Web Programming Assignment Solution

# Introduction

This document provides the full code and structure for the web-based email system assignment using Node.js. The project includes setting up a database, creating routes, and building a simple front-end using EJS.

# Project Directory Structure

wpr<your\_student\_id>/  
│  
├── dbsetup.js  
├── index.js  
├── package.json  
├── .env  
├── views/  
│ ├── layout.ejs  
│ ├── signin.ejs  
│ ├── signup.ejs  
│ ├── inbox.ejs  
│ ├── outbox.ejs  
│ ├── compose.ejs  
│ ├── detail.ejs  
│ └── error.ejs  
├── public/  
│ ├── css/  
│ │ └── styles.css  
│ └── js/  
│ └── scripts.js  
└── uploads/

# dbsetup.js

const mysql = require('mysql2');  
  
const connection = mysql.createConnection({  
 host: 'localhost',  
 user: 'wpr',  
 password: 'fit2024',  
 multipleStatements: true  
});  
  
const dbName = 'wpr<your\_student\_id>';  
  
connection.query(`  
 CREATE DATABASE IF NOT EXISTS ${dbName};  
 USE ${dbName};  
 CREATE TABLE users (  
 id INT AUTO\_INCREMENT PRIMARY KEY,  
 email VARCHAR(255) UNIQUE,  
 password VARCHAR(255),  
 fullname VARCHAR(255)  
 );  
 CREATE TABLE emails (  
 id INT AUTO\_INCREMENT PRIMARY KEY,  
 sender\_id INT,  
 recipient\_id INT,  
 subject VARCHAR(255),  
 body TEXT,  
 attachment VARCHAR(255),  
 timestamp DATETIME  
 );  
 INSERT INTO users (email, password, fullname) VALUES   
 ('a@a.com', '123', 'User A'),  
 ('b@b.com', '123', 'User B'),  
 ('c@c.com', '123', 'User C');  
 INSERT INTO emails (sender\_id, recipient\_id, subject, body, timestamp) VALUES   
 (1, 2, 'Hello', 'This is the first email', NOW()),  
 (2, 1, 'Reply', 'Thanks for your email', NOW()),  
 (1, 3, 'Meeting', 'Let’s schedule a meeting', NOW()),  
 (3, 1, 'Confirmation', 'Meeting confirmed', NOW()),  
 (1, 2, 'Update', 'Here is the update', NOW()),  
 (2, 1, 'Re: Update', 'Got it, thanks', NOW()),  
 (3, 2, 'Reminder', 'Don’t forget our meeting', NOW()),  
 (2, 3, 'Thanks', 'Thank you for your support', NOW());  
`, (err) => {  
 if (err) throw err;  
 console.log('Database setup complete.');  
 connection.end();  
});

# index.js

require('dotenv').config();  
const express = require('express');  
const cookieParser = require('cookie-parser');  
const mysql = require('mysql2/promise');  
const path = require('path');  
const multer = require('multer');  
const app = express();  
  
// Middleware setup  
app.use(express.urlencoded({ extended: true }));  
app.use(express.json());  
app.use(cookieParser());  
app.use(express.static(path.join(\_\_dirname, 'public')));  
app.set('view engine', 'ejs');  
app.set('views', path.join(\_\_dirname, 'views'));  
  
// MySQL Connection Pool  
const pool = mysql.createPool({  
 host: 'localhost',  
 user: 'wpr',  
 password: 'fit2024',  
 database: 'wpr<your\_student\_id>'  
});  
  
// Routes  
app.get('/', async (req, res) => {  
 if (req.cookies.user) {  
 return res.redirect('/inbox');  
 }  
 res.render('signin');  
});  
  
app.post('/signin', async (req, res) => {  
 const { email, password } = req.body;  
 const [rows] = await pool.query('SELECT \* FROM users WHERE email = ? AND password = ?', [email, password]);  
 if (rows.length > 0) {  
 res.cookie('user', rows[0].id, { httpOnly: true });  
 return res.redirect('/inbox');  
 }  
 res.render('signin', { error: 'Invalid credentials' });  
});  
  
app.get('/signup', (req, res) => {  
 res.render('signup');  
});  
  
app.post('/signup', async (req, res) => {  
 const { fullname, email, password, confirmPassword } = req.body;  
 if (password !== confirmPassword) {  
 return res.render('signup', { error: 'Passwords do not match' });  
 }  
  
 try {  
 await pool.query('INSERT INTO users (email, password, fullname) VALUES (?, ?, ?)', [email, password, fullname]);  
 res.redirect('/');  
 } catch (err) {  
 res.render('signup', { error: 'Email already exists' });  
 }  
});  
  
app.get('/inbox', async (req, res) => {  
 if (!req.cookies.user) {  
 return res.status(403).render('error', { message: 'Access Denied' });  
 }  
  
 const userId = req.cookies.user;  
 const [emails] = await pool.query('SELECT \* FROM emails WHERE recipient\_id = ?', [userId]);  
 res.render('inbox', { emails });  
});  
  
app.listen(8000, () => {  
 console.log('Server running on http://localhost:8000');  
});

# .env

DB\_HOST=localhost  
DB\_USER=wpr  
DB\_PASS=fit2024  
DB\_NAME=wpr<your\_student\_id>  
DB\_PORT=3306

# public/css/styles.css

body {  
 font-family: Arial, sans-serif;  
}  
form {  
 margin: 0 auto;  
 padding: 1em;  
 max-width: 300px;  
}