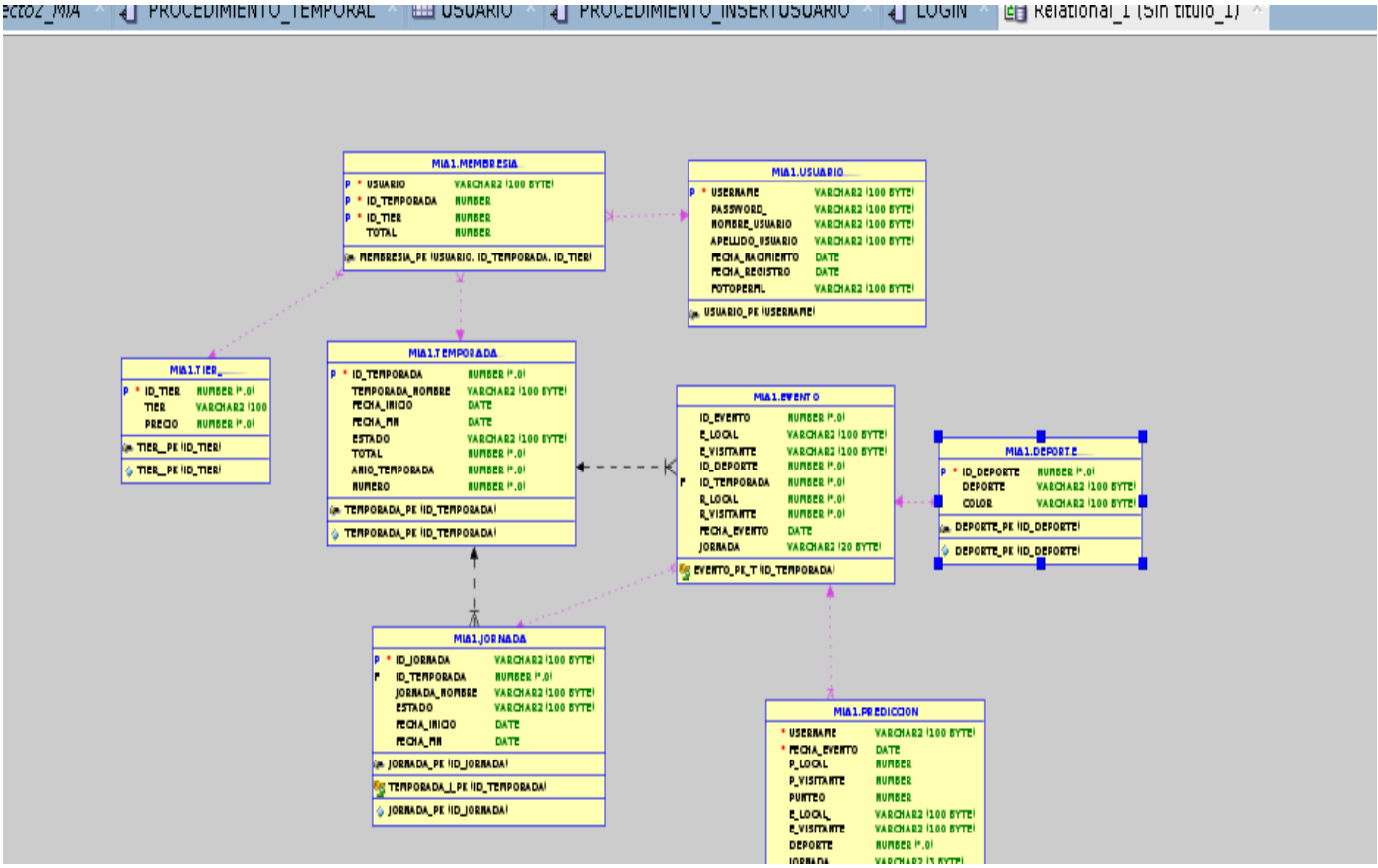
Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Manejo E implementación de Archivos

Manual Tecnico

Juan Diego Alvarado -201807335

Modelo Entidad Relación



Método Carga Masiva

Método de gran utilidad para poder realizar la carga masiva , debido a que este realiza un for para poder recorrer el archivo YAML dado por el proyecto.

```
let Temporal = [];
reader.readAsText(file);
reader.onload = e => {
    try {
        const doc = yaml.load(e.target.result);
        this.load = JSON.stringify(doc, null, 2);

        for (const [e1, e2] of Object.entries(doc)) {

            for (let rec1 = 0; rec1 < e2.resultados.length; rec1++) {

                for (let rec2 = 0; rec2 < e2.resultados[rec1].jornadas.length; rec2++) {


                    for (let rec3 = 0; rec3 < e2.resultados[rec1].jornadas[rec2].predicciones.length; rec3++) {

                        let Carga = {
                            NOMBRE: e2.nombre,
                            APELLIDO: e2.apellido,
                            PASSWORD: e2.password,
                            USERNAME: e2.username,
                            TEMPORADA: e2.resultados[rec1].temporada,
                            TIER: e2.resultados[rec1].tier,
                            JORNADA: e2.resultados[rec1].jornadas[rec2].jornada,
                            DEPORTE: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].deporte,
                            FECHA: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].fecha,
                            E_VISITANTE: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].visitante,
                            E_LOCAL: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].local,
                            P_VISITANTE: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].prediccion.visitante,
                            P_LOCAL: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].prediccion.local,
```

```
                    for (let rec3 = 0; rec3 < e2.resultados[rec1].jornadas[rec2].predicciones.length; rec3++) {

                        let Carga = {
                            NOMBRE: e2.nombre,
                            APELLIDO: e2.apellido,
                            PASSWORD: e2.password,
                            USERNAME: e2.username,
                            TEMPORADA: e2.resultados[rec1].temporada,
                            TIER: e2.resultados[rec1].tier,
                            JORNADA: e2.resultados[rec1].jornadas[rec2].jornada,
                            DEPORTE: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].deporte,
                            FECHA: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].fecha,
                            E_VISITANTE: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].visitante,
                            E_LOCAL: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].local,
                            P_VISITANTE: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].prediccion.visitante,
                            P_LOCAL: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].prediccion.local,
                            R_VISITANTE: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].resultado.visitante,
                            R_LOCAL: e2.resultados[rec1].jornadas[rec2].predicciones[rec3].resultado.local,


                        }
                        //     console.log(Carga.NOMBRE+" --"+Carga.APELLIDO+"---"+Carga.E_VISITANTE)
                        //  console.log(Carga)

                        Temporal.push(Carga);


                    }
```

Método POST

Método post que su función es de mandar una petición directo al servidor de GO

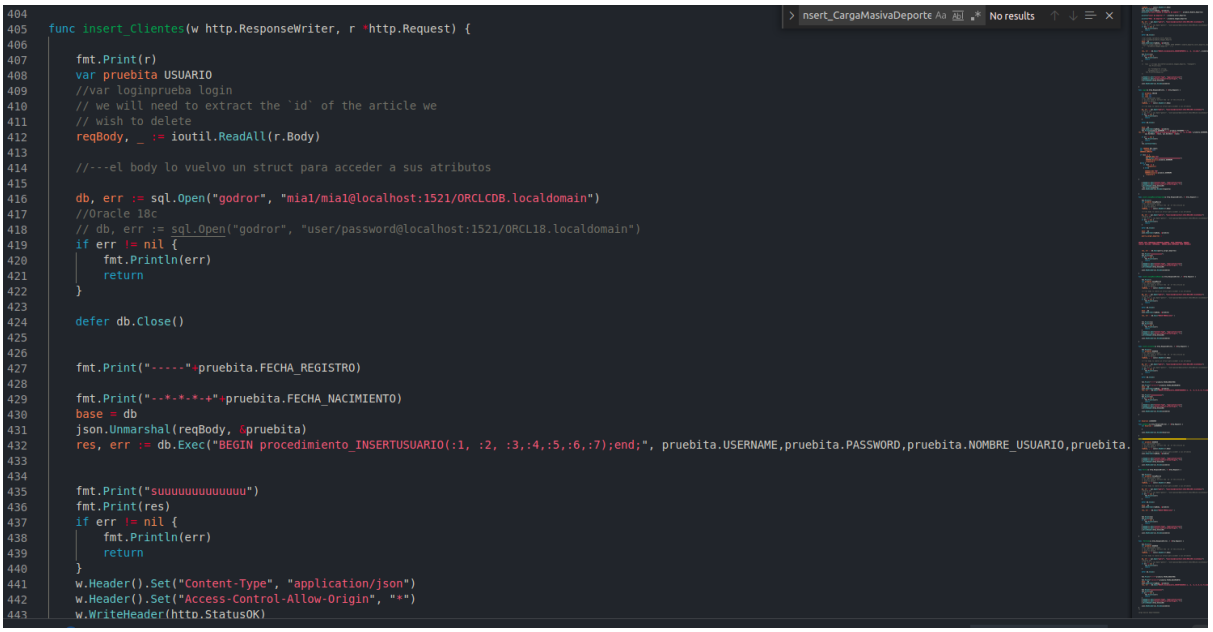
```
sendPost = async (carga) => {
    await axios
        .post("http://localhost:8000/CargaMasiva", carga)
        .then((response) => {
            console.log(response.data);
        })
        .catch((err) => console.error(`Error: ${err}`));

};
sendModelo = async () => {
    await axios
        .post("http://localhost:8000/CargarModelo")
        .then((response) => {
            console.log(response.data);
        })
        .catch((err) => console.error(`Error: ${err}`));

};
```

Método Insert CLIETES

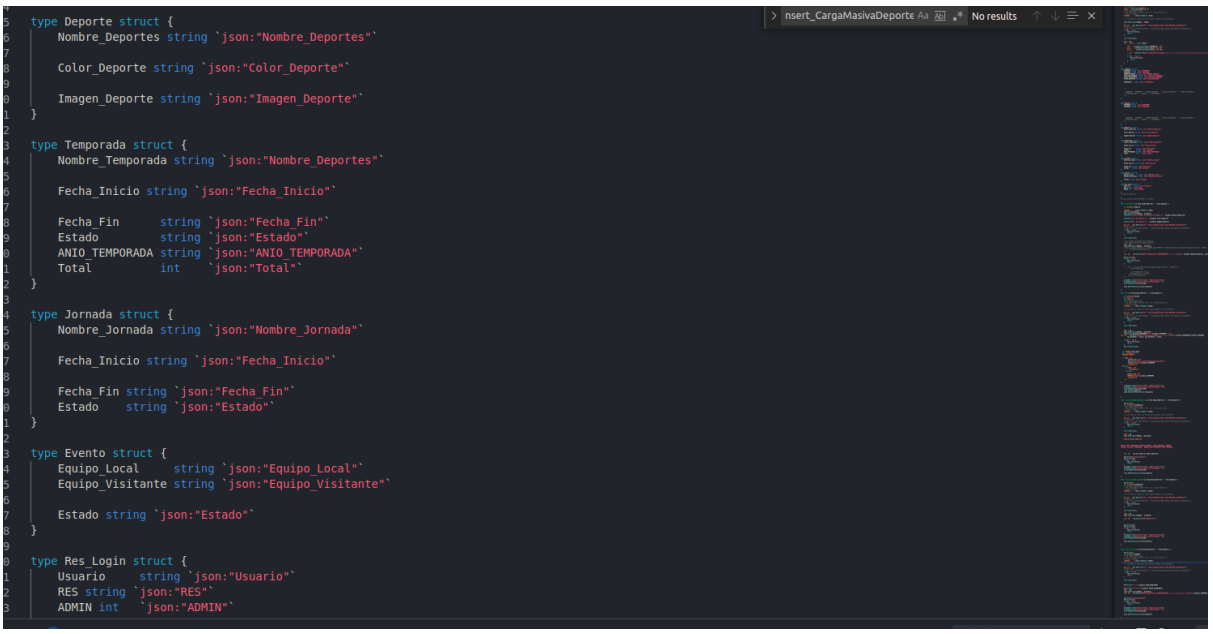
Este método como su nombre lo indica recibe una petición POST que viene directamente del servidor de React , lo cual permite crear nuevos usuarios en la base de datos



```
404
405  func insert_Clientes(w http.ResponseWriter, r *http.Request) {
406
407      fmt.Print(r)
408      var pruebita USUARIO
409      //var loginprueba login
410      // we will need to extract the `id` of the article we
411      // wish to delete
412      reqBody, _ := ioutil.ReadAll(r.Body)
413
414      //---el body lo vuelvo un struct para acceder a sus atributos
415
416      db, err := sql.Open("godror", "mia1/mia1@localhost:1521/ORCLCDB.localdomain")
417      //Oracle 18c
418      // db, err := sql.Open("godror", "user/password@localhost:1521/ORCL18.localdomain")
419      if err != nil {
420          fmt.Println(err)
421          return
422      }
423
424      defer db.Close()
425
426
427      fmt.Print("-----"+pruebita.FECHA_REGISTRO)
428
429      fmt.Print("--*-*-*-*-+"+pruebita.FECHA_NACIMIENTO)
430      base = db
431      json.Unmarshal(reqBody, &pruebita)
432      res, err := db.Exec("BEGIN procedimiento_INSERTUSUARIO(:1, :2, :3,:4,:5,:6,:7);end;", pruebita.USERNAME,pruebita.PASSWORD,pruebita.NOMBRE_USUARIO,pruebita.
433
434
435      fmt.Print("suuuuuuuuuuuuu")
436      fmt.Print(res)
437      if err != nil {
438          fmt.Println(err)
439          return
440      }
441      w.Header().Set("Content-Type", "application/json")
442      w.Header().Set("Access-Control-Allow-Origin", "*")
443      w.WriteHeader(http.StatusOK)
```

Structs

Structs utilizados para almacenar todos los datos que vienen desde el servidor de REACT y así poder llenar la base de datos con lo que venga.



```
5   type Deporte struct {
6       Nombre_Deportes string `json:"Nombre_Deportes"`
7
8       Color_Deporte string `json:"Color_Deporte"`
9
0       Imagen_Deporte string `json:"Imagen_Deporte"`
1   }
2
3   type Temporada struct {
4       Nombre_Temporada string `json:"Nombre_Deportes"`
5
6       Fecha_Inicio string `json:"Fecha_Inicio"`
7
8       Fecha_Fin      string `json:"Fecha_Fin"`
9       Estado         string `json:"Estado"`
0       ANIO_TEMPORADA string `json:"ANIO_TEMPORADA"`
1       Total          int    `json:"Total"`
2   }
3
4   type Jornada struct {
5       Nombre_Jornada string `json:"Nombre_Jornada"`
6
7       Fecha_Inicio string `json:"Fecha_Inicio"`
8
9       Fecha_Fin string `json:"Fecha_Fin"`
0       Estado    string `json:"Estado"`
1   }
2
3   type Evento struct {
4       Equipo_Local     string `json:"Equipo_Local"`
5       Equipo_Visitante string `json:"Equipo_Visitante"`
6
7       Estado string `json:"Estado"`
8   }
9
0   type Res_Login struct {
1       Usuario     string `json:"Usuario"`
2       RES string `json:"RES"`
3       ADMIN int   `json:"ADMIN"`
```

PROCEDMIENTO ALMACENADO INSET_USUARIO

Este procedimiento almacenado es de gran utilidad para poder crear usuarios.

```
create or replace  PROCEDURE   PROCEDIMIENTO_InsertUsuario  (
USERNAME   VARCHAR2,
PASSWORD_ VARCHAR2,
NOMBRE_USUARIO VARCHAR2,
APELLIDO_USUARIO  VARCHAR2,
FECHA_NACIMIENTO  VARCHAR2,
FECHA_REGISTRO  VARCHAR2,
FOTOPERFIL  VARCHAR2

)AS
BEGIN
INSERT INTO USUARIO
VALUES (USERNAME,PASSWORD_ ,NOMBRE_USUARIO,
APELLIDO_USUARIO,TO_DATE(FECHA_NACIMIENTO, 'dd/mm/yyyy'),TO_DATE(FECHA_REGISTRO, 'dd/mm/yyyy'),FOTOPERFIL) ;
END;
```

INSERT INTO PREDICCIÓN

Un insert para poder llenar todas las predicciones con sus diferentes parámetros.

```
INSERT INTO PREDICCION(USERNAME,  PREDICCION.E_LOCAL_, PREDICCION.E_VISITANTE, PREDICCION.P_LOCAL,
PREDICCION.P_VISITANTE, JORNADA, ID_TEMPORADA, FECHA_EVENTO,DEPORTE)
SELECT DISTINCT USERNAME, TEMPORAL.E_LOCAL, TEMPORAL.E_VISITANTE, P_LOCAL, P_VISITANTE, JORNADA, ID_TEMPORADA, TO_DATE(FECHA, 'dd/mm/yyyy HH24:MI'), ID_DEPORTE
FROM TEMPORAL LEFT JOIN TEMPORADA ON TEMPORAL.TEMPORADA = TEMPORADA.TEMPORADA_NOMBRE
LEFT JOIN DEPORTE ON TEMPORAL.DEPORTE = DEPORTE.DEPORTE;


END;
```

CREATE TABLE TEMPORAL

Se crea una tabla temporal para poder almacenar todos los datos que ingresan del archivo de entrada

```
 CREATE TABLE TEMPORAL(
NOMBRE VARCHAR2(100),
APELLIDO VARCHAR2(100),
PASSWORD VARCHAR2 (100),
USERNAME VARCHAR2 (100),
TEMPORADA VARCHAR2 (100),
numero int ,
ANIO_TEMPORADA int,
TIER  VARCHAR2(100),
JORNADA VARCHAR2(100),
DEPORTE VARCHAR2(100),
FECHA VARCHAR2(100),
E_VISITANTE VARCHAR2(100),
E_LOCAL VARCHAR2(100),
P_LOCAL INT  ,
P_VISITANTE INT  ,
R_VISITANTE INT  ,
R_LOCAL INT


);
```