



SOFTWARE DESIGN SPECIFICATION

Web Development  
  
FindMyCampus

|  |  |  |  |
| --- | --- | --- | --- |
| **Created By:** | Sivakumar Thevar | **Approved By:** | Jonathan Dabre |
| **Created On:** | 21-02-2024 | **Approved On:** | 30-04-2024 |

Page left blank intentionally

**INDEX**

[**1** **PURPOSE** 2](#_Toc142418236)

[**2** **PROJECT SCOPE** 2](#_Toc142418237)

[**3** **SYSTEM OVERVIEW** 2](#_Toc142418238)

[**4** **DESIGN CONSIDERATIONS**](#_Toc142418239) 4

[4.1 Requirements](#_Toc142418240) 4

[4.2 Assumptions](#_Toc142418241) 5

[4.3 Dependencies](#_Toc142418242) 5

[**5** **SYSTEM ARCHITECTURE**](#_Toc142418243) 6

[5.1 Architectural Strategies](#_Toc142418244) 7

[5.2 Structure & Relationships](#_Toc142418245) 7

[**6** **DETAILED DESCRIPTION OF COMPONENTS**](#_Toc142418246) 9

[**7** **INTEGRATION**](#_Toc142418247) 10

[**8** **APPENDICES** 1](#_Toc142418248)

[8.1 Appendix A – Detailed Description of Components 1](#_Toc142418249)

**General Instructions for using the Live Project POC Document**

* This template and the subsequent document created using this template is a confidential document and is the intellectual property of Cloud Counselage Pvt. Ltd. Circulating it outside of the organisation without the consent of Cloud Counselage Pvt. Ltd. is the breach of company policies and will lead to legal actions
* The Design Specification of a software forms the basis of development of software
* The **text between inequality (< >) is to be replaced** by relevant text
* Please **remove the yellow highlight on the Text** between the inequality (< >). This is done to help you notice the text to be changed/replaced
* The text in *italics* highlighted in grey is just for reference and should be removed after adding the relevant text

# **PURPOSE**

This document is created based on the requirement specification document. The purpose of this Software Design Specification (SDS) Document is to break down the project into components to describe in detail what the purpose of each component is and how it will be implemented. The SDS will also serve as a tool for verification and validation of the final product.

# **PROJECT SCOPE**

The scope of the FindMyCampus system includes its distinct features, its benefits, and its limitations. The system's distinct features allow it to have a centralized platform for Indian IT and Management college search and discovery with accurate and reliable information about colleges, courses offered, and contact details by using user inputs. The system enables the user to search college and discovery with accurate and reliable information about colleges, courses offered, and contact details based on their preferences.

A centralized platform for students to search and discover colleges and their courses is lacking, with scattered resources and outdated information. Students are bombarded with advertisements, making it difficult for them to make informed decisions about their education. The application is relevant in addressing the lack of a centralized platform for college search and discovery in India. So developing a platform that provides accurate information without persuasive tactics would be beneficial for students.

# **SYSTEM OVERVIEW**

This section will provide an outline of the various components and subsystems of FindMyCampus application.

FindMyCampus application comprises several key components and subsystems that work together to deliver a functional experience. Let’s break it down:

**1. User Interfaces (UI):**

* The UI encompasses everything that users interact with on a web application. It includes visual elements such as web pages, forms, buttons, menus, and other graphical components.
* UI design focuses on creating an intuitive and visually appealing experience for users.
* Displaying college records in responsive and good looking UI with added pagination.
* User will search college according to their preferences by using form input.

**2. Frontend (HTML, CSS, JavaScript):**

* HTML (Hypertext Markup Language): Defines the structure and content of web pages. It creates the layout, forms, button, links, and other elements.
* CSS (Cascading Style Sheets): Styles the HTML elements by controlling their appearance (colors, fonts, spacing, etc.). It ensures a visually appealing user interface.
* JavaScript: Adds interactivity to the web application. It handles user actions (clicks, form submissions) and dynamically updates the page content without requiring a full page reload.

**3. Backend (JavaScript):**

* Server-side Logic: Handles requests from the frontend. It processes data, communicates with databases, and performs business logic. In this case, we’re using Node.js and Express.js for the backend.
* API Endpoints: Allow the frontend to communicate with the backend. These endpoints define the routes for retrieving college data and other web pages.
* Database Interaction: Connects to the MySQL database to store and retrieve information related to colleges.

**4. Database (MySQL):**

* MySQL Database: Stores college-related data such as college names, address (city), and contact number.
* Tables and Schemas: Define the structure of the database. Tables represent entities and schemas organize related tables.
* Queries: Backend code executes SQL queries to retrieve data from the MySQL database.

# **DESIGN CONSIDERATIONS**

This section describes requirements, assumptions and dependencies to be addressed to devise a complete design solution.

## Requirements

**1. Hardware Requirements:**

* Operating System:
  + - Windows: Windows 10 or later.
    - macOS: macOS Catalina 10.15 or later.
    - Linux: 64-bit Ubuntu 18.04+, Debian 10+, openSUSE 15.5+, or Fedora Linux 38+
* Processor (CPU):
  + - An Intel Pentium 4 or later processor is typically sufficient for web development tasks.
* Memory (RAM):
  + - A minimum of 2 GB RAM is required, but it’s recommended to have at least 4 GB for smoother performance.
* Screen Resolution:
  + - Your screen resolution should be 1280x1024 or larger.
* Application Window Size:
  + - The application window size should be 1024x680 or larger.

**2. Software requirements:**

* + IDE: VSCode/Notepad++
  + Front-End: HTML, CSS, JS
  + Back-End: Node.js, Express.js, ejs
  + Browser: Chrome/Firefox/Edge
  + Database: SQL database (XAMPP)

**3. Functional Requirements:**

* + Easy to navigate pages.
  + Smooth experiences with added animation effects.
  + Well-presented records by great UI/UX in the front-end.
  + Pagination for displaying 10 records per page to improve user experience.
  + Easily searchable through form and well-presented upon the search input.

**4. Non-Functional Requirements:**

* + Usability: The user interface should be intuitive and easy to navigate.
  + Reliability: The system should be available and data is accurate and up-to-date to provide reliable search results to users.
  + Responsiveness: Responsive design principles of web application ensure the application is accessible and functional across various devices and screen sizes.

**5. External Interface Requirements:**

* + User interactions: How users interact with the web app through the client-side interface.
  + Databases: Interaction with database systems.

## Assumptions

* Assuming user will give right info in input fields on the form.
* Assuming user will navigate properly in website.
* User will have good internet connection.

## Dependencies

* Web Application is rely on database
* Frontend and Backend is depend on certain packages and libraries

# **SYSTEM ARCHITECTURE**

The software system architecture refers to the logical organization of a distributed system into software components. It defines how components of a software system are assembled, their relationship and communication between them. It serves as a blueprint for software application and development basis for developer team. An effective architecture serves as the conceptual glue that holds every phase of the project together for all of its stakeholders, enabling agility, time and cost savings, and early identification of design risks.

The Software architecture:

* Defines structure of a system
* Defines behaviour of a system
* Defines component relationship
* Defines communication structure
* Balances stakeholder’s needs
* Influences team structure
* Focuses on significant elements
* Captures early design decisions

Below some important characteristics which are commonly considered are explained.

**Operational Architecture Characteristics:**

* Availability
* Performance
* Reliability
* Low fault tolerance
* Scalability

**Structural Architecture Characteristics:**

* Configurability
* Extensibility
* Supportability
* Portability
* Maintainability

**Cross-Cutting Architecture Characteristics:**

* Accessibility
* Security
* Usability
* Privacy
* Feasibility

## Architectural Strategies

The major components of the system architecture for the FindMyCampus web application are:

1. User Interfaces (UI)

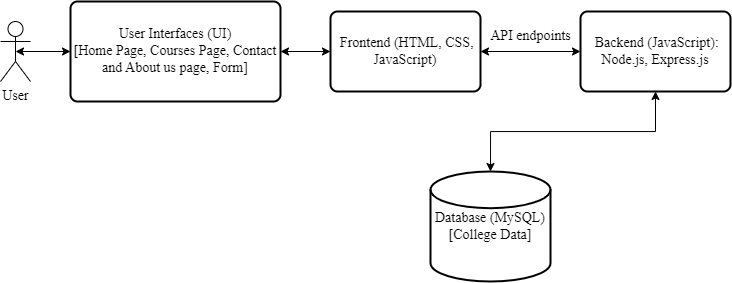
2. Frontend (HTML, CSS, JavaScript)

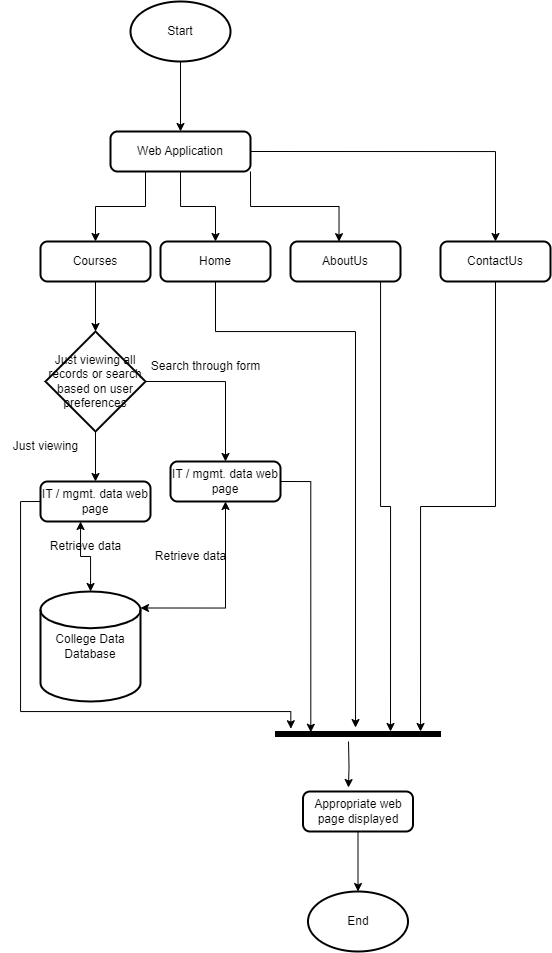
3. Backend (JavaScript): Node.js, Express.js, ejs

4. Database (MySQL)

5. API (Application Programming Interface)

## Structure & Relationships





# **DETAILED DESCRIPTION OF COMPONENTS**

For detailed description of the components, please refer **Appendix A – Detailed Description of Components**

The below template will be used to specify the details of all the components

**Table 1: Detailed Design Specification Template**

|  |  |
| --- | --- |
| **Identification** | The unique name for the component and the location of the component in the system. |
| **Type** | A module, a subprogram, a form, a data file, a control procedure, a class, etc. |
| **Purpose** | Function and performance requirements implemented by the design component, including derived requirements. Derived requirements are not explicitly stated in the SRS - but are implied or adjunct to formally stated SDS requirements. |
| **Subordinates** | The internal structure of the component, the constituents of the component, and the functional requirements satisfied by each part. |
| **Dependencies** | How the component’s function and performance relate to other components. How this component is used by other components. The other components that use this component. Interaction details such as timing, interaction conditions (such as order of execution and data sharing), and responsibility for creation, duplication, use, storage, and elimination of components. |
| **Interfaces** | Detailed description of all external or internal interfaces as well as of any mechanism for communicating through messages, parameters, or common data areas. All error messages and error codes should be identified. All screen formats, interactive messages, and other user interface components (originally defined in the SRS) should be given here. |
| **Resources** | A complete description of all resources (hardware or software) external to the component but required to carry out its functions. |
| **Processing** | A full description of the functions presented in the Function subsection. Pseudocode can be used to document algorithms, equations, and logic. |
| **Data** | For the data internal to the component, describes the representation method, initial values, use, semantics, and format. |

# **INTEGRATIONS**

1. Frontend Development Tools:

* + Frontend development (also known as client-side development) focuses on creating the user interface and making websites visually appealing. It involves using technologies like HTML, CSS, and JavaScript to construct web pages.
  + In our application, the frontend would handle user interactions, display search results, and provide a seamless user experience. It integrate with external css libraries.

2. Backend Development Tools:

* + Backend development (also known as server-side development) deals with making websites work efficiently. It involves managing server-side, handling requests from the frontend, interacting with databases and APIs. It involves JavaScript technology.
  + In our application, the backend would handle search queries, retrieve data from the database, and serve it to the frontend. It integrate with external JS libraries for backend development like ejs, mysql.

3. Database Integration:

* + Database is essential for storing and managing data. We are using SQL database (XAMPP).
  + In our application, the database would store information about colleges, courses, and other relevant data. For example, it would store details like college names, locations, admission criteria, and available programs.

4. API endpoints:

* These endpoints allow the frontend to retrieve data from the database.
* For example, when a user searches for colleges, the frontend sends a request to the backend API, which then queries the database and returns relevant results.

# **APPENDICES**

## Appendix A – Detailed Description of Components

|  |  |
| --- | --- |
| **Identification** | **Home Page** |
| **Type** | Content component |
| **Purpose** | Display essential information and guide users. |
| **Subordinates** | This screen contains links to the following screens:   1. Menu items |
| **Dependencies** | Navigation Menu |
| **Interfaces** | None (static content) |
| **Resources** | Text content, images |
| **Processing** | None (static page) |
| **Data** | None (displays static data) |

|  |  |
| --- | --- |
| **Identification** | **About Us Page** |
| **Type** | Content component |
| **Purpose** | Provide background and mission. |
| **Subordinates** | - |
| **Dependencies** | Navigation Menu |
| **Interfaces** | None (static content) |
| **Resources** | Text content, images |
| **Processing** | None (static page) |
| **Data** | Company information |

|  |  |
| --- | --- |
| **Identification** | **Contact Us Page** |
| **Type** | Interaction component |
| **Purpose** | Allow users to reach out to the company. |
| **Subordinates** | Contact Form |
| **Dependencies** | Navigation Menu |
| **Interfaces** | Form submission |
| **Resources** | Form fields, image |
| **Processing** | - |
| **Data** | User inquiries |

|  |  |
| --- | --- |
| **Identification** | **Courses [BTech, MTech, BBA, MBA] page** |
| **Type** | Interaction component |
| **Purpose** | Display college data of IT/management courses |
| **Subordinates** | Form for user search according to their preferences |
| **Dependencies** | Navigation Menu, database |
| **Interfaces** | API calls for retrieving college data, Form submission to backend |
| **Resources** | Images, CSS styles, form fields, college database’s college and its courses details |
| **Processing** | Processes user input to retrieve relevant results. |
| **Data** | College Data retrieved from database |