Manual Técnico

Karla Fernanda Matías de León 3350910240901

Hania Mirleth Mazariegos Alonzo 3190701150910

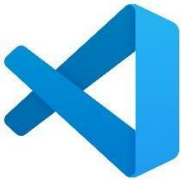
Proyecto 1

Es una aplicación, donde los usuarios pueden verificar estadísticas sobre los aspirantes a la Universidad San Carlos de Guatemala en el año 2023. Este sistema es una aplicación web sencilla, en la que se utilizo como base de datos, MongoDB y Neo4J, son gestores de bases de datos no relacionales, se utilizo también NodeJS para el desarrollo del Backed y para el Fronted se utilizo React que es una biblioteca de JavaScript

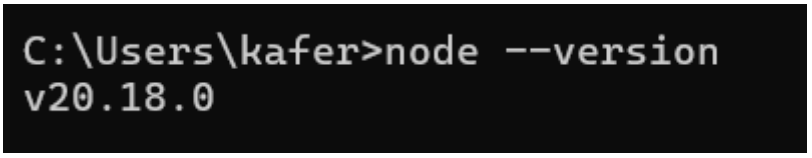
Para desarrollar nuestro proyecto, crearemos una ApiREST. La arquitectura de nuestra página será la siguiente

Teniendo en cuenta nuestro modelo, y las herramientas a trabajar. Para poder desarrollar el proyecto se decidió utilizar el sistema operativo Windows.

Dentro de nuestro sistema operativo, debemos contar con el IDE Visua lStudio Code, en el proyecto se utilizó este IDE, ya que es el ambiente de desarrollo más común.

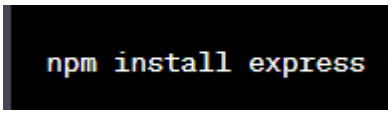


Ya que trabajaremos con NodeJS, se debe contar con una version de este reciente, en este caso se utilizó Node v20.18 Podemos verificar nuestra version de Node desde cmd o el simbolo de Sistema en Windows.

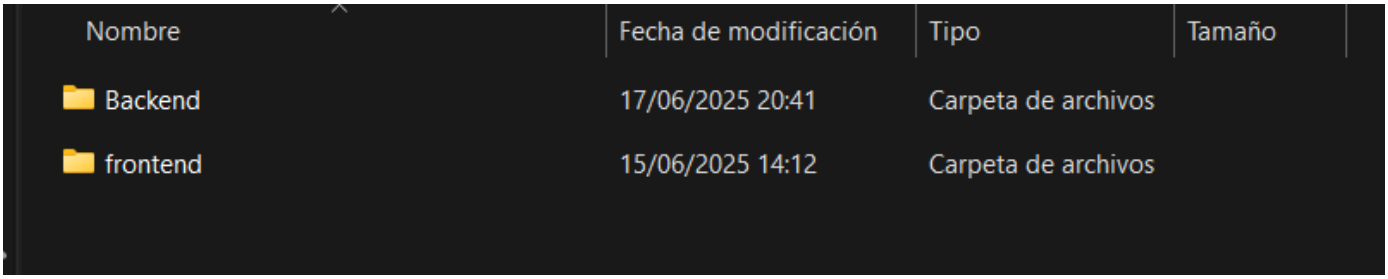


Debemos contar tambien con MongoDB, que sera nuestro gestor de bases de datos a utilizar, en este caso utilizamos la version 2.5 de MongoDB. De igual manera que con Node, podemos verificar si tenemos instalado o que version de MongoDB tenemos, desde la terminal.

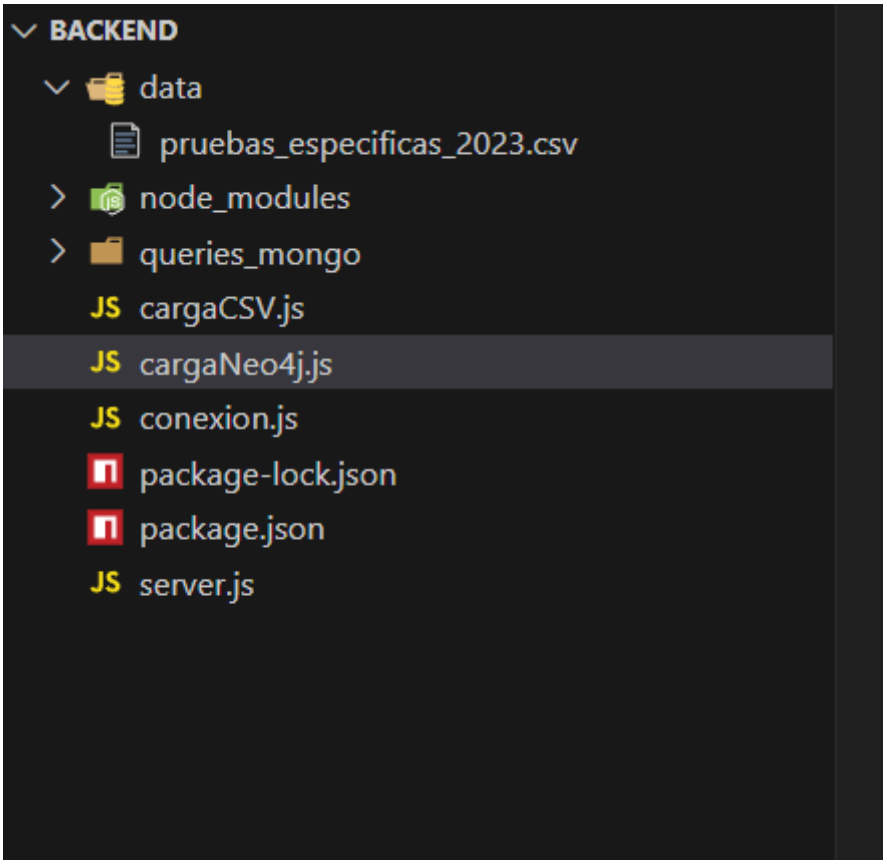
Una vez nuestro ambiente de desarrollo esté listo. Podemos comenzar a desarrollar el proyecto.
Para poder desarrollarlo, debemos contar con la dependencia de express, que es parte de NodeJS, para poder manejar nuestra base de datos. Para instalarla utilizamos el siguiente comando



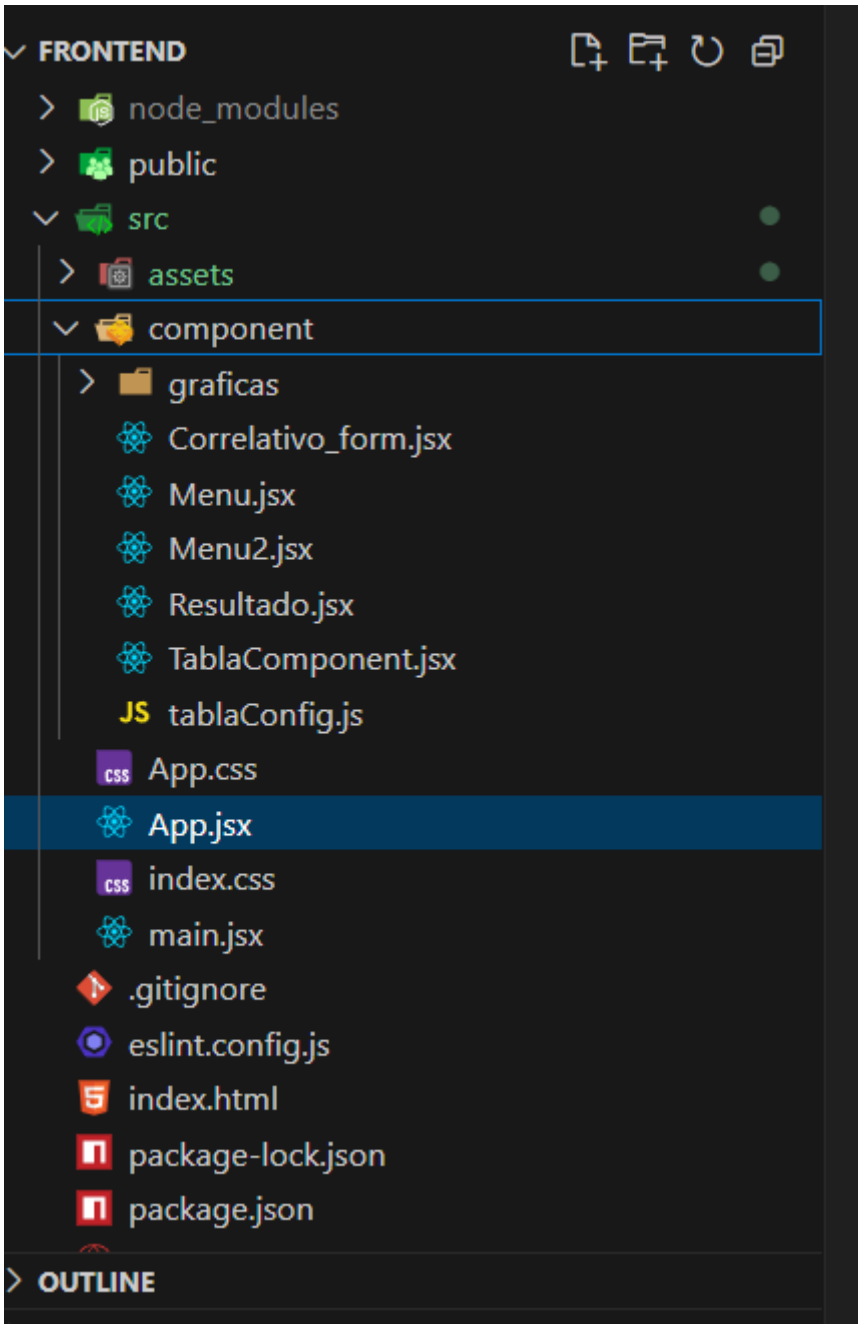
Y para poder desarrollar nuestra página, vamos a crear una carpeta llamada Proyecto1, y dentro crearemos otras dos carpetas una para el Fronted y otra para el Backend. De esta manera trabajaremos por separado ambas partes. Y creando así nuestra Api Rest



La arquitectura de back es la siguiente, tendremos una carpeta llamad data donde almacenaremos nuestro csv a cargar. Luego tenemos una carpeta queries_mongo donde están todas las consultas que haremos a nuestra base de datos de MongoDB. Luego tenemos la conexión y nuestro server que es el que ejecutaremos para levantar el back y tener conexión con nuestro front. Ahí mismo tenemos dos archivos .js uno llamada cargaCSV y otro cargaNeo4j.js. El archivo cargaCSV, es el que tiene toda la estructura para poder cargar nuestro csv a nuestra base de datos MongoDB. Y el de carga Neo4j.js es el encargado de cargar el csv a nuestra base de datos Neo4j



Para el fronted utilizamos la estructura que nos brinda vite+react. Basicamente tenemos en src la estructura de nuestro proyecto, dentro tenemos los assets donde almacenamos imágenes si nos sirven y luego en component tenemos la siguiente estructura, una carpeta llamada graficas donde almacenamos los componentes para nuestras graficas de pie, barras, líneas, etc. Luego afuera de esta carpeta tenemos Correlativo_form, Menu, Menu2, Resultado y TablaComponent, que son componentes que utilizamos en nuestro App.jsx.



Consultas

1. **Aspirantes por tipo de institución educativa**

```
db.aspirantes.aggregate([
  {
    $group: {
      _id: "$tipo_institucion_educativa",
      total_aspirantes: { $sum: 1 }
    }
  }
])
```

2. **Cantidad de aprobados por materia**

```
db.aspirantes.aggregate([
  { $match: { aprobacion: true } },
  { $group: { _id: "$materia", aprobados: { $sum: 1 } } }
])
```

3. **Aprobados por carrera y año**

```
db.aspirantes.aggregate([
  { $match: { aprobacion: true } },
  {
    $group: {
      _id: {
        carrera: "$carrera_objetivo",
        anio: "$anio_de_ingreso"
      },
      aprobados: { $sum: 1 }
    }
  }
])
```

4. Porcentaje de aprobación por materia

```
db.aspirantes.aggregate([
  {
    $group: {
      _id: "$materia",
      total: { $sum: 1 },
      aprobados: {
        $sum: {
          $cond: [ "$aprobacion", 1, 0 ]
        }
      }
    }
  },
  {
    $project: {
      porcentaje_aprobacion: {
        $multiply: [
          { $divide: [ "$aprobados", "$total" ] },
          100
        ]
      }
    }
  }
])
```

5. Promedio de edad por carrera

```
db.aspirantes.aggregate([
  {
    $match: {
      anio_nacimiento: { $exists: true, $type: "number", $ne: NaN }
    }
  },
  {
    $addFields: {
      edad: { $subtract: [new Date().getFullYear(), "$anio_nacimiento"] }
    }
  },
  {
    $group: {
      _id: "$carrera_objetivo",
      promedio_edad_exacto: { $avg: "$edad" }
    }
  },
  {
    $project: {
      _id: 1,
      promedio_edad: { $round: ["$promedio_edad_exacto", 2] }
    }
  },
  { $sort: { _id: 1 } }
])
```

6. Creación de colección auxiliar (resumen_carrera)

```
db.aspirantes.aggregate([
  {
    $group: {
      _id: "$carrera_objetivo",
      total: { $sum: 1 },
      aprobados: {
        $sum: {
          $cond: ["$aprobacion", 1, 0]
        }
      }
    }
  },
  {
    $project: {
      _id: 0,
      carrera: "$_id",
      total: 1,
      aprobados: 1
    }
  },
  {
    $out: "resumen_carrera"
  }
])
```

7. Promedio de edad de los aspirantes que aprobaron por carrera y tipo de institución

```
db.aspirantes.aggregate([
  {
    $match: {
      aprobacion: true,
      anio_nacimiento: { $exists: true, $type: "number", $ne: NaN }
    }
  },
  {
    $addFields: {
      edad: { $subtract: [new Date().getFullYear(), "$anio_nacimiento"] }
    }
  },
  {
    $group: {
      _id: {
        carrera: "$carrera_objetivo",
        tipo_institucion: "$tipo_institucion_educativa"
      },
      promedio_edad_exacto: { $avg: "$edad" }
    }
  },
  {
    $project: {
      _id: 1,
      promedio_edad: { $round: ["$promedio_edad_exacto", 2] }
    }
  },
  {
    $sort: {
      "_id.carrera": 1,
      "_id.tipo_institucion": 1
    }
  }
])
```

8. Distribución de aprobados por municipio y carrera objetivo

```
db.aspirantes.aggregate([
  {
    $match: { aprobacion: true }
  },
  {
    $group: {
      _id: {
        municipio: "$municipio_institucion_",
        carrera: "$carrera_objetivo"
      },
      total_aprobados: { $sum: 1 }
    }
  },
  {
    $sort: {
      "_id.municipio": 1,
      "_id.carrera": 1
    }
  }
])
```

9. Cantidad de evaluaciones por mes y materia, sólo para instituciones públicas

```
db.aspirantes.aggregate([
  {
    $match: {
      tipo_institucion_educativa: "PUBLICO",
      fecha_asignacion: {
        $exists: true,
        $type: "string",
        $ne: null,
        $ne: "",
        $ne: NaN
      }
    }
  },
  {
    $addFields: {
      mes: { $month: { $toDate: "$fecha_asignacion" } }
    }
  },
  {
    $group: {
      _id: {
        mes: "$mes",
        materia: "$materia"
      },
      cantidad: { $sum: 1 }
    }
  },
  {
    $sort: {
      "_id.mes": 1,
      "_id.materia": 1
    }
  }
])
```

10. Top 5 carreras más demandadas por aspirantes entre 16 y 18 años

```
db.aspirantes.aggregate([
  {
    $match: {
      anio_nacimiento: { $exists: true, $type: "number", $ne: null, $ne: NaN }
    }
  },
  {
    $addFields: {
      edad: { $subtract: [new Date().getFullYear(), "$anio_nacimiento"] }
    }
  },
  {
    $match: {
      edad: { $gte: 16, $lte: 18 }
    }
  },
  {
    $group: {
      _id: "$carrera_objetivo",
      total_aspirantes: { $sum: 1 }
    }
  },
  { $sort: { total_aspirantes: -1 } },
  { $limit: 5 }
])
```

11. Historial de desempeño por aspirante con desglose de intentos por materia y resultados

```
db.aspirantes.aggregate([
  {
    $group: {
      _id: {
        aspirante: "$correlativo_aspirante",
        materia: "$materia"
      },
      intentos: { $sum: 1 },
      aprobados: { $sum: { $cond: ["$aprobacion", 1, 0] } },
      reprobados: { $sum: { $cond: ["$aprobacion", 0, 1] } }
    }
  },
  {
    $project: {
      _id: 0,
      correlativo_aspirante: "$_id.aspirante",
      materia: "$_id.materia",
      intentos: 1,
      aprobados: 1,
      reprobados: 1
    }
  },
  {
    $sort: {
      correlativo_aspirante: 1,
      materia: 1
    }
  }
])
```

12. Distribución por sexo y tipo de institución

```
db.aspirantes.aggregate([
  {
    $group: {
      _id: {
        sexo: "$sexo",
        tipo: "$tipo_institucion_educativa"
      },
      cantidad: { $sum: 1 }
    }
  },
  {
    $project: {
      _id: 0,
      sexo: "$_id.sexo",
      tipo_institucion: "$_id.tipo",
      cantidad: 1
    }
  },
  {
    $sort: {
      sexo: 1,
      tipo_institucion: 1
    }
  }
])
```

13. Tasa de aprobación por edad

```
db.aspirantes.aggregate([
  {
    $match: {
      anio_nacimiento: { $exists: true, $type: "number", $ne: null, $ne: NaN }
    }
  },
  {
    $project: {
      edad: { $subtract: [new Date().getFullYear(), "$anio_nacimiento"] },
      aprobacion: 1
    }
  },
  {
    $group: {
      _id: "$edad",
      total: { $sum: 1 },
      aprobados: { $sum: { $cond: ["$aprobacion", 1, 0] } }
    }
  },
  {
    $project: {
      _id: 0,
      edad: "$_id",
      tasa_aprobacion: {
        $round: [
          { $multiply: [{ $divide: ["$aprobados", "$total"] }, 100] },
          2
        ]
      }
    }
  },
  { $sort: { edad: 1 } }
])
```

14. Número promedio de intentos por materia

```
db.aspirantes.aggregate([
{
  $group: {
    _id: {
      materia: "$materia",
      aspirante: "$correlativo_aspirante"
    },
    intentos: { $sum: 1 }
  }
},
{
  $group: {
    _id: "$_id.materia",
    total_intentos: { $sum: "$intentos" },
    total_aspirantes: { $sum: 1 }
  }
},
{
  $project: {
    _id: 0,
    materia: "$_id",
    promedio_intentos: {
      $round: [
        { $divide: ["$total_intentos", "$total_aspirantes"] },
        2
      ]
    }
  }
}
 ])
```

15. Carreras con más aspirantes reprobados en primer intento

```
db.aspirantes.aggregate([
  {
    $match: {
      numero_de_fecha_de_evaluaci: 1,
      aprobacion: false
    }
  },
  {
    $group: {
      _id: "$carrera_objetivo",
      reprobados_primera: { $sum: 1 }
    }
  },
  { $sort: { reprobados_primera: -1 } },
  { $limit: 5 },
  {
    $project: {
      _id: 0,
      carrera: "$_id",
      reprobados_primera: 1
    }
  }
])
```