Escuela de Ingenierías
Industrial, Informática y Aeroespacial

universidad de león

Nyte León

Laura Iglesias Gondar
Universidad de León

Índice

1. Objeto de estudio

Como estudiante de la Universidad de León no procedente de León ni de Castilla y León, cuando llegué el primer año no sabía qué podía hacer en León ni cuáles eran los mejores sitios para visitar y mis amigos tampoco eran de aquí, entonces tuvimos que ir descubriéndolo por nuestra cuenta. Por otra parte, tras la situación excepcional vivida este año debido al Covid-19 a mucha gente no se le ocurrían ideas sobre qué hacer en casa en su tiempo libre. Fue por todo esto que se me ocurrió la idea de crear NyteLeón, una aplicación web en la que se recomiendan actividades que se pueden realizar en casa o exteriores limitándose además a León o a Castilla y León mediante una serie de filtros o elegir una actividad de forma aleatoria.
Respecto al nombre de la aplicación, este viene de la unión de la palabra en noruego Nyte, que significa "disfrutar" y León.

2. Herramientas

En general, hice uso de GitHub para ir registrando los cambios en el código así como del cuaderno de trabajo OSF[1] donde fui registrando qué decisiones tomaba, qué vídeos miraba, cuales eran útiles así como cuáles no.

Para el desarrollo del código usé Visual Studio Code ya que es un editor de código muy útil para la elaboración de aplicaciones web.

2.1. Base de datos

El primer paso que llevé a cabo en el desarrollo de este proyecto fue conocer y entender bien qué era Neo4j para lo que me informé tanto en su canal de Youtube[2] como mediante el libro *Graph Algorithms*[3]. Además, tras escoger el nombre de la aplicación creé el logo en Paint.

En un principio tenía pensado elaborar la base de datos mediante un archivo .csv del que se muestra un fragmento a continuación:

1	visitarCatedralLeon,0,1,1,1,0,0,1,0
2	visitarBasilicaSanIsidoro,0,1,1,1,0,0,1,0
3	visitarConventoSanMarcos,0,1,1,1,0,0,1,0
4	visitarCasaBotinesGaudi,0,1,1,1,0,0,1,0
5	visitarPalacioGuzmanes,0,1,1,1,0,0,1,0
6	visitarPalacioCondeLuna,0,1,1,1,1,0,1,0
7	visitarCentroLeon,0,1,1,1,1,0,1,0
8	museoMUSAC,0,1,1,1,0,0,1,0
9	museoDeLeon,0,1,1,1,0,0,1,0
10	museoFaunaSalvaje,0,1,1,1,0,0,1,0

Pero tras varios problemas intentando importarlo así como a la hora de generar el código, consideré que era mejor idea crearla directamente mediante comandos Cypher en el Browser de Neo4J.

Entonces, elaboré los comandos en Cypher para Neo4j mediante los que se formaría la base de datos para usar en la aplicación.

Esta la creé a mano añadiendo actividades que encontré en una base de datos en Kaggle[4] que contenía una lista de hobbies. En esa lista eliminé las actividades que se repetían o las que en León no se podían hacer como el surf, y añadí otras que no estaban. Para ello me sirvieron de ayuda tanto Google como blogs de viajes por León y Castilla y León.

A continuación se muestra una parte del susodicho comando.

```
CREATE (visitarCatedralLeon:Actividad{nombre:"Visitar la catedral de León", descripción: "Esta es la descripción de la actividad"}),
(visitarBasilicaSanIsidoro:Actividad{nombre:"Visitar la Basílica de San Isidoro", descripción: "Esta es la descripción de la actividad"}),
(visitarConventoSanMarcos: Actividad{nombre:"Visitar Convento San Marcos", descripción: "Esta es la descripción de la actividad"}),
(visitarCasaBotinesGaudi:Actividad{nombre:"Visitar la Casa Botines de Gaudí", descripción: "Esta es la descripción de la actividad"}),
(visitarPalacioGuzmanes:Actividad{nombre:"Visitar el Palacio de los Guzmanes", descripción: "Esta es la descripción de la actividad"}),
(visitarPalacioCondeLuna:Actividad{nombre:"Visitar el Palacio Conde Luna", descripción: "Esta es la descripción de la actividad"}),
(visitarCentroLeon:Actividad{nombre:"Visitar el centro de León", descripción: "Esta es la descripción de la actividad"}),
(museoMUSAC:Actividad{nombre:"Ir al museo MUSAC", descripción: "Esta es la descripción de la actividad"}),
(museoDeLeon:Actividad{nombre:"Ir al museo de León", descripción: "Esta es la descripción de la actividad"}),
(museoFaunaSalvaje:Actividad{nombre:"Ir al museo de la fauna salvaje", descripción: "Esta es la descripción de la actividad"}),
(pasearParqueCandamia:Actividad{nombre:"Pasear por el parque de la Candamia", descripción: "Esta es la descripción de la actividad"}),
(correrParqueCandamia:Actividad{nombre:"Correr por el parque de la Candamia", descripción: "Esta es la descripción de la actividad"}),
(biciParqueCandamia:Actividad{nombre:"Pasear en bici por el parque de la Candamia", descripción: "Esta es la descripción de la actividad"})
```

Tras ello, para introducirla en Neo4J se debe crear una nueva base de datos en Neo4J, después abrir el Neo4J Browser e introducir este comando. La salida gráficamente de algunos de los nodos es la siguiente:

2.2. Backend

La elección de tecnologías para el Backend de la aplicación fueron Javascript para la funcionalidad, el entorno de Javascript Node.js para enviar las respuestas de las peticiones al Frontend, Express para la recepción de consultas del frontend así como un Driver de Neo4J para realizar las consultas a la base de datos del programa.

2.3. Frontend

Respecto al Frontend se usó -al igual que en Backend- JavaScript para la elaboración de la funcionalidad. En un principio empecé a crear la aplicación con Bootstrap para generar de forma más dinámica la interfaz, pero finalmente decidí que sería mejor usar el framework Vuetify y creé un nuevo repositorio con parte del código anterior pero modificando algunas partes. También se usaron Vue-Resource, Vue.js y Electron para crear una aplicación de escritorio.

3. Funcionamiento de la aplicación

Al tratarse de una aplicación web, el funcionamiento de esta se lleva a cabo entre el Frontend y el Backend.

Con el primero -el Frontend- es con quien interactúa el usuario mientras que, cuando éste lleva a cabo alguna petición en la aplicación como puede ser iniciar sesión o buscar actividades, el Frontend envía una petición al Backend para realizar dicha acción donde, una vez obtenida la información solicitada, el Backend se la devolverá al Frontend que mostrará el resultado al usuario.

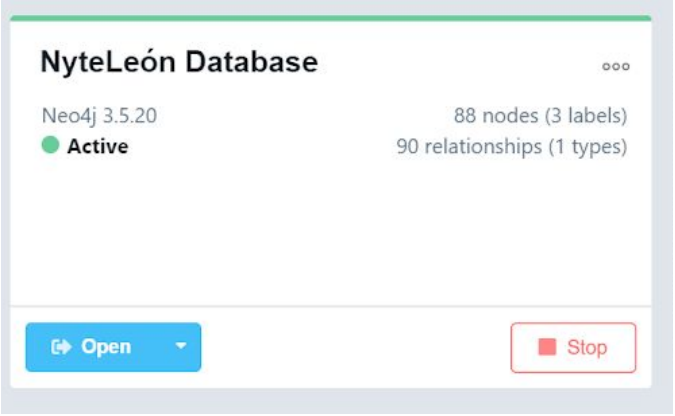
3.1. Base de datos

La base de datos creada a partir de un comando en Neo4J consta de 88 actividades que presentan los siguientes atributos:

-nombre de la actividad.
-modalidad de la actividad que puede tener el valor “Individual” o “Grupo”.
-precio: puede tomar el valor “Gratis” o “Pagando”.
-localización: indica si la actividad es “Dentro de León” o “Fuera de León”.
-actividadFisica: tal y como el nombre da a entender, indica si la actividad requiere esfuerzo físico, siendo sus posibles valores "Con actividad física" o "Sin actividad física".

Además, existen los nodos de tipo Entorno que pueden ser “Interior” o “Exterior” para organizar las actividades de forma más fácil en caso de posible cuarentena, ya que fue el principal motivo por el que decidí crear esta aplicación.

Tal y como se indica en el punto 2.1., se introduce el comando en el Neo4J Browser para crear la base de datos y luego usarla. A continuación se muestra como debería estar la base de datos en Neo4J para poder usar la aplicación.



3.2. Backend

Dentro del Backend se introducen las siguientes líneas de código para poder así mediante el Driver de Neo4J poder acceder a la base de datos.

```
const neo4j = require("neo4j-driver").v1;
const driver = neo4j.driver(
  "bolt://localhost",
  neo4j.auth.basic("neo4j", "123")
);
const session = driver.session();
```

Luego, se compone de cuatro funciones que serán las siguientes:

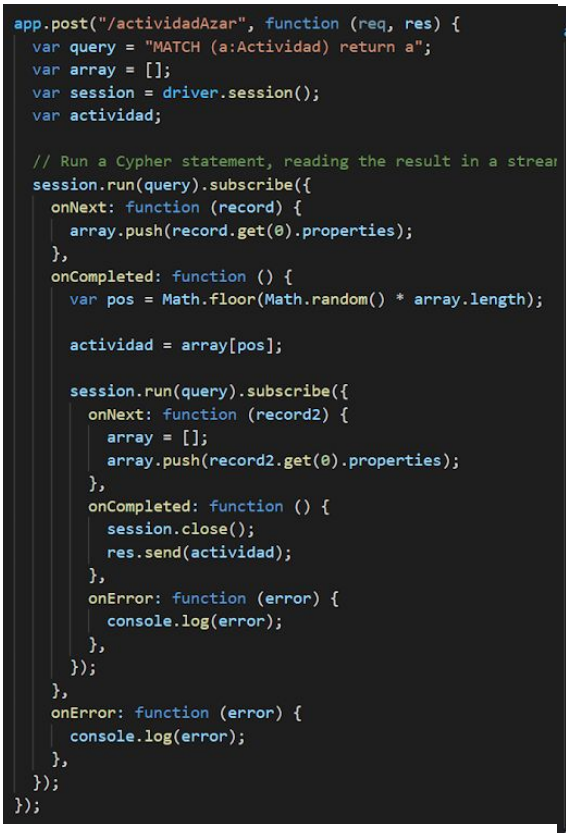
login y usuarios sirven para realizar el registro así como el acceso a la aplicación por parte del usuario, guardando los datos introducidos (nombre, fecha de nacimiento, nombre de usuario y contraseña). Estos datos guardados por login son usados por usuarios a la hora de iniciar sesión. En caso de no existir el usuario o introducir la contraseña de forma incorrecta, la aplicación lanzará un aviso al usuario.

```
app.post("/login", function (req, res) {
  var user = req.body.user;
  var password = req.body.password;

  var query =
    "MATCH (n:Persona) WHERE n.usuario='" +
    user +
    "' AND n.password='" +
    password +
    "' return n";

  const resultPromise = session.run(query);
  resultPromise.then((result) => {
    session.close();

    if (result.records.length == 0) {
      res.send({ message: false });
    } else {
      var record = result.records[0].get(0).properties.usuario;
      res.send(record);
    }
  });
});
```

```
app.post("/usuarios", function (req, res) {
  var user = req.body.usuario;
  var nombre = req.body.nombre;
  var password = req.body.password;
  var nacimiento = req.body.nacimiento;
  var query =
    "CREATE (n:Persona{nombre:'" +
    nombre +
    "', nacimiento:" +
    nacimiento +
    ", usuario:'" +
    user +
    "', password:'" +
    password +
    "'})";

  const resultPromise = session.run(query);
  resultPromise.then((result) => {
    session.close();
    res.send({ message: true });
  });
});
```

La función actividadAzar devuelve una actividad de forma aleatoria entre la lista de todas las posibles cuando se presiona el botón “Actividad al azar”.

Y, por último, la función buscaActividad realiza una búsqueda de todas las actividades posibles en la base de datos que coincidan con los parámetros introducidos por el usuario.



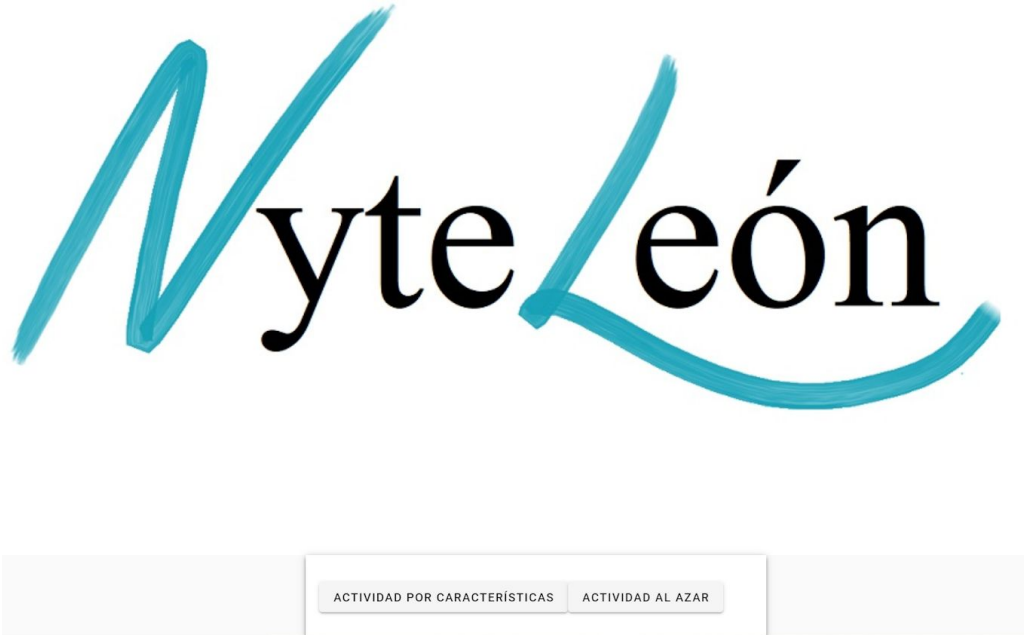
```
app.post("/actividadAzar", function (req, res) {
  var query = "MATCH (a:Actividad) return a";
  var array = [];
  var session = driver.session();
  var actividad;

  // Run a Cypher statement, reading the result in a strea
  session.run(query).subscribe({
    onNext: function (record) {
      array.push(record.get(0).properties);
    },
    onCompleted: function () {
      var pos = Math.floor(Math.random() * array.length);

      actividad = array[pos];

      session.run(query).subscribe({
        onNext: function (record2) {
          array = [];
          array.push(record2.get(0).properties);
        },
        onCompleted: function () {
          session.close();
          res.send(actividad);
        },
        onError: function (error) {
          console.log(error);
        },
      });
    },
    onError: function (error) {
      console.log(error);
    },
  });
});
```

```
app.post("/buscaActividad", function (req, res) {
  var user = req.body.user;
  var entorno = req.body.entorno;
  var modalidad = req.body.modalidad;
  var precio = req.body.precio;
  var localizacion = req.body.localizacion;
  var actividadFisica = req.body.actividadFisica;
  var session = driver.session();

  var query =
    "MATCH(a:Actividad{modalidad:'" +
    modalidad +
    "', precio:'" +
    precio +
    "', localización:'" +
    localizacion +
    "', actividadFisica:'" +
    actividadFisica +
    "'})-[:ES]->(e:Entorno) WHERE e.nombre='" +
    entorno +
    "' return a";

  var array = [];
  var objeto;

  // Run a Cypher statement, reading the result in a stream
  session.run(query).subscribe({
    onNext: function (record) {
      objeto = record.get(0).properties;
      array.push(objeto);
    },
    onCompleted: function () {
      session.close();
      res.send(array);
    },
    onError: function (error) {
      console.log(error);
    },
  });
});

app.listen(port, function () {
  console.log("Example app listening on port " + 3000);
});
```

3.3. Frontend

El Frontend, tal y como he mencionado anteriormente, es la parte de la aplicación que el usuario ve. En NyteLeón, la primera pantalla que el usuario vería sería la siguiente:

Una vez el usuario se registra o inicia sesión en la aplicación pasaría a ver la siguiente pantalla donde se puede elegir una actividad por características o una al azar.



Si se quisiese elegir una actividad por características se mostraría lo siguiente para poder realizar la elección de las características. Aquí se elegirá el tipo de entorno en el que nos gustaría realizar la actividad (interiores o exteriores), la modalidad (si la queremos realizar en grupo o de forma individual), precio (gratis o de pago), localización (para actividades dentro o fuera de León en caso de que se produjese un cierre perimetral de la ciudad debido a la crisis sanitaria) y si queremos que la actividad suponga esfuerzo físico o no.

En cambio, si se escogiese actividad al azar ya se obtendrían los resultados de esa búsqueda sin necesidad de decidir ningún parámetro.

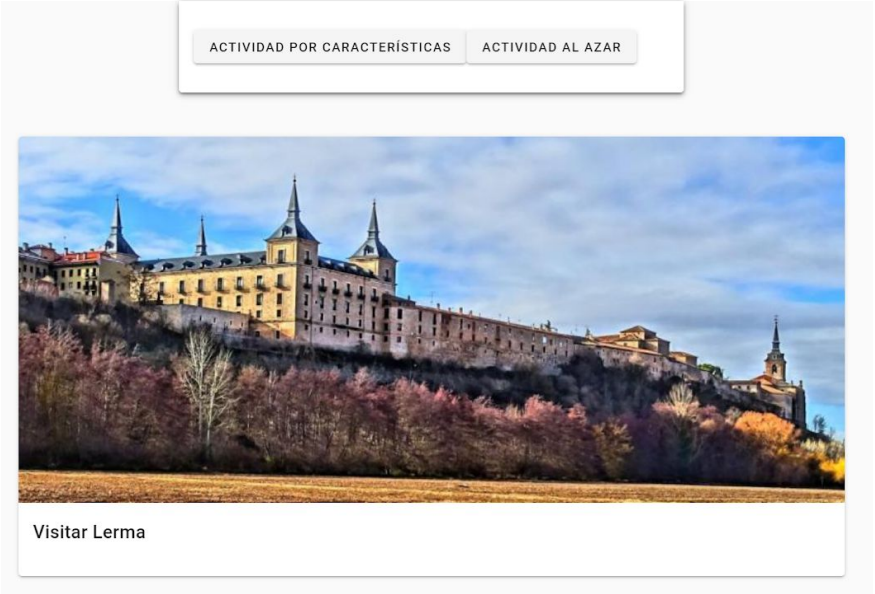
4. Algoritmo de recomendación

El algoritmo en el que se basa esta aplicación consiste en que el usuario seleccione exactamente qué características quiere para su actividad, el Backend envía una petición a la base de datos en Neo4J para obtener los resultados coincidentes con la búsqueda y estas aparecerán listadas con una foto explicativa.

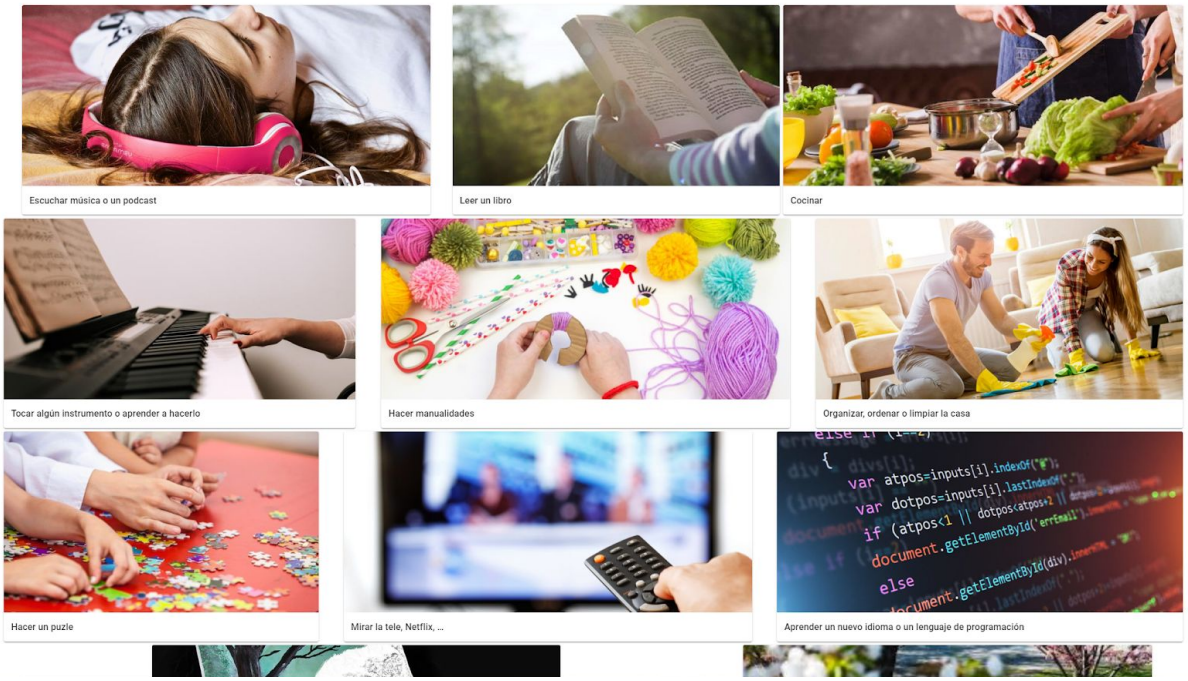
En cambio, si no supiese qué características prefiere para la actividad, podría elegir la opción “Actividad al azar” donde se realiza una búsqueda aleatoria entre la lista de actividades existente mostrando una de cada vez junto con una foto de la actividad.

5. Análisis de resultados

A continuación mostraré el resultado de dos actividades al azar que me ha recomendado la aplicación.



Y ahora algunos de los resultados de buscar actividades en interiores, individuales, gratuitas, dentro de León y sin actividad física.



Y, finalmente, algunos de los resultados de buscar actividades en exteriores, grupales, de pago, fuera de León y con actividad física.

6. Análisis DAFO

Debilidades	Amenazas
La principal debilidad de esta aplicación es que, dado que la base de datos fue elaborada a mano es muy reducida, contando con tan solo 88 posibles actividades.	Una posible amenaza para NyteLeón podrían ser las recomendaciones de qué ver en León que proporciona Google ya que este además recolecta información sobre los usuarios para ofrecerles una recomendación de mayor calidad y precisión.
Fortalezas	**Oportunidades**
NyteLeón es una aplicación innovadora ya que no hay ninguna aplicación en el mercado que proporcione ideas de qué hacer estando en casa o en León en general.	Del mismo modo que a mí me ha servido para descubrir actividades a realizar en León, considero que podría serle útil a mucha otra gente. Por otra parte, se podrían añadir nuevas funcionalidades como la de puntuar las actividades ya realizadas.

7. Líneas de futuro

Tras acabar la aplicación no se me paraban de ocurrir nuevas funcionalidades para añadirle. Una de ellas sería usar la fecha de nacimiento introducida en el registro para mostrar qué actividades suelen realizar más aquellas personas con la misma edad que el usuario.

Otra funcionalidad podría ser valorar las actividades para así mostrarlas por orden de valoraciones en general o por las valoraciones realizadas por gente de la misma edad que tendría el usuario.

8. Lecciones aprendidas

Tras la elaboración de esta aplicación, si algo puedo asegurar es que Ingeniería Informática es una carrera de resistencia, ya que me he visto intentando resolver algunos errores durante días e incluso semanas. A veces, tras resolver esos problemas surgían otros nuevos e incluso me dejó de funcionar la aplicación unos días antes de la fecha de entrega. Tras varios días intentando comprobar qué era resultó ser que simplemente había cambiado la contraseña de la base de datos de 1234 a 123 pero no en el código. Por lo que valoro la importancia de repasar hasta los detalles más ridículos e insignificantes a la hora de buscar errores prestando atención a todos ellos para no perder tiempo.

Por otra parte, no deja de sorprenderme como sin ningún conocimiento previo en aplicaciones web pude realizar esta, ya que la asignatura "Aplicaciones web" la cursé de movilidad internacional por otra semejante pero en la que no se trataba exactamente la materia desde esa perspectiva, entonces el trabajo para realizar esta aplicación web para mi empezó por aprender como estas funcionan, qué es necesario para crearlas y demás.
En resumen, con ganas, trabajo duro y organización se puede acabar resolviendo prácticamente cualquier problema.

9. Bibliografía

[1] https://osf.io/xs7uw/?view_only=b65ab7cb94e048d59f38ac66a72ad5e5

[2] https://www.youtube.com/channel/UCvze3hU6OZBkB1vkhH2lH9Q

[3] https://neo4j.com/graph-algorithms-book/

[4] https://www.kaggle.com/muhadel/hobbies

-Vídeos útiles para la creación de la base de datos:

https://www.youtube.com/watch?v=3VHgmB0SPxQ

-Enlaces útiles para la elaboración del código:

https://vuetifyjs.com/en/components/alerts/

https://www.youtube.com/watch?v=WPlOlF3lWK0&list=PLPl81lqbj-4J-gfAERGDCdOQtVgRhSvIT&index=27

-Enlace al repositorio de GitHub:

https://github.com/lauraiglesiasgondar/nyteleon