132+ Effective Minor Project Ideas for CSE

Leave a Comment / General

Find easy minor project ideas for CSE students. Start coding, build simple tools, and improve your skills with these beginner-friendly computer science projects.

Looking for simple computer science project ideas? Minor projects are a great way to practice coding and learn new skills. These projects don’t have to be big or hard—just simple enough to help you apply what you’ve learned.

You could make a basic app, try out simple algorithms, or build small tools. Here are some easy project ideas to help you get started and build your coding confidence. Let’s begin!

Table of Contents

What is a mini project in CSE?

A mini project in CSE is a small project where students use coding to solve a simple problem or build a basic tool. It's usually easier and quicker than a big project, making it a good way to practice what you've learned.

Mini projects can include things like building an app, using algorithms, or creating simple software. They help students get better at coding and understand how things work in real life. These projects are a good way to try out new programming skills in a short time.

Why Choose a Minor Project for CSE?

A minor project in CSE is a great way to practice and improve your skills. Here's why:

1. **Practice Coding**: You get to work on real coding problems.
2. **Improve Skills**: It helps you get better at programming.
3. **Build Confidence**: Finishing a project makes you feel more confident.
4. **Try New Tools**: You can explore new technologies.
5. **Prepare for Jobs**: It shows you have practical experience for future jobs.

Choosing the Right Minor Project for CSE

Choosing the right minor project in CSE is important for your learning. Here are some tips:

1. **Know Your Skills**: Pick a project that matches your current coding level.
2. **Pick What You Like**: Choose a project that interests you so you stay motivated.
3. **Keep It Simple**: Make sure the project can be finished in the time you have.
4. **Learn Something New**: Try a project that teaches you a new skill or tool.

5. **Think About Use**: Pick a project that can be useful in real life or for future jobs.

Minor Project Ideas for CSE

Here are some of the best minor project ideas for CSE

Web Development Projects

1. Build a personal portfolio website.
2. Create an online quiz app.
3. Design an e-commerce website.
4. Develop a to-do list app.
5. Build a blog website.
6. Design a social media platform.
7. Create a recipe-sharing website.
8. Build an event management website.
9. Design a job portal website.
10. Develop a news portal website

Python Projects

1. Build a library management system.
2. Create a weather app using API.
3. Develop a student information system.
4. Build an expense tracker app.
5. Design a chatbot for customer service.
6. Build a Tic-Tac-Toe game.
7. Develop a task scheduler app.
8. Create a password generator.
9. Design a number guessing game.
10. Build a unit conversion app.

Database Projects

1. Create a hotel management system.
2. Develop an inventory management system.
3. Build a student registration system.
4. Design an employee attendance system.

5. Create an online voting system.
6. Build a library catalog system.
7. Design an employee management system.
8. Develop a movie ticket booking system.
9. Build a school management system.
10. Create a feedback collection system.

Mobile App Development Projects

1. Build a fitness tracker app.
2. Create a recipe finder app.
3. Develop a note-taking app.
4. Design a flashlight app.
5. Build an expense manager app.
6. Create a meditation timer app.
7. Build a weather update app.
8. Design a language learning app.
9. Build a grocery shopping app.
10. Develop a quiz app for mobile.

Machine Learning Projects

1. Build a spam email classifier.
2. Create a handwriting recognition system.
3. Design a face recognition system.
4. Build a recommendation system.
5. Develop a sentiment analysis tool.
6. Create a stock price prediction model.
7. Build an object detection system.
8. Design a voice recognition system.
9. Develop a loan approval prediction system.
10. Create a chatbot using NLP.

Networking Projects

1. Develop a real-time chat application.
2. Build an IP address tracker.
3. Create a file-sharing system over a network.

4. Design a website blocker.
5. Build a network speed monitor.
6. Create a DNS lookup tool.
7. Develop a packet sniffer.
8. Build an intrusion detection system.
9. Design a proxy server.
10. Create a bandwidth manager.

Data Structures and Algorithms Projects

1. Build a sorting algorithms visualizer.
2. Create a maze solver using algorithms.
3. Implement graph traversal (BFS/DFS).
4. Design an AI for Tic-Tac-Toe.
5. Build a calendar application.
6. Create a palindrome checker.
7. Design a prime number finder.
8. Develop a Fibonacci sequence generator.
9. Build a parking management system.
10. Create linked list operations (insertion, deletion).

Security Projects

1. Build a password manager.
2. Create a file encryption system.
3. Design a login authentication system.
4. Implement two-factor authentication.
5. Develop a firewall system.
6. Create a data loss prevention system.
7. Build a secure chat application.
8. Design a digital signature system.
9. Build a secure file sharing system.
10. Create an intrusion detection system.

See also 277+ Creative NLP Project Ideas

Game Development Projects

1. Build a Snake game.
2. Design a Tetris game.
3. Develop a Pong game.
4. Create a Rock, Paper, Scissors game.
5. Build a memory matching game.
6. Design a Flappy Bird clone.
7. Create a Space Invaders game.
8. Develop a 2D platformer game.
9. Build a Sudoku game.
10. Design a Chess game.

IoT Projects

1. Build a smart home system for automation.
2. Create a weather monitoring system.
3. Design a smart garden watering system.
4. Develop a health monitoring system.
5. Build a smart parking system.
6. Create a smart door lock system.
7. Build a smart lighting system.
8. Design a temperature-controlled fan.
9. Develop a home security system.
10. Create a smart waste management system.

Cloud Computing Projects

1. Build a cloud storage system.
2. Create an online backup system.
3. Develop a virtual machine management system.
4. Build a distributed file system.
5. Create a cloud database system.
6. Build a cloud cost estimator tool.
7. Develop cloud synchronization for files.
8. Create a serverless application.
9. Design a cloud load balancer.
10. Build a cloud security system.

Blockchain Projects

1. Develop a blockchain voting system.
2. Build a cryptocurrency wallet.
3. Create a smart contract-based application.
4. Design a supply chain management system.
5. Build a decentralized marketplace.
6. Create a blockchain file sharing system.
7. Design a blockchain-based identity verification system.
8. Build a blockchain payment system.
9. Create custom blockchain tokens.
10. Design a blockchain insurance system.

Robotics Projects

1. Build a line-following robot.
2. Create an obstacle-avoiding robot.
3. Design a voice-controlled robot.
4. Build a remote-controlled robot.
5. Develop a gesture-controlled robot.
6. Create a robotic arm.
7. Design an automated plant watering robot.
8. Build an AI-powered robot.
9. Create a delivery robot.
10. Design a Bluetooth-controlled robot.

Tips for Successful Project Execution

Here are some easy tips for a successful project:

1. **Make a Plan**: Break the project into small tasks and set deadlines.
2. **Stay Organized**: Keep your files and code neat and easy to find.
3. **Start Early**: Begin working on your project ahead of time.
4. **Test Often**: Check your work regularly to catch mistakes early.
5. **Ask for Help**: Get help from teachers or friends if you’re stuck.
6. **Stay Focused**: Work on your project regularly without distractions.
7. **Take Notes**: Write down how your project works so it’s easy to explain later.

Minor Project ideas for cse with source code

1. To-Do List App

Basic HTML + JavaScript for To-Do List:

html

Copy code

```
<!DOCTYPE html>
<html lang=”en”>
<head>
    <meta charset=”UTF-8″>
    <title>To-Do List</title>
</head>
<body>
    <h2>To-Do List</h2>
    <input type=”text” id=”taskInput” placeholder=”Add a task”>
    <button onclick=”addTask()”>Add Task</button>
    <ul id=”taskList”></ul>
    <script>
        function addTask() {
            const taskInput = document.getElementById(“taskInput”);
            const taskList = document.getElementById(“taskList”);
            if (taskInput.value !== “”) {
```

```
        const li = document.createElement("li");
        li.textContent = taskInput.value;
        const deleteButton = document.createElement("button");
        deleteButton.textContent = "Delete";
        deleteButton.onclick = () => li.remove();
        li.appendChild(deleteButton);
        taskList.appendChild(li);
        taskInput.value = "";
      }
    }
  </script>
</body>
</html>
```

This code allows users to add tasks to a list and delete them when done.

2. Quiz Application

Basic HTML + JavaScript for Quiz App:

html

Copy code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
```

```
    <title>Quiz App</title>

</head>

<body>

    <h2>Quiz</h2>

    <p id=”question”></p>

    <button onclick=”answer(true)”>True</button>

    <button onclick=”answer(false)”>False</button>

    <p id=”result”></p>

    <script>

        const questions = [

            { question: “The sky is blue.”, correct: true },

            { question: “Cats can fly.”, correct: false }

        ];

        let currentQuestion = 0;

        let score = 0;

        function loadQuestion() {

            if (currentQuestion < questions.length) {

                document.getElementById(“question”).textContent = questions[currentQuestion].question;

            } else {

                document.getElementById(“result”).textContent = `Your score: ${score}`;

            }

        }
    function answer(isTrue) {
      if (questions[currentQuestion].correct === isTrue) {
        score++;
      }
      currentQuestion++;
      loadQuestion();
    }
    loadQuestion();
  </script>
</body>
</html>
```

This code displays a simple true/false question and tracks the score.

3. Currency Converter

HTML + JavaScript for Currency Conversion (using static rates for simplicity):

html

Copy code

```
<!DOCTYPE html>
<html lang=”en”>
<head>
  <meta charset=”UTF-8″>
  <title>Currency Converter</title>
</head>
```

```
<body>
  <h2>Currency Converter</h2>
  <input type=”number” id=”amount” placeholder=”Amount in USD”>
  <button onclick=”convert()”>Convert to EUR</button>
  <p id=”result”></p>
  <script>
    const conversionRate = 0.85; // Example: 1 USD = 0.85 EUR
    function convert() {
      const amount = parseFloat(document.getElementById(“amount”).value);
      if (!isNaN(amount)) {
        const converted = amount * conversionRate;
        document.getElementById(“result”).textContent = `${amount} USD = ${converted.toFixed(2)} EUR`;
      } else {
        document.getElementById(“result”).textContent = “Please enter a valid number”;
      }
    }
  </script>
</body>
</html>
```

This basic converter uses a static rate for simplicity. You could replace the rate with an API call to get real-time rates.

4. Simple Calculator

HTML + JavaScript for a Basic Calculator:

html

Copy code

```
<!DOCTYPE html>
<html lang=”en”>
<head>
    <meta charset=”UTF-8″>
    <title>Calculator</title>
</head>
<body>
    <h2>Calculator</h2>
    <input id=”num1″ type=”number” placeholder=”First Number”>
    <input id=”num2″ type=”number” placeholder=”Second Number”>
    <button onclick=”calculate(‘+’)”>+</button>
    <button onclick=”calculate(‘-‘)”>-</button>
    <button onclick=”calculate(‘*’)”>*</button>
    <button onclick=”calculate(‘/’)”>/</button>
    <p id=”result”></p>
    <script>
        function calculate(op) {
            const num1 = parseFloat(document.getElementById(“num1”).value);
            const num2 = parseFloat(document.getElementById(“num2”).value);
            let result;
```

```
      if (!isNaN(num1) && !isNaN(num2)) {
        switch(op) {
          case ‘+’: result = num1 + num2; break;
          case ‘-‘: result = num1 – num2; break;
          case ‘*’: result = num1 * num2; break;
          case ‘/’: result = num1 / num2; break;
        }
        document.getElementById(“result”).textContent = `Result: ${result}`;
      } else {
        document.getElementById(“result”).textContent = “Please enter valid numbers”;
      }
    }
  </script>
</body>
</html>
```

This simple calculator can perform addition, subtraction, multiplication, and division on two numbers.

5. Weather App (Simple Display with Static Data)

HTML + JavaScript for a Basic Weather Display (no real API):

html

Copy code

```
<!DOCTYPE html>
<html lang=”en”>
```

```
<head>
  <meta charset=”UTF-8″>
  <title>Weather App</title>
</head>
<body>
  <h2>Weather App</h2>
  <button onclick=”getWeather()”>Get Weather</button>
```

See also 331+ Top-Listed & Best Design Engineering Project Ideas

```
  <p id=”weather”></p>
  <script>
    function getWeather() {
      // Simulating a weather result
      const weatherData = {
        city: “New York”,
        temperature: 22,
        description: “Clear sky”
      };
      document.getElementById(“weather”).textContent =
        `${weatherData.city}: ${weatherData.temperature}°C, ${weatherData.description}`;
    }
  </script>
</body>
```

```
</html>
```

To make this dynamic, you'd need to use an API like OpenWeatherMap to fetch real-time weather data.

These snippets are just starting points to help you get a basic setup running. You can add more functionality, such as form validation, improved design, or error handling, to make these projects more complete.

Mini project **topics** for CSE **3rd year**

1. Online Quiz System

- **Description**: A website where users take quizzes and see their scores.
- **Tech Stack**: HTML, CSS, JavaScript, PHP, MySQL.
- **Skills Learned**: Basic web development, databases.

2. Customer Support Chatbot

- **Description**: A chatbot that answers basic questions.
- **Tech Stack**: Python, Flask.
- **Skills Learned**: Chatbot design, Python.

3. Face Recognition Attendance

- **Description**: A system to mark attendance using face recognition.
- **Tech Stack**: Python (OpenCV), Flask, SQLite.
- **Skills Learned**: Computer vision, attendance system.

4. E-commerce Website

- **Description**: A basic online shopping website.
- **Tech Stack**: HTML, CSS, JavaScript, Node.js, MongoDB.
- **Skills Learned**: Full-stack development.

5. Sentiment Analysis

- **Description**: Analyzes if social media posts are positive or negative.

- **Tech Stack**: Python, TextBlob.
- **Skills Learned**: Data analysis, sentiment detection.

6. Encrypted File Storage

- **Description**: A system to store and encrypt files for security.
- **Tech Stack**: Python, Flask.
- **Skills Learned**: File encryption, security basics.

7. Library System with RFID

- **Description**: A system that tracks books with RFID tags.
- **Tech Stack**: C/C++, Python.
- **Skills Learned**: RFID, basic hardware.

8. Real-Time Bus Tracker

- **Description**: A system to track buses in real-time.
- **Tech Stack**: JavaScript, Google Maps API.
- **Skills Learned**: GPS, real-time tracking.

9. Finance Tracker

- **Description**: An app to track personal spending.
- **Tech Stack**: React Native or Flutter.
- **Skills Learned**: Mobile app development.

10. Quiz Game

- **Description**: A quiz game with a score tracker.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: Basic game design.

11. IoT Home Automation

- **Description**: Control lights and appliances from your phone.
- **Tech Stack**: Arduino, MQTT.
- **Skills Learned**: IoT basics.

12. Virtual Campus Tour

- **Description**: A 3D virtual tour of a campus.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: 3D design, user interaction.

13. Image Filter App

- **Description**: An app that applies filters to photos.
- **Tech Stack**: Python (OpenCV) or JavaScript.
- **Skills Learned**: Image processing.

14. Stock Price Graph

- **Description**: Shows stock prices on a graph.
- **Tech Stack**: JavaScript, Python.
- **Skills Learned**: Data visualization.

15. Health Tracker

- **Description**: Track health metrics like heart rate.
- **Tech Stack**: Arduino, Python.
- **Skills Learned**: IoT, sensor integration.

Mini project **topics** for CSE **2nd year**

1. To-Do List App

- **Description**: An app to add, update, and delete tasks.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: Web design, JavaScript basics.

2. Student Record System

- **Description**: A system to store student details like name and roll number.
- **Tech Stack**: Python, SQLite.
- **Skills Learned**: Database handling, Python.

3. Weather App

- **Description**: Displays current weather for any location.
- **Tech Stack**: HTML, CSS, JavaScript, Weather API.
- **Skills Learned**: API integration, web development.

4. Simple Calculator

- **Description**: A calculator that does basic math operations.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: JavaScript functions.

5. Portfolio Website

- **Description**: A personal website to show your skills and projects.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: Web design.

6. Library Management System

- **Description**: Manages books in a library.
- **Tech Stack**: Java, MySQL.
- **Skills Learned**: Java basics, databases.

7. Digital Clock

- **Description**: Displays current time on a web page.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: JavaScript, time functions.

8. Chat App

- **Description**: A simple app where users can send messages.
- **Tech Stack**: Python (Socket programming).
- **Skills Learned**: Networking, Python.

9. Expense Tracker

- **Description**: Tracks your daily expenses.
- **Tech Stack**: HTML, CSS, JavaScript, Local Storage.
- **Skills Learned**: JavaScript, data handling.

10. Unit Converter

- **Description**: Converts units like length or weight.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: JavaScript basics.

11. Survey Form

- **Description**: A form to collect survey responses.
- **Tech Stack**: HTML, CSS.
- **Skills Learned**: Form design.

12. Password Generator

- **Description**: Generates random passwords.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: JavaScript, security basics.

13. Banking System

- **Description**: Simulates basic banking functions.
- **Tech Stack**: Python, SQLite.
- **Skills Learned**: Python, database handling.

14. Typing Speed Test

- **Description**: Tests how fast you can type.
- **Tech Stack**: HTML, CSS, JavaScript.
- **Skills Learned**: JavaScript, timing functions.

15. Number Guessing Game

- **Description**: A game where you guess a random number.
- **Tech Stack**: Python.

- **Skills Learned**: Python basics, random numbers.

Minor project ideas for cse for **final year**

1. Smart Attendance System

- **Description**: Marks attendance using face recognition.
- **Tech Stack**: Python, OpenCV.
- **Skills Learned**: Computer vision, Python basics.

2. Online Voting System

- **Description**: A simple web app for voting.
- **Tech Stack**: HTML, CSS, PHP, MySQL.
- **Skills Learned**: Web development, database basics.

See also 139+ reMarkable Mini Project Ideas for ECE Students

3. Real-Time Chat Application

- **Description**: Allows users to chat instantly.
- **Tech Stack**: Node.js, Socket.io.
- **Skills Learned**: Real-time communication, JavaScript.

4. E-commerce Website

- **Description**: An online store with products and a cart.
- **Tech Stack**: HTML, CSS, JavaScript, PHP.
- **Skills Learned**: Web design, shopping cart.

5. Sentiment Analysis Tool

- **Description**: Analyzes text to find positive or negative sentiment.
- **Tech Stack**: Python, TextBlob.
- **Skills Learned**: Natural language processing, Python.

6. IoT-Based Home Automation

- **Description**: Control lights and appliances remotely.
- **Tech Stack**: Arduino, IoT.
- **Skills Learned**: IoT basics, sensors.

7. College Event Management System

- **Description**: Manages college events and registrations.
- **Tech Stack**: PHP, MySQL.
- **Skills Learned**: Web development, database handling.

8. Online Exam Portal

- **Description**: A platform to take and manage exams online.
- **Tech Stack**: HTML, CSS, JavaScript, Django.
- **Skills Learned**: Web development, exam management.

9. Disease Prediction System

- **Description**: Predicts diseases based on symptoms.
- **Tech Stack**: Python, Machine Learning.
- **Skills Learned**: Machine learning, data handling.

10. Smart Parking System

- **Description**: Shows available parking spots in real-time.
- **Tech Stack**: IoT, Arduino.
- **Skills Learned**: IoT, real-time data.

11. Blog Website with Admin Panel

- **Description**: A simple blog where admins can post content.
- **Tech Stack**: HTML, CSS, JavaScript, Django.
- **Skills Learned**: Web development, content management.

12. Handwritten Digit Recognition

- **Description**: Recognizes digits from a picture.
- **Tech Stack**: Python, TensorFlow.

- **Skills Learned**: Image processing, machine learning.

13. AI Chatbot for Customer Support

- **Description**: A chatbot to answer common questions.
- **Tech Stack**: Python, NLTK.
- **Skills Learned**: AI, chatbots.

14. News Aggregator

- **Description**: Collects and displays news from various sources.
- **Tech Stack**: Python, Flask.
- **Skills Learned**: Web scraping, data aggregation.

15. Voice-Based Email System

- **Description**: Send emails using voice commands.
- **Tech Stack**: Python, Speech Recognition.
- **Skills Learned**: Voice recognition, email automation.

Mini project **topics** for CSE **3rd year with source code**

1. Library Management System

- **Description**: A system to manage book information, issue dates, and returns.
- **Tech Stack**: Python, MySQL
- **Skills Learned**: Database handling, Python programming

Source Code:
python
Copy code

```
import mysql.connector

conn = mysql.connector.connect(host=”localhost”, user=”root”, password=”password”, database=”library”)

cursor = conn.cursor()

def add_book():
    title = input(“Enter book title: “)
    author = input(“Enter author name: “)
    cursor.execute(f”INSERT INTO books (title, author) VALUES (‘{title}’, ‘{author}’)”)
    conn.commit()
    print(“Book added successfully!”)
# Add more functions for issuing books, returning books, etc.
```

-

2. Weather Forecasting System

- **Description**: A simple app to fetch and display weather data for a specific location.
- **Tech Stack**: Python, OpenWeather API
- **Skills Learned**: API integration, Python programming

Source Code:
python
Copy code

```
import requests

def get_weather(city):
    api_key = “your_api_key”
    base_url = f”http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}”
    response = requests.get(base_url)
    data = response.json()
    if data[“cod”] == 200:
        main = data[“main”]
        temperature = main[“temp”]
        print(f”Temperature: {temperature – 273.15}°C”)
    else:
        print(“City not found.”)
city = input(“Enter city name: “)
get_weather(city)
```

-

3. Student Information System

- **Description**: A system to manage student data, including adding, updating, and displaying student records.
- **Tech Stack**: Python, SQLite
- **Skills Learned**: Database management, Python programming

Source Code:
python
Copy code

```
import sqlite3
def create_table():
    conn = sqlite3.connect(“student_info.db”)
    cursor = conn.cursor()
    cursor.execute(“CREATE TABLE IF NOT EXISTS students (id INTEGER PRIMARY KEY, name TEXT, age INTEGER)”)
    conn.commit()
def add_student(name, age):
    conn = sqlite3.connect(“student_info.db”)
    cursor = conn.cursor()
    cursor.execute(f”INSERT INTO students (name, age) VALUES (‘{name}’, {age})”)
    conn.commit()
```

```
def show_students():
    conn = sqlite3.connect(“student_info.db”)
    cursor = conn.cursor()
    cursor.execute(“SELECT * FROM students”)
    for row in cursor.fetchall():
        print(row)
# Call functions as needed
create_table()
add_student(“John Doe”, 20)
show_students()
```

-

4. To-Do List Application

- **Description**: A simple to-do list where users can add and remove tasks.
- **Tech Stack**: Python, Tkinter
- **Skills Learned**: GUI development, Python programming

Source Code:
python
Copy code

```
import tkinter as tk

def add_task():
    task = entry.get()
    if task:
        listbox.insert(tk.END, task)
        entry.delete(0, tk.END)
```

```
def delete_task():

    listbox.delete(tk.ANCHOR)

window = tk.Tk()

window.title(“To-Do List”)

entry = tk.Entry(window)

entry.pack()

add_button = tk.Button(window, text=”Add Task”, command=add_task)

add_button.pack()

delete_button = tk.Button(window, text=”Delete Task”, command=delete_task)

delete_button.pack()

listbox = tk.Listbox(window)

listbox.pack()

window.mainloop()
```

-

5. Expense Tracker

- **Description**: A simple application to track expenses and balance.
- **Tech Stack**: Python, SQLite
- **Skills Learned**: Database management, Python programming

Source Code:
python
Copy code

```
import sqlite3

def create_table():

    conn = sqlite3.connect(“expense_tracker.db”)
    cursor = conn.cursor()

    cursor.execute(“CREATE TABLE IF NOT EXISTS expenses (id INTEGER PRIMARY KEY, description TEXT, amount REAL)”)

    conn.commit()

def add_expense(description, amount):

    conn = sqlite3.connect(“expense_tracker.db”)

    cursor = conn.cursor()

    cursor.execute(f”INSERT INTO expenses (description, amount) VALUES (‘{description}’, {amount})”)

    conn.commit()

def show_expenses():

    conn = sqlite3.connect(“expense_tracker.db”)

    cursor = conn.cursor()

    cursor.execute(“SELECT * FROM expenses”)

    for row in cursor.fetchall():

        print(row)

# Call functions as needed

create_table()

add_expense(“Lunch”, 10.50)
```

- show_expenses()

Conclusion

In conclusion, minor projects for CSE students are a great way to practice and apply what you’ve learned. They help you develop important skills like coding, problem-solving, and working with different technologies. Projects like a library management system, weather app, or to-do list are simple but useful ways to start. Completing these projects not only builds your confidence but also helps you understand how to approach real-world problems. Working on a project that interests you can also make learning more fun and give you a solid foundation for future studies and your career.

← Previous Post

Related Posts

129+ Innovative MSC Mathematics Project Ideas for Students

Leave a Comment / General / By Adam Tesla

50 Most Innovative SUPW Project Ideas to Test Your Skills

Leave a Comment / General / By Adam Tesla

Leave a Comment

Your email address will not be published. Required fields are marked *

Type here..

Name*

Email*

Website

□ Save my name, email, and website in this browser for the next time I comment.

Post Comment »

Recent Posts

132+ Effective Minor Project Ideas for CSE

135+ Stunning Page Decoration Ideas for Project

234+ Powerful ATL Project Ideas

151+ Most Recent & Unique IB Personal Project Ideas

Top 45+ Unique MERN Project Ideas

Categories

Computer Science

General

Humanities

Mini

Subscribe to Our Newsletter

Subscribe us for lastest project ideas on all subjects into your email.

Email address

Subscribe

Top Pages

Privacy Policy

Disclaimer

Terms And Conditions

Top Categories

Computer Science

General

Humanities

Mini

Follow us on