BÁO CÁO DỰ ÁN WEBSITE THƯƠNG MẠI ĐIỆN TỬ

I. TỔNG QUAN

1.1 Mục Tiêu Dự Án

Mục tiêu của dự án là xây dựng một trang web thương mại điện tử với các trang chính: Trang chủ, Trang sản phẩm, Trang chi tiết sản phẩm, Trang đăng nhập, Trang đăng ký và Trang giỏ hàng. Các chức năng bao gồm tìm kiếm sản phẩm và xác thực người dùng.

1.2 Công Cụ và Ngôn Ngữ

- ReactJS với ViteJS.
- Typescript (.tsx).
- Yup cho validate form.
- Lodash cho xử lý array và object.
- Lazyloader để tối ưu hóa tải trang.

II. PHẦN FRONTEND

2.1 Giao Diện

- Sử dụng TailwindCSS cho giao diện.
- 3 trang chính: Trang chủ, Trang sản phẩm, Trang chi tiết sản phẩm.
- Tìm kiếm sản phẩm với kết quả không mất khi reload trang.

2.2 Đăng Nhập và Đăng Ký

- Trang đăng nhập với 2 form: username, password.
- Trang đăng ký với 4 form: username, password, email, upload IMG.
- Sourch Code:

* Backend

```
static signIn = async ({ email, password, refreshToken = null }) => {
    // 1.
    if (!email) throw new BadRequestError('Email hoặc Mật khẩu không được để trống!');
    const dataUser =await User.findOne({ where: { email: email } })
    if (!dataUser.dataValues) throw new BadRequestError('Người dùng này không tồn tại');
    const foundUser = dataUser.dataValues
    const match = await bcrypt.compare(password, foundUser.password);
    if (!match) throw new AuthFailureError('Mật khẩu không chính xác');

    if (foundUser.status === 'inactive') throw new BadRequestError('Tài khoản của bạn đã bị khóa! Vui lòng liên hệ Admin để biết thêm chi tiết!');

    // 3.
    // created privateKey, publicKey
    const privateKey = crypto.randomBytes(64).toString('hex');
    const publicKey = crypto.randomBytes(64).toString('hex');

    // 4. generate tokens
    const { id: userId } = foundUser;
    const tokens = await createTokenPair(
        { userId, email },
        publicKey, privateKey);

    await KeyTokenService.updateKeyToken({
        refreshToken: tokens.refreshToken,
        privateKey,
        publicKey,
        userId
    });
    return {
        user: getInfoData({
            fileds: ['id', 'email', 'avatar', 'role', 'username'],
            object: foundUser
        }),
        tokens
    }
```

* Fontend:

```
mock.onPost('/jwt/login').reply(async (request) => {
  const { email, password } = JSON.parse(request.data)

  let error = {
    email: ['Something went wrong']
  }

  const login = await fetch(`${api}/auth/signin`, {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
      'x-api-key': 'c6b5a5c98928327f8e1708eddbf6ca1610f74a8eda936399b0dbae23b86f0cfe4b8874b08a5a482239422f06bab34c3056d6ed8fd6d4e709aef0a69c55a23e9e'
    },
    body: JSON.stringify({ email: email, password: password })
  })
  if (login.status == 200) {


    const dataLogin = await login.json();
```

2.3 Giỏ Hàng

- Giỏ hàng hiển thị số lượng sản phẩm và tổng giá tiền.
- Sử dụng phương thức lưu trữ tạm thời để duy trì dữ liệu giỏ hàng.
- Sourch code
- Sourch code:

* Backend:

```
const insertUserCart= async ({userId, productId, qty}) => {

    const userCart = await UserCart.create({
        userId : userId,
        productId,
        qty
    })


    return userCart;
}
```

*Fontend

```
mock.onPost('/apps/ecommerce/addProducts').reply(async config => {
  const {slug, brand, hasFreeShipping, price, name, rating, image, filename, detail} = JSON.parse(config.data)

  // // Get event from post data
  const product = JSON.parse(config.data)
  const user = JSON.parse(localStorage.getItem('userData'));
  const accessToken = JSON.parse(localStorage.getItem('accessToken'));

  const file = base64ToFile(image, filename);
  var data2 = new FormData()
  data2.append('file', file)
  data2.append('slug', slug)
  data2.append('brand', brand)
  data2.append('price', price)
  data2.append('name', name)
  data2.append('rating', rating)
  data2.append('hasFreeShipping', hasFreeShipping)
  data2.append('detail', detail)

  const upCartUser = await fetch(`${api}/product/create`, {
    method: 'POST',
    headers: {
      'x-api-key': 'c6b5a5c98928327f8e1708eddbf6ca1610f74a8eda936399b0dbae23b86f0cfe4b8874b08a5a482239422f06bab34c3056d6ed8fd6d4e709aef0a69c55a23e9e',
      'x-client-id': user.id,
      'authorization': accessToken
    },
    body: data2
  })
  const dataUpUserCart = await upCartUser.json()

  data.userCart.push(dataUpUserCart.data)

  return [201, { ...dataUpUserCart.data }]
})
```

2.4 Lazyloader

- Tối ưu hóa tải trang bằng Lazyloader để cải thiện trải nghiệm người dùng.
- Sourch code:

III. PHẦN BACKEND

3.1 NodeJS/ExpressJS

- Sử dụng NodeJS và ExpressJS để xây dựng server và thực hiện các request HTTP.

3.2 JWT

Sử dụng JWT (JSON Web Token) để xác thực người dùng.

3.3 POSTMAN

- Sử dụng Postman để test api.

IV. PHẦN BACKEND

4.1 Ngày bắt đầu

- Ngày 14/11/2023 -15/11/2023: Setup máy, tìm hiểu về mariodb, heidisql,...

4.2 Phác Thảo Ban Đầu

- Hoàn thành: 16/11/2023 – 22/11/2023
- Ghi chú:
  - Ngày 16/11/2023 - 18/11/2023: Giao diện login, resigter
  - Ngày 20/11/2023 - 21/11/2023: Viết api login, api resigter
  - Ngày 22/11/2023 - 24/11/2023:
    - Tạo giao diện Giỏ hàng, Sản phẩm, Chi tiết Sản phẩm,, Chi tiết giỏ hang, Quản lý sản phẩm
    - Kèm theo đó viết api sản phẩm(Thêm, Xóa, Sửa, ...), api Giỏ hàng(Thêm, Xóa, Sửa, ...)

4.3 Test và Kiểm Thử

- Test trình độ: Trong quá trình vừa làm vừa test chức năng

4.4 Bản Báo Cáo PDF

- Hoàn thành: 27/11/2023
- Ghi chú: Nộp File đi kèm

V. NHẬN XÉT VÀ KHÓ KHĂN

3.1 Nhận Xét Tích Cực

- Tìm hiểu về ngôn ngữ mới database mới

3.2 Khó Khăn Gặp Phải

- Trong giai đoạn đầu có tìm hiểu về ngôn ngữ mới, database mới, cách connect mới nên trong giai đoạn đầu còn làm hơi chậm

VI. DỰ KIẾN GIAI ĐOẠN TIẾP THEO

4.1 Công Việc Dự Kiến

- Fix một số lỗi còn trong sourch, trong quá trình làm

4.2 Ngày Dự Kiến Hoàn Thành

- Ngày dự kiến: 28/11/2023

V. PHẦN DATABASE

4.1 MariaDB

- Sử dụng MariaDB làm cơ sở dữ liệu cho dự án.

VI. KẾT QUẢ

5.1 Bản Báo Cáo PDF

- Đã tạo bản báo cáo với định dạng PDF.

5.2 LAN IP Kết Nối Website

- LAN IP của website: [Địa chỉ IP]