



รายงานโครงงาน

ITDS271 Computer and Communication Security

จัดทำโดย

นาย สิริวิชญ์ ไตรสรินันท์ รหัสนักศึกษา 6487095

เสนออาจารย์ผู้สอน

Dr. Songpon Teerakanok

Dr. Ittipon Rassameeroj

Dr. Assadarat Khurat

โครงงานนี้เป็นส่วนหนึ่งของรายวิชา

ITDS271 Computer and Communication Security

ภาคการศึกษา 2/2022 มหาวิทยาลัยมหิดล

คำนำ

รายงานฉบับนี้จัดทำขึ้นเพื่อเป็นส่วนหนึ่งของรายวิชา ITDS271 Computer and Communication Security โดยมีวัตถุประสงค์เพื่อการเรียนรู้ และสามารถพัฒนาโปรแกรมที่สามารถแสดงหน้าจอการเข้าสู่ระบบเพื่อทำ authentication โดยใช้รหัสผ่านและรูปแบบการพิมพ์รวมถึงสามารถที่จะนำไปประยุกต์ใช้ได้

ทั้งนี้ทางผู้จัดทำขอขอบพระคุณอาจารย์ทั้ง 3 ท่าน ได้แก่ Dr.Songpon Teerakanok และ Dr. Ittipon Rassameeroj และ Dr. Assadarat Khurat ที่ให้ความรู้เกี่ยวกับ เนื้อหาการเรียนในภาคเรียนที่ 2/2565 นี้ ทางคณะผู้จัดทำ คาดหวังอย่างยิ่งว่ารายงานฉบับนี้จะให้ความรู้ และเป็นประโยชน์ในการศึกษาแก่ผู้ที่อ่านและสนใจ หากมี ข้อเสนอแนะประการใด ทางคณะผู้จัดทำขอน้อมรับไว้และขออภัยมา ณ ที่นี้ด้วย

ผู้จัดทำ

นาย สิริวิชญ์ ไตรสินันท์

สารบัญ

Client folder	
Node _module	4
Index.html	4
Manifest.json	5
Homepage.js	5
Login.js	6-12
Example login	13
Register.js	14-18
Example terminal after login	19
MongDB compass app	20
App.css	21
App.js	22
Index.js	23
reportWebVitals.js	23
Package.json	23
Package-lock.json	24
Readme.md	24
Server folder	
User.js	25
Index.js	26
Iservice.js	27
Package.json	27
Package-lock.json	27
Readme.md	27
Link for presentation	28
Reference	28

In client folder contains:

1. node_modules folder : for install modules that use in this project
2. public folder that contains:
 - 2.1 index.html :

```
client > public > index.html > html > head
1  <!DOCTYPE html>
2  <html lang="en">
3    <head>
4      <meta charset="utf-8" />
5      <!-- sets the icon for the website, typically displayed in the browser tab.-->
6      <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
7
8      <!-- sets the viewport size to be the device's width, and sets the initial zoom level to 1.-->
9      <meta name="viewport" content="width=device-width, initial-scale=1" />
10     <!-- sets the color of the browser chrome (such as the address bar) to black.-->
11     <meta name="theme-color" content="#000000" />
12
13     <!--provides a brief description of the web page for search engines and social media.-->
14     <meta
15       name="description"
16       content="Web site created using create-react-app"
17     />
18
19     <!-- links to an external script, in this case Tailwind CSS. -->
20     <script src="https://cdn.tailwindcss.com"></script>
21
22     <!-- sets the title of the web page, which appears in the browser tab. -->
23     <title>Keystroke Login</title>
24   </head>
25
26   <body>
27     <noscript>You need to enable JavaScript to run this app.</noscript>
28     <div id="root"></div>
29   </body>
30 </html>
31
```

2.2 manifest.json : The manifest.json file is the only file that every extension using WebExtension APIs must contain.

```
client > public > {} manifest.json > ...
1  {
2    "short_name": "React App",
3    "name": "Create React App Sample",
4    "icons": [
5      {
6        "src": "favicon.ico",
7        "sizes": "64x64 32x32 24x24 16x16",
8        "type": "image/x-icon"
9      }
10   ],
11   "start_url": ".",
12   "display": "standalone",
13   "theme_color": "#000000",
14   "background_color": "#ffffff"
15 }
16 |
```

3. src folder : for source

3.1 components folder contains:

3.1.1 homepage folder that contains:

3.1.1.1 homepage.js

```
client > src > components > homepage > JS homepage.js > ...
1  import React from 'react'
2  const Homepage = ({...prop}) => {
3    const {name, username, password, userbiokey} = prop
4
5    return (
6      <>
7        <h1>Welcome to Homepage which is only visible when you are logged in ${name}</h1>
8        <h6>${username}</h6>
9        <h6>${password}</h6>
10       <h6>${userbiokey}</h6>
11      </>
12    )
13  }
14  export default Homepage
15
```

Explanation: This is the Homepage functional component in React. It accepts object destructured props such as name, username, password, and userbiokey. It returns JSX code that displays the name, username, password, and userbiokey data on the Homepage component.

3.1.2 login folder that contains:

3.1.2.1 login.js

```
// importing React hooks and axios library
import React, { useEffect, useState } from 'react'
import axios from 'axios';
// importing useHistory hook from react-router-dom
import { useHistory } from "react-router-dom"

const Login = () => {
  // initialize useHistory hook
  const history = useHistory()

  // initialize user state with empty values
  const [user, setUser] = useState({
    username: "",
    password: "",
    userbiokey: -1
  })

  // initialize keyevent state with empty arrays
  const [keyevent, setkeyevent] = useState({
    DU: [], //dwel time
    UD: [], //flight time
    DD: [], //type speed
  })

  var lastup = -1;
  var lastdown= -1;
```

```

// handle form input changes
const handleChange = (key, e) => {
  var value = e
  if (key !== "userbiokey") {
    value = e.target.value
  }
  setUser({
    ...user, //spread operator
    [key]: value,
  })
}

// handle key event changes
const handleSetkeyevent = (key, value) => {
  setkeyevent({
    ...keyevent, //spread operator
    [key]: value,
  })
}

// handle key events during typing
const captureKeyEvent = (e) => {

  if (e.key.length > 1) { // special key
    if (e.key === "Backspace") {
      handleSetkeyevent("DU", []) // reset dwel time
      handleSetkeyevent("UD", []) // reset flight time
      handleSetkeyevent("DD", []) // reset type speed
      return;
    }
  }

  if (e.type === "keydown") { // if keydown event
    if (lastup >= 0) { // if the last key pressed is up

```

```

        const flight = e.timeStamp - lastup // calculate
flight time
        keyevent.UD.push(flight) // set flight time
        console.log("Flight ", keyevent.UD, "key ", e.key);
    }
    if (lastdown >= 0) { // if the last key pressed is down
        const typespeed = e.timeStamp - lastdown // calculate
type speed
        keyevent.DD.push(typespeed) // set type speed
        console.log("Type Speed ", keyevent.DD, "key ",
e.key);
    }
    lastdown = e.timeStamp // set lastdown to current
timestamp
    console.log("lastdown ", lastdown);
} else { // if keyup event
    if (lastdown >= 0) { // if the last key pressed is down
        const dwel = e.timeStamp - lastdown // calculate dwel
time
        keyevent.DU.push(dwel) // set dwel time
    }
    lastup = e.timeStamp // set lastup value
    console.log("lastup ", lastup);
    console.log("Dwell ", keyevent.DU, "key ", e.key);
};
}

useEffect(() => { // add event listeners for key events
    const passwordfeild =
document.getElementById('sign-in-password')

    passwordfeild.onkeydown = captureKeyEvent; // listen for
keydown event
    passwordfeild.onkeyup = captureKeyEvent; // listen for keyup
event

```



```

    }, []);

    const handleRegister = async(e) => {
        e.preventDefault()
        try {
            // Make a POST request to the specified URL with user
            data
            await axios.post("http://localhost:3001/login",
            user).then((res) => {
                // Extract the user data from the response and
                display a welcome message
                var user = res.data;
                alert(`hello ${user.name}`);
                // Log the user data to the console
                console.log({ user });
                // Redirect to the home page
                history.push("/")
            })
        } catch (err) {
            // If there is an error, check if it is an Axios error
            if (axios.isAxiosError(err)) {
                console.log({err});
                // If it is, display the error message from the
                response
                alert(err.response.data.message);
                console.log(user.userbiokey, " not match with
                existing biokey");
                // Redirect to the home page
                history.push("/")
            }
        }
    }

    function handlelogin(e) {
        e.preventDefault()

```

```

        console.log("handle login");
        // Calculate the user's biokey based on the keyevent data
        let sum = keyevent.DU[keyevent.DU.length - 1]; // dwell time
        for (let i = 0; i < keyevent.DD.length; i++) {
            sum += keyevent.DD[i];
        }
        let n = keyevent.DD.length + 1
        user.userbiokey = sum / n
        console.log("userbiokey ", user.userbiokey);
        // Call the handleRegister function to register the user
        handleRegister(e)
    }

    return (
        <div className="flex flex-col w-full max-w-md px-4 py-8
bg-blue-200 rounded-lg shadow sm:px-6 md:px-8 lg:px-10">
            <div className="self-center mb-6 text-xl font-light
text-white sm:text-2xl">
                Login To Your Account
            </div>
            <div className="mt-8">
                <form action="#" autoComplete="off"
onSubmit={handlelogin}>
                    <div className="flex flex-col mb-2">
                        <div className="flex relative ">
                            <span className="rounded-l-md inline-flex
items-center px-3 border-t bg-white border-l border-b
border-gray-300 text-gray-500 shadow-sm text-sm">
                                <svg width="15" height="15"
fill="currentColor" viewBox="0 0 1792 1792"
xmlns="http://www.w3.org/2000/svg">
                                    <path d="M1792 710v794q0 66-47
113t-113 47h-1472q-66 0-113-47t-47-113v-794q44 49 101 87 362 246 497
345 57 42 92.5 65.5t94.5 48 110 24.5h2q51 0 110-24.5t94.5-48
92.5-65.5q170-123 498-345 57-39 100-87zm0-294q0 79-49 151t-122
123q-376 261-468 325-10 7-42.5 30.5t-54 38-52 32.5-57.5 27-50

```

```

9h-2q-23
0-50-9t-57.5-27-52-32.5-54-38-42.5-30.5q-91-64-262-182.5t-205-142.5q
-62-42-117-115.5t-55-136.5q0-78 41.5-130t118.5-52h1472q65 0 112.5
47t47.5 113z">

        </path>
    </svg>
</span>
    <input type="text" id="sign-in-name"
className="rounded-r-lg flex-1 appearance-none border
border-gray-300 w-full py-2 px-4 bg-white text-gray-700
placeholder-gray-400 shadow-sm text-base focus:outline-none
focus:ring-2 focus:ring-purple-600 focus:border-transparent"
name="username" value={user.username} onChange={e =>
handleChange('username', e)} placeholder="Your username" />
</div>
</div>
<div className="flex flex-col mb-6">
    <div className="flex relative ">
        <span className="rounded-l-md inline-flex
items-center px-3 border-t bg-white border-l border-b
border-gray-300 text-gray-500 shadow-sm text-sm">
            <svg width="15" height="15"
fill="currentColor" viewBox="0 0 1792 1792"
xmlns="http://www.w3.org/2000/svg">
                <path d="M1376 768q40 0 68 28t28
68v576q0 40-28 68t-68 28h-960q-40 0-68-28t-28-68v-576q0-40
28-68t68-28h32v-320q0-185 131.5-316.5t316.5-131.5 316.5 131.5
316.5q0 26-19 45t-45 19h-64q-26
0-45-19t-19-45q0-106-75-181t-181-75-181 75-75 181v320h736z">
            </path>
        </svg>
    </span>
    <input id="sign-in-password" className="
rounded-r-lg flex-1 appearance-none border border-gray-300 w-full
py-2 px-4 bg-white text-gray-700 placeholder-gray-400 shadow-sm
text-base focus:outline-none focus:ring-2 focus:ring-purple-600

```

```

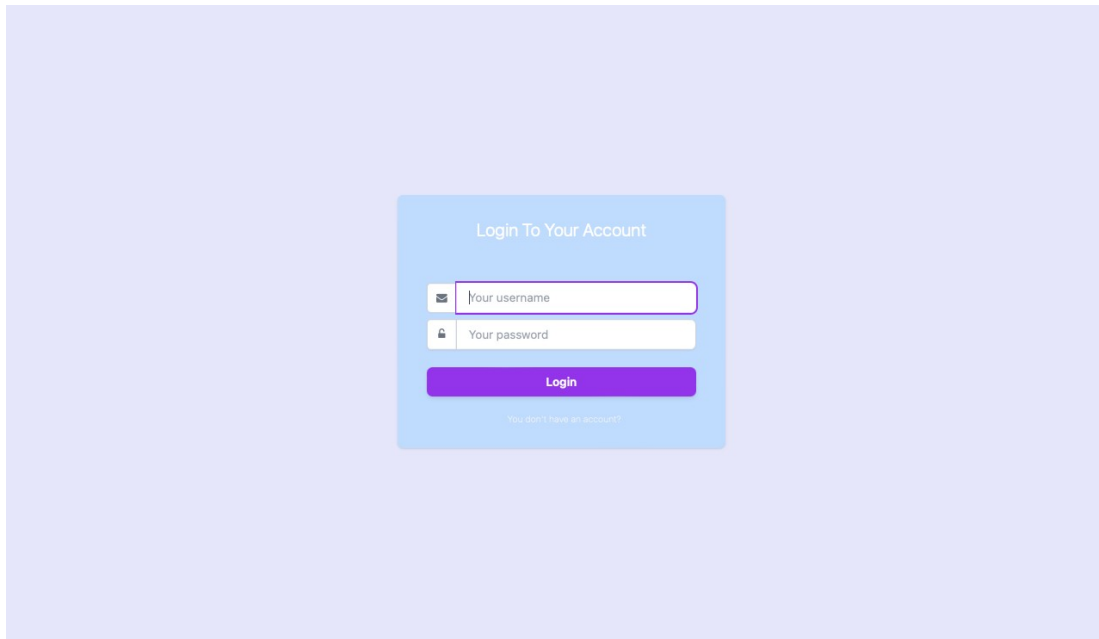
focus:border-transparent"      name="password"      value={user.password}
onChange={e    =>    handleChange('password',    e)}    placeholder="Your
password" />

                                {/* <input type="password"
id="sign-in-password" class=" rounded-r-lg flex-1 appearance-none
border border-gray-300 w-full py-2 px-4 bg-white text-gray-700
placeholder-gray-400 shadow-sm text-base focus:outline-none
focus:ring-2 focus:ring-purple-600 focus:border-transparent"
name="password" value={user.password} onChange={captureInputEvent}
placeholder="Your password" /> */}

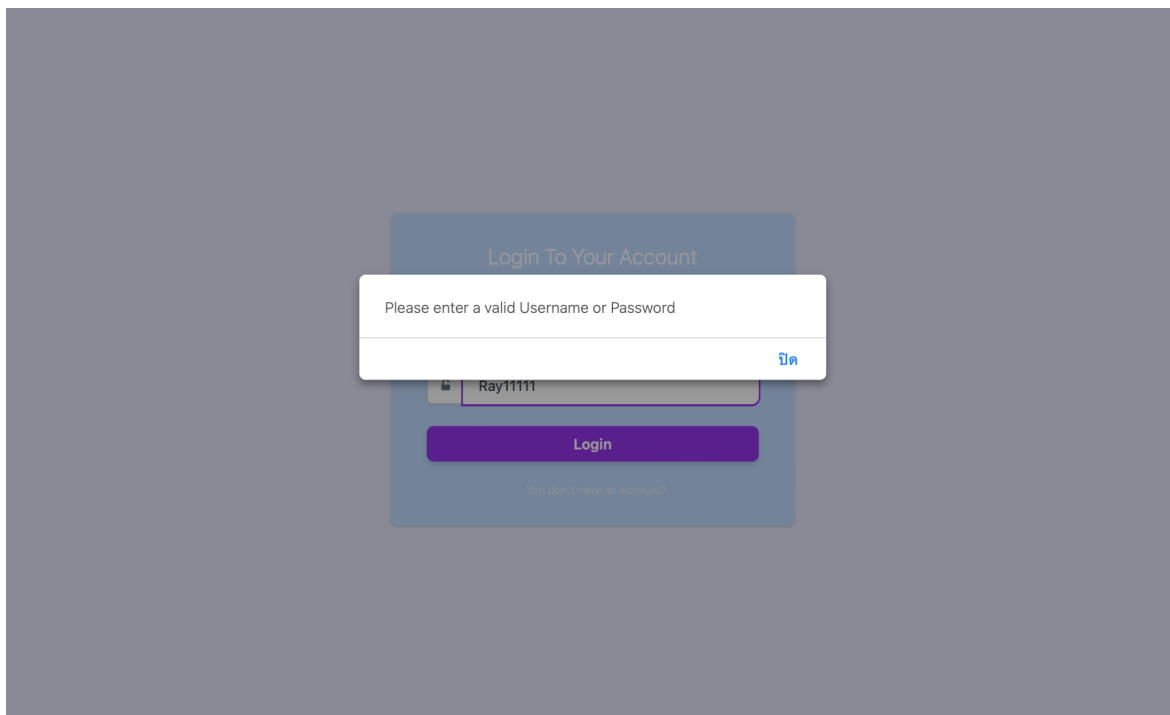
                                </div>
                                </div>
                                <div className="flex w-full">
                                    <button type="submit" className="py-2 px-4
bg-purple-600 hover:bg-purple-700 focus:ring-purple-500
focus:ring-offset-purple-200 text-white w-full transition ease-in
duration-200 text-center text-base font-semibold shadow-md
focus:outline-none focus:ring-2 focus:ring-offset-2 rounded-lg ">
                                        Login
                                    </button>
                                </div>
                                </form>
                                </div>
                                <div className="flex items-center justify-center mt-6">
                                    <a href="/register" className="inline-flex
items-center text-xs font-thin text-center text-gray-100
hover:text-white">
                                        <span className="ml-2">
                                            You don't have an account?
                                        </span>
                                    </a>
                                </div>
                                </div>
                                )
                                }
                                export default Login

```

This is login page after enter <http://localhost:3000/>



In case a user login with invalid username or password, the website will show a notification that requests the user to input valid information.



3.1.3 register folder that contains:

3.1.3.1 register.js

```
import React, { useState } from 'react'
import Axios from "axios";

const Register = () => {
  const [user, setUser] = useState({
    name: "",
    username: "",
    password: "",
    userbiokey: 200,
    Threshold: null,
  })

  const handleChange = (key, e) => {
    var value = e
    if (key !== "userbiokey" && key !== "Threshold") {
      value = e.target.value
    }
    setUser({
      ...user, //spread operator
      [key]: value,
    })
  }

  const onCreateUser = async() => {
    const isDataAvailable = !(user.name && user.username &&
user.password && user.userbiokey && user.Threshold)
    if (isDataAvailable) {
      Axios.post("http://localhost:3001/register", user)
        .then(res => alert("successful create"))
      window.href = "/login"
    }
    else {
      alert("invalid input")
    }
  };
}
```

```

    }

    //register function
    const handleRegister = (e) => {
        e.preventDefault()

        if (user.password.length > 7 && user.password.length < 11){
//8-10
            user.Threshold = 20
        } else if (user.password.length > 10 && user.password.length
< 21){ //11-20
            console.log("here");
            user.Threshold = 30
        } else if (user.password.length > 20 && user.password.length
< 31){ //21-30
            user.Threshold = 40
        }
        console.log(user.Threshold);
        onCreateUser()
    }

    return (
        <div class="flex flex-col max-w-md px-4 py-8 rounded-lg
shadow bg-blue-200 sm:px-6 md:px-8 lg:px-10">
            <div class="self-center mb-2 text-xl font-light
sm:text-2xl text-white">
                Create a new account
            </div>
            <span class="justify-center text-sm text-center
flex-items-center text-gray-400">
                Already have an account ?
                <a href="/" class="text-sm text-blue-500 underline
hover:text-blue-700">
                    Sign in
                </a>
            </span>
        </div>
    )

```

```

<div class="p-6 mt-8">
  <form action="#" onSubmit={handleRegister}>
    <div class="flex flex-col mb-2">
      <div class="relative ">
        <input type="text"
id="create-account-name" class=" rounded-lg border-transparent
flex-1 appearance-none border border-gray-300 w-full py-2 px-4
bg-white text-gray-700 placeholder-gray-400 shadow-sm text-base
focus:outline-none focus:ring-2 focus:ring-purple-600
focus:border-transparent" name="name" value={user.name} onChange={e
=> handleChange("name", e)} placeholder="name" />
      </div>
    </div>
    <div class="flex gap-4 mb-2">
      <div class="relative ">
        <input type="text"
id="create-account-username" class=" rounded-lg border-transparent
flex-1 appearance-none border border-gray-300 w-full py-2 px-4
bg-white text-gray-700 placeholder-gray-400 shadow-sm text-base
focus:outline-none focus:ring-2 focus:ring-purple-600
focus:border-transparent" name="username" value={user.username}
onChange={e => handleChange("username", e)} placeholder="username"
/>
      </div>
    </div>
    <div class="flex flex-col mb-2">
      <div class="relative ">
        <input id="create-password" class="
rounded-lg border-transparent flex-1 appearance-none border
border-gray-300 w-full py-2 px-4 bg-white text-gray-700
placeholder-gray-400 shadow-sm text-base focus:outline-none
focus:ring-2 focus:ring-purple-600 focus:border-transparent"
name="password" value={user.password} onChange={e =>
handleChange("password", e)} placeholder="password" />
      </div>
    </div>
  </form>
</div>

```



```

        </div>
        <br />
        <span class="justify-center text-sm text-center flex-items-center text-gray-400">
            Confirm Password*
        </span>
        <div class="flex flex-col mb-2">
            <div class="relative ">
                <input id="create-password" class="rounded-lg border-transparent flex-1 appearance-none border border-gray-300 w-full py-2 px-4 bg-white text-gray-700 placeholder-gray-400 shadow-sm text-base focus:outline-none focus:ring-2 focus:ring-purple-600 focus:border-transparent" name="password" value={user.password} onChange={e => handleChange("password", e)} placeholder="password" />
            </div>
        </div>
        <div class="flex flex-col mb-2">
            <div class="relative ">
                <input id="create-password" class="rounded-lg border-transparent flex-1 appearance-none border border-gray-300 w-full py-2 px-4 bg-white text-gray-700 placeholder-gray-400 shadow-sm text-base focus:outline-none focus:ring-2 focus:ring-purple-600 focus:border-transparent" name="password" value={user.password} onChange={e => handleChange("password", e)} placeholder="password" />
            </div>
        </div>
        <div class="flex w-full my-4">
            <button type="submit" class="py-2 px-4 bg-purple-600 hover:bg-purple-700 focus:ring-purple-500 focus:ring-offset-purple-200 text-white w-full transition ease-in duration-200 text-center text-base font-semibold shadow-md focus:outline-none focus:ring-2 focus:ring-offset-2 rounded-lg " >
                Register
            </button>

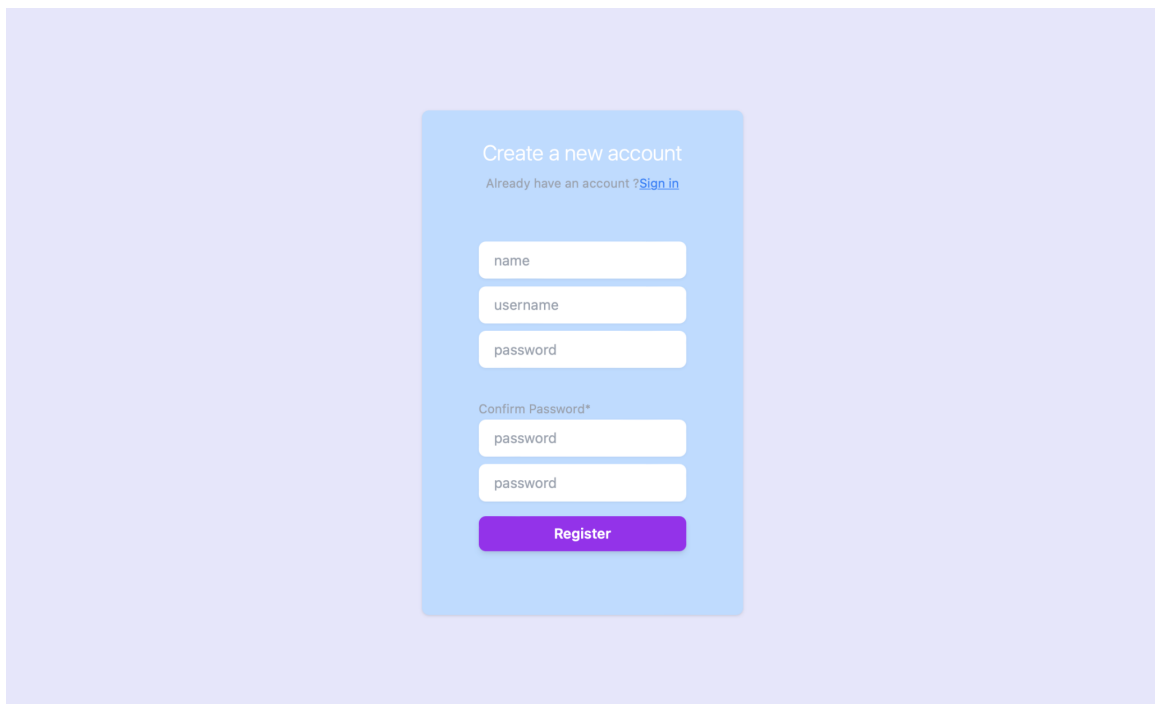
```

```

        </div>
      </form>
    </div>
  </div>
)
}
export default Register

```

After you click 'You don't have an account?', the website will bring users to create a new account page.



Create a new account

Already have an account? [Sign in](#)

name

username

password

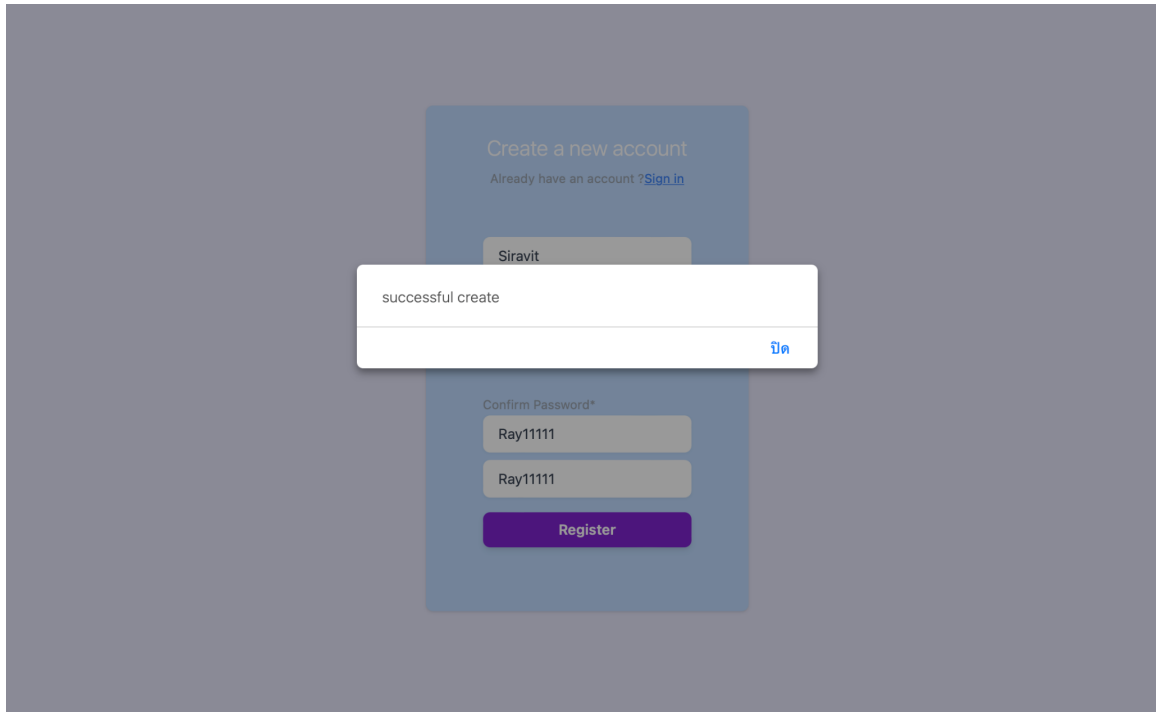
Confirm Password*

password

password

Register

Then you have to fill in all the required information such as name, username, and password.



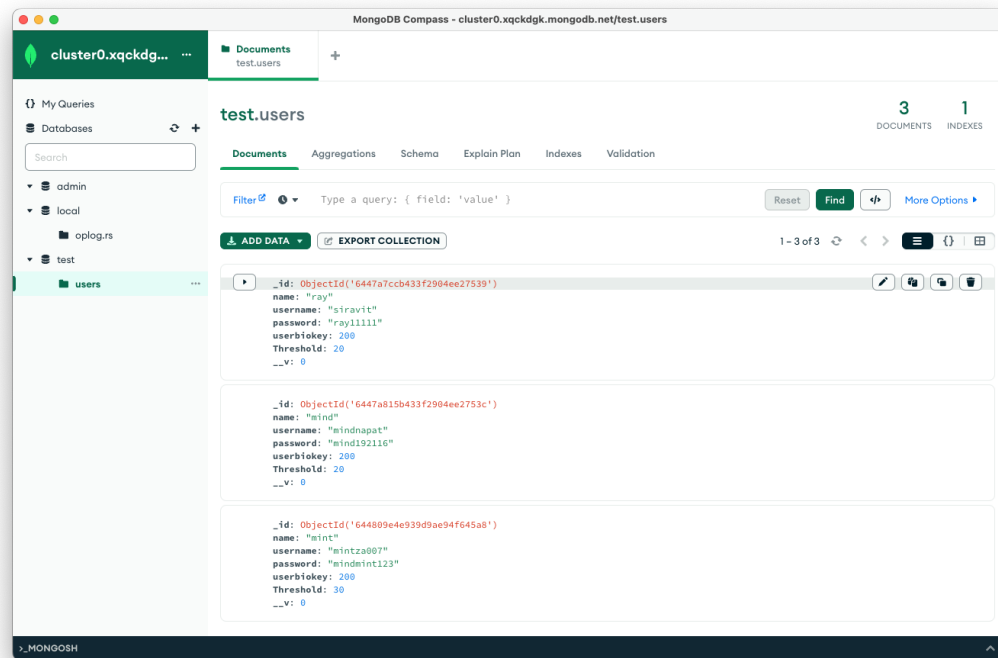
If you create a new account successful, the terminal will show like this

```
Username "mindnapat" is already taken
on creating...
creating...
█
```

If you create a new account unsuccessful, the terminal will show like this

```
Username "mindnapat" is already taken
on creating...
creating...
```

After registering, the MongoDB Compass will update new users in the database.



3.2 App.css

```
client > src > # App.css > ...
1  ∨ .App {
2    |   text-align: center;
3    |   align-content: center;
4    | }
5
6  ∨ @media (prefers-reduced-motion: no-preference) {
7    ∨ .App-logo {
8      |   animation: App-logo-spin infinite 20s linear;
9      | }
10   }
11
12  ∨ .App-header {
13    |   background-color: #394867;
14    |   display: flex;
15    |   flex-direction: column;
16    |   align-items: center;
17    |   justify-content: center;
18    |   font-size: calc(10px + 2vmin);
19    |   color: #F1F6F9;
20    | }
21
22  ∨ .App-link {
23    |   color: #61dafb;
24    | }
25
26  ∨ @keyframes App-logo-spin {
27    ∨ from {
28      |   transform: rotate(0deg);
29      | }
30    ∨ to {
31      |   transform: rotate(360deg);
32      | }
33  }
```

3.3 App.js

```
client > src > JS App.js > [⌕] default
1  // import the Login and Register components from their respective files
2  import Login from "../components/login/login"
3  import Register from "../components/register/register"
4
5  // import the necessary components from the react-router-dom package
6  import {
7    BrowserRouter as Router,
8    Switch,
9    Route
10 } from "react-router-dom";
11
12 // define the App component
13 function App() {
14   return (
15     // wrap everything in a div with a lavender background color
16     <div className="App" style={{backgroundColor: "lavender"}}>
17       <div className="bg-lightslategray h-screen font-sans">
18         <div className="container mx-auto h-full flex justify-center items-center">
19           <Router>
20             <Switch>
21               <Route exact path="/"><Login /></Route>
22               <Route path="/register"><Register /></Route>
23             </Switch>
24           </Router>
25         </div>
26       </div>
27     </div>
28   );
29 }
30
31 // export the App component so it can be used in other files
32 export default App;
```

3.4 index.js

```
client > src > JS index.js
1 // Importing necessary modules
2 import React from "react"; // Module for building user interfaces
3 import ReactDOM from "react-dom"; // Module for rendering React components
4 import App from "./App"; // Importing App component from App.js file
5
6 import reportWebVitals from "./reportWebVitals"; // Module for reporting web vitals to a third-party service
7
8 // Rendering the App component to the DOM
9 ReactDOM.render(
10   <>
11     <App />
12   </>,
13   // Selecting the root DOM element to render the App component into
14   document.getElementById('root')
15 );
16
17 // Calling the function to report web vitals.
18 reportWebVitals();
```

3.5 reportWebVitals.js

```
client > src > JS reportWebVitals.js > ...
1 const reportWebVitals = onPerfEntry => {
2   if (onPerfEntry && onPerfEntry instanceof Function) {
3     import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {
4       getCLS(onPerfEntry);
5       getFID(onPerfEntry);
6       getFCP(onPerfEntry);
7       getLCP(onPerfEntry);
8       getTTFB(onPerfEntry);
9     });
10  }
11 };
12
13 export default reportWebVitals;
14
```

4. Package.json: The package. json file is the heart of any Node project. It records important metadata about a project which is required before publishing to NPM, and also defines functional attributes of a project that npm uses to install dependencies, run scripts, and identify the entry point to our package.

5. Package-lock.json

6. Readme.md

```
# How to run file
1 download this zip file to your machine.
2 open folder "ITCS271_6487095"
3 open new terminal
4 cd /pathfile of client/
5 yarn start
6 open another terminal
7 cd /pathfile of serve/
8 npm start
9 then go to safari and input this URL: http://localhost:3000
```


In server folder contains:

1. Models folder contains:

1.1 users.js

```
server > models > JS Users.js > ...  
1  const mongoose = require("mongoose");  
2  
3  const UserSchema = new mongoose.Schema({  
4    name: String,  
5    username: String,  
6    password: String,  
7    userbiokey: Number,  
8    Threshold: Number,  
9  });  
10  
11 const UserModel = mongoose.model("users", UserSchema);  
12 module.exports = UserModel;
```

This code is for set user schema in Mongoose which requires name as a string, username as a string, password as a string, password as a string, userbiokey as a number, and threshold as a number.

This is example of successful user schema

```
_id: ObjectId('6447a7ccb433f2904ee27539')
name: "ray"
username: "siravit"
password: "ray11111"
userbiokey: 200
Threshold: 20
__v: 0
```

2. node_modules folder : for install modules that use in this project

3. Index.js

```
server > ./index.js > ...
1  const express = require("express");
2  const app = express();
3  const mongoose = require("mongoose");
4  const iService = require("./iservice");
5
6  const cors = require("cors");
7
8  app.use(express.json());
9  app.use(cors());
10
11 // connection to the cluster
12 mongoose.connect(
13   "mongodb+srv://adminray:ray11111@cluster0.xqckdkg.mongodb.net/?retryWrites=true&w=majority"
14 );
15
16 app.listen(3001, () => {
17   console.log("SERVER RUNS PERFECTLY!");
18 });
19
20 app.post("/register", register)
21 app.post("/login", login)
22
23 async function login(req, res) {
24   console.log("on searching for username...");
25   const user = await iService.getByUsername(req.body)
26   if (user == null) return res.status(400).json({message: "Please enter a valid Username or Password"})
27
28   if ((user.userbiokey-user.Threshold < req.body.userbiokey) && (user.userbiokey+user.Threshold > req.body.userbiokey)) {
29     return res.json(user)
30   } else {
31     return res.status(400).json({message: "Error: Please enter Password again"})
32   }
33 }
34
35 function register(req, res, next) {
36   console.log("on creating...");
37   iService.create(req.body)
38     .then(() => res.json({}))
39     .catch(err => next(err));
40 }
41
```

4. iservice .js

```
server > JS iservice.js > ...
1  const UserModel = require("./models/Users");
2
3  async function getByUsername(userParam) {
4      console.log("searching for username...");
5      const response = await UserModel.findOne({ username: userParam.username, password: userParam.password });
6      return response;
7  }
8
9  async function create(userParam) {
10     console.log("creating...");
11     // validate
12     if (await UserModel.findOne({ username: userParam.username })) {
13         throw 'Username "' + userParam.username + '" is already taken';
14     }
15     module.exports = {
16         getByUsername,
17         create,
18     };
19     var module: {
20         exports: typeof module.exports;
21     }
22     module.exports = {
23         getByUsername,
24         create,
25     };
26 }
```

5. Package.json : The package. json file is the heart of any Node project. It records important metadata about a project which is required before publishing to NPM, and also defines functional attributes of a project that npm uses to install dependencies, run scripts, and identify the entry point to our package.
6. Package-lock.json
7. Readme.md

```
# key-stroke-dynamic-login
A Term Project of ITDS271 Computer and Communication Security
# A Group Member
- 6487095 Siravit Trisirinan

The presentation slide is available
[https://www.canva.com/design/DAFhHx1T3RU/FR6aY1FM7Rc1wlsAe7P7Gg/edit?utm_content=DAFhHx1T3RU&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton] (link here).
```

Link for presentation:

https://www.canva.com/design/DAFhHx1T3RU/dlqG6VGTLoxu3C8C9j2Onw/view?utm_content=DAFhHx1T3RU&utm_campaign=designshare&utm_medium=link&utm_source=publishsharelink

Reference:

<https://classic.yarnpkg.com/lang/en/docs/cli/run/>

<https://www.youtube.com/watch?v=bhiEJW5poHU>

<https://www.youtube.com/watch?v=JLwwQMU6Ru0&t=195s>

<https://www.youtube.com/watch?v=YT8OM8lcTTw&t=595s>

<https://www.youtube.com/watch?v=xTrjjLX6p-M>

<https://dev.to/crackingdemon/register-and-login-system-in-mern-stack-1n98>

<https://levelup.gitconnected.com/handling-errors-in-mongoose-express-for-display-in-react-d966287f573b>

<https://www.section.io/engineering-education/nodejs-mongoosejs-mongodb/>