62160002 นาย วิชญ์ ชาญเฉลิมชัย

ไดอะแกรมของโปรแกรม

ส่งข้อมูลคำสั่งไปแสดง

รับข้อมูล user name และ email แล้วส่งไปให้

login.ejs

เรียกใช้ขุ้อมูลจาก database

เชื่อมต่อ database เพื่อเขียนคำสั่ง

index.js

เชื่อมต่อ database เพื่อดึงข้อมูล User และ email มาใช้

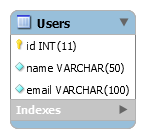
database.js

จัดรูปแบบ

home.ejs

style.css

รูป ER DIAGRAM



คำอธิบายข้อมูล(Data Dictionary)

ไฟล์ใน Folder node\_modules ทั้งหมดเป็นไฟล์ที่เกี่ยวกับ node js ทั้งหมดเพื่ออัปหน้าเว็บขึ้น web server

----------------------------------------------------------------------------------------------------------

ไฟล์ index.js

มีหน้าที่

* รับข้อมูลที่ผู้ใช้พิมพ์ทั้งในหน้า home.ejs และ login.ejs บันทึกเข้าไปใน database
* ส่งข้อความการพิมพ์โต้ตอบระหว่างผู้ใช้มาแสดงผลในหน้า home.ejs
* const socket = require('socket.io');
* const express = require('express');
* const path = require('path');
* const cookieSession = require('cookie-session');
* const dbConnection = require('./database');
* const {body, validationResult} = require('express-validator');
* const { isObject } = require('util');
* const app = express();
* app.use(express.urlencoded({extended:false}));
* app.set('view engine','ejs');
* app.set('views', path.join(\_\_dirname,'views'));
* app.use(express.static(path.join(\_\_dirname, 'public')));
* app.use(cookieSession({
* name:'session',
* keys:['key1','key2'],
* maxAge:3600\*1000 // 1hr
* }));
* // DECLARING CUSTOM MIDDLEWARE
* const ifNotLoggedin = (req, res, next) => {
* if(!req.session.isLoggedIn){
* return res.render('login');
* }
* next();
* }
* const ifLoggedin = (req,res,next) => {
* if(req.session.isLoggedIn){
* return res.redirect('/home');
* }
* next();
* }
* // END OF CUSTOM MIDDLEWARE
* // ROOT PAGE
* app.get('/', ifNotLoggedin, (req,res,next) => {
* dbConnection.execute("SELECT `name` FROM `users` WHERE `id`=?",[req.session.userID])
* .then(([rows]) => {
* res.render('home',{
* name:rows[0].name
* });
* });
* });
* // END OF ROOT PAGE
* ตรวจสอบการ login ใช้คู่กับไฟล์ login.ejs
* เช็คว่าอีเมลล์ที่ใส่มามีใน database หรือไม่ หากไม่มีให้ขึ้นข้อความ อีเมลล์ไม่ถูกต้อง(Invalid Email)
* เช็คว่าผู้ใช้ได้ใส่ชื่อผู้ใช้มาไหม ถ้าไม่ได้ใส่ ขึ้นข้อความว่า โปรดกรอก User name ไม่เกิน 8 ตัวอักษร

app.post('/', ifLoggedin, [

body('user\_name','โปรดกรอก User name ไม่เกิน 8 ตัวอักษร').trim().not().isEmpty().isLength({ max: 8 }),

body('user\_email').custom((value) => {

return dbConnection.execute('SELECT email FROM users WHERE email=?', [value])

.then(([rows]) => {

if(rows.length == 1){

return true;

}

return Promise.reject('อีเมลล์ไม่ถูกต้อง(Invalid Email)');

});

}),

],

* อัปเดตชื่อ username ตามที่ผู้ใช้กรอกลงไปใน database

(req, res,next) => {

const validation\_result = validationResult(req);

const {user\_name, user\_email} = req.body;

if(validation\_result.isEmpty()){

dbConnection.execute("SELECT \* FROM `users` WHERE `email`=?",[user\_email])

.then(([rows]) => {

req.session.isLoggedIn = true;

req.session.userID = rows[0].id;

res.redirect('/');

}).catch(err => {

if (err) throw err;

});

dbConnection.execute("SELECT \* FROM `users` WHERE `email`=?",[user\_email])

.then(([rows]) => {

dbConnection.query("UPDATE `users` SET `name`=? WHERE `email`=? ", [user\_name, user\_email])});

}

else{

let allErrors = validation\_result.errors.map((error) => {

return error.msg;

});

// REDERING login-register PAGE WITH LOGIN VALIDATION ERRORS

res.render('login',{

login\_errors:allErrors

});

}

});

// END OF LOGIN PAGE

* ใช้สำหรับ logout ออกในหน้าแชท เพื่อกลับไปหน้า login ใหม่

// LOGOUT

app.get('/logout',(req,res)=>{

//session destroy

req.session = null;

res.redirect('/');

});

// END OF LOGOUT

* ใช้สำหรับกรณีที่หน้าเพจมีปัญหา จะขึ้นข้อความ 404 Page Not Found!
* app.use('/', (req,res) => {
* res.status(404).send('<h1>404 Page Not Found!</h1>');
* });
* ตั้งค่า port localhost:5000 เพื่อให้เข้าถึง server ได้

const PORT = 5000;

const server = app.listen(PORT, function () {

console.log(`Listening on port ${PORT}`);

console.log(`http://localhost:${PORT}`);

});

* เป็นส่วนของฟังก์ชั่นที่ส่งค่าไปให้กับ server เมื่อมีผู้ใช้พิมพ์ข้อความ มีคน connect เข้าแชทหรือมีคน disconnect ออกจากแชท
* // Socket setup
* const io = socket(server);
* const activeUsers = new Set();
* io.on("connection", (socket) => {
* console.log("Connected");
* socket.on("new user", function (data) {
* socket.userId = data;
* activeUsers.add(data);
* io.emit("new user", [...activeUsers]);
* });
* socket.on("disconnect", () => {
* activeUsers.delete(socket.userId);
* io.emit("user disconnected", socket.userId);
* });
* socket.on("chat message", function (data) {
* io.emit("chat message", data);
* });
* socket.on("typing", function (data) {
* socket.broadcast.emit("typing", data);
* });
* });

---------------------------------------------------------------------------------------------------------

ไฟล์ home.ejs

มีหน้าที่

* แสดงรายชื่อผู้ใช้งานที่เชื่อมต่อเข้ามาโดยรับข้อมูลจาก database
* แสดงหน้าต่างแชทแบบ realtime
* แสดงข้อความว่ามีใครกำลังพิมพ์แชทอยู่
* มีช่องสำหรับผู้ใช้เอาไว้พิมพ์ข้อความโดยผู้ใช้สามารถกดส่งข้อความได้โดยการกด enter
* แสดงจำนวนตัวอักษรที่ผู้ใช้สามารถพิมพ์ได้
* <body>
* <h1><center>REAL TIME CHAT</center></h1>
* <div class="container">
* <div class="inbox">
* <div class="inbox\_\_people-list">
* <h4>UserOnlines</h4>
* <div class="inbox\_\_people"></div>
* </div>
* <div class= "inbox\_\_messages ">
* <div class="messages\_\_history"></div>
* <div class="fallback"></div>
* </div>
* <a href="/logout" class="btn btn-danger inbox\_logout"><p><span class="bg"></span><span class="base"></span><span class="text">logout</span></p></a>
* </div>
* <div class="charnum">
* <p id="charNum">พิมพ์ได้อีก 100 ตัว</p>
* </div>
* <form class="message\_form inbox\_input" >
* <input type="text" class="message\_form\_\_input" maxlength="100" placeholder="Type a message" onkeyup="countChars(this);" />
* </form>
* </div>
* Link style ไปยังไฟล์ style.css

<link rel="stylesheet" href="/css/style.css">

* เชื่อมต่อกับ socket.io
* <script src="https://cdn.socket.io/3.1.3/socket.io.min.js" integrity="sha384-cPwlPLvBTa3sKAgddT6krw0cJat7egBga3DJepJyrLl4Q9/5WLra3rrnMcyTyOnh"
* crossorigin="anonymous"></script>
* ฟังก์ชั่นสำหรับนับจำนวนตัวอักษรโดยที่ค่าเต็มของตัวอักษรที่สามารถพิมพ์ได้คือ 100 ตัว
* function countChars(obj) {
* var maxLength = 100;
* var strLength = obj.value.length;
* var charRemain = (maxLength - strLength);
* document.getElementById("charNum").innerHTML = 'พิมพ์ได้อีก ' + charRemain + ' ตัว';
* }
* กำหนดตัวแปรสำหรับ div แต่ละช่อง
* const socket = io();
* const inboxPeople = document.querySelector(".inbox\_\_people");
* const inputField = document.querySelector(".message\_form\_\_input");
* const messageForm = document.querySelector(".message\_form");
* const messageBox = document.querySelector(".messages\_\_history");
* const fallback = document.querySelector(".fallback");
* let userName = "";
* ฟังก์ชั่นแสดงผู้ใช้ที่พึ่งเข้ามาและผู้ใช้ที่อยู่ในแชททั้งหมด
* const newUserConnected = (user) => {
* userName = user || "<%=name%>";
* socket.emit("new user", userName);
* addToUsersBox(userName);
* };
* const addToUsersBox = (userName) => {
* if (!!document.querySelector(`.${userName}-userlist`)) {
* return;
* }
* const userBox = `
* <div class="chat\_ib ${userName}-userlist">
* <h2>${userName}</h2>
* </div><br><br>`;
* inboxPeople.innerHTML += userBox;
* };
* ฟังก์ชั่นแสดงชื่อผู้ใช้ที่พิมพ์ข้อความและวันเวลาที่พิมพ์ข้อความ และ ประวัติข้อความที่พิมพ์ไปแล้ว
* const addNewMessage = ({ user, message }) => {
* const time = new Date();
* const formattedTime = time.toLocaleString("en-US", { hour: "numeric", minute: "numeric", day: "numeric" ,month: "2-digit", year: "2-digit"});
* const receivedMsg = `
* <div class="incoming\_\_message">
* <div class="received\_\_message">
* <p>${message}</p>
* <div class="message\_\_info">
* <span class="message\_\_author">${user}</span>
* <span class="time\_date">${formattedTime}</span>
* </div>
* </div>
* </div>`;
* const myMsg = `
* <div class="outgoing\_\_message">
* <div class="sent\_\_message">
* <p>${message}</p>
* <div class="message\_\_info">
* <span class="time\_date">${formattedTime}</span>
* </div>
* </div>
* </div>`;
* messageBox.innerHTML += user === userName ? myMsg : receivedMsg;
* };
* ฟังก์ชั่นแสดงการทำงานทั้งหมดของหน้าต่าง real time chat
* newUserConnected();
* messageForm.addEventListener("submit", (e) => {
* e.preventDefault();
* if (!inputField.value) {
* return;
* }
* socket.emit("chat message", {
* message: inputField.value,
* nick: userName,
* });
* inputField.value = "";
* });
* inputField.addEventListener("keyup", () => {
* socket.emit("typing", {
* isTyping: inputField.value.length > 0,
* nick: userName,
* });
* });
* socket.on("new user", function (data) {
* data.map((user) => addToUsersBox(user));
* });
* socket.on("user disconnected", function (userName) {
* document.querySelector(`.${userName}-userlist`).remove();
* });
* socket.on("chat message", function (data) {
* addNewMessage({ user: data.nick, message: data.message });
* });
* socket.on("typing", function (data) {
* const { isTyping, nick } = data;
* if (!isTyping) {
* fallback.innerHTML = "";
* return;
* }
* fallback.innerHTML = `<p>${nick} กำลังพิมพ์...</p>`;
* });
* </script>

-------------------------------------------------------------------------------------------------------

ไฟล์ database.js

* เชื่อมต่อกับ database ใน http://localhost/phpmyadmin/db\_structure.php?server=1&db=realtimechat
* const mysql = require('mysql2');
* const dbConnection = mysql.createPool({
* host : 'localhost', // MYSQL HOST NAME
* user : 'root', // MYSQL USERNAME
* password : '', // MYSQL PASSWORD
* database : 'realtimechat' // MYSQL DB NAME
* }).promise();
* module.exports = dbConnection;

-------------------------------------------------------------------------------------------------------

ไฟล์ login.ejs

ในส่วนของ body จะมี

* ช่องรับข้อมูล username
* ช่องรับข้อมูล email
* ปุ่ม Enter Chat

เมื่อใส่ข้อมูลครบและถูกต้องแล้วกดปุ่ม Enter Chat จะส่งข้อมูลไปยัง database และ เปลี่ยนไปเป็นหน้าแชท

<body>

<div class="col-lg-6 col-md-8 login-box">

<h1 class="text-center mb-5 text-uppercase text-muted">Real Time Chat</h1>

<div class="text-uppercase center">

<form action="" method="POST">

<div class="form-group">

<label for="\_email">User Name</label>

<input type="text" name="user\_name" id="\_rname" class="form-control" placeholder="Name" maxlength="8"

rnodeequired>

</div>

<div class="form-group">

<label for="\_email">Email</label>

<input type="text" name="user\_email" id="\_email" class="form-control" placeholder="Email" rnodeequired>

</div>

<% if (locals.login\_errors) {

login\_errors.forEach(function(error\_msg){ %>

<div class="alert alert-danger" role="alert">

<%= error\_msg %>

</div>

<% });

} %>

<a class = "horizontal">

<button type="submit" class="text">Enter Chat</button>

</a>

</form>

</div>

</div>

</body>

ในส่วนของ style

.center {

margin: auto;

width: 50%;

border: 3px;

padding: 10px;

text-align: center;

}

* กำหนดให้ tag div เป็น block

div {

display: block;

}

* กำหนดสีพื้นหลังและ font
* body {
* background: #222D32;
* font-family: 'Roboto', sans-serif;
* }
* กำหนดสีให้ USER NAME และ EMAIL
* label{
* color: #dedede;
* }
* กำหนด style ให้ login-box
* .login-box {
* margin-top: 50px;
* margin-left: 25%;
* height: auto;
* background: #1A2226;
* text-align: center;
* box-shadow: 0 3px 6px rgba(0, 0, 0, 0.16), 0 3px 6px rgba(0, 0, 0, 0.23);
* position: relative;
* width: 100%;
* min-height: 1px;
* padding-right: 15px;
* padding-left: 15px;

} .login-btm {

float: left;

}

.login-button {

padding-right: 0px;

text-align: right;

margin-bottom: 25px;

}

.login-text {

text-align: left;

padding-left: 0px;

color: #A2A4A4;

}

.loginbttm {

padding: 0px;

}

* กำหนด style ให้ช่องใส่ Name และ Email
* input[type=text] {
* background-color: #1A2226;
* border: none;
* border-bottom: 2px solid #0DB8DE;
* border-top: 0px;
* border-radius: 0px;
* font-weight: bold;
* outline: 0;
* margin-bottom: 20px;
* padding-left: 0px;
* color: #ECF0F5;
* }
* input[type=password] {
* background-color: #1A2226;
* border: none;
* border-bottom: 2px solid #0DB8DE;
* border-top: 0px;
* border-radius: 0px;
* font-weight: bold;
* outline: 0;
* padding-left: 0px;
* margin-bottom: 20px;
* color: #ECF0F5;
* }
* กำหนดstyleและระยะหว่างระหว่าง Name และ Email

.form-group {

margin-bottom: 40px;

outline: 0px;

}

.form-control:focus {

border-color: inherit;

-webkit-box-shadow: none;

box-shadow: none;

border-bottom: 2px solid #0DB8DE;

outline: 0;

background-color: #1A2226;

color: #ECF0F5;

}

input:focus {

outline: none;

box-shadow: 0 0 0;

}

label {

margin-bottom: 0px;

}

.form-control-label {

font-size: 10px;

color: #6C6C6C;

font-weight: bold;

letter-spacing: 1px;

}

* กำหนด Style ให้ปุ่ม Enter Chat

button {

background-color: Transparent;

background-repeat:no-repeat;

border: none;

cursor:pointer;

overflow: hidden;

outline:none;

width: 100%;

height: 100%;

} :root {

     --base-color: rgba(255, 255, 255, 1);

     --hover-color: rgba(220, 120, 150, 1);

}

.horizontal {

     display: flex;

     justify-content: center;

     align-items: center;

     position: relative;

width: 400px;

right: 35px;

     height: 120px;

margin: 20px;

     border: 10px solid var(--base-color);

     font-size: 4rem;

cursor: pointer;

}

.horizontal > .text {

     position: relative;

     color: transparent;

     background-image: linear-gradient(90deg, var(--hover-color) 0%, var(--hover-color) 50%, var(--base-color) 50%, var(--base-color) 100%);

     background-repeat: repeat;

     background-size: 200%;

     background-position: 100% 0;

     -webkit-background-clip: text;

     background-clip: text;

     transition: background-position 300ms;

}

.horizontal::before {

     content: '';

     position: absolute;

     top: 0;

     left: 0;

     width: 100%;

     height: 100%;

     background: var(--base-color);

     transform-origin: 100% 0;

     transform: scale3d(0, 1, 1);

     transition: transform 300ms;

}

.horizontal:hover .text {

     background-position: 0 0;

}

.horizontal:hover::before {

     transform-origin: 0 0;

     transform: scale3d(1, 1, 1);

}

--------------------------------------------------------------------------------------------------------

ไฟล์ style.css

กำหนด Style กับ home.ejs

* กำหนด font พื้นฐาน
* \* {
* margin: 0;
* padding: 0;
* box-sizing: border-box;
* font-family: 'Nunito', sans-serif;
* }
* กำหนดสีพื้นหลังหน้าเว็บ
* html, body {
* background: linear-gradient(120deg, rgba(23, 190, 187, 1), rgba(240, 166, 202, 1));
* overflow: hidden;
* }
* กำหนด style ให้ div container
* .container{
* display: flex;
* justify-content: center;
* align-items: center;
* flex-direction: column;
* height: 100vh;
* width: 100vw;
* }
* กำหนด style ให้หน้าต่างแชท
* .inbox {
* width: 600px;
* height: 65%;
* border-radius: 0.2em;
* position: relative;
* box-shadow: rgba(0, 0, 0, 0.35) 0px 5px 15px;

} .inbox\_input {

width: 600px;

border-radius: 0.1em;

position: absolute;

bottom: 42px

}

.inbox\_logout {

margin: 40px;

bottom: -20px;

position: absolute;

right: 0;

}

* กำหนด style ให้กับช่องพิมพ์ข้อความ
* .message\_form\_\_input{
* background: rgba(60, 241, 238, 0.02);
* position: absolute;
* bottom: 18px;
* border: none;
* width: 73%;
* padding: 1.2em;
* outline: none;
* color: rgba(3, 1, 1, 0.9);
* font-weight: 300;
* }
* กำหนด style หั้บช่องแสดง UserOnlines หรือช่องแสดงผู้ที่ใช้งานอยู่
* .chat\_ib{
* float: left;
* text-align: left;
* margin: -0.25em 2em;
* font-size: 0.7em;
* font-weight: 300;
* color: rgb(255, 255, 255);
* width: 200px;
* background:rgb(0, 183, 0);
* width: 0.7em;
* height: 0.7em;
* border-radius: 100%;
* margin: 1em 0.7em;
* position: relative

}

.inbox\_\_people-list-bot{

border-radius: 0.2em;

position: relative;

}

.inbox\_\_people-list {

background: rgba(255, 255, 255, 0.1);

width: 25%;

height: 100%;

float: right;

border-top-right-radius: 0.2em;

border-bottom-right-radius: 0.2em;

}

.inbox\_\_people-list h4 {

background: rgba(255, 255, 255, 0.05);

color: rgba(255, 255, 255, 0.9);

font-size: 0.9em;

padding: 1em;

margin: 0;

font-weight: 300;

text-align: center;

}

* กำหนด scoll bar ให้กับหน้าต่างแชท
* inbox\_\_messages::-webkit-scrollbar-track
* {
* -webkit-box-shadow: inset 0 0 6px rgba(0,0,0,0.3);
* border-radius: 10px;
* background-color: #F5F5F5;
* }
* .inbox\_\_messages::-webkit-scrollbar
* {
* width: 12px;
* background-color: #F5F5F5;
* }
* .inbox\_\_messages::-webkit-scrollbar-thumb
* {
* border-radius: 10px;
* -webkit-box-shadow: inset 0 0 6px rgba(0,0,0,.3);
* background-color: #555;
* }
* กำหนด style ให้หน้าต่างแชท
* .inbox\_\_messages{
* overflow-y: scroll;
* overflow-x: hidden;
* width: 450px;
* background: rgba(0, 0, 0, 0.1);
* height: 91%;
* border-top-right-radius: 0.2em;
* border-bottom-right-radius: 0.2em;
* }
* กำหนด style ให้กับกล่องข้อความ ทั้งข้อความเข้า และ ข้อความที่ส่งออกไป
* .outgoing\_\_message , .incoming\_\_message {
* background: rgb(255,193,193);
* background: linear-gradient(90deg, rgba(255,193,193,1) 27%, rgba(115,255,213,1) 100%);
* padding: 1em 0;
* height: auto;
* width: 60%;
* border-radius: 5px;
* margin: 2em 1em;
* }
* .outgoing\_\_message {
* float: right;
* }
* .incoming\_\_message {
* float: left;
* }
* .sent\_\_message {
* color: rgb(0, 0, 0);
* font-size: 1em;
* margin: 0 1em;
* }
* .received\_\_message {
* color: rgb(0, 0, 0);
* font-size: 1em;
* margin: 0 1em;
* }
* กำหนด style ให้กับข้อมูลชื่อผู้ส่งข้อความและวันเวลาที่ส่งข้อความ
* .message\_\_info{
* margin: 0 2em;
* }
* .message\_\_author,.time\_date{
* color: rgb(0, 0, 0);
* font-size: 10px;
* margin: 1 0em;
* position: relative;
* left:73px ;
* top: 10px;
* }
* กำหนด style ให้tag h1 และ h2

h1{

display: block;

font-size: 2em;

margin-block-start: 0.67em;

margin-block-end: 0.67em;

margin-inline-start: 0px;

margin-inline-end: 0px;

font-weight: bold;

color: rgb(177, 230, 255);opacity:0.01;

}

h2{

position: relative;

text-align: left;

left: 32px;

bottom: 5px;

}

* กำหนด style ให้กับ fallback(ข้อความที่บอกว่าใครกำลังพิมพ์ข้อความในแชทอยู่)
* .fallback{
* text-align: left;
* position: fixed;
* bottom: 40px;
* left:100;
* }
* กำหนด style ให้กับข้อความบอกจำนวนตัวอักษรที่สามารถพิมพ์ได้
* .charnum{
* position: absolute;
* right: 41%;
* bottom: 30px;
* }
* กำหนด style ให้กับปุ่ม LOGOUT
* div a {
* width: 25%;
* max-width: 240px;
* height: 54px;
* padding: 8px;
* font-size: 0.8rem;
* font-weight: 900;
* color: #ff4655;
* text-align: center;
* text-transform: uppercase;
* text-decoration: none;
* box-shadow: 0 0 0 1px inset rgba(236, 232, 225, 0.3);
* position: relative;
* margin: 10px 0;
* left: 410px;
* }
* div a.white:hover > p {
* color: #ece8e1;
* }
* div a.white > p {
* background: #ece8e1;
* color: #0f1923;
* }
* div a.white > p span.base {
* border: 1px solid transparent;
* }
* div a.transparent:hover > p {
* color: #ece8e1;
* }
* div a.transparent:hover > p span.text {
* box-shadow: 0 0 0 1px #ece8e1;
* }
* div a.transparent > p {
* background: #0f1923;
* color: #ece8e1;
* }
* div a.transparent > p span.base {
* border: 1px solid #ece8e1;
* }
* div a:after, div a:before {
* content: "";
* width: 1px;
* position: absolute;
* height: 8px;
* background: #0f1923;
* left: 0;
* top: 50%;
* transform: translateY(-50%);
* }
* div a:before {
* right: 0;
* left: initial;
* }
* div a p {
* margin: 0;
* height: 54px;
* line-height: 54px;
* box-sizing: border-box;
* z-index: 1;
* left: 0;
* width: 100%;
* position: relative;
* overflow: hidden;
* }
* div a p span.base {
* box-sizing: border-box;
* position: absolute;
* z-index: 2;
* width: 100%;
* height: 100%;
* left: 0;
* border: 1px solid #ff4655;
* }
* div a p span.base:before {
* content: "";
* width: 2px;
* height: 2px;
* left: -1px;
* top: -1px;
* background: #0f1923;
* position: absolute;
* transition: 0.3s ease-out all;
* }
* div a p span.bg {
* left: -5%;
* position: absolute;
* background: #ff4655;
* width: 0;
* height: 100%;
* z-index: 3;
* transition: 0.3s ease-out all;
* transform: skewX(-10deg);
* }
* div a p span.text {
* z-index: 4;
* width: 100%;
* height: 100%;
* position: absolute;
* left: 0;
* top: 0;
* }
* div a p span.text:after {
* content: "";
* width: 4px;
* height: 4px;
* right: 0;
* bottom: 0;
* background: #0f1923;
* position: absolute;
* transition: 0.3s ease-out all;
* z-index: 5;
* }
* div a:hover {
* color: #ece8e1;
* }
* div a:hover span.bg {
* width: 110%;
* }
* div a:hover span.text:after {
* background: #ece8e1;
* }