Literature Review

Subjective questionnaires on the comfort sensation and satisfaction degree as well as objective physiological measurements were made in different conditions (office with different plants or without any plants). The following conclusions can be drawn.

1. The high satisfaction degree and comfort sensation demonstrate that an environment with plants can effectively improve human comfort.

2. The environment assessed to be the most comfortable condition was the one with green, slightly scented and small-size plants.

3. The EEG and oxyhaemoglobin saturation in selected six physiological indexes (EEG, ECG, oxyhaemoglobin saturation, fingertip blood flow, skin resistance

1. Larsen L, Adams J, Deal B, et al. Plants in the workplace—the effects of plant density on productivity, attitudes, and perceptions. Environ Behav 1998; 30: 261–281.

2. Han KT. Influence of limitedly visible leafy indoor plants on the psychology, behavior, and health of students at a junior high school in Taiwan. Environ Behav 2009; 41: 658–692.

3. Roelofsen P. The impact of office environments on employee performance: the design of the workplace as a strategy for productivity enhancement. J FacilManag 2002; 1: 18.

4. Wells M. Office clutter or meaningful personal displays: the role of office personalization in employee and organizational wellbeing. J Environ Psychol 2000; 20: 17.

5. Conklin E. Interior plantings bring nature indoors. Amer Nurseryman 1974; 139: 10.

Conclusion: The Plant Information Website will provide a valuable resource for anyone interested in plants, offering a wealth of information and helpful resources in a user-friendly format. It will be an essential tool for gardeners, horticulturists, and anyone else seeking reliable and up-to-date information about plants.

Proposed Work

Purpose: The purpose of this website is to provide a comprehensive source of information about plants, including details about the appearance, habitat, and care requirements of each plant. The website aims to meet the needs of a wide range of users, including gardeners, horticulturists, botanists, and anyone else with an interest in plants.

Intended Audience: The intended audience for this website includes people of all ages and genders who have an interest in plants. This includes gardeners, horticulturists, botanists, and anyone else who is looking for reliable and up-to-date information about plants.

Key Features:

A database of plant information, including details about the appearance, habitat, and care requirements of each plant.

A plant search function, allowing users to find plants based on specific criteria such as scientific name, common name, or region of origin.

Plant identification tools, including a plant identification quiz and a plant identification guide.

Plant care tips and advice, including information on watering, fertilizing, pruning, and other aspects of plant care.

Design and User Interface: The design of the website will be visually appealing, with clear and easy-to-navigate menus and pages. The user interface will be intuitive and user-friendly, with clear instructions and helpful illustrations.

Proposed Method for Development: The website will be developed using PERN stack which consists of developing the front end using basic technologies such as HTML, CSS, and JavaScript. Along with this in order to provide a smooth user experience the project also uses jQuery and ajax for performing asynchronous database operation.

The project is backed by Node framework which is a runtime environment based on javascript , it also express framework which provide greater functionalities to develop a new web application with all required features

In order to provide the database operation service we are using postgres as our database system which a free and open source database management system.

A content management system (CMS) such as PgAdmin is used to manage the database of plant information and allow for easy updates and additions. The development process will involve designing and coding the website, populating the database with plant information, and testing and debugging the website to ensure it is fully functional.

The Project is divided into certain modules which helps in development of the project with ease.

User Signup Module

User Login module

Dashboard module

Admin Login module

Admin content management module

The exact working of all these modules is described in the next section.

Front End:

HTML (HyperText Markup Language) is the standard markup language for creating web pages and web applications. It is used to define the structure and content of a web page, including the text, images, and other media.

CSS (Cascading Style Sheets) is a stylesheet language used for describing the look and formatting of a document written in HTML. CSS is used to control the layout and design of a webpage, including the font, color, and size of text, the placement of elements on the page, and the responsive design of the page for different screen sizes.

JavaScript is a programming language that is commonly used in web development. It is used to add interactivity and dynamic behavior to web pages, such as animations, form validation, and creating event-driven applications.

jQuery is a JavaScript library that makes it easier to work with HTML, CSS, and JavaScript. It provides a number of useful functions and methods for manipulating the DOM (Document Object Model) and handling events, as well as for making AJAX requests.

AJAX (Asynchronous JavaScript and XML) is a technique used for making requests to a server from a client-side web application, without the need to refresh the entire page. It allows web pages to be more interactive and responsive by updating only specific parts of the page based on the data received from the server. AJAX is often used with jQuery, which provides a number of functions and methods for making AJAX requests.

These technologies are all commonly used in front-end web development, and together they form the core of many modern web applications. I hope this helps! Let me know if you have any questions.

Backend:

[Node.js](https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs) is a server-side platform based on the [JavaScript](https://www.simplilearn.com/tutorials/javascript-tutorial/introduction-to-javascript) Engine in Google Chrome. It was created by Ryan Dahl in 2009, and the most recent version is v0.10.36. This is a cross-platform runtime environment for developing server-side and networking applications that are open source. Node.js programs are written in JavaScript and run on the Node.js runtime on OS X, Microsoft Windows, and Linux. Node.js also comes with a big library of JavaScript modules, which makes developing [Node.js web applications](https://www.simplilearn.com/tutorials/nodejs-tutorial/node-app) much easier.

The Node js program runs in a single process rather than establishing a new thread for each request. Blocking behavior is the exception rather than the rule in Node.js, because the standard library offers a set of asynchronous I/O primitives that prevent JavaScript code from blocking, and libraries in Node.js are frequently written using non-blocking paradigms. The popularity of Node.js is skyrocketing right now. Netflix, Uber, PayPal, Twitter, and more well-known companies are presently using Node.js. According to StackOverflow's 2021 Developer Survey, Node.js is the 6th most popular technology among [programmers](https://www.simplilearn.com/job-roles-for-programmers-article), with nearly one-third of professional developers putting it as their first preference.

[Node.js](https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs) is a server-side platform based on the [JavaScript](https://www.simplilearn.com/tutorials/javascript-tutorial/introduction-to-javascript) Engine in Google Chrome. It was created by Ryan Dahl in 2009, and the most recent version is v0.10.36. This is a cross-platform runtime environment for developing server-side and networking applications that are open source. Node.js programs are written in JavaScript and run on the Node.js runtime on OS X, Microsoft Windows, and Linux. Node.js also comes with a big library of JavaScript modules, which makes developing [Node.js web applications](https://www.simplilearn.com/tutorials/nodejs-tutorial/node-app) much easier.

The Node js program runs in a single process rather than establishing a new thread for each request. Blocking behavior is the exception rather than the rule in Node.js, because the standard library offers a set of asynchronous I/O primitives that prevent JavaScript code from blocking, and libraries in Node.js are frequently written using non-blocking paradigms. The popularity of Node.js is skyrocketing right now. Netflix, Uber, PayPal, Twitter, and more well-known companies are presently using Node.js. According to StackOverflow's 2021 Developer Survey, Node.js is the 6th most popular technology among [programmers](https://www.simplilearn.com/job-roles-for-programmers-article), with nearly one-third of professional developers putting it as their first preference.

Node.js is sometimes misunderstood by developers as a [backend framework](https://www.simplilearn.com/tutorials/programming-tutorial/what-is-backend-development) that is exclusively used to construct servers. This is not the case; Node.js can be used on the frontend as well as the backend. The event-driven, non-blocking nature of Node.js frameworks is one of the reasons it is a popular choice for developers designing a flexible and scalable backend.

Some of the reasons why Node.js is suitable for both backend and [frontend development](https://www.simplilearn.com/how-to-become-a-front-end-developer-article) are:

Reusability

With the support of frameworks like Express.js and Meteor.js, JavaScript is a common language for writing both backend and frontend code. Express.js is used as a backend in certain popular stacks, such as MERN (a Node.js framework). Between the frontend and the backend, multiple components can be reused.

Productivity and Developer Efficiency

A significant amount of developer time can be saved by reducing context switching between several languages. Because many technologies are common for both backend and frontend, using JavaScript for both leads to enhanced efficiency.

Node.js Frameworks

Some jobs are still difficult to accomplish with Node.js, so various frameworks have been created to help.

The following are some of the most popular Node.js frameworks:

Nest.js - This is a powerful Node.js backend framework that is appropriate for constructing enterprise-level projects. It has a large number of libraries that implement Typescript, Model-View-Presenter (MVP), and integrated Object-Oriented-Programming (OOP), Function-Point (FP), and Functional-Reactive-Programming (FRP) principles (FRP).

Express.js -is It  a lightweight, minimally designed framework with a large set of HTTP helpers. It is used by developers who do not require a lengthy and costly development procedure. It's also ideal for creating APIs, mobile apps, and web applications.

Socket.io - Its user-friendliness makes it simple to utilise across a variety of platforms. It is primarily concerned with bi-directional real-time connectivity. Reconnection, binary, and multiplexing are also supported.

Meteor.js - This enables real-time functionality, dynamic imports, front-end-back-end connectivity, and API protection.

Koa.js - This uses asynchronous methods to make error handling easier and improve the performance of the application.

Loopback.io - It enables developers to quickly create APIs by providing a number of features that make the process easier. Ad-hoc queries and storage services are supported. It supports a variety of REST services as well as a number of well-known databases.

Express Js

Express is a popular web framework for Node.js. It is a minimalist framework that provides a set of functions and middleware for building web applications and APIs.

Some key features of Express include:

Routing: Express makes it easy to define and handle different routes (URL paths) in a web application.

Middleware: Express allows developers to use middleware functions to perform tasks such as parsing request bodies, serving static files, and handling errors.

Templates: Express supports the use of template engines such as EJS, Pug, and Handlebars, which allow developers to generate dynamic HTML pages based on data from the server.

HTTP helpers: Express provides a number of functions for handling common HTTP tasks, such as redirecting requests, setting response headers, and sending responses.

Together, Node.js and Express form a powerful combination for building modern web applications. They are widely used in the development of server-side JavaScript applications, and are a popular choice for building APIs and micro services.

ER diagram

An entity relationship (ER) diagram is a graphical representation of the entities and relationships in a database. It is used to model the data and how they are related to each other in a database.

An ER diagram consists of several components:

Entities: An entity is a real-world object or concept that is represented in a database. For example, in a database for a library, entities might include books, patrons, and checkouts. Each entity is represented by a rectangular box in the ER diagram.

Attributes: An attribute is a piece of data that describes an entity. For example, the "title" attribute describes a book entity. Attributes are represented by ovals in the ER diagram.

Relationships: A relationship is a connection between two or more entities. For example, a "checkout" relationship connects a patron entity to a book entity. Relationships are represented by a diamond in the ER diagram.

Cardinality: Cardinality refers to the number of entities that can be associated with a relationship. There are two types of cardinality: one-to-one (1:1), one-to-many (1:M), and many-to-many (M:M). These cardinalities are represented by lines connecting the entities and relationships in the ER diagram.

ER diagrams are useful for visualizing and understanding the structure of a database. They can help designers create a database that is easy to use and maintain.

Client server

In a client-server architecture, a client is a computer or device that sends a request to a server, and the server is a computer or device that responds to the request. The client and server communicate with each other over a network, such as the internet.

There are several advantages to using a client-server architecture:

1. Centralized resource management: The server manages and stores shared resources, such as databases and files, which can be accessed by multiple clients. This centralizes resource management and makes it easier to update and maintain the resources.
2. Improved security: The server can be configured to allow only authorized clients to access the resources, improving security.
3. Scalability: The client-server architecture allows for easy scalability by adding more servers as the number of clients increases.
4. Improved reliability: If one client or server fails, the other components can continue to function, improving reliability.

There are also several types of client-server architectures, including two-tier, three-tier, and n-tier architectures. In a two-tier architecture, the client communicates directly with the server, while in a three-tier architecture, the client communicates with an application server, which in turn communicates with the server. In an n-tier architecture, there are multiple layers of servers and application servers.