

## LỜI CẢM ƠN

Sau hơn hai tháng thực hiện thì bài niên luận cơ sở của chúng em đã được hoàn thành, thể hiện những kiến thức mà chúng em đã học được cùng với những gì đã tìm hiểu. Ngoài sự cố gắng hết mình của bản thân, chúng em đã được sự khích lệ rất nhiều từ phía nhà trường, thầy cô, gia đình và bạn bè.

Chính vì thế mà ngày hôm nay, khi hoàn tất xong niên luận này, chúng em muốn gửi lời biết ơn chân thành đến ba mẹ đã luôn động viên và tạo điều kiện cho con học tập. Chúng em xin gửi lời cảm ơn đến Quý Thầy Cô thuộc khoa Công Nghệ Thông Tin và Truyền Thông đã tận tình truyền đạt những kiến thức quý báu cho chúng em trong quá trình học tập.

Đặc biệt chúng em xin cảm ơn thầy Phạm Thế Phi đã tận tình hướng dẫn chúng em trong suốt quá trình thực hiện đề tài niên luận.

Để đáp lại tấm lòng chân tình đó, chúng em sẽ cố gắng vận dụng các kiến thức mà mình đã được trang bị vào thực tiễn cuộc sống một cách hiệu quả nhất nhằm đem lại lợi ích cho mình và cho người khác.

*Xin chân thành cảm ơn!*

Sinh viên thực hiện

Trần Văn Châu

Trà Minh Trí

## MỤC LỤC

CHƯƠNG 1. TỔNG QUAN .....	1
1.1 Đặt vấn đề .....	1
1.2 Mục tiêu cần đạt được .....	1
1.3 Hướng giải quyết .....	2
CHƯƠNG 2. NỘI DUNG NGHIÊN CỨU.....	3
2.1 Giới thiệu GOOGLE MAPS APIs .....	3
2.1.1 Google Maps API là gì? .....	3
2.1.2 Một số ứng dụng của Google Map API.....	3
2.2 Cách sử dụng và phát triển GOOGLE MAPS APIs .....	4
2.2.1 Tạo một Google Maps đơn giản.....	4
2.2.2 Google Maps Overlays .....	7
2.2.3 Google Maps Event.....	9
2.2.4 Google Maps Directions .....	11
2.2.5 Google Maps Places Library .....	13
2.2.6 Google Maps Autocomplete .....	14
CHƯƠNG 3. XÂY DỰNG ỨNG DỤNG DEMO .....	15
3.1 Load bản đồ .....	15
3.2 Thêm và xóa Marker .....	15
3.3 Xác định vị trí hiện tại của người dùng (Geolocation) .....	16
3.4 Tạo hộp tìm kiếm (SearchBox) và chỉ đường.....	17
3.5 Tìm kiếm theo khu vực .....	19
CHƯƠNG 4. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	22
4.1 Kết luận.....	22
4.2 Hướng phát triển .....	22
TÀI LIỆU THAM KHẢO .....	23

# CHƯƠNG 1. TỔNG QUAN

## 1.1 Đặt vấn đề

Như chúng ta đã biết bản đồ từ thời xa xưa giúp con người có thể xác định được phương hướng, vị trí chính xác nơi mình muốn đến, cần đến, giúp cho họ có thể hiểu biết đầy đủ về vùng địa lý mà họ tìm hiểu. Ngày nay với sự giúp đỡ của công nghệ thông tin người ta có thể ngồi tại một chỗ và tìm kiếm chính xác đến một vùng nào đó trên thế giới bằng bản đồ trực tuyến. Hiện nay, bản đồ trực tuyến là một trong những lĩnh vực phát triển mạnh mẽ được hầu hết các nơi trên thế giới xây dựng và Việt Nam không nằm ngoài số đó.

Ngày nay với sự bùng nổ của Internet, có rất nhiều bản đồ số được xây dựng. Nó rất quan trọng cho con người và công việc. Nhu cầu con người ngày càng lớn, ai cũng cần đến bản đồ. Vì vậy ai, đơn vị, tổ chức nào sẽ cung cấp cho chúng ta?

Trên thế giới Google đã xây dựng được một bản đồ trực tuyến của toàn thế giới và cung cấp các API cho người lập trình để có thể tự xây dựng bản đồ trực tuyến ở đất nước mình. Với sự phát triển mạnh mẽ trong lĩnh vực bản đồ các công ty ở Việt Nam như bamboo, địa danh đã xây dựng cơ sở dữ liệu của mình, hiển thị thông tin dựa trên bản đồ của Google. Chúng ứng dụng bởi rất nhiều công nghệ khác nhau như .NET, PHP, JSP...

## 1.2 Mục tiêu cần đạt được

Tìm hiểu các vấn đề xung quanh Google Maps APIs và các dịch vụ mà Google Map hỗ trợ. Áp dụng kiến thức tìm hiểu được xây dựng hệ thống chạy trên Web để giải quyết các vấn đề sau:

- Tìm kiếm: ATM, nhà hàng, khách sạn..., tìm kiếm theo địa chỉ được cho sẵn.
- Hướng dẫn đường đi.
- Tính toán lộ trình và thời gian đường đi.
- Định vị người dùng.

### 1.3 Hướng giải quyết

Áp dụng nội dung tìm hiểu để xây dựng các chức năng:

- Đánh dấu các địa điểm trên bản đồ cùng các thông tin cho địa điểm: các khu vui chơi giải trí, nhà hàng khách sạn, ATM, ngân hàng...
- Chỉ dẫn đường đến các địa điểm cần tìm, chỉ dẫn đường giao thông công cộng, có thể là các địa điểm cung cấp như trên. Ở đây sử dụng các Service Google cung cấp.
- Khoanh vùng khu vực: các trung tâm kinh tế, khu đô thị...
- Tình trạng giao thông các khu vực. Đưa ra các giải pháp có thể... Còn rất nhiều ứng dụng cho phép bạn xây dựng. Quan trọng là đều mang lại lợi ích cho người cung cấp dịch vụ và người sử dụng dịch vụ. Có thể đem lại lợi ích kinh tế nếu như ứng dụng áp dụng tốt trong thực tế!

## **CHƯƠNG 2. NỘI DUNG NGHIÊN CỨU**

### **2.1 Giới thiệu GOOGLE MAPS APIs**

#### **2.1.1 Google Maps API là gì?**

Google Map là một dịch vụ ứng dụng vào công nghệ bản đồ trực tuyến trên web miễn phí được cung cấp bởi Google, hỗ trợ nhiều dịch vụ khác của Google đặc biệt là dò đường và chỉ đường; hiển thị bản đồ đường sá, các tuyến đường tối ưu cho từng loại phương tiện, cách bắt xe và chuyển tuyến cho các loại phương tiện công cộng (xe bus, xe khách ...), và những địa điểm (kinh doanh, trường học, bệnh viện, cây ATM...) trong khu vực cũng như khắp nơi trên thế giới. Vậy Map API là gì?

Map API là một phương thức cho phép 1 website B sử dụng dịch vụ bản đồ của website A (gọi là Map API) và nhúng vào website của mình (site B). Site A ở đây là google map, site B là các website cá nhân hoặc tổ chức muốn sử dụng dịch vụ của google (di chuột, room, đánh dấu trên bản đồ...)

Các ứng dụng xây dựng trên maps được nhúng vào trang web cá nhân thông qua các thẻ Javascripts do vậy việc sử dụng API Google rất dễ dàng.

Google Maps APIs đã được nâng cấp lên phiên bản v3 không chỉ hỗ trợ cho các máy để bàn truyền thống mà cho cả các thiết bị di động; các ứng dụng nhanh hơn và nhiều hơn .

Các dịch vụ hoàn toàn miễn phí với việc xây dựng một ứng dụng nhỏ. Trả phí nếu đó là việc sử dụng cho mục đích kinh doanh, doanh nghiệp.

Google phát triển Google Maps APIs dành cho 4 mảng chính đó là:

- Các ứng dụng trên Android.
- Các ứng dụng trên IOS.
- Các ứng dụng Web.
- Các ứng dụng Webservice.

#### **2.1.2 Một số ứng dụng của Google Map API**

- Đánh dấu các địa điểm trên bản đồ kèm theo thông tin cho địa điểm đó : khu vui chơi giải trí, nhà hàng khách sạn, cây ATM, bệnh viện, trường học,... bất cứ địa điểm nào bạn muốn.
- Chỉ dẫn đường đến các địa điểm cần tìm(đường tối ưu và nhiều option khác), chỉ dẫn đường giao thông công cộng, có thể là các địa điểm cung cấp như trên. Ở đây sử dụng các service google cung cấp.
- Khoanh vùng khu vực: các trung tâm kinh tế, khu đô thị, khu ô nhiễm...

## 2.2 Cách sử dụng và phát triển GOOGLE MAPS APIs


### 2.2.1 Tạo một Google Maps đơn giản

Để sử dụng API của Google Maps, chúng ta phải đăng ký dự án ứng dụng trên Bảng điều khiển API của Google và nhận một khóa Google API để thêm vào ứng dụng của mình. Khóa API này cho phép chúng ta theo dõi và kiểm soát việc sử dụng các dịch vụ API trong các ứng dụng.

Có nhiều gói giới hạn sử dụng cho các dịch vụ của Google Maps API, ở đây chúng ta sử dụng gói giới hạn tiêu chuẩn. Gói này cho phép tải miễn phí 25,000 bản đồ mỗi ngày với API JavaScript của Google Maps mới, API của Maps tĩnh và các hiện thực của API Street View.

Các bước đăng kí khóa API:

- Truy cập vào <https://code.google.com/apis/console> và đăng nhập bằng tài khoản gmail của mình.
- Chọn hoặc tạo Project mới để sử dụng dịch vụ API.
- Vào mục APIs & services chọn Library.
- Chọn mục Credentials để tạo khóa API mới.

<input type="checkbox"/> Name	Creation date ▾	Restrictions	Key
<input type="checkbox"/>  API key 1	1 Oct 2017	None	AIzaSyBC6o_LJLNuTqIbUFTg3cqKeE_hYaAW-U

Trong nội dung đề tài này chủ yếu tìm hiểu về Map JavaScript API, do đó cần kích hoạt dịch vụ Google Maps JavaScript API trong bản điều khiển của Google. Các bước thực hiện:

- Tại mục Library chọn Google Maps JavaScript API.
- Bật Enable dịch vụ.

Lời khuyên từ nhà cung cấp dịch vụ:

- Khai báo ứng dụng dưới dạng HTML5, sử dụng `<!DOCTYPE html>`.
- Tạo một thẻ `<div>` để chứa bản đồ.
- Định nghĩa một hàm JavaScript tạo ra một bản đồ trong `<div>`.
- Tải API JavaScript Maps bằng một thẻ `<script>`.

#### a. Khai báo ứng dụng dưới dạng HTML5

Với khai báo `<!DOCTYPE html>`, hầu hết các trình duyệt sẽ hiển thị nội dung trong chế độ tiêu chuẩn, nếu trình duyệt không hiểu, nó sẽ bỏ qua và sử dụng "quirks mode" để hiển thị nội dung của chúng. Do có sự xung đột về CSS giữa chế độ "quirks mode" và chế độ tiêu chuẩn nên nhà phát triển đã đề nghị sử dụng các thuộc tính style như sau:

```
1. <style>
2.   #map {
3.     height: 100%;
4.   }
5.   html, body {
6.     height: 100%;
7.     margin: 0;
8.     padding: 0;
9.   }
10. </style>
```

Khai báo này chỉ ra rằng vùng chứa bản đồ `<div>` sẽ chiếm 100% chiều cao của thẻ HTML.

#### b. Tải API JavaScript của Google Maps

Để tải API JavaScript của Google Maps, sử dụng thẻ `<script>` như sau:

```
1. <script async defer src=https://maps.googleapis.com/maps/api/js?key=Y
   OUR_API_KEY &callback=initMap">
2. </script>
```

URL chứa trong thẻ <script> là vị trí của tệp JavaScript, sẽ tải tất cả các kí hiệu và định nghĩa bạn cần cho việc sử dụng các dịch vụ API JavaScript của Google Maps. Thẻ <script> này là bắt buộc.

Thuộc tính “async” cho phép trình duyệt hiển thị phần còn lại của trang web trong khi tải API JavaScript của Maps. Khi API đã sẵn sàng, nó sẽ gọi hàm được chỉ định bằng cách sử dụng tham số “callback”.

Tham số “key” chứa khóa API của ứng dụng đã đăng kí.

Việc bảo mật trên web là rất quan trọng, do đó nhà phát triển đã thực hiện tất cả các API JavaScript Maps bằng HTTPS. Sử dụng HTTPS làm cho trang web an toàn hơn.

### c. Tùy chọn bản đồ

```
1. map = new google.maps.Map(mapDiv, {
2.   center: new google.maps.LatLng(21.033333 ,105.849998),
3.   zoom: 8,
4.   mapTypeId: 'ROADMAP'
5. });
```

Có 2 tùy chọn bắt buộc cho mỗi bản đồ: center và zoom.

- Center: là thuộc tính xác định vị trí trung tâm của bản đồ tương ứng với kinh độ và vĩ độ đã nhập.
- Zoom: thuộc tính xác định cấp độ phóng to của bản đồ. “zoom: 0” sẽ hiển thị bản đồ trái đất ở dạng thu nhỏ hoàn toàn. Cấp độ của thuộc tính zoom càng cao thì độ phóng to bản đồ càng lớn. Mức độ thu phóng được xác định dưới dạng số nguyên.
- mapTypeId: là thuộc tính xác định kiểu bản đồ hiển thị. Có các loại bản đồ sau được hỗ trợ:
  - ROADMAP: chế độ đường mặc định.
  - SATELLITE: hiển thị ảnh vệ tinh của Google Earth.
  - HYBRID: hiển thị hỗn hợp chế độ đường và vệ tinh.



- TERRAIN: hiển thị bản đồ vật lý dựa trên thông tin địa hình.

#### d. Đối tượng bản đồ

```
1. map = new google.maps.Map(document.getElementById("map"), {...});
```

Lớp JavaScript đại diện cho một bản đồ là lớp Map. Đối tượng của lớp này xác định một bản đồ duy nhất trên một trang. Khi tạo bản đồ mới, chúng ta có thể chỉ định thẻ <div> với id của bản đồ để xác định vùng chứa của bản đồ. Chúng ta cũng có thể tạo nhiều bản đồ cùng một lúc với tên các id khác nhau, khi sử dụng bản đồ nào thì chỉ cần gọi tên id tương ứng của nó.

#### e. Thư viện

Khi tải API JavaScript Maps qua URL, chúng ta có thể tải các thư viện bổ sung bằng cách sử dụng tham số “libraries”. Thư viện là các mô đun của mã cung cấp chức năng bổ sung cho API JavaScript của bản đồ chính nhưng không được tải trừ khi bạn yêu cầu chúng cụ thể.

Các thư viện có sẵn:

- Drawing: cung cấp một giao diện đồ họa cho người sử dụng để vẽ đa giác, hình chữ nhật, đa giác, vòng tròn và điểm đánh dấu trên bản đồ.
- Geometry: bao gồm các chức năng tiện ích để tính các giá trị hình học vô hạn (như khoảng cách và diện tích) trên bề mặt trái đất.
- Places: cho phép ứng dụng của bạn tìm kiếm các địa điểm như cơ sở, vị trí địa lý, hoặc điểm quan tâm nổi bật, trong một khu vực xác định.
- Visualization: cung cấp bản mô phỏng nhiệt cho dữ liệu.

### 2.2.2 Google Maps Overlays

Overlays là các đối tượng trên bản đồ được ràng buộc với tọa độ kinh độ và vĩ độ.

Google Maps có một số kiểu Overlays:

- Marker: đánh dấu một vị trí trên bản đồ.

- Polyline: là một đường đi được vẽ qua một loạt các toạ độ đã cho. Nó hỗ trợ một loạt các thuộc tính: path - các điểm sẽ đi qua, strokeColor - định màu cho đường đi, strokeOpacity - xác định độ sáng rõ của đường đi, strokeWeight - quy định độ to nhỏ của đường đi.
- Polygon: chuỗi các đường thẳng trên bản đồ và các khối.
- Circle: hỗ trợ các thuộc tính: center, radius, strokeColor, strokeOpacity, strokeWeight, ... Cho phép khoanh tròn vị trí center đã khai báo.
- Info Windows: Cho phép hiển thị thông tin phía trên vị trí đã đánh dấu.

#### a. Marker

Marker được thiết kế để đánh dấu một điểm theo toạ độ xác định trên bản đồ.

```
1. var marker = new google.maps.Marker({
2.     position: myLatLng,
3.     map: map,
4.     title: 'Hello World!'
5. });
```

Để hiển thị Marker trên bản đồ, chúng ta cần quan tâm 3 tham số chính như sau:

- Position (bắt buộc): chỉ định vị trí ban đầu của điểm đánh dấu (LatLng).
- Map: xác định bản đồ mà marker sẽ hiển thị. Nếu không chỉ định bản đồ, marker được tạo ra sẽ không hiển thị trên bản đồ, thay vào đó có thể thêm marker bằng cách gọi phương thức setMap().
- Title: thêm chú thích cho marker.

Để loại bỏ một điểm đánh dấu khỏi bản đồ, có thể gọi phương thức setMap() với tham số null.

```
1. marker.setMap(null);
```

Tuy nhiên phương pháp này không xóa các điểm đánh dấu mà chỉ loại bỏ các điểm đó khỏi bản đồ. Nếu muốn xóa các điểm đánh dấu, sau khi loại bỏ khỏi bản đồ cần thiết lập điểm đánh dấu là null.

Ngoài ra, Marker options còn cho phép chúng ta thực hiện một số tùy chỉnh cho marker như:

- Icon: là đường dẫn đến hình ảnh để thay thế icon mặc định.
- Draggable: cho phép kéo thả marker trên bản đồ. Nó có kiểu dữ liệu true hay false, nếu giá trị là true thì marker có thể kéo thả được.
- Animation: hiệu ứng chuyển động của marker. Có hai loại Drop và Bounce. Drop sẽ chỉ ra rằng điểm đánh dấu được thả từ trên cùng của bản đồ xuống vị trí cuối cùng của điểm đánh dấu trong quá trình khởi tạo. Bounce thể hiện một điểm đánh dấu sẽ nảy liên tục.

## b. InfoWindow

Là một cửa sổ chứa nội dung miêu tả cho một điểm cố định nào đó trên bản đồ.

Đối tượng InfoWindowOptions có các thành phần như sau:

- Content: chứa một chuỗi văn bản, một nút DOM hoặc mã HTML để hiển thị khi InfoWindow được mở.
- Position: là đối tượng kiểu LatLng thể hiện tọa độ hiển thị của InfoWindow. Position có thể gắn liền với một marker.
- maxWidth: xác định chiều rộng tối đa của cửa sổ InfoWindow tính theo đơn vị pixel.

```
1. infowindow.setContent(place.name);
2. infowindow.open(map, this);
```

Khi tạo một InfoWindow, nó không được hiển thị tự động trên bản đồ. Chúng ta cần phải gọi phương thức open(), truyền cho nó tham số map để mở và kèm theo marker hay vị trí cần hiển thị.

Theo mặc định, một InfoWindow sẽ mở cho đến khi người dùng tắt nó (nhấp vào dấu x ở góc phía trên bên phải của cửa sổ thông tin). Nhưng cũng có thể đóng cửa sổ InfoWindow bằng cách gọi phương thức close().

## 2.2.3 Google Maps Event

JavaScript trong trình duyệt được điều khiển theo sự kiện, có nghĩa là nó sẽ thao tác với người dùng thông qua các sự kiện và ngôn ngữ này sẽ lắng nghe những thao tác của người dùng thông qua sự kiện đó. Có 2 loại sự kiện:

- UI Events: lắng nghe sự kiện từ người dùng.
- MVC State Change: lắng nghe sự kiện từ sự thay đổi giá trị của các thuộc tính trên Map.

Để lắng nghe sự kiện, chúng ta có thể sử dụng phương thức `addListener()`. Phương thức này nhận vào một đối tượng, một kiểu sự kiện để lắng nghe và một phương thức xử lý khi sự kiện xảy ra.

#### **a. UI Events**

Một số đối tượng trong Maps API được thiết kế để đáp ứng với sự kiện người sử dụng chẳng hạn như các sự kiện từ chuột hoặc bàn phím. Một đối tượng `google.maps.Marker` có thể lắng nghe người sử dụng các sự kiện sau đây:

- Click
- Dblclick
- Mouseup
- Mousedown
- Mouseover
- Mouseout.

Những sự kiện này có thể trông giống như các sự kiện của DOM nhưng trong thực tế có những sự kiện chỉ dành riêng cho Google API Map. Vì các trình duyệt có cách lắng nghe sự kiện khác nhau nên API JavaScript của Maps cung cấp các cơ chế này để lắng nghe và phản hồi các sự kiện đó mà không cần phải kiểm tra trình duyệt.

#### **b. MVC State Change**

Các sự kiện:

- `zoom_changed`: sự kiện sẽ thực thi khi ta zoom size trên Map.
- `center_changed`: sự kiện sẽ thực thi khi thuộc tính `center` của Map thay đổi.
- `title_changed`: sự kiện sẽ thực thi khi thuộc tính `title` của Map thay đổi

- `heading_changed`: sự kiện sẽ thực thi khi thuộc tính `heading` của Map thay đổi
- `dragstart`: sự kiện sẽ thực thi khi người dùng bắt đầu drag bản đồ
- `drag`: sự kiện sẽ thực thi khi người dùng drag bản đồ
- `dragend`: sự kiện sẽ thực thi khi người dùng kết thúc drag bản đồ
- `maptypeid_changed`: sự kiện sẽ thực thi khi thuộc tính `maptypeid` của map thay đổi.

Bất cứ khi nào thuộc tính của đối tượng bị thay đổi, API JavaScript của Google Maps sẽ kích hoạt sự kiện tương ứng. Ví dụ: API sẽ kích hoạt sự kiện `zoom_changed` trên bản đồ khi mức độ thu phóng của bản đồ thay đổi.

### c. Xử lý sự kiện và tham số.

Để đăng kí thông báo sự kiện, sử dụng phương thức `addListener()`.

```
1. addDomListener(instance:Object, eventName:string, handler:Function)
```

Các thao tác của người dùng thường sẽ thông qua một tham số đại diện cho thao tác đó. Google Maps Events sẽ nhận diện thao tác đó thông qua tham số này. Ví dụ Google Maps sẽ lắng nghe sự kiện “click” lên Marker và hiển thị một `InfoWindow` tại Marker đó.

## 2.2.4 Google Maps Directions

`Directions` là chức năng quan trọng của Google Maps cho phép chúng ta vẽ đường đi từ điểm A đến điểm B. Chúng ta khởi tạo yêu cầu chỉ đường với đối tượng `DirectionsService`, đối tượng này sẽ trả về kết quả chỉ đường là một `DirectionsResult` hoặc một trạng thái `DirectionsStatus`.

Để bắt đầu với dịch vụ chỉ đường, trước hết chúng ta khởi tạo một `DirectionsService`, sau đó gọi tới phương thức `route(request, display)`. Phương thức này gồm 2 thành phần là `Request` chứa yêu cầu của người dùng và `display` để hiển thị kết quả chỉ đường.

### a. Directions Requests

DirectionsRequest gồm các thành phần chính như:

- Origin(bắt buộc): LatLng chứa tọa độ điểm đầu của đường đi.
- Destination(bắt buộc): LatLng chứa tọa độ điểm cuối của đường đi.
- Travelmode: xác định phương thức di chuyển trong việc tính toán đường đi, gồm các loại:
  - DRIVING(mặc định): đường lái xe tiêu chuẩn sử dụng mạng lưới đường bộ.
  - BICYCLING: yêu cầu đường đi xe đạp.
  - TRANSIT: yêu cầu chỉ đường thông qua các tuyến vận chuyển công cộng.
  - WALKING: yêu cầu hướng đi bộ và vỉa hè.

### b. Rendering Directions

Rendering Directions thể hiện kết quả trả về khi yêu cầu DirectionsRequest được gọi. Kết quả trả về gồm một DirectionsStatus và một DirectionsResult.

DirectionsStatus(trạng thái chỉ đường) có thể trả về một số giá trị như:

- OK: cho biết kết quả trả về trong DirectionsRequest là hợp lệ.
- NOT\_FOUND: cho biết điểm đầu hoặc điểm cuối không hợp lệ.
- ZERO\_RESULTS: không có tuyến đường nào được tìm thấy.
- INVALID\_REQUEST: cho biết yêu cầu DirectionsRequest không hợp lệ.
- OVER\_QUERY\_LIMIT: cho biết trang web đã gửi quá nhiều yêu cầu trong khoảng thời gian cho phép.
- REQUEST\_DENIED: cho biết trang web không cho phép sử dụng dịch vụ chỉ đường.
- UNKNOWN\_ERROR: cho biết yêu cầu chỉ đường không được xử lý do lỗi máy chủ, có thể thành công nếu thử lại.

DirectionsResult(kết quả chỉ đường) có thể tự động hiển thị kết quả trên bản đồ bằng việc sử dụng đối tượng DirectionsRenderer. Các bước thực hiện:

- Tạo một đối tượng DirectionsRenderer.
- Gọi setMap() để gắn đối tượng vào bản đồ.
- Gọi setDirection() trên renderer để gắn nó vào DirectionsResult.

Một DirectionsRenderer còn có thể xử lý văn bản hiển thị chi tiết các bước trong việc chỉ đường, để làm như vậy chỉ cần gọi phương thức setPanel() cho DirectionsRenderer, sau đó chuyển nó vào thẻ <div> để hiển thị.

### 2.2.5 Google Maps Places Library

Các chức năng trong thư viện Places cho phép ứng dụng có thể tìm kiếm địa điểm nằm trong khu vực xác định, để tải thư viện chúng ta thêm tham số libraries trong URL khởi tạo bản đồ:

```
1. <script async defer src=https://maps.googleapis.com/maps/api/js?key=Y
   OUR_API_KEY &libraries=places">
2. </script>
```

Dịch vụ Places cung cấp 4 loại tìm kiếm địa điểm:

- Nearby Search: trả về danh sách địa điểm gần vị trí người dùng.
- Text Search: trả về danh sách địa điểm gần đó dựa trên chuỗi tìm kiếm.
- Radar Search: trả về danh sách địa điểm lớn hơn trong bán kính được chỉ định.
- Place Details Request: trả về thông tin chi tiết của một địa điểm cụ thể.

Trong nội dung đề tài chỉ tìm hiểu về loại tìm kiếm Nearby Search. Phương pháp tìm kiếm này phải luôn bao gồm một vị trí, có thể xác định bằng hai cách:

- LatLngBounds: một hình hộp chữ nhật với tọa độ địa lý.
- Một vòng tròn là sự kết hợp của thuộc tính location, trung tâm là một LatLng, một bán kính xác định được đo bằng mét.

Nơi tìm kiếm sẽ gọi đến PlacesService của phương thức nearbySearch(request, callback), sau đó trả về một mảng các đối tượng PlacesResults.

Một số request thường được sử dụng như:

- Location: là một đối tượng LatLng.

- Radius: là một số nguyên cụ thể đại diện cho bán kính của vòng tròn, đơn vị mét. Bán kính tối đa là 50 000m
- Type: xác định nội dung tìm kiếm, ví dụ zoo, hotel..

### 2.2.6 Google Maps Autocomplete

Tự động điền là một trong những tính năng của thư viện Places, khi người dùng bắt đầu gõ một địa chỉ, tính năng này sẽ tự động điền vào phần còn lại.

API cung cấp hai loại công cụ tự động hoàn thành nội dung tìm kiếm thông qua lớp Autocomplete và SearchBox.

- Autocomplete: tạo trường nhập văn bản trên trang web, cung cấp dự đoán về các địa điểm trong danh sách chọn UI và trả về chi tiết địa điểm theo yêu cầu. Mỗi mục trong danh sách chọn tương ứng với một địa điểm.
- SearchBox: tương tự như Autocomplete nhưng cung cấp một danh sách dự đoán mở rộng hơn, bao gồm địa điểm cộng với cụm từ tìm kiếm đề xuất. Khi người dùng chọn một địa điểm từ danh sách, thông tin về địa điểm đó sẽ được trả về đối tượng SearchBox và có thể lấy ra sử dụng.

Một SearchBox gồm hai đối số:

- Input: chứa nội dung cần tìm kiếm trên hộp SearchBox.
- Options: chứa thuộc tính bounds, xác định khu vực tìm kiếm địa điểm.

Để thay đổi vùng tìm kiếm cho SearchBox, gọi phương thức setBounds() và bỏ qua thuộc tính bounds ở trên.

Khi người dùng chọn một địa điểm dự đoán từ hộp SearchBox, dịch vụ sẽ kích hoạt sự kiện places\_changed. Qua đó có thể gọi getPlaces() trên đối tượng SearchBox để lấy một mảng PlacesResults.



## CHƯƠNG 3.XÂY DỰNG ỨNG DỤNG DEMO

Áp dụng các kiến thức tìm hiểu ở trên để xây dựng một website gồm nhiều chức năng sử dụng các dịch vụ của Google Maps API như:

- Xác định vị trí hiện tại của người dùng
- Tìm kiếm địa điểm bất kì bằng cách nhập nội dung vào ô tìm kiếm.
- Chỉ đường giữa 2 địa điểm bằng cách nhập địa chỉ đầu và địa chỉ cuối vào 2 ô tìm kiếm, có hỗ trợ chi tiết đường đi.
- Tìm kiếm các dịch vụ như khách sạn, bệnh viện, ATM,.. theo từng khu vực xác định.

Các bước xây dựng JavaScript:

### 3.1 Load bản đồ

Đặt trung tâm bản đồ tại vị trí myLatLng, mức zoom 8, kiểu bản đồ roadmap.

```
1. var mapDiv = document.getElementById('map');
2.     var myLatLng = new google.maps.LatLng(10.850, 106.650);
3.     initMap = function(){
4. map = new google.maps.Map(mapDiv, {
5.     center: myLatLng,
6.     zoom: 8,
7.     mapTypeId: 'roadmap'
8. });
9. };
```

### 3.2 Thêm và xóa Marker

Tạo một mảng chứa các Marker, thêm các Marker vào bản đồ:

```
1. function setMapOnAll(map) {
2.     for (var i = 0; i < markers.length; i++) {
3.         markers[i].setMap(map);
4.     }
5. }
```

Loại bỏ và xóa các Marker khỏi bản đồ:

```
1. function clearMarkers() {
2.     setMapOnAll(null);
3. }
```

```

4. function deleteMarkers() {
5.     clearMarkers();
6.     markers = [];
7. }

```

### 3.3 Xác định vị trí hiện tại của người dùng (Geolocation)

```

1. function geolocate() {
2.     if (navigator.geolocation) {
3.         navigator.geolocation.getCurrentPosition(function(position) {
4.             console.log(position);
5.             pos = {
6.                 lat: position.coords.latitude,
7.                 lng: position.coords.longitude
8.             };
9.             map.setCenter(pos);
10.         }, );
11.     } else {
12.         alert("no location");
13.     }
14. }

```

Thuộc tính `navigator.geolocation` dùng để kiểm tra trang web có được cho phép truy cập vị trí của thiết bị hay không, nếu được sẽ chạy tiếp phương thức `getCurrentPosition()` để lấy vị trí hiện tại của người dùng.

Tọa độ của vị trí hiện tại được lưu vào biến `pos`, sau đó đặt tọa độ đó vào giữa bản đồ bằng phương thức `setCenter()`. Tiếp đến chúng ta sẽ đặt một Marker đánh dấu vị trí đó:

```

1. function markerLocation() {
2.     var imgMylocation = 'locationicon.png'
3.     map.setZoom(16);
4.     var marker = new google.maps.Marker({
5.         icon:imgMylocation,
6.         position: pos,
7.         animation: google.maps.Animation.DROP,
8.         map :map
9.     });
10. }

```

Hàm `GeolocationControl()` sẽ điều khiển việc click một đối tượng trên bản đồ và xác định vị trí hiện tại:

```

1. function GeolocationControl() {

```

```

2.     var geoButton = document.getElementById('mylocation');
3.     google.maps.event.addDomListener(geoButton, 'click', geolocate);
4.     google.maps.event.addDomListener(geoButton, 'click', markerLocatio
n);
5.     };

```

### 3.4 Tạo hộp tìm kiếm (SearchBox) và chỉ đường

Tạo hộp tìm kiếm:

```

1. function mapSearchbox() {
2.     var input = document.getElementById('pacinput');
3.     var searchBox = new google.maps.places.SearchBox(input);
4.     map.controls[google.maps.ControlPosition.TOP_LEFT].push(input);
5.     map.addListener('bounds_changed', function() {
6.         searchBox.setBounds(map.getBounds());
7.     });
8.     var markers = [];
9.     //Lắng nghe để kích hoạt sự kiện khi người dùng chọn một dự đoán
10.    searchBox.addListener('places_changed', function() {
11.        var places = searchBox.getPlaces();
12.        if (places.length == 0) {
13.            return;
14.        }
15.        // Xóa Marker đã tìm kiếm trước đó
16.        markers.forEach(function(marker) {
17.            marker.setMap(null);
18.        });
19.        markers = [];
20.        // Thêm icon, name cho vị trí
21.        var bounds = new google.maps.LatLngBounds();
22.        places.forEach(function(place) {
23.            if (!place.geometry) {
24.                console.log("Returned place contains no geometry");
25.                return;
26.            }
27.            var icon = {
28.                url: place.icon,
29.                size: new google.maps.Size(71, 71),
30.                origin: new google.maps.Point(0, 0),
31.                anchor: new google.maps.Point(17, 34),
32.                scaledSize: new google.maps.Size(25, 25)
33.            };
34.            // Tạo Marker cho vị trí
35.            markers.push(new google.maps.Marker({

```

```

36.         map: map,
37.         icon: icon,
38.         title: place.name,
39.         position: place.geometry.location
40.     }));
41.     if (place.geometry.viewport) {
42.         // Xác định chế độ xem
43.         bounds.union(place.geometry.viewport);
44.     } else {
45.         bounds.extend(place.geometry.location);
46.     }
47. });
48. map.fitBounds(bounds);
49. });
50. }

```

Để chỉ đường với hộp tìm kiếm, chúng ta tạo thêm 2 hộp SearchBox, sau đó tạo 2 đối tượng là điểm đầu và điểm cuối của việc chỉ đường: LatLng1 và LatLng2, sau đó gán vị trí cho 2 đối tượng này bằng việc sử dụng thuộc tính geometry của Places và thêm vào hàm mapSearchBox():

```

51. function mapSearchbox1() {
52.     .....
53.     .....
54.     // Thêm icon, name cho vị trí
55.     var bounds = new google.maps.LatLngBounds();
56.     places.forEach(function(place) {
57.         // Lấy vị trí chỉ đường thứ nhất
58.         LatLng1 = place.geometry.location;
59.         if (!place.geometry) {
60.             console.log("Returned place contains no geometry");
61.             return;
62.         }
63.         .....
64.         .....
65.     }
66. function mapSearchbox2() {
67.     .....
68.     .....
69.     // Thêm icon, name cho vị trí
70.     var bounds = new google.maps.LatLngBounds();
71.     places.forEach(function(place) {
72.         // Lấy vị trí chỉ đường thứ hai

```

```

73.         LatLng2 = place.geometry.location;
74.         if (!place.geometry) {
75.             console.log("Returned place contains no geometry");
76.             return;
77.         }
78.         .....
79.         .....
80.     }

```

Khởi tạo chỉ đường bằng việc xây dựng hàm calculateAndDisplayRoute() với điểm đầu và điểm cuối là LatLng1 và LatLng2, hiện thông tin chi tiết chỉ đường sử dụng phương thức setPanel():

```

1. var directionsDisplay = new google.maps.DirectionsRenderer;
2. var directionsService = new google.maps.DirectionsService;
3. directionsDisplay.setMap(map);
4. directionsDisplay.setMap(map);
5. directionsDisplay.setPanel(document.getElementById('right-panel'));
6. function calculateAndDisplayRoute(directionsService, directionsDisplay
   ) {
7.     directionsService.route({
8.         origin: LatLng1,
9.         destination: LatLng2,
10.        travelMode: 'DRIVING'
11.    }, function(response, status) {
12.        if (status === 'OK') {
13.            directionsDisplay.setDirections(response);
14.        } else {
15.            window.alert('Directions request failed due to ' + status);
16.        }
17.    });
18. }

```

### 3.5 Tìm kiếm theo khu vực

Sử dụng phương thức tìm kiếm nearbySearch để tìm các dịch vụ như ATM, ngân hàng, khách sạn theo từng khu vực cụ thể. Trước hết chúng ta xây dựng thư viện khu vực và dịch vụ như sau:

Thư viện khu vực, file JavaScript:

```

1. var countries = {
2.     'hn': {
3.         center: {lat: 21.025679, lng: 105.837227},
4.         zoom: 12

```

```

5.         },
6.         'hp': {
7.             center: {lat: 20.847489, lng: 106.698188},
8.             zoom: 12
9.         },
10.        'tb': {
11.            center: {lat: 20.463752, lng: 106.362309},
12.            zoom: 12
13.        },
14.        'na': {
15.            center: {lat: 18.754104, lng: 105.529658},
16.            zoom: 12
17.        },
18.    };

```

File html:

```

1. <select class="select-find" id="country">
2.     <option selected>Chọn địa điểm</option>
3.     <option value="hn">Hà Nội</option>
4.     <option value="hp">Hải Phòng</option>
5.     <option value="tb">Thái Bình</option>
6.     <option value="na">Nghệ An</option>
7. </select>

```

Thư viện dịch vụ, file html:

```

1. <select class="select-find" id="choose">
2.     <option value="atm" selected>Chọn dịch vụ</option>
3.     <option value="atm">ATM</option>
4.     <option value="bank">Ngân hàng</option>
5.     <option value="cafe">Cafe</option>
6.     <option value="hospital">Y tế</option>
7. </select>

```

Xây dựng hàm tìm kiếm find() lấy giá trị khu vực và dịch vụ được chọn ở trên:

```

1. function find() {
2.     var country = document.getElementById('country').value;
3.     var choosee = document.getElementById('choose').value;
4.     var service = new google.maps.places.PlacesService(map);
5.     service.nearbySearch({
6.         location: countries[country].center,
7.         radius: 20000,
8.         type: choosee
9.     }, callback);
10. }

```

```

11. function callback(results, status) {
12.     if (status === google.maps.places.PlacesServiceStatus.OK) {
13.         for (var i = 0; i < results.length; i++) {
14.             // createMarker(results[i]);
15.             addMarker(results[i]);
16.         }
17.     }
18. }

```

Thêm Marker đánh dấu các địa điểm tìm được:

```

1. function addMarker(place) {
2.     var placeLoc = place.geometry.location;
3.     var marker = new google.maps.Marker({
4.         position: place.geometry.location,
5.         map: map
6.     });
7.     markers.push(marker);
8.     google.maps.event.addListener(marker, 'click', function() {
9.         infowindow.setContent(place.name);
10.        infowindow.open(map, this);
11.    });
12. }

```

## **CHƯƠNG 4.KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

### **4.1 Kết luận**

Tóm lại, bài báo cáo đã cung cấp kiến thức cơ bản về bản đồ trực tuyến, các công nghệ hỗ trợ Web, Google Map API. Cụ thể, bài báo cáo niên luận gồm những phần sau:

- Tổng quan về Google Map API.
- Xây dựng một hệ thống gồm các chức năng tìm kiếm địa điểm, xác định vị trí, tìm kiếm các dịch vụ theo khu vực.

Do hạn chế về mặt thời gian cũng như kiến thức, bài niên luận còn nhiều sai sót và thiếu chi tiết, rất mong được sự đóng góp của các thầy/cô và các bạn để bài niên luận được hoàn thiện hơn.

### **4.2 Hướng phát triển**

- Với ứng dụng trên sẽ giúp ích cho người sử dụng dễ dàng tìm được địa điểm như nhà hàng hay một địa điểm du lịch nào đó để thỏa mãn nhu cầu của mình. Hoặc những người chủ của các cửa hàng hay các khách sạn muốn cho người nào có nhu cầu sử dụng tìm đến dịch vụ của họ, thì chỉ việc post địa điểm của mình lên ứng dụng Map này. Do Google Map không hề cho các địa điểm đó lên Map của họ nên từ đó người xây dựng ứng dụng sẽ được một khoản chi phí nho nhỏ từ những người chủ này.
- Xây dựng trang web hoàn thiện hơn về mặt giao diện cũng như chức năng.



## **TÀI LIỆU THAM KHẢO**

### **Tiếng Việt:**

1. TÌM HIỂU VỀ GOOGLE MAP API - <https://goo.gl/iAG34D>
2. Google Map Events Freetuts.net - <https://goo.gl/kVgzxN>

### **Tiếng Anh:**

1. Google Maps APIs Tutorial - <https://goo.gl/uc8nL>
2. Google Maps Reference - <https://goo.gl/cy3xAL>