



# Four-Year Course Planning Assistant for Pepperdine University

Frankie Mundo, Eric Njuku

Mentor: Dr. Scalzo

Pepperdine University, 24255 Pacific Coast Highway, Malibu, CA 90263

## Introduction

Pepperdine students need an efficient tool to help them organize a course plan to ensure that they graduate within four years.

Our Objective: Develop a web-based application to help students quickly and easily generate a four-year course plan based on their major or minor.

## Methods

Database Design: Designed relational database using MySQL Workbench:

- Two tables: Courses (Course ID, course title, number of units, and if GE requirement) and Majors\_Minors (program name and if major/minor)
- Created a new table called Courses\_Has\_Majors\_Minors with an n:m relationship, where the two elements are primary keys from other tables
- Utilized scripts to fill tables with information from Pepperdine course catalog.

Backend:

- Used the Node.js runtime environment to start server and run on local port.
- Wrote the code for the backend in a file called PeppAI.js.
  - connects to our MySQL database and uses Express.js framework to handle HTTP requests, outputting plan based on four-year 128 credit plan
- Created a script.js file to handle the fetch request to the backend endpoint:
  - request loops through the course plan data to display each course, evenly divided among four years

Frontend: Created a file called index.html to manage raw text on the local webpage and file called styles.css to handle the visual styling of the webpage.

```
.JS PeppAI.js x index.html JS script.js # styles.css
Users > ericnjuku > Desktop > COSC490 > public > JS PeppAI.js > connection.connect() callback
1 // Import required modules
2 const mysql = require('mysql');
3 const readline = require('readline');
4
5 const express = require('express');
6 const path = require('path');
7
8 const app = express();
9 const port = 3000;
10
11 // Serve static files from the 'public' directory
12 app.use(express.static(path.join(__dirname, 'public')));
13
14 // Define a route to handle requests to the root URL
15 app.get('/', (req, res) => {
16   res.sendFile(path.join(__dirname, 'public', 'index.html'));
17 });
18
19 // Create MySQL connection
20
21 const connection = mysql.createConnection({
22   host: 'localhost', // MySQL host
23   user: 'root', // MySQL username
24   password: 'Karethis15', // MySQL password
25   database: 'Pepperdine_Database' // MySQL database name
26 });
```

```
.JS PeppAI.js x index.html JS script.js # styles.css
Users > ericnjuku > Desktop > COSC490 > public > index.html > ...
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Course Plan Generator</title>
7   <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10   <script src="script.js"></script>
11   <div class="container">
12     <h1>Pepperdine Course Planning Assistant</h1>
13     <label for="major-minor">Enter your major or minors</label>
14     <input type="text" id="major-minor" placeholder="e.g., Accounting">
15     <button onclick="generatePlan()">Generate Plan</button>
16   </div>
17 </body>
18 </html>
```

```
.JS PeppAI.js x index.html JS script.js # styles.css
Users > ericnjuku > Desktop > COSC490 > public > JS script.js > generatePlan > then() callback
1 function generatePlan() {
2   // Retrieve the value of the input field where users enter their major/minor
3   const majorMinor = document.getElementById('major-minor').value;
4
5   // Make a fetch request to the backend endpoint to generate the course plan
6   fetch('/course-plan/has-prerequisites?major=minor')
7     .then(response => {
8       // Check if the fetch request was successful (status code 200)
9       if (!response.ok) {
10         throw new Error('Network response was not ok!');
11       }
12       // Parse the JSON response
13       return response.json();
14     })
15     .then(data => {
16       // Access the course plan data from the response
17       const coursePlanObj = document.getElementById('course-plan');
18       coursePlanDiv.innerHTML = ''; // Clear previous content
19
20       // Check if the response contains an error message
21       if (data.error) {
22         coursePlanDiv.textContent = data.error; // Display error message
23       } else {
24         const coursePlan = data.coursePlan;
25
26         // Calculate the number of courses per year
27         const coursesPerYear = Math.ceil(coursePlan.length / 4);
28
29         // Initialize variables to keep track of the displayed year title
30         let currentYearIndex = 0;
31         let yearTitleDisplayed = false;
32
33         // Loop through the course plan data and display each course
34         for (let i = 0; i < coursePlan.length; i++) {
35           // Calculate the year index based on the current index and courses per year
36           const yearIndex = Math.floor(i / coursesPerYear);
```

```
.JS PeppAI.js x index.html JS script.js # styles.css
Users > ericnjuku > Desktop > COSC490 > public > # styles.css > button
1 body {
2   font-family: Arial, sans-serif;
3   margin: 0;
4   padding: 0;
5
6
7   .container {
8     max-width: 600px;
9     margin: 50px auto;
10    padding: 20px;
11    border: 1px solid #ccc;
12    border-radius: 5px;
13
14    input[type="text"] {
15      width: 100%;
16      padding: 10px;
17      margin-bottom: 10px;
18    }
```

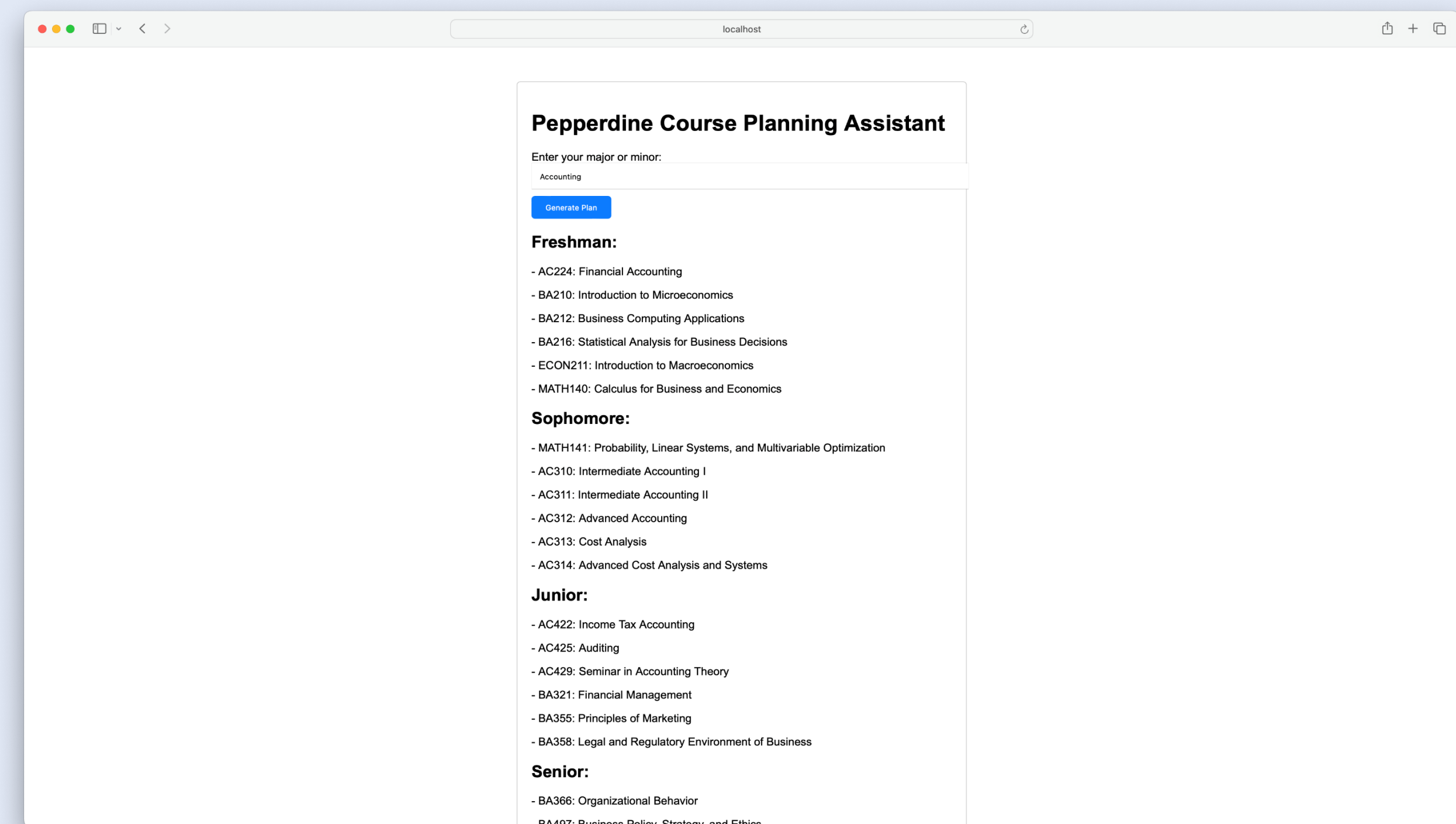
## Results

Our result is a local web application with an input field for users to enter their major or minor.

- User inputs their program of choice and clicks the “Generate Plan” button and a list of required classes for their program is displayed.
- The courses are organized by year

The database is fully implemented, with all information present, and the code is fully functioning with no errors, but we were unable to add handling for GE courses on the front end and prerequisites in general.

```
Server running at http://localhost:3000
Connected to MySQL database
Enter your major or minor: Computer Science/Mathematics
Course plan for Computer Science/Mathematics:
Freshman:
- COSC101: Programming Principles I with Python
- COSC105: Programming Principles I with R
- COSC121: Programming Principles II
- MATH150: Calculus I
- MATH151: Calculus II
- MATH220: Formal Methods
- MATH221: Discrete Structures
```



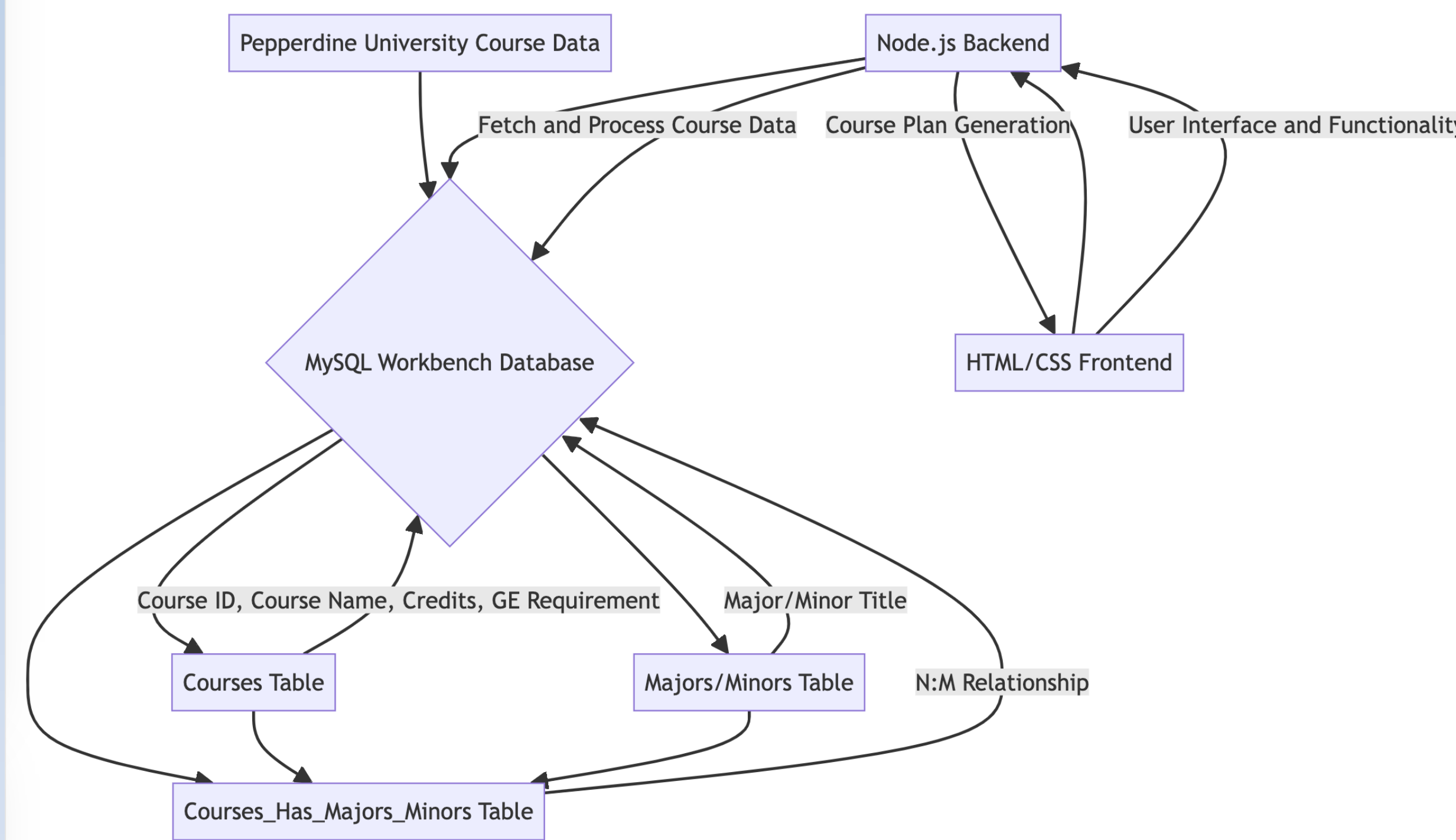
## Discussion

Our tool effectively displays the courses necessary for each major or minor at Pepperdine and the code itself displays the list of the classes in addition to potential GE classes that the student can take.

Future improvements:

- Account for "or" classes, where there are multiple options for a class to take
- Add the GE classes to the front-end user interface
- Implement a table in the database to account for prerequisite courses and add the code to handle this
- Improve the user interface to make it more visually appealing

### Pepperdine Course Planning Assistant Architecture



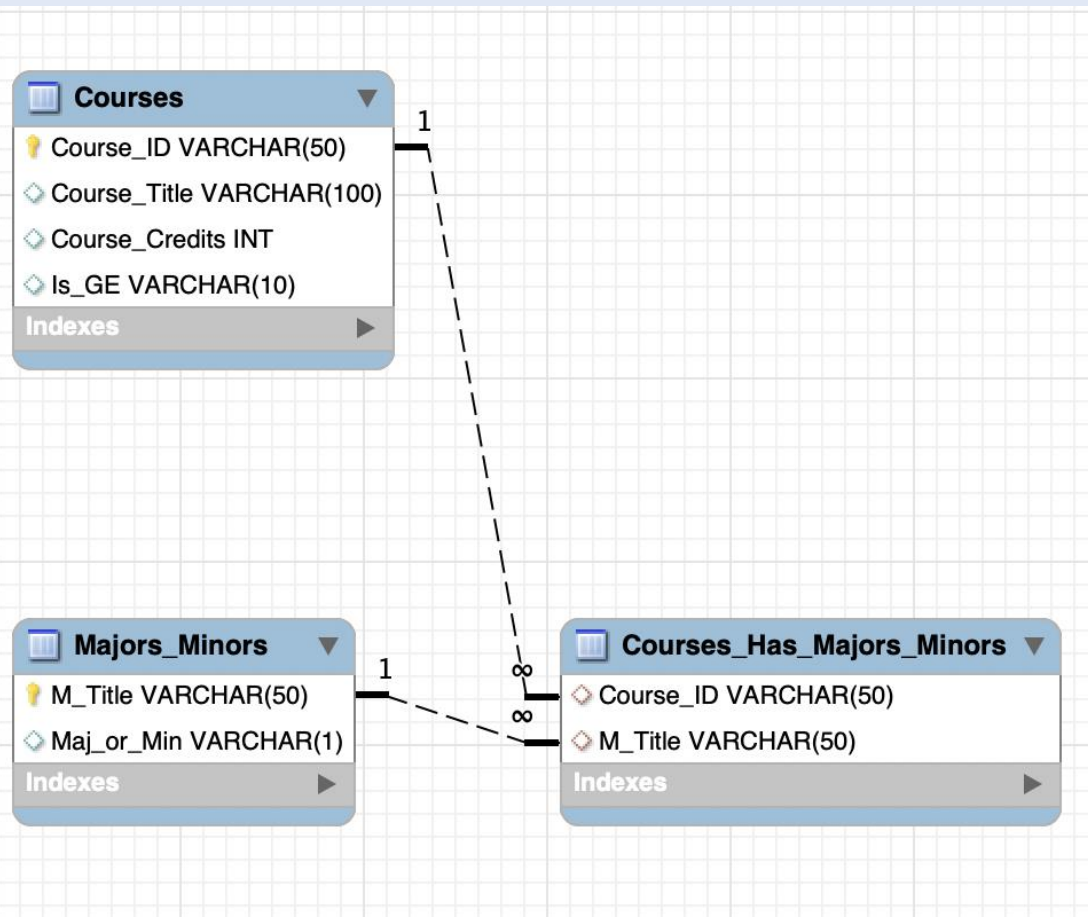
The diagram above illustrates the flow of data and the interactions between the different components of the application, showing the integration of a MySQL Workbench database, a Node.js backend, and an HTML/CSS frontend.

## Conclusion

We have effectively constructed a Pepperdine course planning assistant on the full stack:

- Constructed a database using MySQL and utilized scripts to populate the tables within
- Created a Node.js server to run a local webpage
- Managed information with JavaScript
- Developed the front end of the user interface with HTML and CSS.

This webpage will allow students to generate a four-year course plan based on their major/minor, with plans to implement more frontend/backend improvements to the application in the future.



```
Administration Schemas PeppAI* Majors_Minors Courses Courses_Has_Majors_Minors
SCHEMAS
Filter objects
Pepperdine_Database
Tables
Courses
Courses_Has_Majo...
Majors_Minors
Views
Stored Procedures
Functions
sys
Result Grid
Filter Objects Edit Export/Import
Course_ID Course_Title Course_Credits Is_GE
AAS200 Introduction to African American Studies 4 N
AAS292 Selected Topics 4 N
AAS299 Directed Studies 4 N
AAS431 African American Cinema 4 N
AAS592 Selected Topics 4 N
AAS599 Directed Studies 4 N
AC224 Financial Accounting 4 N
AC225 Managerial Accounting 3 N
AC292 Special Topics 4 N
AC310 Intermediate Accounting I 4 N
AC311 Intermediate Accounting II 4 N
AC312 Advanced Accounting 4 N
AC313 Cost Analysis 3 N
AC314 Advanced Cost Analysis and Systems 3 N
AC422 Income Tax Accounting 4 N
AC425 Auditing 4 N
AC429 Seminar in Accounting Theory 3 N
AC492 Selected Topics 4 N
AC495 Experiential Learning for Accounting 4 N
AC499 Directed Studies 4 N
AC501 Ethics for Accounting 4 N
AC506 Advanced Taxation and Ethics 4 N
AC511 Accounting for Nonbusiness Organizat... 4 N
AC517 Financial Statement Analysis 4 N
AC524 Advanced Auditing 4 N
```

```
1 CREATE DATABASE Pepperdine_Database;
2 USE Pepperdine_Database;
3
4 CREATE TABLE Courses (Course_ID VARCHAR(50),
5   Course_Title VARCHAR(100),
6   Course_Credits INT(1),
7   Is_GE VARCHAR(10),
8   PRIMARY KEY (Course_ID));
9
10 INSERT INTO Courses (Course_ID, Course_Title, Course_Credits, Is_GE)
11 VALUES ("AC224", "Financial Accounting", 4, "N"),
12 ("AC225", "Managerial Accounting", 3, "N"),
13 ("AC292", "Special Topics", 4, "N"),
```