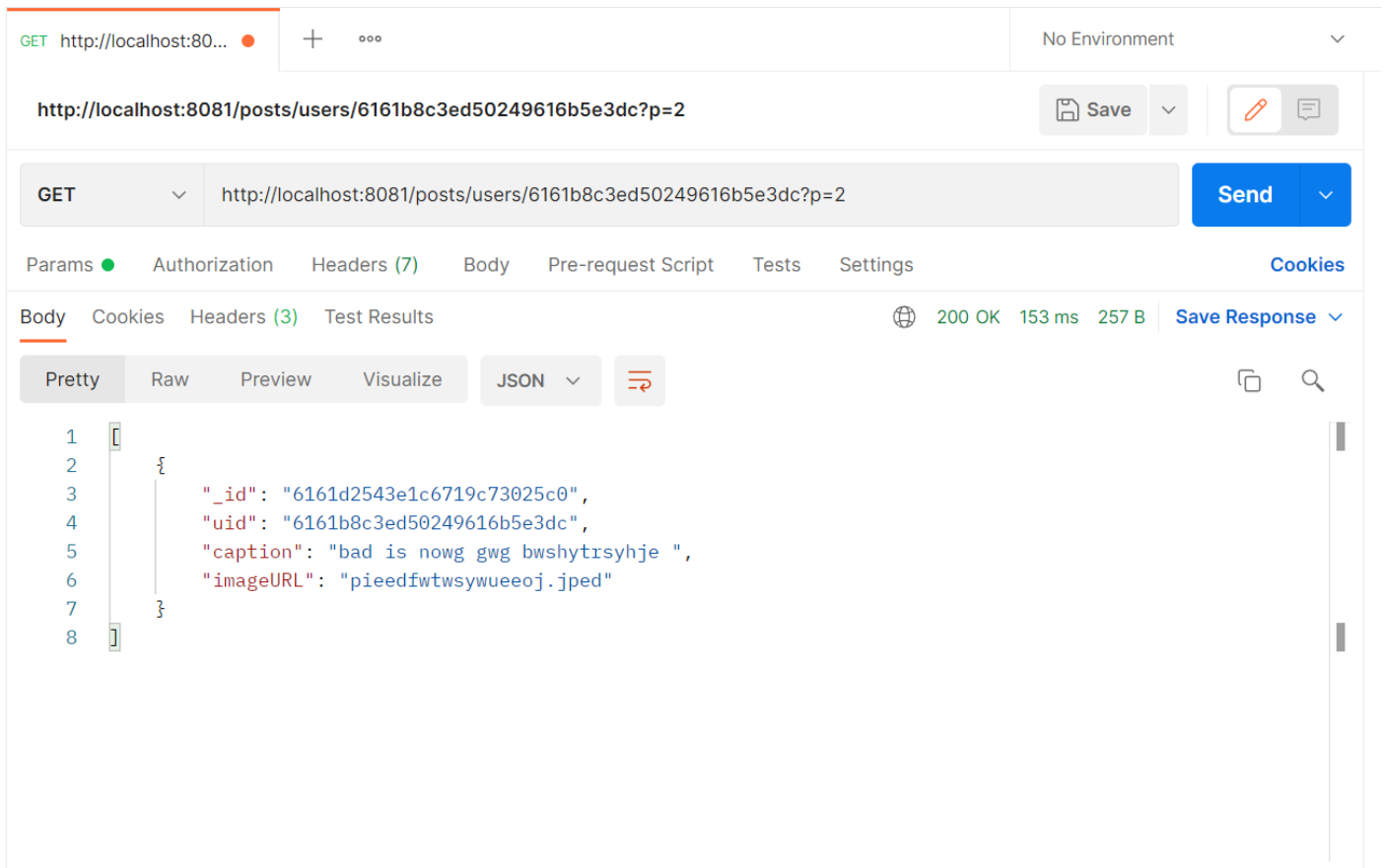
GET http://localhost:8081/posts/users/6161b8c3ed50249616b5e3dc?p=1 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Body Cookies Headers (3) Test Results 200 OK 225 ms 593 B Save Response

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "_id": "6161ba26ed50249616b5e3e2",
4     "uid": "6161b8c3ed50249616b5e3dc",
5     "caption": "bad",
6     "imageURL": "pie.jpg"
7   },
8   {
9     "_id": "6161d22d3e1c6719c73025bd",
10    "uid": "6161b8c3ed50249616b5e3dc",
11    "caption": "bad is 2nd now",
12    "imageURL": "pieedfweojferoi.jpg"
13  },
14  {
15    "_id": "6161d2393e1c6719c73025be",
16    "uid": "6161b8c3ed50249616b5e3dc",
17    "caption": "bad is 3rd now",
18    "imageURL": "pieedfweojferoi.jpg"
19  }
20 ]
```



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%.