

NAME: SHIVANSH BAJAR

EMAIL: shivansh.bajar@gmail.com

COLLEGE: Vellore Institute of Technology, Chennai

REGISTRAION NUMBER: 19BCE1638

## SCREENSHOTS OF TASK:

### CREATE USER:

```
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
        }
        fmt.Fprintf(w, "%v\n", r.PostForm)
        name := r.FormValue("name")
        email := r.FormValue("mail")
        password := r.FormValue("pass")

        md5HashInBytes := md5.Sum([]byte(password))
        md5HashInString := hex.EncodeToString(md5HashInBytes[:])

        fmt.Fprintf(w, "Name = %s\n", name)
        fmt.Fprintf(w, "Address = %s\n", email)
        fmt.Fprintf(w, "Password = %s\n", password)

        clientOptions := options.Client().
            ApplyURI("mongodb+srv://shivanshbajar:123@cluster0.helmj.mongodb.net/insta?retryWrites=true&w=majority")
        ctx, cancel := context.WithTimeout(context.Background(), 10*time.Second)
        defer cancel()
        client, err := mongo.Connect(ctx, clientOptions)
        if err != nil {
            log.Fatal(err)
        }

        collection := client.Database("insta").Collection("user")
        user := bson.D{{Key: "Name", Value: name},
            {Key: "Email", Value: email}, {Key: "Password", Value: md5HashInString}}

        res, insertErr := collection.InsertOne(ctx, user)
        if insertErr != nil {
            log.Fatal(insertErr)
        }
        fmt.Println(res)
    } else {
        http.ServeFile(w, r, "userform.html")
    }
}
```

### FORM:

```
main.go postform.html userform.html X
userform.html > html
1 <html>
2   <head>
3   </head>
4   <body>
5     <div>
6       <form method="POST" action="/users">
7         <label>Name</label><input name="name" type="text" value=""/><br>
8         <label>Email</label><input name="mail" type="text" value=""/><br>
9         <label>Password</label><input name="pass" type="text" value=""/><br>
10        <input type="submit" value="submit"/>
11      </form>
12    </div>
13  </body>
14 </html>
```

## GET USER USING ID

```
func getUser(w http.ResponseWriter, r *http.Request) {
    fmt.Println("method:", r.Method)
    if r.Method == http.MethodGet {
        if err := r.ParseForm(); err != nil {
            fmt.Fprintf(w, "ParseForm() err: %v", err)
            return
        }
        id := r.FormValue("id")

        clientOptions := options.Client().
            ApplyURI("mongodb+srv://shivanshbajar:123@cluster0.helmj.mongodb.net/insta?retryWrites=true&w=majority")
        ctx, cancel := context.WithTimeout(context.Background(), 10*time.Second)
        defer cancel()
        client, err := mongo.Connect(ctx, clientOptions)
        if err != nil {
            log.Fatal(err)
        }

        collection := client.Database("insta").Collection("user")

        objectId, err := primitive.ObjectIDFromHex(id)
        if err != nil {
            log.Println("Invalid id")
        }

        var user bson.M
        if err = collection.FindOne(ctx, bson.M{"_id": objectId}).Decode(&user); err != nil {
            log.Fatal(err)
        }
        fmt.Fprintf(w, "USER: %s", user)
    }
}
```

## CREATE A POST

```
func addPosts(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
        }
        fmt.Fprintf(w, "%v\n", r.PostForm)
        name := r.FormValue("name")
        caption := r.FormValue("cap")
        url := r.FormValue("url")
        timestamp := time.Now()

        fmt.Fprintf(w, "Name = %s\n", name)
        fmt.Fprintf(w, "Caption = %s\n", caption)
        fmt.Fprintf(w, "Photo Url = %s\n", url)
        fmt.Fprintf(w, "Timestamp = %s\n", timestamp)

        clientOptions := options.Client().
            ApplyURI("mongodb+srv://shivanshbajar:123@cluster0.helmj.mongodb.net/insta?retryWrites=true&w=majority")
        ctx, cancel := context.WithTimeout(context.Background(), 10*time.Second)
        defer cancel()
        client, err := mongo.Connect(ctx, clientOptions)
        if err != nil {
            log.Fatal(err)
        }

        collection := client.Database("insta").Collection("posts")

        post := bson.D{{Key: "Name", Value: name},
            {Key: "Caption", Value: caption}, {Key: "Post URL", Value: url}, {Key: "TimeStamp", Value: timestamp}}

        res, insertErr := collection.InsertOne(ctx, post)
        if insertErr != nil {
            log.Fatal(insertErr)
        }
        fmt.Println(res)
    } else {
        http.ServeFile(w, r, "postform.html")
    }
}
```

## FORM

```
main.go  postform.html  userform.html
postform.html > html
1  <html>
2      <head>
3      </head>
4      <body>
5          <div>
6              <form method="POST" action="/posts">
7                  <label>Name</label><input name="name" type="text" value=""/><br>
8                  <label>Caption</label><input name="cap" type="text" value=""/><br>
9                  <label>Image URL</label><input type="url" name="url", value=""/><br>
10                 <input type="submit" value="submit"/>
11             </form>
12         </div>
13     </body>
14 </html>
```

## GET A POST USING ID

```
func getPost(w http.ResponseWriter, r *http.Request) {
    fmt.Println("method:", r.Method)
    if r.Method == http.MethodGet {
        if err := r.ParseForm(); err != nil {
            fmt.Fprintf(w, "ParseForm() err: %v", err)
            return
        }
        postid := r.FormValue("postid")

        clientOptions := options.Client().
            ApplyURI("mongodb+srv://shivanshbajjar:123@cluster0.helmj.mongodb.net/insta?retryWrites=true&w=majority")
        ctx, cancel := context.WithTimeout(context.Background(), 10*time.Second)
        defer cancel()
        client, err := mongo.Connect(ctx, clientOptions)
        if err != nil {
            log.Fatal(err)
        }

        collection := client.Database("insta").Collection("posts")

        objectId, err := primitive.ObjectIDFromHex(postid)
        if err != nil {
            log.Println("Invalid id")
        }

        var post bson.M
        if err = collection.FindOne(ctx, bson.M{"_id": objectId}).Decode(&post); err != nil {
            log.Fatal(err)
        }
        fmt.Fprintf(w, "POST: %s", post)
    }
}
```

## LIST ALL POSTS OF A USER

```
func getPostList(w http.ResponseWriter, r *http.Request) {
    fmt.Println("method:", r.Method)
    if r.Method == http.MethodGet {
        if err := r.ParseForm(); err != nil {
            fmt.Fprintf(w, "ParseForm() err: %v", err)
            return
        }
        key := r.FormValue("id")
        //fmt.Fprintf(w, key)

        clientOptions := options.Client().
            ApplyURI("mongodb+srv://shivanshbajjar:123@cluster0.helmj.mongodb.net/insta?retryWrites=true&w=majority")
        ctx, cancel := context.WithTimeout(context.Background(), 10*time.Second)
        defer cancel()
        client, err := mongo.Connect(ctx, clientOptions)
        if err != nil {
            log.Fatal(err)
        }

        collection := client.Database("insta").Collection("posts")

        /*
            objectId, err := primitive.ObjectIDFromHex(key)
            if err != nil {
                log.Println("Invalid id")
            }
        */
        postslst, err := collection.Find(ctx, bson.M{"Name": key})
        if err != nil {
            log.Fatal(err)
        }
        var listofpost []bson.M
        if err = postslst.All(ctx, &listofpost); err != nil {
            log.Fatal(err)
        }
        fmt.Fprintf(w, "All posts by user: %s", listofpost)
    }
}
```

## HANDLE REQUEST AND MAIN FUNTION:

```
func handleRequests() {
    http.HandleFunc("/users", addUsers)
    http.HandleFunc("/posts", addPosts)
    http.HandleFunc("/users/", getUser)
    http.HandleFunc("/posts/", getPost)
    http.HandleFunc("/posts/users/", getPostList)
    log.Fatal(http.ListenAndServe(":8082", nil))
}

func main() {
    handleRequests()
}
```

## PASSWORD ENCRYPTION:

THIS HAS BEEN DONE IN ADD USER FUNCTION:

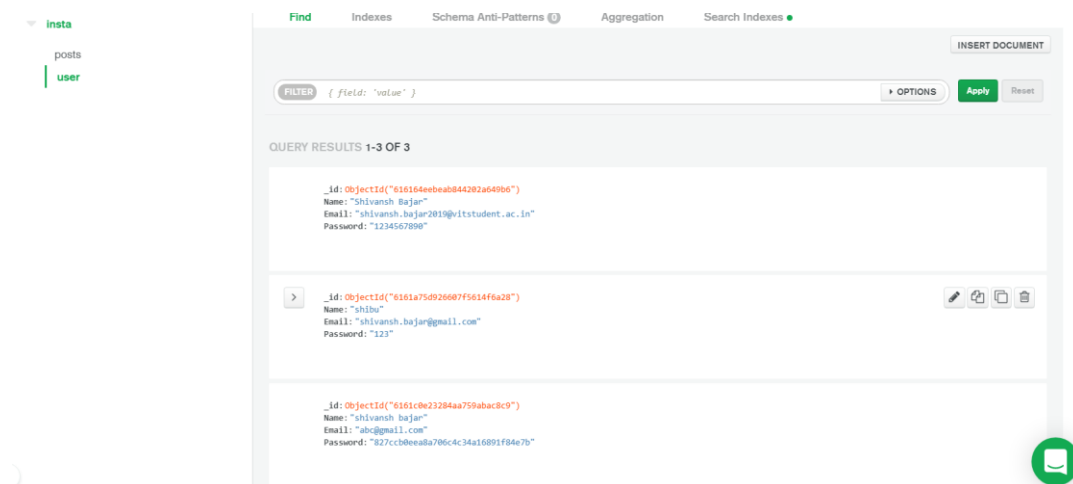
```
password := r.FormValue("pass")

md5HashInBytes := md5.Sum([]byte(password))
md5HashInString := hex.EncodeToString(md5HashInBytes[:])
```

## MONGO DB:

INSTA IS THE DATABASE WHICH HAS TWO CLUSTERS THAT ARE: **USERS** and **POSTS**

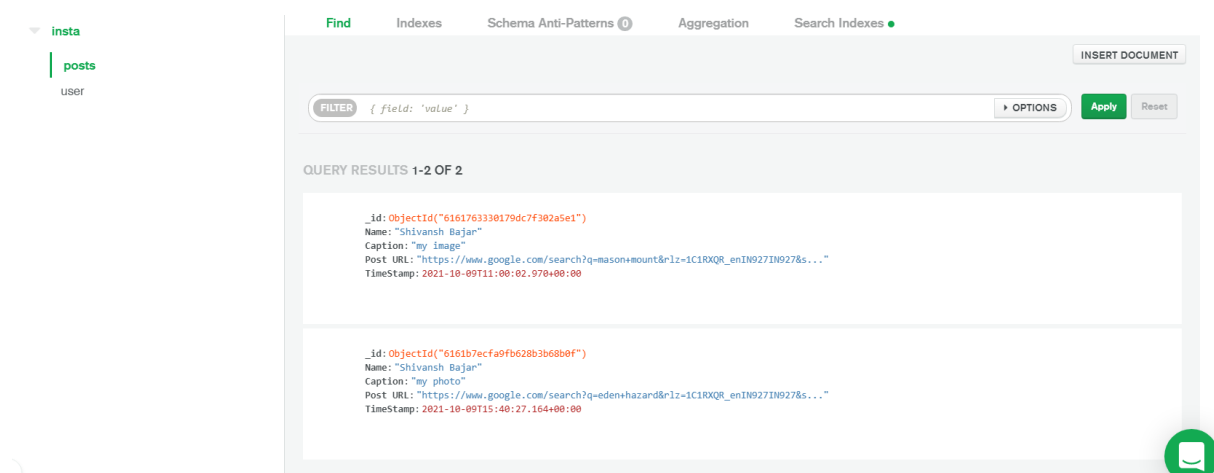
### USER CLUSTER:



The screenshot shows the MongoDB Compass interface for the 'insta' database, specifically the 'user' collection. The left sidebar shows the database structure with 'insta' expanded, containing 'posts' and 'user' collections. The main panel shows the 'Find' tab with a filter of '{ field: 'value' }'. The query results show three documents:

- Document 1:   
\_id: ObjectId("616164eeba0844202a6406")  
Name: "Shivansh Bajar"  
Email: "shivansh.bajar2019@vitstudent.ac.in"  
Password: "1234567890"
- Document 2:   
\_id: ObjectId("6161a75d926607f5614fe28")  
Name: "shilpa"  
Email: "shivansh.bajar@gmail.com"  
Password: "123"
- Document 3:   
\_id: ObjectId("6161c0e23284aa759abac8c9")  
Name: "Shivansh Bajar"  
Email: "abc@gmail.com"  
Password: "927cch8ee8a706c4c34a16091f04e7b"

### POSTS CLUSTER:



The screenshot shows the MongoDB Compass interface for the 'insta' database, specifically the 'posts' collection. The left sidebar shows the database structure with 'insta' expanded, containing 'posts' and 'user' collections. The main panel shows the 'Find' tab with a filter of '{ field: 'value' }'. The query results show two documents:

- Document 1:   
\_id: ObjectId("6161763330179dc7f302a5e1")  
Name: "Shivansh Bajar"  
Caption: "my image"  
Post URL: "https://www.google.com/search?q=mason+mount&rlz=1C1RXQR\_enIN927IN9278s..."  
Timestamp: 2021-10-09T11:00:02.970+00:00
- Document 2:   
\_id: ObjectId("6161b7ecfa9fb628b3b68b0f")  
Name: "Shivansh Bajar"  
Caption: "my photo"  
Post URL: "https://www.google.com/search?q=eden+hazard&rlz=1C1RXQR\_enIN927IN9278s..."  
Timestamp: 2021-10-09T15:40:27.164+00:00