🎓 SBH - Study Buddy Hub - MERN STACK PROJECT

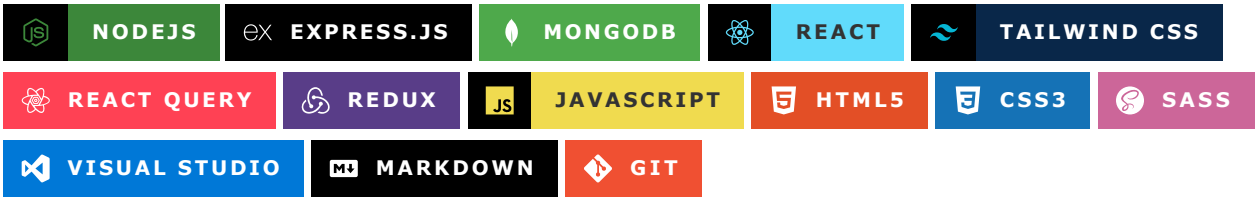
🇮🇹 Se si utilizza la app di prova on line e si riscontrano dei ritardi nella risposta al primo accesso aspettare 1 minuto e riprovare probabilemente il server di render.com gratuito e' andato in sleep mode pochi secondi e dovrebbe partire alla prima richiesta ricevuta
Questa è la documentazione GENERALE del progetto , nelle sottocartelle front e back si troveranno le documentazioni dettagliate per il back end ed il front end. menter nella versione `/assets/documentation.pdf` e `/assets/documentation.md` si trovano i 3 readme generale,back-end e front-end tutti insieme uno sotto l'altro.

🇬🇧 If you use the online test app and experience delays in the response at the first login, wait 5 minutes and try again, probably the free render.com server went into sleep mode for a few seconds and should start at the first request received

💥 Try Now!

StudyBuddyHub

🔨 Tools



❓ Why

This project is the final MERN project for start2impact full stack developer master course.

👊 General

🇮🇹 La richiesta principale dell'esercizio era sviluppare l'autentificazione al sistema e la registrazione utente inoltre bisognava dare qualche funzionalita a piacere . Ispirato da quest'ultimi anni di studio e dagli obiettivi onu a cui a sua volta si inspira s2i e dalle persone conosciute in questa avventura ho voluto fare qualcosa che stimolasse lo studio di gruppo visto che io in prima persona e ho visto tanti altri dopo di me soprattuto all'inizio di un corso si trovano molto propensi allo studio di gruppo , lanciano qualche richiesta sul discord ma non sempre e' facile coordinare le tempistiche ma secondo me e' un esigenza servizio che potrebbe riempire questa piccola lacuna , supportare nell'incontro e nella creazione di piccoli gruppi studio. seguiranno ulteriori dettagli

🇬🇧 The main request of the exercise was to develop system authentication and user registration and it was also necessary to provide some functionality as desired. Inspired by these last years of intense study and by the UN objectives which in turn inspire s2i and by the people I met in this adventure, I wanted to do something that stimulated group study since I personally and I have seen many others afterwards of me,

especially at the beginning of a course, they find themselves very inclined towards group study, they launch some requests on the discord but it is not always easy to coordinate the timing but in my opinion it is a need for a service that could fill this small gap, support in ' meeting and creating small study groups. further details will follow

🎨 Pricipal Functions

🇮🇹 -Gli utenti possono iscriversi o essere fondatori di tre gruppi al massimo contemporaneamente -Gli utenti possono creare un gruppo studio solo se prima scelgono un corso su cui appogiare il gruppo. -Un partecipante ad un gruppo puo' accedere alla chat privata del gruppo. -Un utente puo' cancellarsi da un gruppo e se e' il fondatore eliminare il gruppo. -Solo gli admin possono acceder alla Admin Panel per creare Scuole,Master e Corsi su cui gli utenti potranno appoggiare i loro gruppi studio

-Login tramite mail e password -Sign up che crea la scheda utente ma in stato Pending. poi manda una email che poi andra' aperta e cliccato il link di conferma per rendere l'account Active -Funzione di Forgot password che mandera' una mail con link che ha validita' 10 minuti per eventuale cambio password.

🇬🇧 Users can join or be founders of up to three groups at the same time -Users can create a study group only if they first choose a course on which to base the group. -A participant in a group can access the group's private chat. -A user can unsubscribe from a group and if he is the founder delete the group. -Only admins can access the Admin Panal to create Schools, Masters and Courses on which users can support their study groups

-Login via email and password

- Sign up which creates the user card but in Pending status sends an email which will then be opened and the confirmation link must be clicked to make the account Active -Forgot password function which will send an email with a link valid for 10 minutes for any password change.

📐 General Architecture

Alt text

🇮🇹 Il back end e' basato su NODE-JS ed EXPRESS per la gestione del server , utilizziamo MONGOOSE per aiuatrci a gestire il database MONGO-DB. Il front end e' sviluppato in React con diverse librerie , fra le principal cito ReactQuery(tenstackquery) ed Axios per la gestione delle fetch , context api e reducer per la gestione degli stati, Cookies e Jwt token per la gestione delle autorizzazoni. Tailwind e' la scelta principale per la gestione dello stile.

🇬🇧 The back end is based on NODE-JS and EXPRESS for server management, we use MONGOOSE to help us manage the MONGO-DB database. The front end is developed in React with various libraries, the main ones being ReactQuery (tenstackquery) and Axios for managing fetches, context api and reducer for managing states, Cookies and Jwt tokens for managing authorizations. Tailwind is the leading choice for style management.

🚪 Security

Alt text

🇮🇹 La sicurezza per l'accesso e' gestita con token JWT / HTTP ONLY COOKIES per l'autorizzazione principale alle api. La password viene registrata sul database encryptata.(bcrypt). Invce i token di conferma account e cambio password scordata visto che sono temporanei e gia' spediti via mail ho deciso per questo progetto di lasciarli solo bcrytati senza gestione jwt. C'e un controllo quindi sia lato back end che front end sull'autorizzazione solo gli utenti loggati possono vedere alcuni parti del menu front end ma per molte operazioni sensibili il controllo viene effettuato anche lato back. Una volta registrato un nuovo utente sara' in "status : pending" finche' non clicchera' il link di autorizzazione nella mail ricevuta sull'indirizzo di registrazione. Una volta cliccato sul link lo stato diventera' "Active". I tokens con il link di conferma via mail dopo la registrazione e anche quello per il cambio passowrd provvisorio (che ha validita- di 10 minuti) hanno solo uno livello di crypting con Bcrypt ma non sono jwt come il gettone principale di autentificazione.

Avevo per motivi di studio anche implementato l'accesso tramite "google auth" ed in locale funziona con alcune limitazioni per esempio 100 indirizzi di "test" massimo che possono interagire col progetto , e con notifica aggiuntiva di google che l'app non 'e sicura seguendo queste info per chi fosse interessato ad approfondire : Google OAuth2. In locale funziona bene , e fa riferimento a localhost:4000 (nel mio caso), invece per renderlo operativo con i siti dove ho deployato il back end ed il front end al momento mi e' impossibile , perche' google richiede solo domini di primo livello e SSL e https , cosa che al momento mi e' impossibile o molto difficile gratuitamente . Anche usando servizi di reindirizzamento serve montare un server Linux per esempio che faccia il reindirezzamento.

🇬🇧 Access security is managed with a JWT token. The password is recorded in the encrypted database. However, since the account confirmation and forgotten password change tokens are temporary and already sent via email, for this project I decided to leave them only bcrytated without jwt management. There is therefore a control on both the back end and front end on the authorization only logged in users can see some parts of the front end menu but for many sensitive operations the control is also carried out on the back side. Once registered, a new user will be in "status: pending" until they click the authorization link in the email received on the registration address. Once you click on the link the status will become "Active". The tokens with the confirmation link via email after registration and also the one for the temporary password change (which is valid for 10 minutes) only have one level of encryption with Bcrypt but are not jwt like the main authentication token.

For study reasons I had also implemented access via "google auth" and locally it works with some limitations for example 100 maximum "test" addresses that can interact with the project, and with additional notification from Google that the app is not secure by following this info for those interested in learning more: Google OAuth2. Locally it works well, and refers to localhost:4000 (in my case), however it is impossible for me to make it operational with the sites where I have deployed the back end and the front end at the moment, because Google only requires first domains level and SSL and https, which at the moment is impossible or very difficult for me for free. Even using redirection services you need to set up a Linux server for example that does the redirection.

```
### :triangular_flag_on_post: Note

:it:
Per leggere i campi dipendenti da altre' entita' ricordarsi di usare "populate"
come in questo estratto del progetto :
```

```
:uk:
To read fields dependent on other entities, remember to use "populate" as in this
project excerpt:
```

{ const populateOptions = [{ path: 'school', select: 'name' }, { path: 'master', select: 'name' }, { path: 'course', select: 'name' }, { path: 'participants.user', select: 'userName' }, // Aggiunta per popolare i partecipanti { path: 'founder', select: 'userName' }, // Aggiunta per popolare i partecipanti];

}

🚀 Chat

🇮🇹 la chat funziona di base con dei socket in ascolto sul server. la chat e' in tempo reale ma viene anche registrata sul database in un campo relativo ad ogni gruppo di studio
essendo una funziona sperimentale e non oggeto prncipale del progetto scolastico in essere non entro nei particolari qui nella documentazione ma rimango a disposizione nei contatti in calce per ulteriori info

:ukn: the chat basically works with sockets listening on the server. the chat is in real time but is also recorded on the database in a field relating to each study group
Since it is an experimental function and not the main object of the existing school project, I will not go into detail here in the documentation but I remain at your disposal with the contacts at the bottom for further information

💡 IDEAS

🇮🇹 Il progetto potrebbe avere anche una gestione delle foto profilo ma diventava una scelta da gestire che complicava ancora il progetto , quindi ho fatto un po' di ricerche ma per il momento ho voluto sorvolare. In futuro si potrebbero implementare dei punteggi bonus / malus , che si ottengono da gli altri utenti del gruppo magari in base alla frequenza nel gruppo o altri fattori, si potrebbe cosi decidere se entrare o fare entrare qualcuno nel gruppo con una bassa reputazione. Si potrebbe aggiungere un controllo su quando qualcuno o tutti nel gruppo superano l'esame relativo con punti o badge dedicati.

Si potrebbe pensare di monetizzare in almeno 3 modi:

- funzioni aggiuntive per gli utenti premium , per esempio adesso il numero massimo di gruppi contemporanei e' 3, si potrebbe rendere piu alto o manipolare questo numero dietro un abbonamento premium (occasione per approfondire e testare Stripe o altre librerie per i pagamenti reali) o per esempio rendere il fatto di creare piu' di un gruppo una funzione premium invce iscriversi una funzione free ecc ecc
- gestione dei tutor , utenti esperti che abbiano dimostrato frequenza costanza ed affidabilita' coi punteggi e badge interni, una volta completato un master per esempio potrebbero diventare Tutor, essere sceltri da un gruppo che andranno a seguire e che paghera' una somma extra che andra' al tutor ed in precentuale al sito
- servizio esterno per le scuole , una scuola per inserirsi nell'elenco e aggiunegre i suoi master ed i suoi corsi dovrebbe pagare una somma annuale al sito , ed aggiungere cosi un servizio molto fidelizzante

fra quelli proposti ... in questo caso si potrebbe escludere dalla vista dell'user la scelta delle altre scuole per esempio.

🇬🇧 The project could also have profile photo management but it became a choice to manage which further complicated the project, so I did a bit of research but for the moment I wanted to ignore it. In the future, bonus/malus scores could be implemented, which are obtained from other users in the group, perhaps based on frequency in the group or other factors, so it could be decided whether to enter or have someone with a low reputation join the group. You could add a check on when someone or everyone in the group passes the relevant exam with dedicated points or badges.

You could think of monetizing in at least 3 ways:

- additional functions for premium users, for example now the maximum number of simultaneous groups is 3, this number could be made higher or manipulated behind a premium subscription (an opportunity to delve deeper and test Stripe or other libraries for real payments) or for example, making the fact of creating more than one group a premium function instead of signing up a free function etc etc
- management of tutors, expert users who have demonstrated frequency, consistency and reliability with internal scores and badges, for example once they have completed a master's degree they could become tutors, be chosen by a group that they will follow and who will pay an extra sum which will ' to the tutor and in percentage to the site
- external service for schools, for a school to be included in the list and add its masters and courses it would have to pay an annual sum to the site, and thus add a very loyalty-building service among those offered... in this case it could be excluded from the user's view the choice of other schools for example.

📺 External Service

🇮🇹

Per il database come gia' detto utilizzo il servizio base di Cloud di MongoDb
Per il deploy del backend il servizio gratuito base di render.com
Per il deploy del frontend il servizio gratuito base di netlify.com
Per il servizio di mail transictional il servizio base gratuito di brevo.com
Per il servizio di mail usato in develpoment local mode il servizio gratuito base mailtrap.io

🇬🇧

For the database, as already mentioned, I use the basic Cloud service of MongoDb
For the deployment of the backend the free basic service of render.com
For the deployment of the frontend the free basic service of netlify.com
For the transitional email service, the free basic service of brevo.com
For the email service used in local development mode, the free basic service mailtrap.io

if you want to immediately test the endpoints use this link for the test response from browser or with a GET:

test backend

if you want to immediately see the site : test frontend

💾 Installation

First of all, you need Node.js installed. If you don't have it, you can download it here: Node.js After the installation, you're ready to go. you will find instructions for local installation or deployment both in the \front and \back folders in the readme.md relating to the front-end and back-end in detail.

MIT

📧 Contact Me

Any questions? Send me an e-mail here: claudiodallara77@gmail.com
You can find my Linkedin profile here: https://www.linkedin.com/in/claudio-dall-ara-244816175/

🎓 SBH - Study Buddy Hub - BACK END

🎓 SBH - Study Buddy Hub - BACK END

🇮🇹

Questa è la documentazione BACK END del progetto , nella root la documentazione generale e nella cartella front la documentazione FRONT END

🇬🇧

This is the BACK documentation of the project, in the main folder the GENERAL doc and in front folder the FRONT END doc.

👮🏻 API - Main endpoints

🇮🇹 il back end e' in grado di gestire altri endpoint ma al momento spiego l'utilizzo di quelli utilizzati nel front end

🇬🇧 the back end is able to manage many other endpoints but at the moment I will explain the use of those used in the front end for what is needed for the project.

General

🇮🇹 La `root` sara' per esempio
`localhost:3005` (se il server e' in ascolto come nel mio caso sulla porta 3005)
oppure il il dominio che ospita il server on line nel nostro caso
`https://s2i-study-buddy-hub-4coach.onrender.com/`.
Le rotte pincipali sono 5:

🇬🇧 For example, the `root` will be
`localhost:3005` (if the server is listening on port 3005 as in my case)
or the domain that hosts the online server in our case
> `https://s2i-study-buddy-hub-4coach.onrender.com/`.

`/test<br>` /api/v1/users`<br>` /api/v1/schools`<br>` /api/v1/masters`<br>` /api/v1/courses`<br>` /api/v1/groups`

/test

🇮🇹 La prima rotta e' molto semplice facendo una `GET` rispondera' con 200 SUCCESS quando il server e' on line . la rotta e' pubblica (le rotte pubbliche sono comunque gestite dal limiter per evitare abusi).

🇬🇧 The first route is very simple doing a `GET` will respond with 200 SUCCESS when the server is online. the route is public (public routes are however managed by the limiter to avoid abuse).

/users

🇮🇹 La seconda rotta principale `user` contiene :

🇬🇧 The second main route `user` contains :

```
{

 // THIS ROUTES ARE NOT PROTECT
  GET   - users/confirmAccount/:activeToken
  POST  - users/signup
  POST  - users/login
  GET   - users/forgotPassword
  PATCH - users/resetPassword/:token

  // THIS ROUTE IS PROTECT ONLY WITH JWT IN HEADERS REQ (bearer token)
  GET   - users/validateToken

}
```

GET - users/confirmAccount/:activeToken

The active token is the link receveid in the confirmation email after the signup.

For 200 SUCCES case received the JWT token and activate the account from "Pending" to "Active" state in the database.

POST - users/signup

`body req:{password:string(min 8) , passwordConfirm(equal to passord) , userName:string(min3,max30), email:string-valid-format-Email-Address}`

For 200 SUCCESS case receive the email for Activate the account

POST - users/login

`body req:{password:string(min 8) ,email:string-valid-format-Email-Address}`

For 200 SUCCESS case receive the JWT token for access the site app.

POST - users/forgotPassword

`body req:{email:string-valid-format-Email-Address}`

For 200 SUCCESS case receive the email with the temporary link for change the password.

PATCH- users/resetPassword:token

the token is the token in the link received via mail after the "forget password request" `body req:{password:string(min8),passwordConfirm(string equal to password field )}`

For 200 SUCCESS case receive set the new password in the database.

GET - users/validateToken

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.

`body req:{password:string(min8),passwordConfirm(string equal to password field )}`
`header: jwt bearer token`

For 200 SUCCESS case receive the object with the user details:
Here a response example

```
{
  status: 'success',
  message: 'Token is valid',
  userName: "claudio dallara",
  email: "claudiodallara77@gmail.com"
  role: "user"
  _id: 55556545d54s5d45,
}
```

/schools

🇮🇹 il prossimo endpoint e' /schools . e' un endpoint protetto, bisogna sempre mettere jwt come bearer token nell'headere della richiesta. possono accedere solo gli admin al POST , possono accedere tutti al GET 🇬🇧 the next endpoint is /schools.it is a protected endpoint, you must always put jwt as a bearer token in the request header. Only admins can access POST, everyone can access GET

GET - schools/

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.

For 200 SUCCESS case receive the object data whit an array of the groups details:
Here a response example :

```
    {
    status: 'succes',
    results: doc.length,
    data: { data: doc },
    requestedAt: req.requestTime,
    }
```

POST - schools/

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.
This endpoint is for create a new schools, PROTECT only for "admin"

```
body req:{name:string,site:valid-email-format)}
header: jwt bearer token
```

For 200 SUCCESS case cerate a new school in database and receive the new object create

/schools

🇮🇹 il prossimo endpoint e' /schools . e' un endpoint protetto, bisogna sempre mettere jwt come bearer token nell'headere della richiesta. possono accedere solo gli admin al POST , possono accedere tutti al GET 🇬🇧 the next endpoint is /schools.it is a protected endpoint, you must always put jwt as a bearer token in the request header. Only admins can access POST, everyone can access GET

GET - /schools

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.

For 200 SUCCESS case receive the object data with an array of the groups details:
Here a response example :

```
    {
    status: 'succes',
    results: doc.length,
    data: { data: doc },
    requestedAt: req.requestTime,
    }
```

POST - /schools

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.
This endpoint is for create a new schools, PROTECT only for "admin"

```
body req:{name:string,site:valid-email-format)}
header: jwt bearer token
```

For 200 SUCCESS case create a new school in database and receive the new object create

/masters

🇬🇧 the next endpoint is /masters is a protected endpoint, you must always put jwt as a bearer token in the request header. Only admins can access POST, everyone can access GET

GET - /masters

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.

For 200 SUCCESS case receive the object data with an array of the masters details:
Here a response example :

```
{
status: 'succes',
results: doc.length,
data: { data: doc },
requestedAt: req.requestTime,
}
```

POST - /masters

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.
This endpoint is for create a new master, PROTECT only for "admin"
the relevant school to insert is the index `_id` of the home school to be taken from the "schools" table

```
body req:{name:string,school:objectId}
header: jwt bearer token
```

For 200 SUCCESS case create a new master in database and receive the new object create

/courses

🇬🇧 the next endpoint is /course is a protected endpoint, you must always put jwt as a bearer token in the request header. Only admins can access POST, everyone can access GET

GET - /courses

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.

For 200 SUCCESS case receive the object data with an array of the courses details:
Here a response example :

```
{
status: 'succes',
results: doc.length,
data: { data: doc },
requestedAt: req.requestTime,
}
```

POST - /courses

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.
This endpoint is for create a new master, PROTECT only for "admin"
the relevant school to insert is the index `_id` of the home school to be taken from the "schools" table the relevant master to insert is the index `_id` of the home master to be taken from the "masters" table

```
body req:{name:string,school:objectId),master:objectId)}
header: jwt bearer token
```

For 200 SUCCESS case create a new course in database and receive the new object create

/groups

🇬🇧 the next endpoint is /groups is a protected endpoint, you must always put jwt as a bearer token in the request header. Everyone (use,admin,mod ecc)can access GET and POST.

GET - /groups

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.

For 200 SUCCESS case receive the object data with an array of the groups details:
Here a response example :

```
{
status: 'succes',
results: doc.length,
data: { data: doc },
requestedAt: req.requestTime,
}
```

POST - /courses

This endpoint is referred to with a get and putting req in the heder. as bearer token the authorization jwt.
This endpoint is for create a new group
the relevant school to insert is the index `_id` of the home school to be taken from the "schools" table the relevant master to insert is the index `_id` of the home master to be taken from the "masters" table the relevant course to insert is the index `_id` of the home course to be taken from the "courses" table

```
body req:{name:string,school:objectId,master:objectId,course:objectId}
header: jwt bearer token
```

For 200 SUCCESS case create a new group in database (the founder is the user who create the group)and receive the new object create

Other Endpoint

🇮🇹 ci sono altri endpoint pronti nel backend come per esempio, il getOne per le scuole/master/corsi/gruppi, il deleteMe , getMe per l'user, il cambio password ma da loggati ed altro ... ma visto che non sono sviluppati sul front end del progetto sorvolo

🇬🇧 there are other endpoints ready in the backend such as, for example, getOne for schools/masters/courses/groups, deleteMe, getMe for the user, changing password but when logged in and more... but since they are not developed on the front end of the flyover project

📘 DataBase MONGO DB MONGOOSE

🇮🇹 Il database e' sviluppato in MongoDb con servizio cloud integrato.

🇬🇧 The database is developed in MongoDb with integrated cloud service. There are 5 main tables : -User -Group -Course -Master -School

SCHOOL SCHEMA (MONGOOSE) :

```
{
  name: {
  type: String,
  required: true,
  unique: true,
  },
  site: {
  type: mongoose.SchemaTypes.Url,
  }
}
```

MASTER SCHEMA (MONGOOSE) :

```
{
  name: {
    type: String,
    required: true,
  },
  school: {
    type: mongoose.Schema.ObjectId,
    ref: 'School',
    required: [true, 'Course must belong to a school.'],
  }
  }
```

COURSE SCHEMA (MONGOOSE) :

```
{
   name: {
    type: String,
    required: true,
  },
  master: {
    type: mongoose.Schema.ObjectId,
    ref: 'Master',
    required: [true, 'Course must belong to a master.'],
  },
  school: {
    type: mongoose.Schema.ObjectId,
    ref: 'School',
    required: [true, 'Course must belong to a school.'],
  },
  }
```

GROUP SCHEMA (MONGOOSE) :

```
{
  name: {
    type: String,
    required: [true, 'Group must have a name'],
    unique: true,
    index: true, // altrimenti unique non funziona
  },
  course: {
    type: mongoose.Schema.ObjectId,
    ref: 'Course',
    required: [true, 'Group must refer to a course'],
  },
  master: {
    type: mongoose.Schema.ObjectId,
    ref: 'Master',
    required: [true, 'Group must refer to a master'],
  },
  school: {
    type: mongoose.Schema.ObjectId,
    ref: 'School',
    required: [true, 'Group must refer to a school'],
  },
  founder: {
    type: mongoose.Schema.ObjectId,
    ref: 'User',
    required: [true, 'Group must refer to a founder'],
  },
  participants: [
    {
      user: {
        type: mongoose.Schema.ObjectId,
```

```
          ref: 'User',
        },

        dateStart: { type: Date, default: Date.now() },
        dateEnd: { type: Date, default: null },
      },
    ],

    maxParticipants: {
      type: Number,
      default: 2,
    },
    currentParticipantsNumber: {
      type: Number,
      virtual: true,
      get: function () {
        return this.participants.length;
      },
    },
    chat: [
      {
        user: {
          type: String,
          required: [true, 'Chat message must have a user'],
        },
        message: {
          type: String,
          required: [true, 'Chat message must have a message'],
        },
        date: {
          type: Date,
          default: Date.now,
          required: [true, 'Chat message must have a date'],
        },
      },
    ]
    }
```

USER SCHEMA (MONGOOSE) :

```
  {
   userName: {
      type: String,
      required: [true, 'A user must have a name'],
      minlength: 3,
      maxlength: 30,
    },
    email: {
      type: String,
      required: [true, 'Please provide email'],
      unique: true,
```

```
      lowercase: true,
      validate: [validator.isEmail, 'Please provide a valid email'],
    },
    role: {
      type: String,
      enum: ['user', 'mod', 'admin', 'tutor'],
      default: 'user',
    },
    password: {
      type: String,
      required: [true, 'Please provide a password'],
      minlength: 8,
      select: false,
    },
    passwordConfirm: {
      type: String,
      required: [true, 'Please provide a confirm password'],
      validate: {
        // This only works on CREATE and SAVE !!!
        validator: function (el) {
          return el === this.password;
        },
        message: 'Passwords are not the same!',
      },

      select: false,
    },
    passwordChangedAt: {
      type: Date,
      select: false,
    },
    passwordResetToken: {
      type: String,
      select: false,
    },
    passwordResetExpires: {
      type: Date,
      select: false,
    },
    activeToken: {
      type: String,
      select: false,
    },
    status: {
      type: String,
      enum: ['Pending', 'Active', 'Ban'],
      default: 'Pending',
    }
  }

  ## :hammer: Tools

  ![Javascript](https://img.shields.io/badge/Javascript-F0DB4F?style=for-the-
  badge&labelColor=black&logo=javascript&logoColor=F0DB4F)
```

```
![Nodejs](https://img.shields.io/badge/Nodejs-3C873A?style=for-the-
badge&labelColor=black&logo=node.js&logoColor=3C873A)
![Express.js](https://img.shields.io/badge/Express.js-000000?style=for-the-
badge&logo=express&logoColor=white)
![MongoDB](https://img.shields.io/badge/MongoDB-4EA94B?style=for-the-
badge&logo=mongodb&logoColor=white)
![Markdown](https://img.shields.io/badge/Markdown-000000?style=for-the-
badge&logo=markdown&logoColor=white)
![VSCode](https://img.shields.io/badge/Visual_Studio-0078d7?style=for-the-
badge&logo=visual%20studio&logoColor=white)
![Git](https://img.shields.io/badge/Git-F05032?style=for-the-
badge&logo=git&logoColor=white)

## :dart: Settings for env file e Render.com

:it:
Per semplicita' ho aggiunto un file `config.fake` dovrebbe servire per
semplificare la stesura del file stesso in locale o nel modo in qui si settono le
variabile d'ambiente nel sistema di deploy scelto. Nel mio caso su render si puo'
caricare un file intero con copia incolla del suo contenuto (prima tolgo le righe
commentate ) ed in un solo copia incolla si riescono a caricare tutte le variabili
d'ambiente altrimenti si possono caricare una alla volta.
:uk:
For simplicity I added a `config.fake` file which should serve to simplify the
drafting of the file itself locally or in the way in which the environment
variables are set in the chosen deployment system. In my case on render you can
load an entire file with copy and paste of its contents (first I remove the
commented lines) and in a single copy and paste you can create all the environment
variables otherwise they can be loaded one at a time.

![Screen Render Env](/assets/pictures/pictures/renderEnv.png 'Screen Render Env')

:it:
Per il settings del progetto Render.com ci chiedera' di scegliere una repo da
Github, nel nostro progetto indicheremo la cartella back perche' il link della
repo punta all'intero progetto invece noi vogliamo scendere nella cartella
`\back`.
Il progetto e' da lasciare cosi come e' sara' cura di Render.com creare la build ,
e poi deployare e lanciare il server.
Inoltre attenzione ad inserire come build command "yarn" e come start command
`node server.js`nel mio caso o il file principale del back in generale.
Scegliamo yarn anche se in locale usaimo npm perche render funziona meglio cosi
per node js.
:uk:
For the project settings Render.com will ask us to choose a repo from Github, in
our project we will indicate the back folder because the repo link points to the
entire project instead we want to go to the `\back` folder.
Also be careful to insert "yarn" as the build command and `node server.js` as the
start command in my case or the main back file in general.
We choose yarn even if we use npm locally because render works better this way for
node js.

![Setting Render 1](/assets/pictures/pictures/render1Setting.png 'Setting Render
1')
```

```
![Setting Render 2](/assets/pictures/pictures/render2Setting.png 'Setting Render
2')

:it:
La variabile NODE_ENV e' impostata su develpment nell'esempio questo vuol dire che
puntera' al frontend locale impostato su localhost:4000 .
Se si vuole puntare al front end di produzione in questo progetto
https://studybuddyhub.netlify.app allora commentare la riga # NODE_ENV=development
e toglierw il commento # alla riga NODE_ENV=production.
Per lavorare su server locale far partire il programma da locale ricordarsi di
coordinare il frontend in tal caso .
Altra differenza in cui incide la variabile NODE_ENV e' che se in production
allora usa il servizio di BREVO e manda mail reali se invece in development
utilizza il servizio fittizio di MAILTRAP.
:uk:
The NODE_ENV variable is set to developpment in the example, this means that it
will point to the local frontend set to localhost:4000.
If you want to point to the production front end in this project
https://studybuddyhub.netlify.app then comment out the line # NODE_ENV=development
and uncomment # the line NODE_ENV=production.
To work on a local server, start the program locally, remember to coordinate the
frontend in this case.
Another difference affected by the NODE_ENV variable is that if in production it
uses the BREVO service and sends real emails while in development it uses the
fictitious MAILTRAP service.
```

{ # NODE_ENV=production NODE_ENV=development

```
# PORT SETTING
PORT=3005

# NODE VERSION IMPORTANT FOR SETTING
# RENDER ON COM DEVELOP
NODE_VERSION=18.17.1

# GENERAL FRONT SIDE ADDRESS
FRONT_SITE_WEB=https://studybuddyhub.netlify.app
FRONT_SITE_LOCAL=http://localhost:4000

# VARIABLE AFFECTED ONLY FOR GOOGLE AUTH
# WHICH IS CURRENTLY LOCALLY ONLY IN DEV MODE
BACK_SITE_WEB=http://localhost:3005
CLIENT_ID=866088888888-k7s0obpjca3rj75mjnaah0dk704rrrw3.apps.googleusercontent.com
CLIENT_SECRET=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

# MONGODB SETTINGS
USER_NAME=claudiodallara77
DATABASE_PASSWORD=sD1K22PwkBertyKs
DATABASE=mongodb+srv://claudiodallara77:
```

```
<password>@cluster0.2efcc4w.mongodb.net/S2iStudyBuddyHub?
retryWrites=true&w=majority

# BREVO SETTINGS
API_KEY_BREVO=xkeysib-
6d64537333e73730129b8rtdgf785458758cf756621984cd2bf7777777bdb666-yTvfhlT2vyjSa3KX

# JWT SETTINGS
JWT_SECRET=fullstack-project-secret.007-dallaRussiaConFurore-Cesena
JWT_EXPIRES_IN=90d
JWT_COOKIE_EXPIRES_IN=90

# MAIL TRAP SETTINGS
EMAIL_USERNAME=9e77fd5c0a564e
EMAIL_PASSWORD=80127255r6b921
EMAIL_HOST=sandbox.smtp.mailtrap.io
EMAIL_PORT=25
```

}

```
:it:
Ho voluto provare anche a gestire una chat. Ho scoperto l'utilizzo dei socket ed
ho inziato a sperimentare un po' ... il file principale che gestisce la chat e'
socketManager.js

:uk:
I also wanted to try managing a chat. I discovered the use of sockets and started
experimenting a bit... the main file that manages the chat is socketManager.js

## :floppy_disk: Installation

:it:
Prima di tutto, è necessario che Node.js sia installato.
Se non ce l'hai puoi scaricarlo qui:
[Node.js](https://nodejs.org/it/download/)
Dopo l'installazione, sei pronto per partire.

:uk:
First of all, you need Node.js installed.
If you don't have it, you can download it here:
[Node.js](https://nodejs.org/it/download/)
After the installation, you're ready to go.

### 1 - Clone the repository

`git clone https://github.com/boobaGreen/S2I-STUDY_BUDDY_HUB_4COACH`

IMPORTANT!! - NOW go to the FOLDER "back" :
`cd back`
```

```
### 2 - Install the dependencies

REMEBER: we are in the "back" folder now!

`npm install`

### 3 - Setting the config.env file

:it:
Sopra abbiamo già elencato tutte le variabili d'ambiente da impostare.
Ricordo di lasciare decommentata la modalità desiderata se sviluppo punterà al
front end locale e utilizzerà "mailtrap" di default, se produzione allora punterà
al front end indicato come web e utilizzerà "brevo" per inviare email vere e
proprie

:uk:
Above we have already listed all the environment variables to be set.
I remember to leave the desired mode uncommented if development will point to the
local front end and will use "mailtrap" by default, if production then it will
point to the front end indicated as web and will use "brevo" to send real emails
```

{ # NODE_ENV=production NODE_ENV=development` }

```
### 4 - Start it

add this scripts at your "package.json" file :
```

"scripts": { "start": "SET NODE_ENV=development&&nodemon server.js", "start:prod": "SET NODE_ENV=production&&nodemon server.js", },

```
`npm start` - start in DEV mode default (error's message are set for developers)
`npm start:prod ` - start in PROD mode (error's messages are set for clients)

[MIT](https://choosealicense.com/licenses/mit/)

## :e-mail: Contact Me

Any questions? Send me an e-mail here: claudiodallara77@gmail.com <br>
You can find my Linkedin profile here: https://www.linkedin.com/in/claudio-dall-
ara-244816175/
```

🎓 SBH - Study Buddy Hub - Front End

🇮🇹

Questa è la documentazione di FronEnd React per un progetto MERN più ampio.

🇬🇧

This is the FronEnd React Documentation for a larger MERN project.

❓ Why

This project is the final MERN project for start2impact full stack developer master course.

💡 How it Works

🇮🇹 La app avra' una Cover che e' la "Home page" quando non si e' loggati. Le altre voci del menu per i non loggati saranno "Sign Up" , "Login" ed "About" che e' l'unica pagina tra queste insieme a "Page Not Found" per i path non gestiti che saranno uguali per gli utenti autenfificati e no. Il menu per gli utenti autentificati invece sara' "Home"(dove si gestiranno i gruppi studio a cui si e' iscritti), "Groups" dove si potra creare un nuovo gruppo ed iscriversi ad altri oltre che navigare fra i gruppi.Ci sara' la pagina "About" che come gia' detto sara' l'unica (se non consideriamo "Page not found")accessibile allo stesso modo da utenti autentificati o no.Ci sara' ovviamente la pagina per il "Logout" ed infine solo per utenti "admin" un pannello per creare nuove scuole , master e corsi a cui i gruppi faranno riferimento.

:en: The app will have a Cover which is the "Home page" when you are not logged in. The other menu items for non-logged in users will be "Sign Up", "Login" and "About" which is the only page among these together with "Page Not Found" for unmanaged paths which will be the same for authenticated users and no. The menu for authenticated users will instead be "Home" (where you will manage the study groups you have signed up to), "Groups" where you can create a new group and subscribe to others as well as navigate between the groups. There will be ' the "About" page which, as already mentioned, will be the only one if we do not consider "Page not found" accessible in the same way by authenticated or non-authenticated users. There will obviously be the "Logout" page and finally only for users "admin" a panel to create new schools, masters and courses to which the groups will refer.

:: ScreenShot

🇮🇹 la app oltre alla gestione dell'autentificazione piuttosto complessa lato gestione ma che in realta' non richiede un manuale per l utente particolare...ha il resto dell'app e' appena accennata, idee per renderla piu completa e complessa nel readme del progetto principale nella root folder.

🇬🇧 the app in addition to the rather complex authentication management on the management side but which in reality does not require a manual for the particular user... the rest of the app is just mentioned, ideas to make it more complete and complex in the project readme root in the root folder.

Example screenshot>

Setting Render 2 cover about signup login cardmygroup chat groups logout

🎨Styles

🇮🇹 Per Gestire lo stile ho usato Tailwind. Lo stile e' volutamente molto semplice ma ho voluto cercare di rendere il codice leggibile senza astrarre troppo . Ho quindi creato Elementi riutilizzabili alcuni anche personalizzabili , ed ho usato comunque una serie di variabili colore di base personalizzati per rendere piu veloce possibile eventual cambiamenti futuri , la tabella si trova nel file `index.css` nella root. Le mediaquery sono molto semplici ed anche il menu' principale diventa a scomparsa in caso di schermo piccoli.

:en: To manage the style I used Tailwind. The style is deliberately very simple but I wanted to try to make the code readable without abstracting too much. I therefore created reusable elements, some of which were also customizable, and I still used a series of customized basic color variables to make any future changes as fast as possible, the table is found in the index.css file in the root. The media queries are very simple and even the main menu disappears on small screens.

📚 Dependencies

🇮🇹 spiegazione delle librerie usate per scopo: 🇬🇧 explanation of libraries used for purpose:

general react

"react": "^18.2.0" "react-dom": "^18.2.0" "react-router-dom": "^6.18.0"

fetching

For react query and axios settings "@tanstack/react-query": "^5.17.19", "@tanstack/react-query-devtools": "^5.17.21" "axios": "^1.6.1"

Jwt Token and http only cookies

"js-cookie": "^3.0.5", "jsonwebtoken": "^9.0.2"

form

"react-hook-form": "^7.48.2", "@hookform/resolvers": "^3.3.2", "yup": "^1.3.2"

toggle dark mode

"react-dark-mode-toggle": "^0.2.0"

icons

"react-icons": "^4.11.0"

spinners

"react-loader-spinner": "^6.1.6" "react-spinners": "^0.13.8",

carousel

"react-slick": "^0.30.1", "slick-carousel": "^1.8.1",

chat

"socket.io-client": "^4.7.2", "react-scroll-to-bottom": "^4.2.0",

🐾 Settings config.env

🇮🇹 Di file config.env ne ho pensati 2 , sono da creare quindi 2 file : il primo si chiamra' `.env.development` ed avra' nel mio caso co il server in locale che gira su localhost:3005 :

🇬🇧 I thought of 2 config.env files, so 2 files need to be created: the first will be called `.env.development` and in my case will have the local server running on localhost:3005

```
{VITE_APP_BASE_URL=http://localhost:3005/api/v1
VITE_APP_BASE_URL_SOCKET=http://localhost:3005}
```

🇮🇹 il secondo si chiamera' `.env.production` ed avra' nel mio caso:

🇬🇧 the second will be called `.env.production` and will have in my case:

```
{VITE_APP_BASE_URL=https://s2i-study-buddy-hub-4coach.onrender.com/api/v1
VITE_APP_BASE_URL_SOCKET=https://s2i-study-buddy-hub-4coach.onrender.com}
```

📺 Deploy on netlify

Deploy for free on netlify.com

if you want to immediately test the on line app : STUDYBUDDYHUB

💾 Installation

First of all, you need Node.js installed.
If you don't have it, you can download it here: Node.js
After the installation, you're ready to go.

1 - Clone the repository

`git clone https://github.com/boobaGreen/S2INodeJsPOF`

IMPORTANT!! - NOW go to the FOLDER "front"" : `cd back`

2 - Install the dependencies

REMEBER: we are in the "front" folder now!

`npm install`

3 - Start it

`npm run dev` for development mode, open the browser on http://localhost:4000/

📄 License

MIT

📧 Contact Me

Any questions? Send me an e-mail here: claudiodallara77@gmail.com
You can find my Linkedin profile here: https://www.linkedin.com/in/claudio-dall-ara-244816175/