# Intro

Good evening Dr. Tu, Thank you for joining me today. My name is Tran Duc Anh and today I'm very happy to be here to present to you about my pre-thesis project name Bus Route Lookup WebSite. My prresentation have 6 part, the first part is Introduction, the second part I will talk about literature review, the third part is about methodology of the project, the fourth part will contain implement & result, the fifth part I will show you about the conclusion and my future work, and the last part is demostration. Now I will start my presentation with the first part that is introduction.

## Background

Bus Routine App is a reliable tool that allows people to look up bus routes nearby using a computer or smartphone. It brings many benefits when users can search for bus routes on a map, select time slots, book bus tickets online or moving to the bus stations. Additionally, the website can show the user the shortest path on a map between their current position and the bus stations they choose, for that people can save time and money to get to bus stations as quickly as possible.

## Scope Objecttive

About the scope and objective of this project, The Bus Lookup Website is an easy-to-use self-service system which enables the customer look for bus routing via map that are display on the website's user interface. After that user can also buys bus ticket online on website. Customers can register and login to the website. Customer has to check the availability of the bus ticket before they buy the bus ticket. After process buys bus ticket is successfully, the website will notification that you have successfully booked your ticket and return you to the home page. Customers can choose departure place, arriving place, departure date,departure time and ticket number, the bus ticket also wills states the seat number that assigned by the system automatically.

The Bus Lookup website can allow user to:

- Allows users to search for bus stops around their area.

- Provide a web-based buying bus ticket functions. Customer can buy busticket through the online system and no need to queue up to buy bus ticket inthe counter.

- Enable customer to check the availability of the bus ticket online. Customercan check the time departure and arrival for every bus stops through the system.

## Assumption Solution

On this project, I have assume that users of the bus route lookup website may vary in their technological proficiency and accessibility needs. Some users might prefer mobile access, while others might rely on desktop computers. Additionally, certain users may require features such as screen reader compatibility for visually impaired individuals. And the solution for that is To address the assumption of user accessibility, the website should be designed responsively to ensure compatibility across various devices, including desktops, tablets, and smartphones. Additionally, features such as adjustable font sizes and compatibility with screen readers should be implemented to accommodate users with different accessibility needs.

## Literature Review

Now let me move on to the second part that is Literature Review. In this project I have used the following languages and technologies to build the project such as ReactJS, Javascript and Bootstrap.

ReactJS is a free and open-source front-end JavaScript library for building user interfaces based on components. It is maintained by Meta (formerly Facebook) and a community of individual developers and companies. It was first released in 2013 and has since gained significant popularity in the web development community due to its declarative approach to building UI components and its focus on efficiency and performance.

JavaScript (JS) is a computer programming language used to make websites and applications dynamic and interactive. Along with hypertext markup language (HTML) and cascading style sheets (CSS), JavaScript is one of the most commonly used programming languages of the internet.

And Bootstrap is the most popular CSS Framework for developing responsive and mobile-first websites. Bootstrap is a popular open-source front-end framework primarily used for designing responsive and mobile-first websites and web applications.

For the Back-end of this project, I have use MongoDB. One of MongoDB's hallmark features is its scalability, achieved through horizontal scaling with sharding. By distributing data across multiple servers, MongoDB can handle large volumes of data and high throughput requirements with ease. This scalability, coupled with its high performance, makes MongoDB an ideal choice for applications dealing with massive datasets and demanding workloads.

Now I will move on to the third part of my presentation that call Methodology. Firstly about user requirement alalysist.  For users, the primary requirements revolve around ease of use, accessibility, and the ability to efficiently plan their journeys. This includes features such as a user-friendly interface for searching bus stations, selecting routes based on preferences like timings and stops, and seamless booking and payment processes for tickets.

Additionally, users expect real-time updates on bus status and relevant information such as delays or cancellations. User registration and login functionalities should be secure yet straightforward, allowing for personalized experiences such as storing favorite routes or booking history.

## Use case

And here is the use case diagram for my project. This is a use case diagram for Bus Route Lookup Website illustrates the interactions between users and the system, as well as the roles and functionalities available. Users can perform various actions such as searching for bus stations, selecting routes, logging in or registering, booking bus tickets, and checking bus status and information. The diagram shows that users have access to functionalities crucial for their journey planning and ticket booking needs. Additionally, the diagram depicts an admin actor who has specific privileges like adding users, managing login and registration processes, registering routes, and adding stations. This highlights the administrative control over the system's core functionalities, ensuring smooth operations and user management. Overall, the use case diagram provides a clear overview of how different actors interact with the Bus Route Lookup Website and the functionalities available to them.

## System Design

About the system design, here is a class diagram that I have build for my project. The class diagram provided illustrates the database schema for the Bus Route Lookup Website. Each class within the diagram corresponds to a table in the underlying database schema. Attributes within each class represent the fields contained within the respective tables. The connections between classes delineate relationships between different entities. For instance, the Booking class possesses foreign keys, such as UserID and RouteID, linking it to the User and Route tables, respectively. This signifies a one-to-many relationship, indicating that while a user can have multiple bookings, each booking pertains to only one user and one route. Similarly, the Route class exhibits a one-to-many relationship with the Booking class, implying that a route can have numerous bookings, but each booking correlates with a single route.

## Implementation & result

In the fourth part I will present to you about the implement and result of my project. Let me show you some of the user inteface of the project, the first one that is login UI. The login form is centered on the page and consists of input fields for username and password. The login logic checks user credentials against database and displays appropriate error messages.

Second is the bus list UI. The Bus List component is responsible for rendering a list of available buses. Each bus item displays details such as the bus name, source, destination, departure time, arrival time, price, and bus type. A "Book Now" button is provided for each bus item, allowing users to navigate to the booking page for the selected bus.

And this is a booking bus layout UI. The Bus Layout component is responsible for displaying the layout of available seats on a selected bus. The component dynamically generates the seat layout based on the selected bus's configuration, allowing users to visualize available, booked, and selected seats on the website. Users can select seats by clicking on them, with the selected seats highlighted in blue. Additionally, the component displays information about the selected bus, including its name, bus type, and the number of available, booked, and selected seats. A "Book Now" button is provided at the bottom, enabling users to proceed with booking the selected seats.

## Conclusion & Future Work

Move on to the fifth part of my presentation, that is Conclusion & Future Work. In conclusion, the Bus Lookup Website serves as a convenient web-based platform enabling users to effortlessly check bus routes, ticket availability, and even purchase tickets online. The prevalence of E-ticketing, particularly in developed nations, underscores the growing significance of online ticketing solutions. Moreover, with the widespread accessibility of the internet and electronic devices worldwide, this website emerges as an exceptionally user-friendly tool, offering numerous advantages to its users.

In future I would like to add some more features and improve weaknesses of the website such as:

- Improve user interface, that is enhancing the user interface (UI) is paramount to ensuring a seamless and intuitive user experience. Future iterations of the website should focus on refining the UI to make it more visually appealing, responsive, and user-friendly.

- Completing Login/Register function, the enhancements should focus on simplifying the registration form with minimal fields, and integrating multi-factor authentication methods can bolster security while maintaining user-friendliness.

- Upgrade, improve the HomePage, the hompage can be improve with some features, can display popular routes, current promotions, and essential information about the service.

- Database Completion: A robust and comprehensive database is the backbone of any bus route lookup website. In future I will focus on completing and optimizing the database to ensure the accuracy and reliability of the information provided.

- And the last things is Adding more features to the websites. To further enrich the user experience and meet evolving user needs, additional features can be integrated into the website, such as notification, displays the route from your location to the bus stop, payment platforms.

## References

There are some references that I have consulted and used during the project. And now let me move on to the last part of my presentation today, that is demostration.

# Map Component

This is a constructive for this component:

```
constructor(props){
    super();
    this.state = {
        latitude: 10.99835602,
        longitude: 77.01502627,
        buses: [],
        selectedBusId: null,
        markerClick: false,
        searchText: '',
        distance: 40,
    }
}
```

Buses is a state variable that hold bus data

Sample bus data:

```
let busData = [
    {
        id: '1',
        name: 'Ben xe khu A',
        latitude: 10.874297392244264,
        longitude: 106.8025951292285,
    },

    {
        id: '2',
        name: '33, 53, 99, 30',
        latitude: 10.876741049919326,
        longitude: 106.80247374631635,
    },

    {
        id: '3',
        name: 'Dai hoc Quoc Te',
        latitude: 10.879107142550048,
        longitude: 106.80290114373884,
    },

    {
        id: '4',
```

Each bus object has an id, name, latitude, longtitude

And here I leverage the component did mount to get the location

The coordinate espeacially latitude and longtitude can also be add in here to get the location

```
    //get location
    componentDidMount = () => {
        navigator.geolocation.getCurrentPosition(
            (position) => {
                console.log(position.coords)
                this.setState({
                    latitude: position.coords.latitude,
                    longitude: position.coords.longitude,
                    // buses,
                })
            },

            (error) => {
                console.log('Error getting location' + error.message)
            }
        )
    }
```

Location the bus station on the map:

```
{
    this.state.buses.map((bus) => {
        return (
            <LocationOnIcon color={'secondary'}
                lat={bus.latitude}
                lng={bus.longitude}
                // text = {bus.name}
                onClick={() => { this.setState({ selectedBusId: bus.id, markerClick: true }) }}
            />
        )
    })
}
```

Go this state -> buses -> map, and then for each bus in this array we will return a component . A component that will just be a pin icon that will display the bus station on the map

When you search anything on the web app you always use the lowercase to search so that you excludes any errors that relates to the case of the search . so this function will give you the buses that contain that substring then you can set that as the variable here buses equal to filterBuses

```
const handleSearch = () => {
    let filteredBuses = busData.filter(
        g =>
            g.name.toLowerCase().includes(this.state.searchText.toLowerCase())
            &&
            (getDistanceFromLatLonInkm(this.state.latitude, this.state.longitude, g.latitude, g.longitude) < t
    )
    this.setState({
        buses: filteredBuses
    })
}
```

This method is a mathematical equation that given to latitudes and longtitudes it will give you the distance between them in kilometers

```
const getDistanceFromLatLonInkm = (lat1, lon1, lat2, lon2) => {
    const deg2rad = (deg) => { return deg * (Math.PI / 180) }
    var R = 6371; // Radius of the earth in km
    var dLat = deg2rad(lat2 - lat1); // deg2rad below
    var dLon = deg2rad(lon2 - lon1);
    var a =
        Math.sin(dLat / 2) * Math.sin(dLat / 2) +
        Math.cos(deg2rad(lat1)) * Math.cos(deg2rad(lat2)) *
        Math.sin(dLon / 2) * Math.sin(dLon / 2)
        ;
    var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
    var d = R * c; // Distance in km
    return d;
}
```

The resetAll component will reset the state of  everything back to normal

```
const resetAll = () => {
    this.setState({
        buses: busData,
        distance: 40,
        searchText: ''
    })
}
```

Ànd the last things is I have handle the action that when I click the bus station icon on the map, I will go to another page that will display bus booking places. For that I implement the method here call handleBusClick this will take in the bus parameter and this will replace the current url with the url of the bus booking pages

```
const handleBusClick = (bus) => {
    window.location.replace('/bus/' + bus.id)
}
```