# RedBit

The documentation for users can be found at the repositories [wiki page](https://github.com/RedBit-devs/RedBit/wiki)

## [About](https://github.com/RedBit-devs/RedBit/wiki)

This project was made as a project for school.

## Goals

We aim to make a fully open source chat. With minimal data collection, and being fully transparent about that minimal data towards the user. We believe that everyone who uses a program should be given the opportunity to modify the software they use. (or contribute to the main project)

# Starter

Look at the [Nuxt documentation](https://nuxt.com/docs/getting-started/introduction) to learn more.

## Setup

Make sure to install dependencies:

npm install

## Development Server

Start the development server on http://localhost:3000:

npm run dev

## Production

Build the application for production:

**Warning:** ***There should be no spaces in the filepath when building***

npm run build

Locally preview production build:

npm run preview

Check out the [deployment documentation](https://nuxt.com/docs/getting-started/deployment) for more information.

# Development

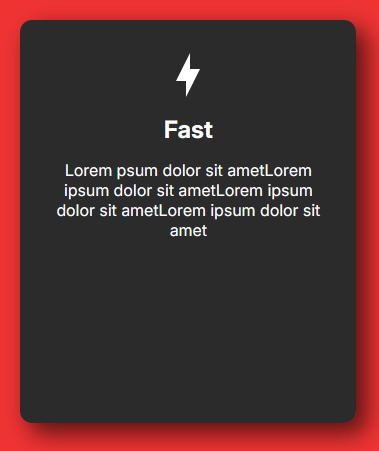
The documentation for the API can be found in the [user documentation on the wiki page](https://github.com/RedBit-devs/RedBit/wiki/API)

## Components

### [BasicShowcaseCard](https://github.com/RedBit-devs/RedBit/blob/main/components/BasicShowcaseCard.vue)

It is used for displaying information in a card form on the landingpage. For the “picture” eather an icon or an image can be provided. To use an icon you must start it with icon:

Picture

#### Props

"cardData": {  
 type: Object as PropType<CardData>  
 }

Types

type CardData = {  
 imageUrl?: string;  
 headerText?: string;  
 bubbles?: Bubble[];  
 description?: string;  
};  
type Bubble = {  
 url: string;  
 name: string;  
 imageUrl: string;  
};

### [ChatCard](https://github.com/RedBit-devs/RedBit/blob/main/components/ChatCard.vue)

Picture

kép

kép

#### Props

'name': {  
 type: String  
},  
'picture': {  
 type: String,  
 default: ""  
},  
'activity': {  
 type: String,  
 default: ""  
}

### [ChatMessage](https://github.com/RedBit-devs/RedBit/blob/main/components/ChatMessage.vue)

used for displaying messages sent. ### Props

authorImage: {  
 type: String,  
 required: true,  
 default: ""  
},  
authorName: {  
 type: String,  
 required: true  
},  
message: {  
 type: String,  
 required: true  
},

### [ChatInputField](https://github.com/RedBit-devs/RedBit/blob/main/components/ChatInputFiled.vue)

The input field whre the user can type in thier messages and send them to other users or to a chat room. ### Props

"send": {  
 type: Function,  
 default: () => { }  
 },  
 "route": {  
 type: String,  
 default: ""  
 }

### [CheckFriendProfileCard](https://github.com/RedBit-devs/RedBit/blob/main/components/CheckFriendProfileCard.vue)

It’s a dialog what the user can open by clicking on it’s friend’s user name. It shows some informaiton abaout the profile what the user clicked on. The user can navigate between four pages where it can see the common servers or friends and can manage the friendship.

### Props

closeDialogFunc: {  
 type: Function,  
 default: () => { }  
 },  
  
 isDialogOpen: {  
 type: Boolean,  
 default: ref(false)  
 }

### [CreateServer](https://github.com/RedBit-devs/RedBit/blob/main/components/CreateServer.vue)

It’s a modal where the user can create it’s own serever what opens up after the user clicked the create server button.

### Props

isShown: {  
 type: Boolean,  
 default: false,  
 },  
 closeDialogFunc: {  
 type: Function,  
 default: () => { },  
 }

## Database

For the database we use [prisma.io](https://www.prisma.io/) ORM. The database schema can be found in the projects root folder [/prisma/schema.prisma](https://github.com/RedBit-devs/RedBit/blob/main/prisma/schema.prisma)

## External

### [Mailing](https://support.google.com/a/answer/176600)

# Hosting

For production, after building the project successfully the builded files can be found at .output/server/index.mjs. It can be run with node

node .output/server/index.mjs

## ENV

An [example for the env](https://github.com/RedBit-devs/RedBit/blob/main/.env.example) is found in the projects root directory. - DATABASE\_URL - A connection string for the database used - Can use connection pooling

* DIRECT\_URL
  + Direct connection to the database. Used for migrations.
  + A connection string for the database used
  + Do not use connection pooling
* JWT\_SECRET
  + JWT token encription key
* NUXT\_NODEMAILER\_FROM
  + Email address used for sending emails to the user
* NUXT\_NODEMAILER\_AUTH\_PASS
  + Mailing secret

[SMTP server provider used](https://support.google.com/a/answer/176600)