

KHOA KỸ THUẬT VÀ CÔNG NGHỆ
BỘ MÔN CÔNG NGHỆ THÔNG TIN



THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH
HỌC KỲ I, NĂM HỌC 2022-2024

**NGHIÊN CỨU NODE.JS VÀ XÂY DỰNG
ỨNG DỤNG QUẢN LÝ ĐỐI TÁC BỘ MÔN
CÔNG NGHỆ THÔNG TIN**

Giáo viên hướng dẫn:
TS. Nguyễn Bảo Ân

Sinh viên thực hiện:
Họ tên:Trần Ngọc Mai
MSSV:110121062
Lớp:DA21TTA

Trà Vinh, tháng 1 năm 2024

KHOA KỸ THUẬT VÀ CÔNG NGHỆ
BỘ MÔN CÔNG NGHỆ THÔNG TIN



THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH
HỌC KỲ I, NĂM HỌC 2022-2024

**NGHIÊN CỨU NODE.JS VÀ XÂY DỰNG
ỨNG DỤNG QUẢN LÝ ĐỐI TÁC BỘ MÔN
CÔNG NGHỆ THÔNG TIN**

Giáo viên hướng dẫn:
TS. Nguyễn Bảo Ân

Sinh viên thực hiện:
Họ tên:Trần Ngọc Mai
MSSV:110121062
Lớp:DA21TTA

Trà Vinh, tháng 1 năm 2024

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

[illegible]

Trà Vinh, ngày tháng năm

Giáo viên hướng dẫn

(Ký tên và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

This image shows a full page of primary-ruled paper. It features approximately 20 horizontal dotted lines spaced evenly down the page, providing a guide for handwriting practice. The paper is otherwise blank, with no margins or additional markings.

Trà Vinh, ngày tháng năm

Giáo viên hướng dẫn
(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Kính gửi: thầy Nguyễn Bảo Ân

Trước hết, em xin gửi lời tri ân và lòng biết ơn sâu sắc nhất tới thầy về sự hỗ trợ và sự chia sẻ kiến thức quý báu trong suốt quá trình thực hiện đồ án cơ sở ngành.

Những giờ giảng của thầy không chỉ là những bài giảng mà còn là những nguồn động viên, là động lực giúp em vượt qua những thách thức khó khăn. Sự tận tâm và sự kiên nhẫn của thầy đã giúp em hiểu sâu và áp dụng kiến thức vào thực tế một cách linh hoạt và sáng tạo.

Em cũng xin bày tỏ lòng biết ơn tới tất cả bạn bè, những người đã chia sẻ kiến thức, kinh nghiệm và hỗ trợ em trong những tình huống khó khăn. Sự đồng lòng và giúp đỡ từ mọi người là nguồn động viên quan trọng, giúp em vượt qua mọi khó khăn.

Em xin hứa sẽ tiếp tục nỗ lực, không ngừng học hỏi và phấn đấu để trở thành một người học viên tích cực và có ảnh hưởng tích cực trong ngành nghề của mình.

Trân trọng,

Trần Ngọc Mai

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN	8
1.1 Mô tả bài toán	8
1.2 Công cụ và cách thức thực hiện	8
1.3 Xác định mô hình dữ liệu	8
1.3.1 Kết quả nghiên cứu	8
1.3.2 Kết quả đạt được	8
CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT	9
2.1 Nghiên cứu RESTful API	9
2.1.1 Các thành phần của RESTful API	9
2.1.2 RESTful hoạt động như thế nào?	10
2.1.3 Các trạng thái trả về	10
2.2 Kiến thức về framework Express.js	11
2.3 Hệ quản trị cơ sở dữ liệu MongoDB	11
CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU	12
3.1 Đặc tả các chức năng hệ thống	12
3.1.1 Chức năng của người dùng	12
3.2 Thiết kế hệ thống	12
3.2.1 Lược đồ mô tả hệ thống	12
3.2.2 Lược đồ cơ sở dữ liệu	13
3.3 Thiết kế mã nguồn	13
3.3.1 Mã nguồn hiển thị trang quản lý đối tác	13
3.3.2 Mã nguồn hiển thị trang thông tin đối tác	14
3.3.3 Mã nguồn hiển thị trang chỉnh sửa đối tác	14
3.3.4 Mã nguồn xử lý chỉnh sửa đối tác	15
3.3.5 Mã nguồn xử lý thêm đối tác mới	16
CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU	17
4.1 Sản phẩm đạt được sau quá trình nghiên cứu	17
4.2 Về hiệu năng	17
4.3 Về trải nghiệm người dùng	17
4.4 Kết quả thử nghiệm	17
4.4.1 Chức năng của người dùng	17
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN	23
5.1 Kết luận	23
5.1.1 Kết quả đạt được	23
5.1.2 Những đóng góp mới	23
5.2 Hướng phát triển	23
DANH MỤC TÀI LIỆU THAM KHẢO	24
PHỤ LỤC	25

DANH MỤC HÌNH ẢNH

Hình 1: RESTful API	9
Hình 2: Mô hình MVC	12
Hình 3: Lướt đồ cơ sở dữ liệu	13
Hình 4: Mã nguồn hiển thị trang quản lý đối tác	13
Hình 5: Mã nguồn hiển thị trang thông tin đối tác	14
Hình 6: Mã nguồn hiển thị trang chỉnh sửa đối tác	14
Hình 7: Mã nguồn xử lý chỉnh sửa đối tác	15
Hình 8: Mã nguồn xử lý thêm đối tác mới	16
Hình 9: Mã nguồn xóa đối tác	16
Hình 10: Chức năng thêm đối tác cá nhân	18
Hình 11: Chức năng thêm đối tác đơn vị	18
Hình 12: Xem đối tác cá nhân	19
Hình 13: Xem đối tác đơn vị	20
Hình 14: Xóa đối tác	20
Hình 15: Xóa đối tác	21
Hình 16: Chức năng chỉnh sửa thông tin loại cá nhân	22
Hình 17: Chức năng chỉnh sửa thông tin loại đơn vị	22

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

Đề tài: Nghiên cứu Node.js và xây dựng ứng dụng quản lý lịch đối tác bộ môn công nghệ thông tin

Mục tiêu: Nghiên cứu về Node.js, một nền tảng phát triển ứng dụng web và di động chủ yếu dựa trên ngôn ngữ lập trình JavaScript. Xây dựng ứng dụng quản lý lịch đối tác bộ môn công nghệ thông tin dựa trên các công cụ Node.js & Express.js và hệ quản trị cơ sở dữ liệu MongoDB.

Hướng tiếp cận: Nghiên cứu lý thuyết: Tiếp cận vào thế giới của Node.js, Express.js, MongoDB, và quản lý lịch đối tác thông qua việc tìm hiểu sâu vào tài liệu chuyên sâu, tham gia cộng đồng nghiên cứu. Nghiên cứu thực nghiệm: Xây dựng ứng dụng quản lý lịch đối của bộ môn công nghệ thông tin.

Cách giải quyết vấn đề: Tận dụng sức mạnh của Node.js, một nền tảng linh hoạt và dễ sử dụng. Sử dụng framework Express xây dựng phần back-end một cách nhanh chóng và dễ dàng mở rộng. Sự linh hoạt và hiệu quả của MongoDB, một hệ quản trị cơ sở dữ liệu NoSQL, sẽ giúp đáp ứng tốt các yêu cầu của dự án.

Kết quả đạt được: Ứng dụng quản lý lịch đối tác được xây dựng thành công, đáp ứng được các yêu cầu đề ra. Ứng dụng có giao diện thân thiện, dễ sử dụng. Ứng dụng có khả năng mở rộng, dễ dàng tích hợp với các hệ thống khác.

Bài học kinh nghiệm: Qua trình thực hiện dự án, tôi đã nhận ra sự quan trọng của việc đọc hiểu kỹ và nghiên cứu trước về đề tài. Cần lập kế hoạch thực hiện chi tiết và khoa học. Cần kiên trì và nỗ lực để hoàn thành đề tài.

MỞ ĐẦU

Lý do chọn đề tài:

Trong thời buổi công nghệ hiện đại việc quản lý đối tác của bộ môn công nghệ thông tin bằng văn bản thực sự tốn thời gian và khó quản lý vì thế nên xây dựng một ứng dụng quản lý đối tác.

Mục đích nghiên cứu:

Với mong muốn giúp đỡ bộ môn công nghệ thông tin giải quyết vấn đề này tôi đã nghiên cứu, tìm hiểu và phát triển ứng dụng “Quản lý đối tác của bộ môn công nghệ thông tin”. Ứng dụng này không chỉ giúp bộ môn công nghệ thông tin tiết kiệm thời gian mà còn dễ dàng quản lý.

Kết quả đạt được:

Xây dựng thành công ứng dụng quản lý đối tác cho bộ môn công nghệ thông tin sử dụng NodeJS, ExpressJS và MongoDB để triển khai ứng dụng. Với giao diện thân thiện dễ dàng thao tác giúp người dùng tiếp cận ứng dụng một cách dễ dàng hơn.

CHƯƠNG 1: TỔNG QUAN

1.1 Mô tả bài toán

Trong thời buổi công nghệ hiện đại việc quản lý đối tác hiện đang là vấn đề phức tạp, có quá nhiều đối tác không thể quản lý và ghi nhớ trên văn bản hết được vì vậy ứng dụng quản lý đối tác là một ứng dụng cần thiết giúp người dùng và doanh nghiệp dễ dàng quản lý lịch gặp đối tác bao gồm cả thời gian và loại đối tác cần gặp. Đây là một hệ thống giúp quản lý, theo dõi và tối ưu hóa lịch trình làm việc với các đối tác.

1.2 Công cụ và cách thức thực hiện

Để giải quyết bài toán trên tôi đã tìm hiểu và nghiên cứu các công cụ có thể giải quyết vấn đề quản lý đối tác này và tôi chọn Nodejs, Express.js và cơ sở dữ liệu MongoDB.

1.3 Xác định mô hình dữ liệu

Mô hình dữ liệu gồm 2 bảng gồm: đối tác cá nhân và đối tác đơn vị.

1.3.1 Kết quả nghiên cứu

Sau quá trình nghiên cứu và thực hiện đồ án xây dựng ứng dụng quản lý đối tác này tôi đã tiếp thu thêm được nhiều kiến thức đồng thời học được cách tương tác với NodeJs, Express.js và cơ sở dữ liệu mongodb, qua đó tích lũy thêm được nhiều kinh nghiệm để phát triển các dự án sau này tốt hơn.

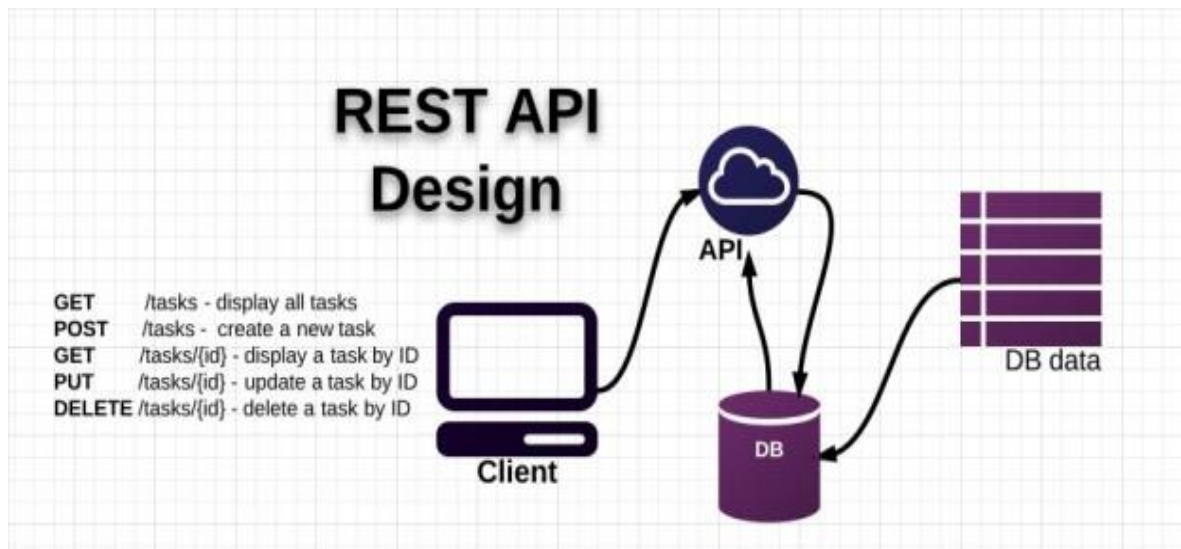
1.3.2 Kết quả đạt được

Sau quá trình nghiên cứu và tìm hiểu thì ứng dụng đã được thành công, với đầy đủ các chức năng cơ bản như thêm, xem, sửa, xóa đối tác, ứng dụng chạy nhanh chóng, giao diện thân thiện với người dùng và dễ sử dụng.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1 Nghiên cứu RESTful API

RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động...), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.



Hình 1: RESTful API

2.1.1 Các thành phần của RESTful API

Tìm hiểu sơ bộ về các thành phần, cấu trúc của RESTful API để nắm chắc định nghĩa trên.

Application Programming Interface được viết tắt là API, tập hợp các quy tắc, cơ chế, dựa vào đó, một ứng dụng hay thành phần tương tác với ứng dụng, thành phần khác. Với kiểu dữ liệu phổ biến như JSON, XML, API có thể trả dữ liệu về ứng dụng bạn cần.

Representational State Transfer được viết tắt thành REST, dạng chuyển đổi cấu trúc dữ liệu, kiểu kiến trúc để viết API. Dùng phương thức HTTP đơn giản để tạo giao tiếp giữa các máy. REST gửi yêu cầu HTTP như GET, POST,... đến một URL để xử lý dữ liệu.

RESTful API, tiêu chuẩn trong thiết kế API ở các ứng dụng web nhằm quản lý resource. RESTful API là kiểu thiết kế được nhiều ứng dụng khác nhau (web, mobile,...) dùng để giao tiếp.

2.1.2 RESTful hoạt động như thế nào?

RESTful API chia một giao dịch thành nhiều module nhỏ để dễ giải quyết phần cơ bản của transaction. Điều này làm tăng sự linh hoạt nhưng có thể gây khó khăn cho developer khi muốn thiết kế REST API từ đầu.

Amazon S3, CDMA, OpenStack Swift là những nơi cung cấp model cho các developer sử dụng.

RESTful API chia một transaction (giao dịch) ra thành nhiều module nhỏ, mỗi module giải quyết một phần cơ bản của transaction. Việc này giúp tăng tính linh hoạt nhưng đôi khi lại tương đối khó khăn cho các developer khi muốn thiết kế REST API từ đầu. Một vài phương thức HTTP mà RESTful API hay sử dụng:

- GET: trả về một tài nguyên
- POST: tạo mới một tài nguyên.
- PUT: cập nhật thông tin cho tài nguyên.
- DELETE: xóa một tài nguyên.

Các hoạt động trên thường gọi là CRUD tương đương với create – tạo, read – đọc, update – sửa, delete – xóa.

2.1.3 Các trạng thái trả về

Khi chúng ta request một API nào đó thường thì sẽ có vài status code để nhận biết sau:

- 200 OK – Trả về thành công cho những phương thức GET, PUT, PATCH hoặc DELETE.
- 201 Created – Trả về khi một Resource vừa được tạo thành công.
- 204 No Content – Trả về khi Resource xóa thành công.
- 304 Not Modified – Client có thể sử dụng dữ liệu cache.
- 400 Bad Request – Request không hợp lệ 401 Unauthorized – Request cần có auth.
- 403 Forbidden – bị từ chối không cho phép.
- 404 Not Found – Không tìm thấy resource từ URI.
- 405 Method Not Allowed – Phương thức không cho phép với user hiện tại.
- 410 Gone – Resource không còn tồn tại, Version cũ đã không còn hỗ trợ.
- 415 Unsupported Media Type – Không hỗ trợ kiểu Resource này.
- 422 Unprocessable Entity – Dữ liệu không được xác thực.

- 429 Too Many Requests – Request bị từ chối do bị giới hạn.

2.2 Kiến thức về framework Express.js

ExpressJS là một framework miễn phí và mã nguồn mở dành cho việc xây dựng ứng dụng web trên nền tảng Node.js. Với ExpressJS, người dùng có thể nhanh chóng thiết kế và phát triển các ứng dụng web và API một cách dễ dàng và tiện lợi. Điều này đặc biệt hữu ích cho các nhà phát triển và lập trình viên đã quen thuộc với JavaScript.

Vì ExpressJS là một framework phát triển ứng dụng web trên Nodejs, các lập trình viên có thể sử dụng mã đã có sẵn để xây dựng các ứng dụng web đơn trang (SPA), đa trang hoặc kết hợp cả hai. Ngoài ra, ExpressJS còn cung cấp một kiến trúc MVC (Model-View-Controller) hữu ích để tổ chức các ứng dụng web phía máy chủ.

2.3 Hệ quản trị cơ sở dữ liệu MongoDB

- MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSql và được hàng triệu người sử dụng, MongoDB là một database hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON. MongoDB được phát triển bởi MongoDB Inc. và được cấp phép theo Giấy phép Công cộng phía Máy chủ (SSPL).
- Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ (như MySQL hay SQL Server...) sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là collection thay vì bảng.
- So với RDBMS thì trong MongoDB collection ứng với table, còn document sẽ ứng với row, MongoDB sẽ dùng các document thay cho row trong RDBMS.
- Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định.

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

3.1 Đặc tả các chức năng hệ thống

Hệ thống có các chức năng cơ bản như nhập thông tin, xem thông tin, thêm, xóa, sửa đối tác. Ngoài ra, người dùng còn có thể chọn loại đối tác cá nhân và đơn vị để có thể dễ dàng quản lý hơn.

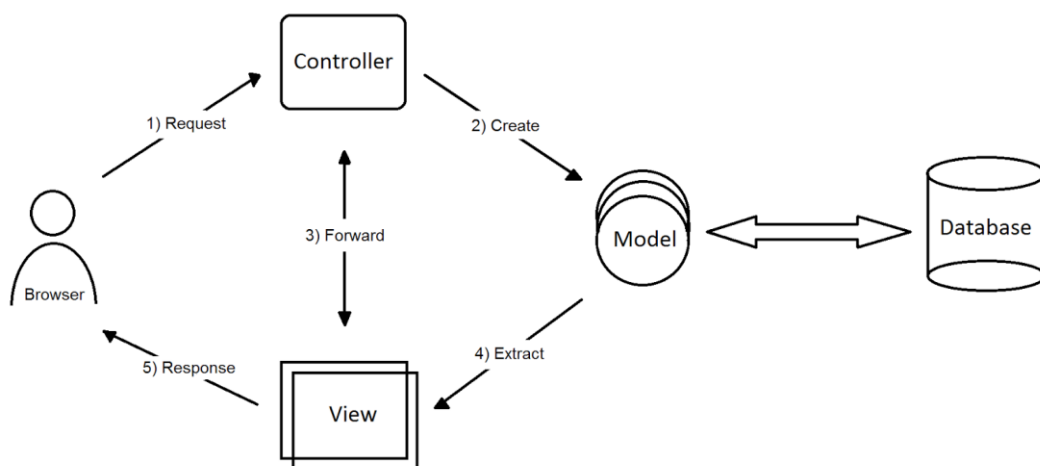
Giao diện người dùng sử dụng công cụ view engine để thiết kế các form nhập thông tin chi tiết cho đối tác. Giao tiếp giữa backend và giao thức HTTPS được đảm bảo thông qua RESTful API. Sử dụng MongoDB để lưu trữ thông tin đối tác, hệ thống này đạt được sự linh hoạt và hiệu quả bằng cách kết hợp NodeJS với cơ sở dữ liệu

3.1.1 Chức năng của người dùng

Người dùng có thể xem danh sách toàn bộ đối tác đã được nhập, bao gồm cả đối tác cá nhân và đơn vị, có thể xem chi tiết đối tác bằng cách chọn một đối tác cụ thể từ danh sách, chi tiết này sẽ hiển thị tất cả các thông tin liên quan đến đối tác đó. Tạo mới thông tin cho đối tác cá nhân hoặc đơn vị. Cung cấp khả năng chỉnh sửa thông tin của đối tác cá nhân hoặc đơn vị. Cho phép người dùng xóa đối tác khỏi hệ thống nếu cần thiết, đồng thời cung cấp xác nhận trước khi thực hiện hành động này.

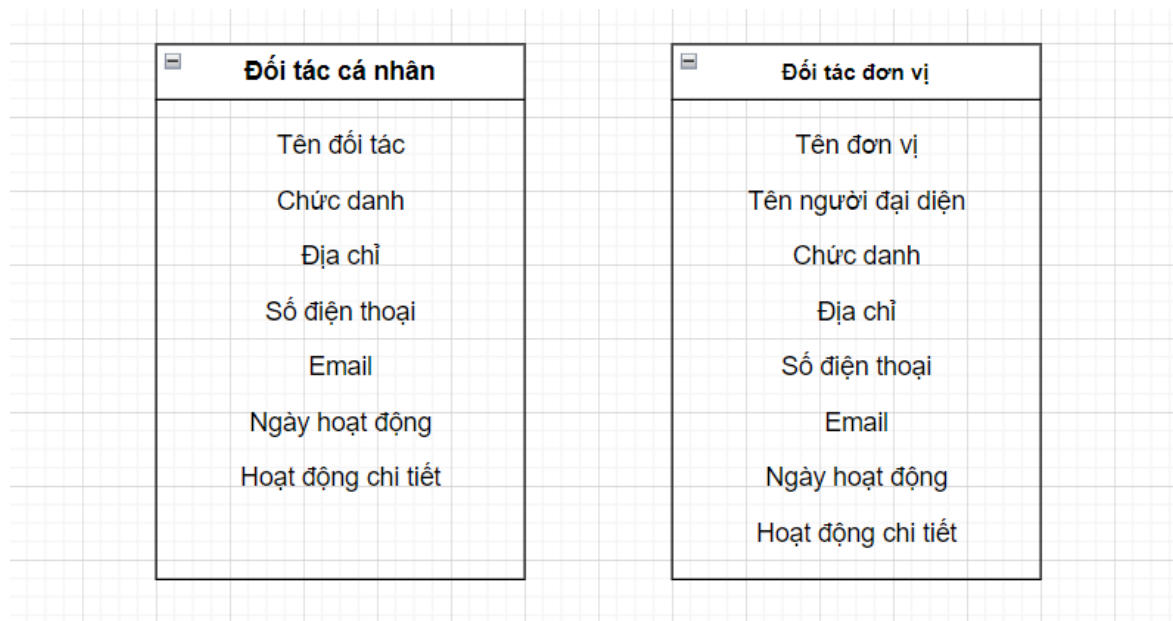
3.2 Thiết kế hệ thống

3.2.1 Lược đồ mô tả hệ thống



Hình 2: Mô hình MVC

3.2.2 Lược đồ cơ sở dữ liệu



Hình 3: Lược đồ cơ sở dữ liệu

Bảng “Đối tác cá nhân” lưu trữ các thông tin đối tác cá nhân như: Tên đối tác, chức danh, địa chỉ, số điện thoại, email, ngày hoạt động, hoạt động chi tiết.

Bảng “Đối tác đơn vị” lưu trữ các thông tin đối tác cá nhân như: Tên đơn vị, tên người đại diện, chức danh, địa chỉ, số điện thoại, email, ngày hoạt động, hoạt động chi tiết.

3.3 Thiết kế mã nguồn

3.3.1 Mã nguồn hiển thị trang quản lý đối tác

```
app.get('/partners/manage', async (req, res) => {
  try {
    const partners = await Partner.find();
    res.render('manage', { partners });
  } catch (error) {
    res.status(500).send('Lỗi Máy Chủ');
  }
});
```

Hình 4: Mã nguồn hiển thị trang quản lý đối tác

3.3.2 Mã nguồn hiển thị trang thông tin đối tác

```
app.get('/partners/view/:id', async (req, res) => {  
  try {  
    const partner = await Partner.findById(req.params.id);  
    res.render('view', { partner, partnerId: req.params.id });  
  } catch (error) {  
    res.status(500).send('Lỗi Máy Chủ');  
  }  
});
```

Hình 5: Mã nguồn hiển thị trang thông tin đối tác

3.3.3 Mã nguồn hiển thị trang chỉnh sửa đối tác

```
app.get('/partners/edit/:id', async (req, res) => {  
  try {  
    const partner = await Partner.findById(req.params.id);  
    res.render('edit', { partner });  
  } catch (error) {  
    res.status(500).send('Lỗi máy chủ');  
  }  
});
```

Hình 6: Mã nguồn hiển thị trang chỉnh sửa đối tác

3.3.4 Mã nguồn xử lý chỉnh sửa đối tác

```
app.post('/api/partners/:id', async (req, res) => {
  try {
    const { partnerType, ...updateData } = req.body;
    let updatedPartner;
    if (partnerType === 'individual') {
      updatedPartner = await Partner.findByIdAndUpdate(
        req.params.id,
        { $set: { partnerType, individualData: { ...updateData } } },
        { new: true }
      );
    } else if (partnerType === 'organization') {
      updatedPartner = await Partner.findByIdAndUpdate(
        req.params.id,
        { $set: { partnerType, organizationData: { ...updateData } } },
        { new: true }
      );
    } else {
      return res.status(400).json({ error: 'Loại đối tác không hợp lệ' });
    }
    console.log('Cập nhật đối tác:', updatedPartner);
    const redirectPath = req.query.redirect || '/partners/manage';
    res.redirect(redirectPath);
  } catch (error) {
    console.error('Lỗi cập nhật đối tác:', error);
    res.status(500).json({ error: 'Lỗi máy chủ' });
  }
});
```

Hình 7: Mã nguồn xử lý chỉnh sửa đối tác

3.3.5 Mã nguồn xử lý thêm đối tác mới

```
app.post('/api/partners', async (req, res) => {
  try {
    const { partnerType } = req.body;

    let savedPartner;

    if (partnerType === 'individual') {
      const newIndividual = new Partner({
        partnerType: 'individual',
        individualData: { ...req.body },
      });

      savedPartner = await newIndividual.save();
    } else if (partnerType === 'organization') {
      const newOrganization = new Partner({
        partnerType: 'organization',
        organizationData: { ...req.body },
      });

      savedPartner = await newOrganization.save();
    } else {
      return res.status(400).json({ error: 'Loại đối tác không hợp lệ' });
    }
    res.status(201).json({ success: true, partner: savedPartner });
  } catch (error) {
    console.error('Lỗi xử lý dữ liệu biểu mẫu:', error);
    res.status(500).json({ error: 'Lỗi máy chủ' });
  }
});
```

Hình 8: Mã nguồn xử lý thêm đối tác mới

3.3.5 Mã nguồn xóa đối tác

```
app.delete('/api/partners/:id', async (req, res) => {
  try {
    await Partner.findByIdAndDelete(req.params.id);
    res.json({ success: true });
  } catch (error) {
    res.status(500).json({ error: 'Lỗi máy chủ' });
  }
});
```

Hình 9: Mã nguồn xóa đối tác

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1 Sản phẩm đạt được sau quá trình nghiên cứu

Ứng dụng quản lý đối tác được xây dựng thành công với đầy đủ các chức năng cơ bản, đáp ứng đầy đủ các nhu cầu được đề ra. Ứng dụng chạy nhanh chóng, giao diện thân thiện dễ sử dụng.

4.2 Về hiệu năng

Node.js cung cấp một nền tảng linh hoạt và hiệu quả để xây dựng ứng dụng quản lý đối tác với khả năng xử lý đồng thời mạnh mẽ và sự linh hoạt trong tương tác với cơ sở dữ liệu và các tác vụ I/O. Tuy nhiên, tôi cần thêm nhiều thời gian để đảm bảo hiệu suất tốt nhất, việc tối ưu hóa mã nguồn, quản lý tài nguyên, và thường xuyên kiểm tra là rất quan trọng.

4.3 Về trải nghiệm người dùng

Giao diện thân thiện dễ dàng thao tác, tối ưu hóa việc sử lý các thao tác một cách nhanh chóng.

Hiện thị đầy đủ các thông tin giúp người dùng có thể thích nghi nhanh được với ứng dụng.

4.4 Kết quả thử nghiệm

4.4.1 Chức năng của người dùng

Chức năng thêm đối tác cá nhân

Trang hiển thị đầu tiên của hệ thống là loại cá nhân bao gồm các thông tin cá nhân của đối tác sau khi nhập toàn bộ thông tin cần thiết vào rồi ấn nút “Thêm đối tác cá nhân”, hệ thống sẽ thông báo “Thêm đối tác thành công” mọi thông tin đã thêm sẽ được ghi gọn bên dưới đối với đối tác cá nhân sẽ hiển thị “Họ tên-Xem”.

The screenshot shows a web interface titled "Quản lý đối tác" (Partner Management). At the top, there is a dropdown menu labeled "Chọn loại đối tác:" (Select partner type) with "Cá nhân" (Individual) selected. Below this is a form with the following fields: "Họ tên:" (Full name), "Chức danh:" (Position), "Địa chỉ:" (Address), "Email:", "Số điện thoại:" (Phone number), "Ngày hoạt động:" (Start date) with a date picker showing "dd/mm/yyyy", and "Hoạt động chi tiết:" (Detailed activity) with a text area. A blue button labeled "Thêm đối tác cá nhân" (Add individual partner) is at the bottom of the form. At the very bottom of the page, there are two links: "Trần Ngọc Mai - Xem" and "Vĩnh Long - Nguyễn Văn B - Xem".

Hình 10: Chức năng thêm đối tác cá nhân

Chức năng thêm đối tác đơn vị

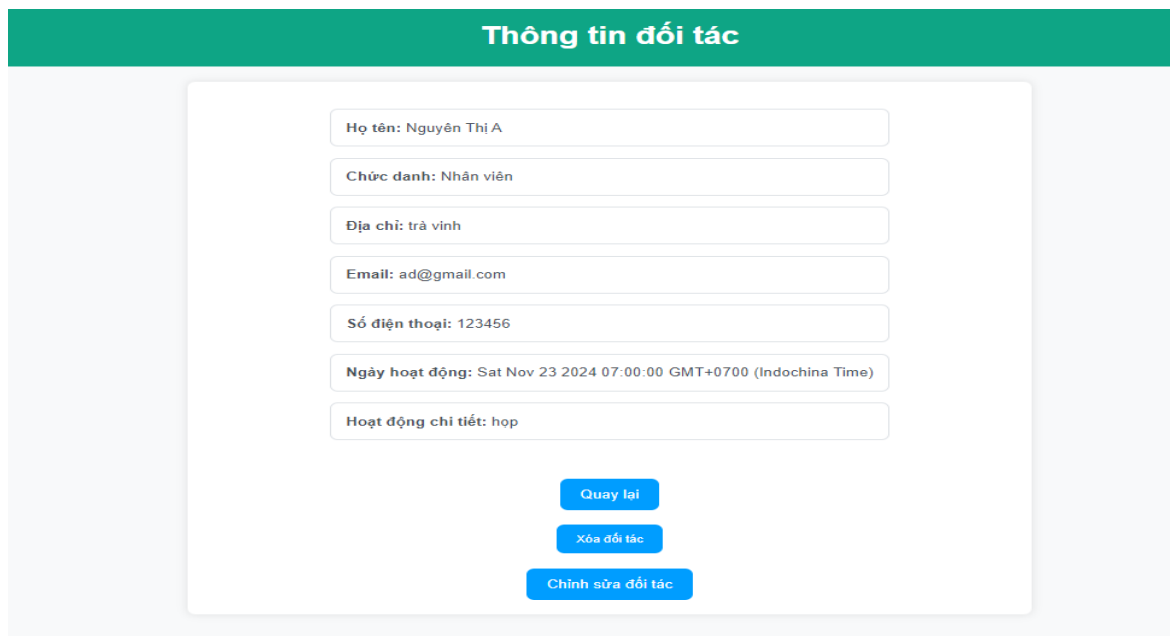
Trang tiếp theo là trang loại đơn vị cũng tương tự như loại cá nhân, ở phần thông tin của đơn vị có sự thay đổi là "Tên đơn vị và Tên người đại diện" còn lại mọi thông tin đều giống với loại cá nhân sau khi thêm hệ thống vẫn sẽ thông báo "Thêm đối tác thành công" và thông tin sẽ được lưu gọn bên dưới đối với đối tác đơn vị sẽ hiển thị "Tên đơn vị-Tên người đại diện-Xem".

The screenshot shows a web interface titled "Quản lý đối tác" (Partner Management). At the top, there is a dropdown menu labeled "Chọn loại đối tác:" (Select partner type) with "Đơn vị" (Unit) selected. Below this is a form with the following fields: "Tên đơn vị:" (Unit name), "Tên Người đại diện:" (Representative name), "Chức danh (Đơn vị):" (Position (Unit)), "Địa chỉ:", "Email (Đơn vị):" (Email (Unit)), "Số điện thoại (Đơn vị):" (Phone number (Unit)), "Ngày hoạt động:" (Start date) with a date picker showing "dd/mm/yyyy", and "Hoạt động chi tiết:" (Detailed activity) with a text area. A blue button labeled "Thêm đối tác đơn vị" (Add unit partner) is at the bottom of the form. At the very bottom of the page, there are two links: "Trần Ngọc Mai - Xem" and "Vĩnh Long - Nguyễn Văn B - Xem".

Hình 11: Chức năng thêm đối tác đơn vị

Xem đối tác cá nhân

Khi người dùng nhấn “Họ tên-xem” đối với đối tác cá nhân thì hệ thống sẽ chuyển đến trang thông tin đối tác hiển thị ra thông tin chi tiết của đối tác cá nhân, bên dưới sẽ có các nút như “Quay lại, xóa đối tác và chỉnh sửa đối tác”.



Thông tin đối tác

Họ tên: Nguyễn Thị A

Chức danh: Nhân viên

Địa chỉ: trà vinh

Email: ad@gmail.com

Số điện thoại: 123456

Ngày hoạt động: Sat Nov 23 2024 07:00:00 GMT+0700 (Indochina Time)

Hoạt động chi tiết: hợp

Quay lại

Xóa đối tác

Chỉnh sửa đối tác

Hình 12: Xem đối tác cá nhân

Xem đối tác đơn vị

Khi người dùng nhấn “Tên đơn vị-Tên người đại diện-xem” đối với đối tác đơn vị thì hệ thống sẽ chuyển đến trang thông tin đối tác hiển thị ra thông tin chi tiết của đối tác đơn vị, bên dưới cũng sẽ có các nút như “Quay lại, xóa đối tác và chỉnh sửa đối tác”.

Thông tin đối tác

Tên đơn vị: vĩnh long

Người đại diện: Nguyễn Văn B

Chức danh (Đơn vị): nhân viên

Địa chỉ(đơn vị): vĩnh long

Email(đơn vị): ad@gmail.com

Số điện thoại(đơn vị): 09009999292

Ngày hoạt động: Fri Jan 12 2024 07:00:00 GMT+0700 (Indochina Time)

Hoạt động chi tiết: ĐI KHẢO SÁT

Quay lại

Xóa đối tác

Chỉnh sửa đối tác

Hình 13: Xem đối tác đơn vị

Xóa đối tác

Chức năng xóa đối tác của loại cá nhân và đơn vị như nhau, khi người dùng nhấn vào nút xóa thì hệ thống sẽ hỏi “ Bạn có chắc chắn muốn xóa đối tác này không?” người dùng bấm “OK” thì hệ thống sẽ thông báo “Đối tác đã được xóa” và quay lại trang “Thêm đối tác”.

localhost:9999 cho biết

Bạn có chắc chắn muốn xóa đối tác này không?

OK

Hủy

Người đại diện: Nguyễn Văn B

Chức danh (Đơn vị): nhân viên

Địa chỉ(đơn vị): vĩnh long

Email(đơn vị): ad@gmail.com

Số điện thoại(đơn vị): 09009999292

Ngày hoạt động: Fri Jan 12 2024 07:00:00 GMT+0700 (Indochina Time)

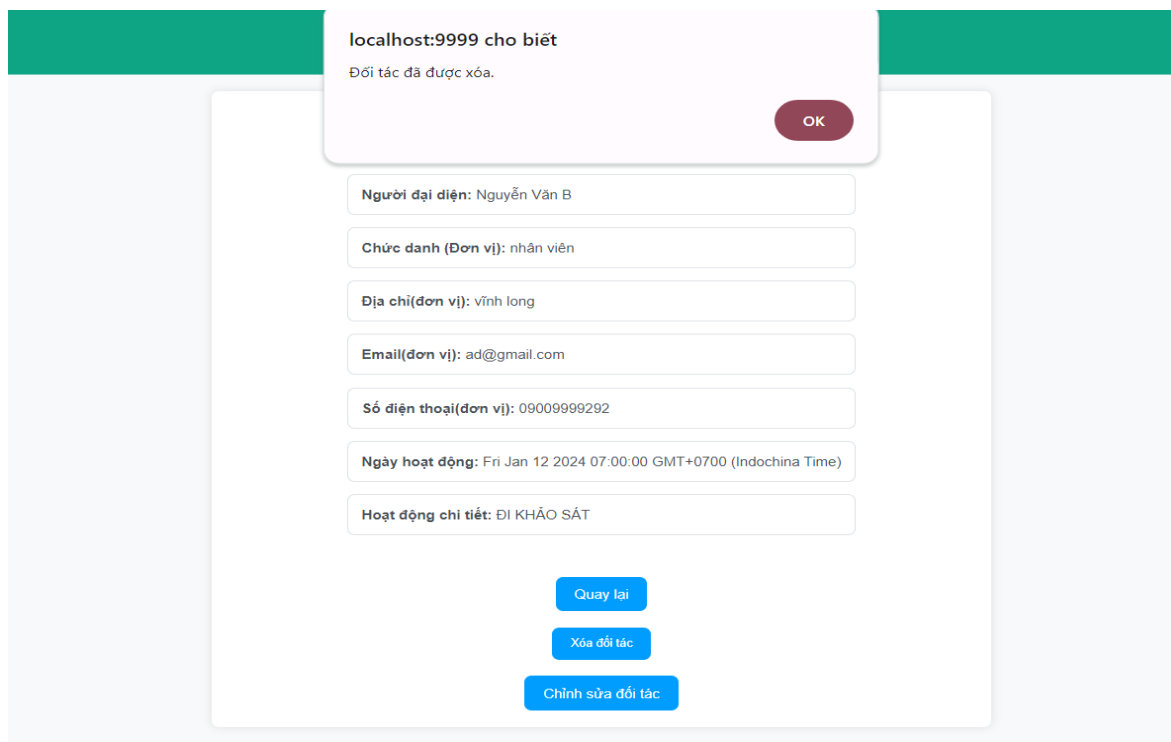
Hoạt động chi tiết: ĐI KHẢO SÁT

Quay lại

Xóa đối tác

Chỉnh sửa đối tác

Hình 14: Xóa đối tác



Hình 15: Xóa đối tác

Chức năng chỉnh sửa thông tin loại cá nhân

Khi người dùng đang xem thông tin đối tác cá nhân nếu muốn chỉnh sửa thông tin người dùng nhấn vào nút “Chỉnh sửa đối tác” hệ thống sẽ chuyển trang đến trang chỉnh sửa thông tin đối tác cá nhân, sau khi chỉnh sửa theo nhu cầu và nhấn nút “Cập nhật” hệ thống sẽ cập nhật thay đổi và chuyển đến trang “Thêm đối tác”.

Chỉnh sửa thông tin đối tác

Loại đối tác:

Họ tên:

Nguyễn Thị A

Chức danh:

Nhân viên

Địa chỉ:

trà vinh

Email:

ad@gmail.com

Số điện thoại:

123456

Ngày hoạt động:

dd/mm/yyyy

Hoạt động chi tiết:

hợp

Quay lại

Cập nhật

Hình 16: Chức năng chỉnh sửa thông tin loại cá nhân

Chức năng chỉnh sửa thông tin loại đơn vị

Khi người dùng đang xem thông tin đối tác đơn vị nếu muốn chỉnh sửa thông tin người dùng nhấn vào nút “Chỉnh sửa đối tác” hệ thống sẽ chuyển trang đến trang chỉnh sửa thông tin đối tác đơn vị, sau khi chỉnh sửa theo nhu cầu và nhấn nút “Cập nhật” hệ thống sẽ cập nhật thay đổi và chuyển đến trang “Thêm đối tác”.

Thông tin đối tác

Tên đơn vị: Vĩnh Long

Người đại diện: Nguyễn Văn A

Chức danh (Đơn vị): Nhân viên

Địa chỉ(đơn vị): Vĩnh Long

Email(đơn vị): tv23@gmail.com

Số điện thoại(đơn vị): 058 4902190

Ngày hoạt động: Wed Jan 03 2024 07:00:00 GMT+0700 (Indochina Time)

Hoạt động chi tiết: đi họp

Quay lại

Xóa đối tác

Chỉnh sửa đối tác

Hình 17: Chức năng chỉnh sửa thông tin loại đơn vị

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết luận

5.1.1 Kết quả đạt được

Ứng dụng quản đối tác của bộ môn công nghệ thông tin được xây dựng thành công và đáp ứng được các yêu cầu đề ra. Ứng dụng có khả năng xử lý nhanh chóng, giao diện thân thiện và dễ sử dụng.

5.1.2 Những đóng góp mới

Ứng dụng quản lý đối tác của bộ môn công nghệ thông tin là một ứng dụng hữu ích, góp phần nâng cao hiệu quả công tác quản lý đối tác của bộ môn. Ứng dụng có những đóng góp mới sau:

Sử dụng các công nghệ hiện đại như Node.js, Express.js và MongoDB, giúp ứng dụng có khả năng xử lý nhanh chóng và có khả năng mở rộng cao.

Thiết kế giao diện đơn giản, thân thiện và dễ sử dụng, giúp người dùng dễ dàng tìm thấy các chức năng cần thiết.

5.2 Hướng phát triển

Ứng dụng quản lý đối tác của sinh viên bộ môn công nghệ thông tin có thể được phát triển thêm theo các hướng sau:

Thêm chức năng thông báo để thông báo cho người dùng về các sự kiện quan trọng như hạn chót thanh toán, cuộc họp, hoặc các thay đổi quan trọng khác.

Thêm chức năng tìm kiếm và lọc để người dùng có thể dễ dàng tìm kiếm đối tác theo các tiêu chí như tên, địa chỉ hoặc thông tin liên hệ.

Tối ưu hóa hiệu năng của ứng dụng để có thể xử lý được nhiều yêu cầu hơn.

Việc phát triển ứng dụng theo các hướng trên sẽ giúp ứng dụng đáp ứng được nhu cầu sử dụng ngày càng cao của người dùng.

DANH MỤC TÀI LIỆU THAM KHẢO

- [1] [express - npm \(npmjs.com\)](https://npmjs.com)
- [2] [Lập trình web với Nodejs, Express, MongoDB \(gitiho.com\)](https://github.com)
- [3] [How to install Node.js | Node.js \(nodejs.org\)](https://nodejs.org)
- [4] [Express web framework \(Node.js/JavaScript\) - Learn web development | MDN \(mozilla.org\)](https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First/Express)

PHỤ LỤC