Project Step 7: Group 112: CRUD Database Project

By:

Jada Young & Omar Garza Cantu (Group 112)

Oregon State University

Project URL: http://flip1.engr.oregonstate.edu:7473/

Report Contents

Executive Summary

Step 1 & 2 Feedback:

Corrections were made to the attribute names which all involved capitalized first letters on each word separated with an underscore. It was also decided that the primary key of the CurEnroll and StuAccts tables should not have to be a separate auto incrementing value, but could for the StuAccts table be the student id, while for CurEnroll it all the values would work as the primary key.

Step 3 Feedback:

It was pointed out that CurEnroll was the only table that was not plural, which was an oversight which was corrected by renaming it to CurEnrolls.

Step 4 Feedback:

The alter table query for the data definition queries was not implemented for the CurEnrolls as it was missing a foreign key with Student_ID on the Students table. The other issues were related to the sample data that we provided which did not have matching foreign keys. Because the work was split different sample table entries were used which led to the discrepancy.

Step 5 & 6 Feedback:

A validation function was implemented to deal with the format of email and phone numbers provided. There were also some issues with the sample data being stored in an incorrect format. There were issues related to having separate sites for the project which will be consolidated for the final project, so it was of minor concern. It was decided that payments made or removed would affect the StuAcct with queries to keep it updated.

Project Overview & Database Outline

Project Overview:

Tech School serves 100 students a year. They need a database that tracks students, courses, instructors, and payments. Additionally, student balances need to be retrieved and updated with payments processed via one of 4 methods (Cash, Credit/Debit, Check, and Wire) or with the tuition cost of enrollment into one or more classes. Tech School expects to process 1,000 payments and have around 250 enrollments. Student balances are expected to be less than 20 thousand dollars as students who join the technical school also work either full time or part time and usually enroll in just one class per semester. A database-driven website will keep record of all students, which courses students are and were previously enrolled in, courses offered, and the student account balance (payments received and balance still owed).

Database Outline:

Students: records the details of the students previously and currently enrolled.

- **Student_ID:** int, auto_increment, unique, not NULL, PK
- **Email:** varchar, not NULL
- **First_Name:** 15 char, not NULL
- **Last_Name:** 15 char, not NULL
- **Phone:** 10 char, not NULL
- **Address_ID:** int, not NULL
- **Relationship:** A M:M relationship between Students and Courses is implemented with Student_ID as an FK inside of CurEnrolls. Students can be enrolled in multiple courses. A 1:1 relationship between Students and StuAccts is implemented with Student_ID as an FK inside StuAccts.(Note: Students expected to be Domestic as this is a brick and mortar location). A 1:1 relationship between Students and Addresses is implemented using Address_ID.

- **Addresses: records the details of a student’s address.**
    - Address_ID: int, unique, auto_increment, not NULL, PK
    - Street: varchar, not NULL
    - City: varchar, not NULL
    - State: varchar, not NULL
    - Zip: 9 char
    - Country, varchar, not NULL
    - **Relationship:** 1:1 relationship between Students and Addresses is implemented using Address_ID as FK in Students.

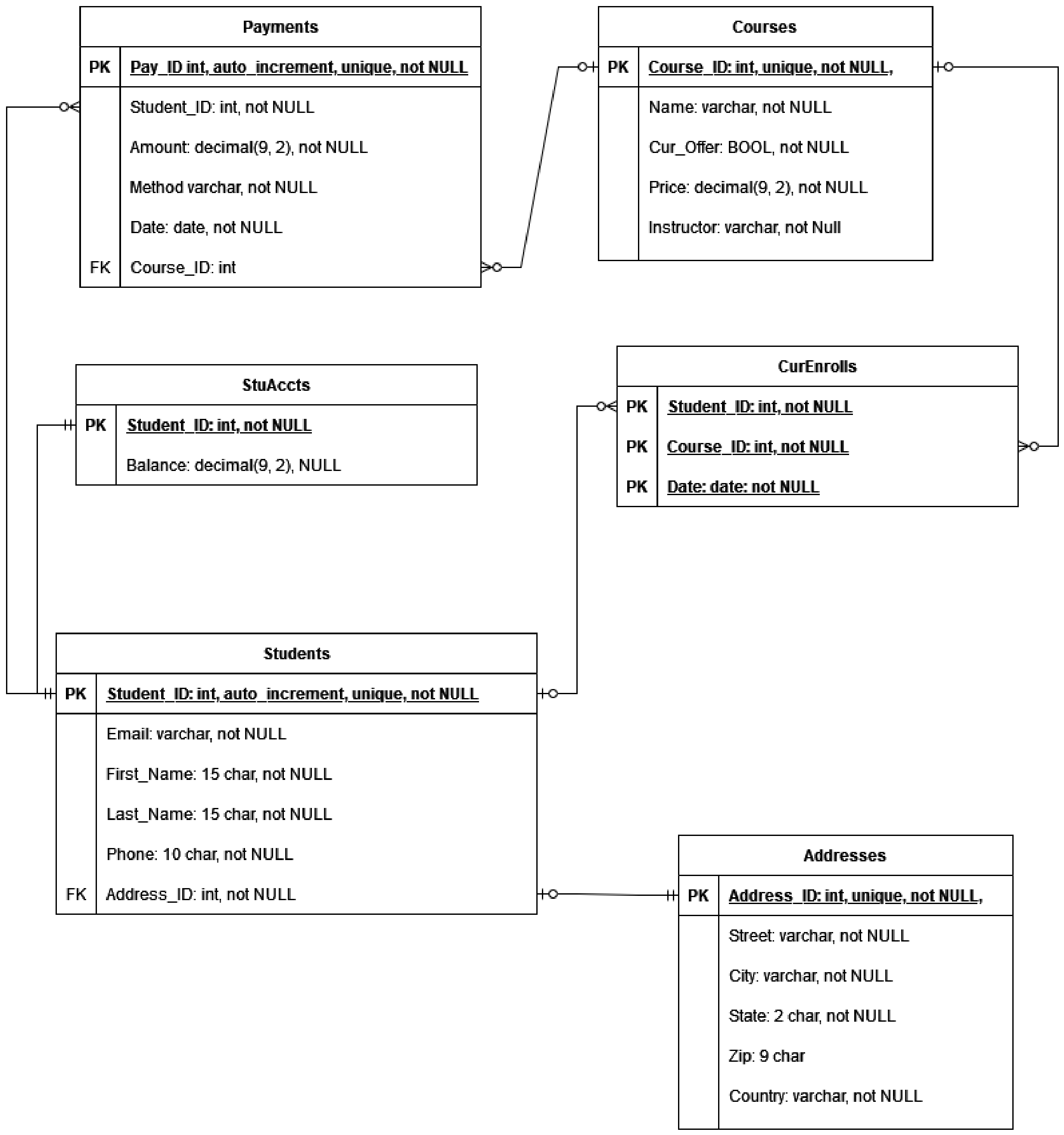
- **CurEnrolls: records ID of students and courses that they are currently enrolled in.**
    - **Student_ID:** int, not NULL, PK
    - **Course_ID:** int, not NULL, PK
    - **Date:** date: not NULL, PK
    - **Relationship:** A M:M relationship between Students and Courses is implemented using Student_ID and Course_ID as FKs and PKs in CurEnrolls. To generate a unique row and to keep track of courses that may have been taken more than once, the Date attribute is added.

- **StuAccts: holds the records for payments and the tuition cost of the student in order to generate a current student account balance.**
    - **Student_ID:** int, not NULL, PK
    - **Balance:** decimal(9, 2), not NULL
    - **Relationship:** A M:M relationship between Students and Payments is implemented using Student_ID and Pay_ID as PK in StuAccts.

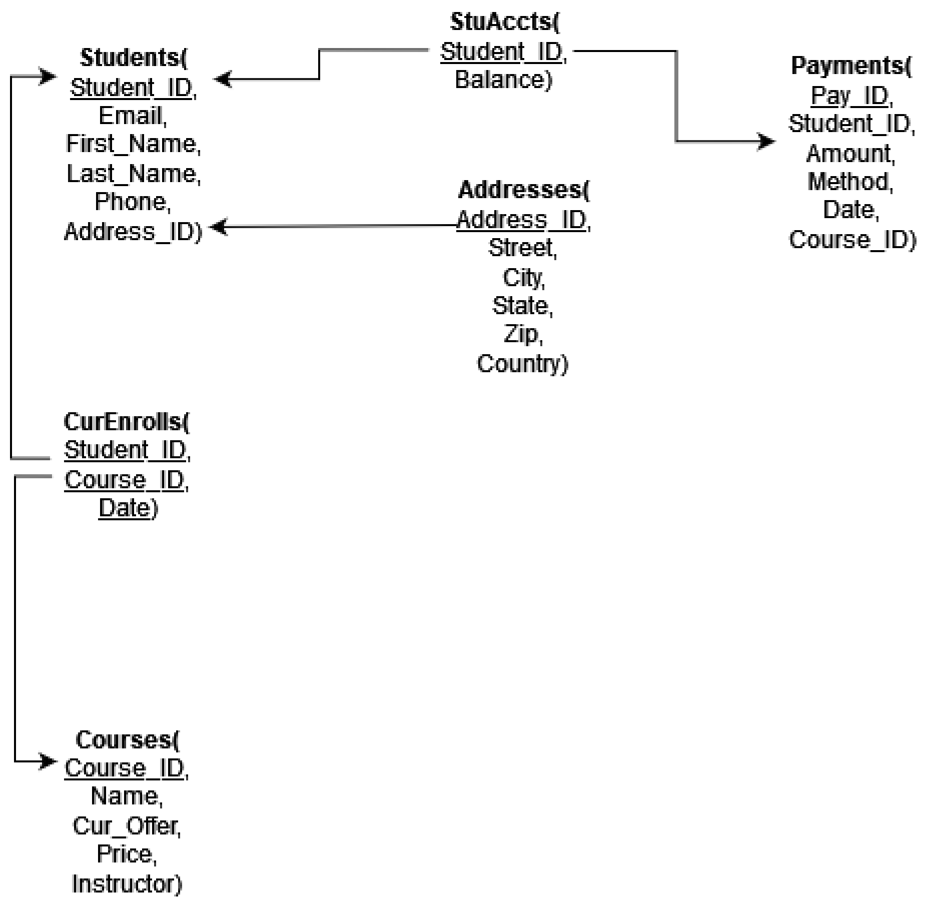
- **Payments: holds the records for payments made towards tuition.**
    - **Pay_ID:** int, auto_increment, unique, not NULL
    - **Student_ID:** int, not NULL
    - **Amount:** decimal(9, 2), not NULL
    - **Method:** varchar, not NULL
    - **Date:** date, not NULL
    - **Course_ID:** int
    - **Relationship:** A M:M relationship between Payments and Students is implemented using Student_ID as an FK inside Payments. A 1:M relationship is implemented between StuAccts and Payments using Student_ID as a PK in StuAccts.

- **Courses: records courses offered by the school**
    - **Course_ID:** int, unique, not NULL, PK
    - **Name:** varchar, not NULL
    - **Cur_Offer:** BOOL, not NULL
    - **Price:** decimal(9, 2), not NULL
    - **Instructor:** varchar, not Null
    - **Relationship:** A M:M relationship is implemented between Courses and Students using Course_ID as a FK inside CurEnrolls.

Entity Relationship Diagram


Payments
PK Pay_ID int, auto_increment, unique, not NULL
Student_ID: int, not NULL
Amount: decimal(9, 2), not NULL
Method varchar, not NULL
Date: date, not NULL
FK Course_ID: int
Courses
PK Course_ID: int, unique, not NULL,
Name: varchar, not NULL
Cur_Offer: BOOL, not NULL
Price: decimal(9, 2), not NULL
Instructor: varchar, not Null
StuAccts
PK Student_ID: int, not NULL
Balance: decimal(9, 2), NULL
CurEnrolls
PK Student_ID: int, not NULL
PK Course_ID: int, not NULL
PK Date: date: not NULL
Students
PK Student_ID: int, auto_increment, unique, not NULL
Email: varchar, not NULL
First_Name: 15 char, not NULL
Last_Name: 15 char, not NULL
Phone: 10 char, not NULL
FK Address_ID: int, not NULL
Addresses
PK Address_ID: int, unique, not NULL,
Street: varchar, not NULL
City: varchar, not NULL
State: 2 char, not NULL
Zip: 9 char
Country: varchar, not NULL

Schema

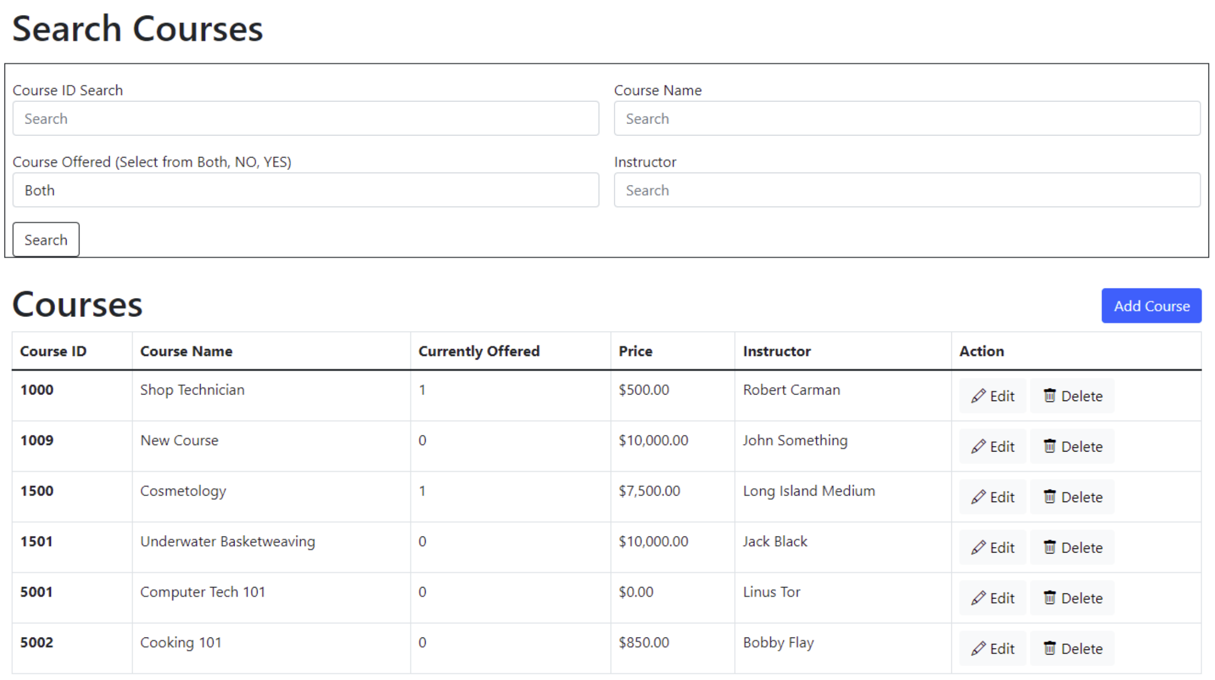


StuAccts(Student_ID, Balance)
Students(Student_ID, Email, First_Name, Last_Name, Phone, Address_ID)
Payments(Pay_ID, Student_ID, Amount, Method, Date, Course_ID)
Addresses(Address_ID, Street, City, State, Zip, Country)
CurEnrolls(Student_ID, Course_ID, Date)
Courses(Course_ID, Name, Cur_Offer, Price, Instructor)

UI Pages with CRUD Steps

Courses - Read & Delete:

The search box at the top affects the courses displayed below and each row has a delete button.



Search Courses

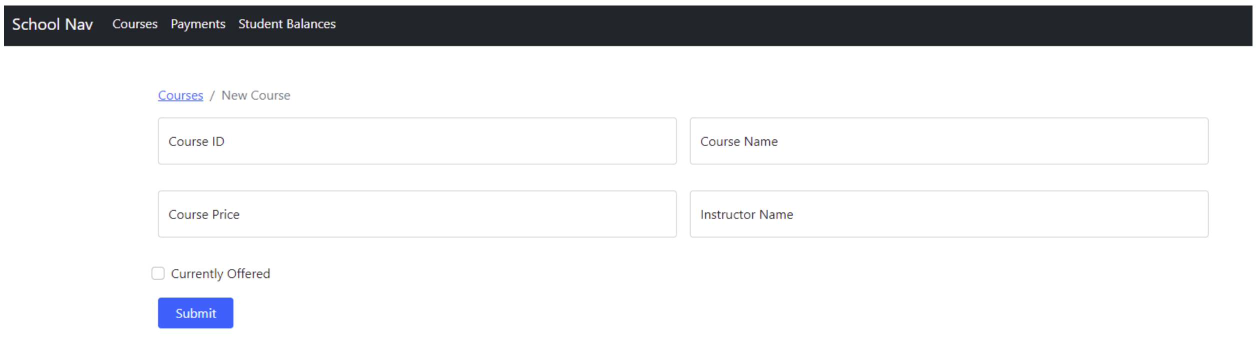
Course ID Search: Search	Course Name: Search
Course Offered (Select from Both, NO, YES): Both	Instructor: Search

Courses

Course ID	Course Name	Currently Offered	Price	Instructor	Action
1000	Shop Technician	1	$500.00	Robert Carman	Edit Delete
1009	New Course	0	$10,000.00	John Something	Edit Delete
1500	Cosmetology	1	$7,500.00	Long Island Medium	Edit Delete
1501	Underwater Basketweaving	0	$10,000.00	Jack Black	Edit Delete
5001	Computer Tech 101	0	$0.00	Linus Tor	Edit Delete
5002	Cooking 101	0	$850.00	Bobby Flay	Edit Delete

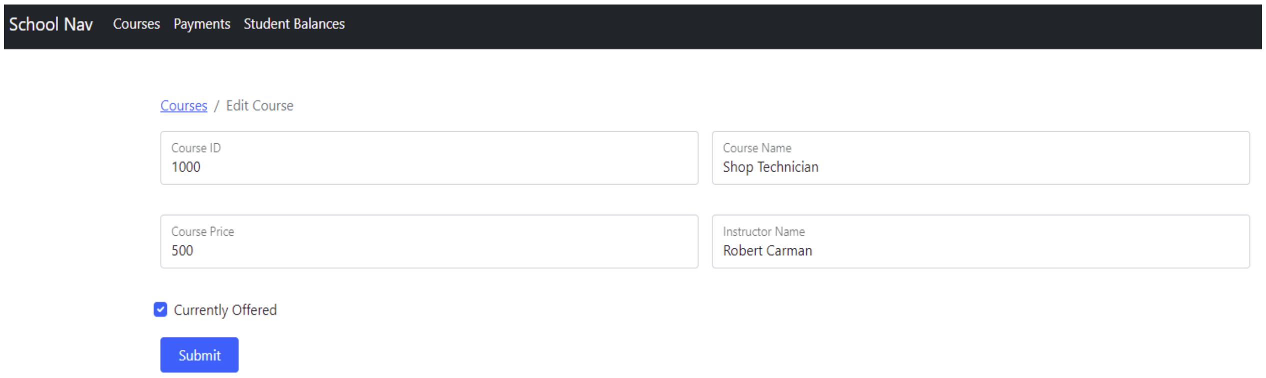
Courses - Create:

When clicking on Add Course you will see the below page.



Courses - Update:

Clicking on the edit button on the Create step shows the below page:



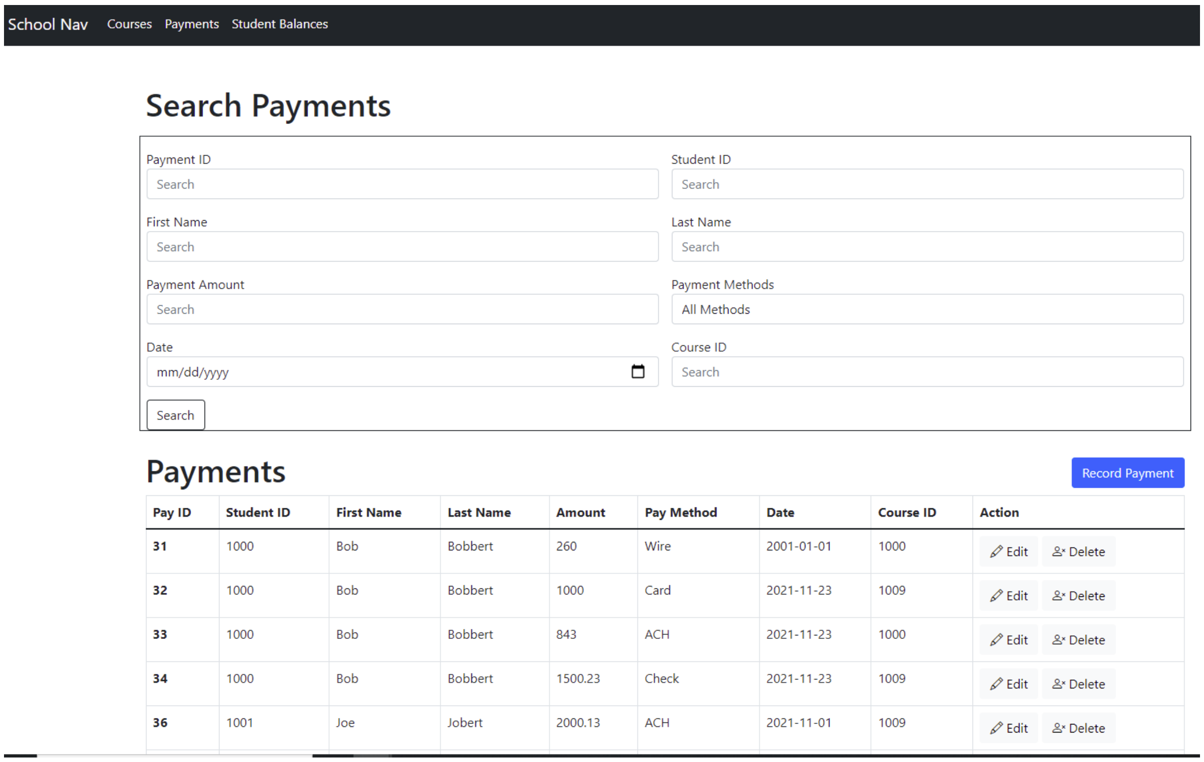
Courses / Edit Course

Course ID: 1000	Course Name: Shop Technician
Course Price: 500	Instructor Name: Robert Carman

☑ Currently Offered

Payments - Read & Delete:

A search field is provided at the top to change the output displayed below, and a delete button is provided on each row.



Search Payments

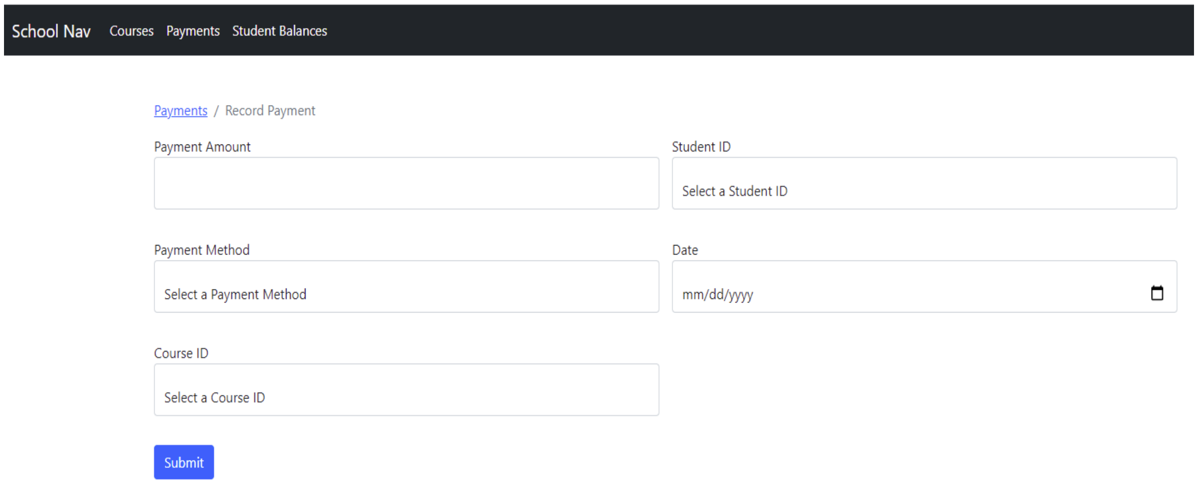
Payment ID: Search	Student ID: Search
First Name: Search	Last Name: Search
Payment Amount: Search	Payment Methods: All Methods
Date: mm/dd/yyyy	Course ID: Search

Payments

Pay ID	Student ID	First Name	Last Name	Amount	Pay Method	Date	Course ID	Action
31	1000	Bob	Bobbert	260	Wire	2001-01-01	1000	Edit Delete
32	1000	Bob	Bobbert	1000	Card	2021-11-23	1009	Edit Delete
33	1000	Bob	Bobbert	843	ACH	2021-11-23	1000	Edit Delete
34	1000	Bob	Bobbert	1500.23	Check	2021-11-23	1009	Edit Delete
36	1001	Joe	Jobert	2000.13	ACH	2021-11-01	1009	Edit Delete

Payments - Create:

Clicking on the Record Payment button above will lead to the form below.


School Nav Courses Payments Student Balances

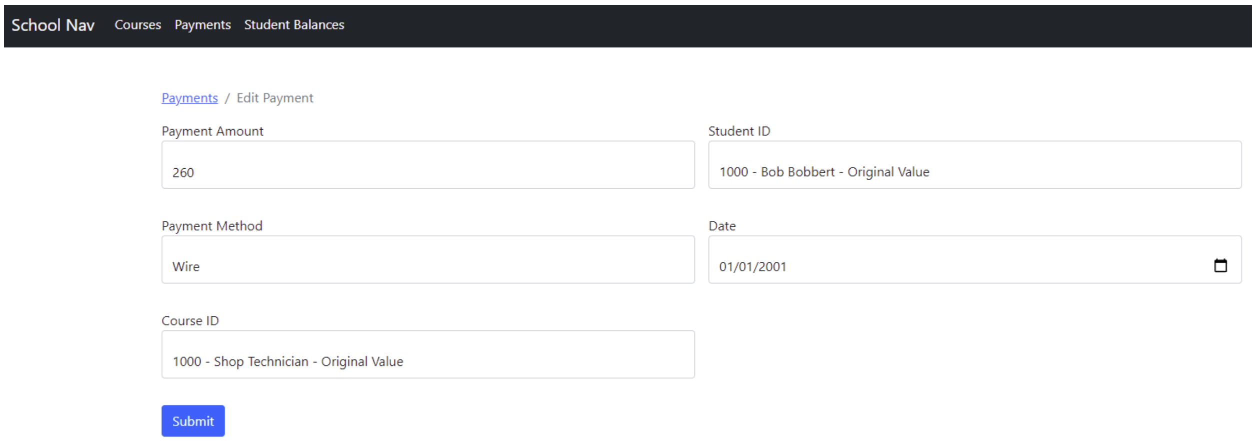
Payments / Record Payment

Payment Amount	Student ID
	Select a Student ID
Payment Method	Date
Select a Payment Method	mm/dd/yyyy
Course ID	
Select a Course ID	

Submit

Payments - Update:

Clicking on the Edit button on each row will lead to the Edit Payment page below.


School Nav Courses Payments Student Balances

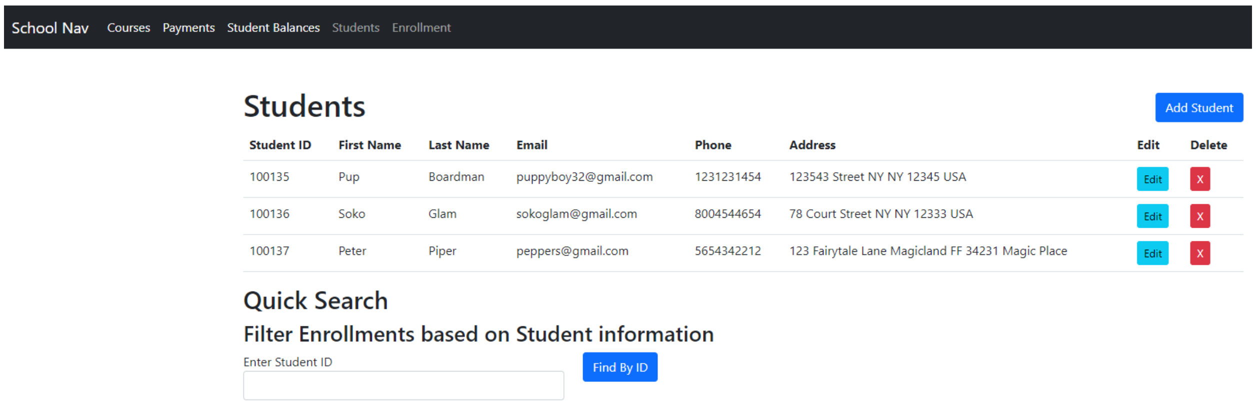
Payments / Edit Payment

Payment Amount	Student ID
260	1000 - Bob Bobbert - Original Value
Payment Method	Date
Wire	01/01/2001
Course ID	
1000 - Shop Technician - Original Value	

Submit

Students - Read & Delete:

The Table information is displayed above and then there is a search field at the bottom. Much like other pages there is a delete button that will remove a row of students. The Addresses table is also queried and displayed.



Students

Student ID	First Name	Last Name	Email	Phone	Address	Edit	Delete
100135	Pup	Boardman	puppyboy32@gmail.com	1231231454	123543 Street NY NY 12345 USA	Edit	X
100136	Soko	Glam	sokoglam@gmail.com	8004544654	78 Court Street NY NY 12333 USA	Edit	X
100137	Peter	Piper	peppers@gmail.com	5654342212	123 Fairytale Lane Magicland FF 34231 Magic Place	Edit	X

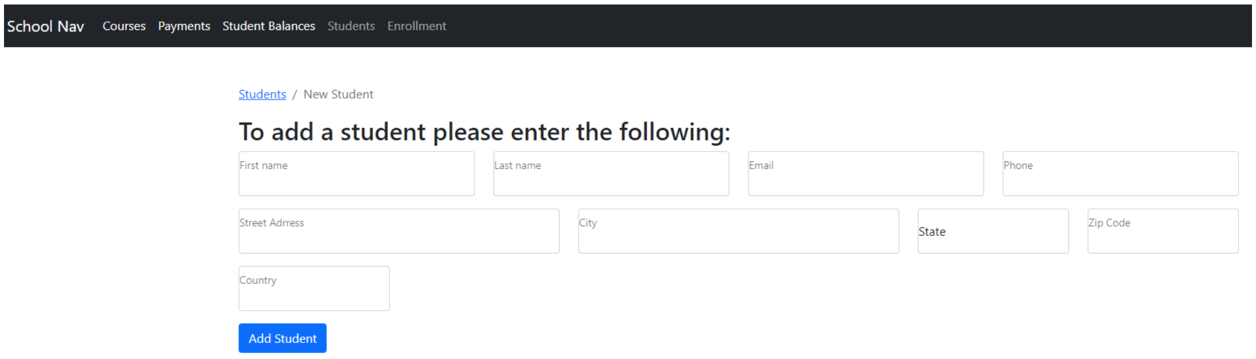
Quick Search

Filter Enrollments based on Student information

Enter Student ID

Students - Create

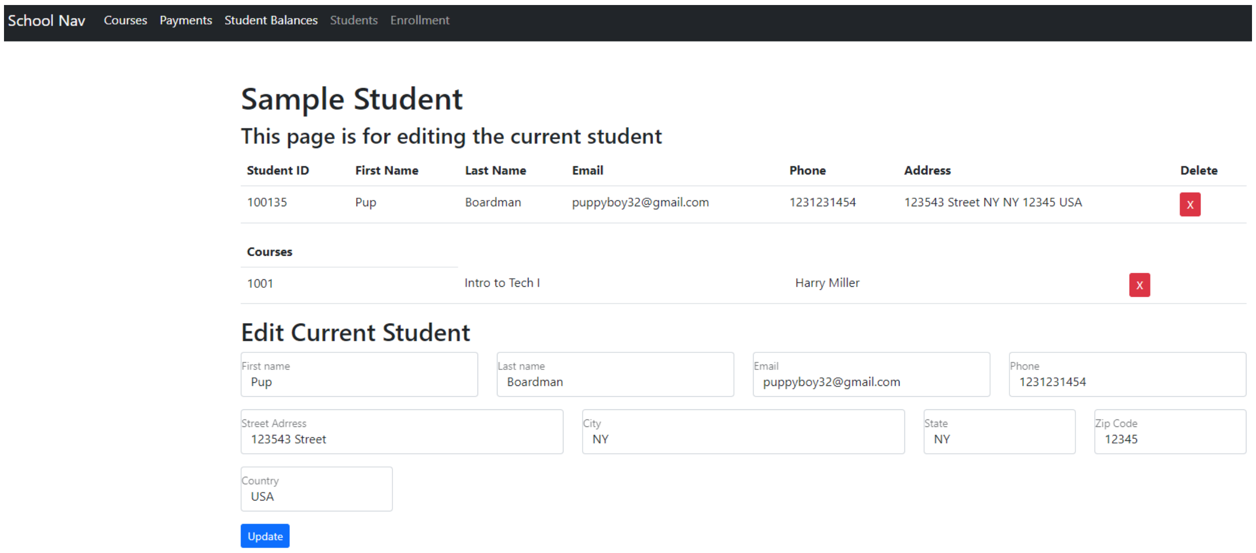
New student information is entered below to be entered into the Students table.



To add a student please enter the following:

Students - Update

Student information can be updated by changing the information in the form.



Sample Student

This page is for editing the current student

Student ID	First Name	Last Name	Email	Phone	Address	Delete
100135	Pup	Boardman	puppyboy32@gmail.com	1231231454	123543 Street NY NY 12345 USA	x

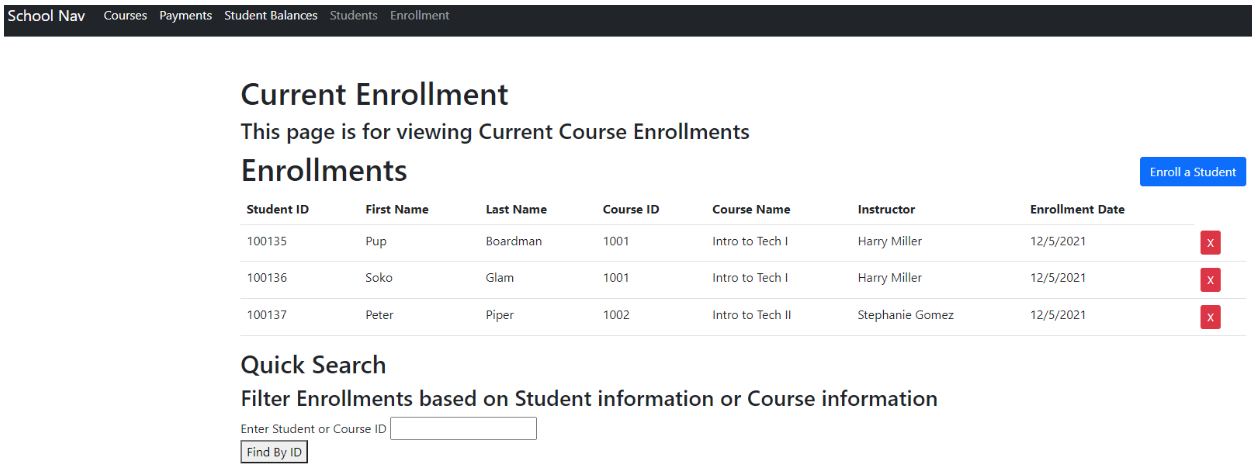
Courses			
1001	Intro to Tech I	Harry Miller	x

Edit Current Student

First name: Pup	Last name: Boardman	Email: puppyboy32@gmail.com	Phone: 1231231454
Street Adrress: 123543 Street	City: NY	State: NY	Zip Code: 12345
Country: USA			

CurEnrolls - Read & Delete

Current enrollment is displayed above with a search feature below and delete buttons on each row, to remove the relationship.



Current Enrollment

This page is for viewing Current Course Enrollments

Enrollments

Student ID	First Name	Last Name	Course ID	Course Name	Instructor	Enrollment Date	
100135	Pup	Boardman	1001	Intro to Tech I	Harry Miller	12/5/2021	x
100136	Soko	Glam	1001	Intro to Tech I	Harry Miller	12/5/2021	x
100137	Peter	Piper	1002	Intro to Tech II	Stephanie Gomez	12/5/2021	x

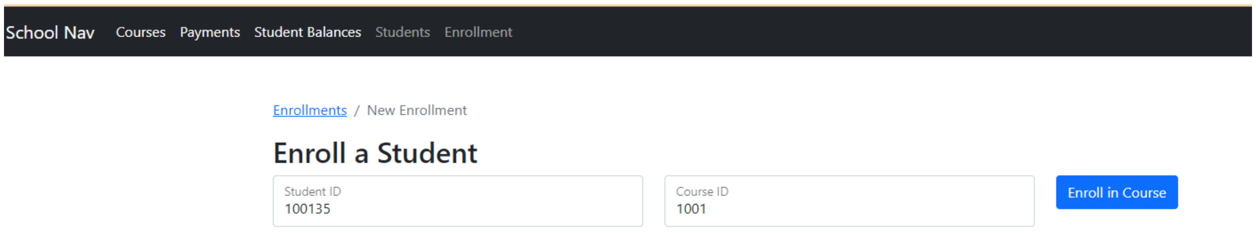
Quick Search

Filter Enrollments based on Student information or Course information

Enter Student or Course ID

CurEnrolls - Create

Current enrollment creating a relationship between students and courses



Enroll a Student

Student ID
100135

Course ID
1001

Sources Cited

Array.prototype.foreach() - javascript: MDN. JavaScript | MDN. (n.d.). Retrieved November 27, 2021, from https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach

chovy. (1960, July 1). *Express JS - EJS with layout template*. Stack Overflow. Retrieved November 30, 2021, from https://stackoverflow.com/questions/10942997/express-js-ejs-with-layout-template

Eernisse, M. (n.d.). EJS. Retrieved November 15, 2021, from https://ejs.co/ . Code type: EJS for templating engine.

Javascript.Info. (2021, October 24). *Promises chaining*. The Modern JavaScript Tutorial. Retrieved November 20, 2021, from https://javascript.info/promise-chaining

Kyeck, P. (1959, May 1). *Node.js - ejs - including a partial*. Stack Overflow. Retrieved November 30, 2021, from https://stackoverflow.com/questions/5404830/node-js-ejs-including-a-partial

RaddyTheBrand (2021, Feb 12). *User Management System*. YouTube, Video Medium, Code type: html, javascript, express-handlebars & node.js, Retrieved November 8, 2021 from https://www.youtube.com/watch?v=1aXZQcG2Y6I&list=PLYqkl7FT2ig8QES9nkovc04GrnRJUhEb&index=1&t=2836s&ab_channel=RaddyTheBrand

Noach, S. (1960, August 1). *MySQL join on vs using?* Stack Overflow. Retrieved November 27, 2021, from https://stackoverflow.com/questions/11366006/mysql-join-on-vs-using

ORACLE. (n.d.). *MySQL 8.0 Reference Manual :: 13.1.20 create table statement*. MySQL. Retrieved November 6, 2021, from https://dev.mysql.com/doc/refman/8.0/en/create-table.html

Thornton, J., & Otto, M. (n.d.). *Form controls, Icons, Navbar* . Bootstrap v5.1. Retrieved November 8, 2021, from https://getbootstrap.com/docs/5.1/forms/form-control/