1.

```
<!DOCTYPE html>
<html lang="en">
<head>
	<meta charset="UTF-8">
	<meta name="viewport" content="width=device-width, initial-scale=1.0">
	<title>Admin Dashboard</title>
	<!-- Bootstrap CSS -->
	<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
	<!-- Font Awesome -->
	<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
	<style>
	:root {
	--sidebar-width: 250px;
	--sidebar-bg: #2c3e50;
	--sidebar-active: #34495e;
	--primary-color: #3498db;
	--success-color: #2ecc71;
	--warning-color: #f39c12;
	--danger-color: #e74c3c;
	}

	body {
	font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
	background-color: #f5f5f5;
	overflow-x: hidden;
	}

	/* Sidebar */
	.sidebar {
	width: var(--sidebar-width);
	min-height: 100vh;
	background: var(--sidebar-bg);
	color: white;
	position: fixed;
	transition: all 0.3s ease;
	z-index: 1000;
	}

	.sidebar-header {
	padding: 20px;
```

```css
border-bottom: 1px solid rgba(255,255,255,0.1);
height: 70px;
}

.sidebar-menu {
list-style: none;
padding: 0;
margin-top: 10px;
}

.sidebar-menu li a {
display: flex;
align-items: center;
padding: 12px 20px;
color: #ecf0f1;
text-decoration: none;
transition: all 0.3s ease;
border-left: 3px solid transparent;
}

.sidebar-menu li a:hover,
.sidebar-menu li a.active {
background: var(--sidebar-active);
color: white;
border-left: 3px solid var(--primary-color);
}

.sidebar-menu li a i {
width: 24px;
text-align: center;
margin-right: 12px;
font-size: 1.1rem;
}

/* Main Content */
.main-content {
margin-left: var(--sidebar-width);
padding: 20px;
transition: all 0.3s ease;
min-height: 100vh;
}

/* Top Navigation */
.top-nav {
```

```
background: white;
padding: 15px 20px;
border-radius: 8px;
box-shadow: 0 2px 4px rgba(0,0,0,0.05);
margin-bottom: 20px;
display: flex;
justify-content: space-between;
align-items: center;
position: sticky;
top: 0;
z-index: 100;
}

/* Cards */
.stat-card {
border: none;
border-radius: 10px;
box-shadow: 0 4px 6px rgba(0,0,0,0.05);
transition: all 0.3s ease;
margin-bottom: 20px;
height: 100%;
overflow: hidden;
}

.stat-card:hover {
transform: translateY(-5px);
box-shadow: 0 10px 15px rgba(0,0,0,0.1);
}

.stat-card .card-body {
padding: 20px;
position: relative;
}

.stat-card .icon {
position: absolute;
right: 20px;
top: 20px;
font-size: 2.5rem;
opacity: 0.2;
transition: all 0.3s ease;
}

.stat-card:hover .icon {
```

```
opacity: 0.4;
transform: scale(1.1);
}

.stat-card h6 {
font-size: 0.9rem;
font-weight: 500;
margin-bottom: 10px;
color: rgba(255,255,255,0.9);
}

.stat-card h3 {
font-size: 1.8rem;
font-weight: 700;
margin-bottom: 0;
}

/* Card Colors */
.bg-primary { background: var(--primary-color) !important; }
.bg-success { background: var(--success-color) !important; }
.bg-warning { background: var(--warning-color) !important; }
.bg-danger { background: var(--danger-color) !important; }

/* Tables */
.data-card {
border: none;
border-radius: 10px;
box-shadow: 0 4px 6px rgba(0,0,0,0.05);
margin-bottom: 20px;
}

.data-card .card-header {
background: white;
border-bottom: 1px solid rgba(0,0,0,0.05);
font-weight: 600;
padding: 15px 20px;
border-radius: 10px 10px 0 0 !important;
}

.data-card .table {
margin-bottom: 0;
}

.data-card .table th {
```

```
border-top: none;
font-weight: 500;
color: #6c757d;
}

.data-card .table td {
vertical-align: middle;
}

/* Badges */
.badge {
padding: 6px 10px;
font-weight: 500;
font-size: 0.75rem;
}

/* Responsive Adjustments */
@media (max-width: 992px) {
.stat-card h3 {
        font-size: 1.5rem;
}
}

@media (max-width: 768px) {
.sidebar {
        transform: translateX(-100%);
        width: 280px;
}

.sidebar.active {
        transform: translateX(0);
}

.main-content {
        margin-left: 0;
}

.top-nav {
        padding: 12px 15px;
}
}

@media (max-width: 576px) {
.stat-card .card-body {
			padding: 15px;
		}

		.stat-card h3 {
			font-size: 1.4rem;
		}

		.stat-card .icon {
			font-size: 2rem;
			top: 15px;
			right: 15px;
		}
		}
		</style>
</head>
<body>
	<div class="wrapper d-flex">
	<!-- Sidebar -->
	<nav class="sidebar">
	<div class="sidebar-header d-flex align-items-center">
		<h4 class="mb-0">Admin Panel</h4>
	</div>
	<ul class="sidebar-menu">
		<li><a href="#" class="active"><i class="fas fa-tachometer-alt"></i>
Dashboard</a></li>
		<li><a href="#"><i class="fas fa-shopping-cart"></i> E-commerce</a></li>
		<li><a href="#"><i class="fas fa-graduation-cap"></i> College</a></li>
		<li><a href="#"><i class="fas fa-book"></i> Exams</a></li>
		<li><a href="#"><i class="fas fa-users"></i> Users</a></li>
		<li><a href="#"><i class="fas fa-cog"></i> Settings</a></li>
	</ul>
	</nav>

	<!-- Main Content -->
	<div class="main-content">
	<!-- Top Navigation -->
	<div class="top-nav">
		<button class="btn btn-sm d-lg-none" id="sidebarToggle">
		<i class="fas fa-bars"></i>
		</button>
		<div class="d-flex align-items-center">
		<div class="input-group ms-3" style="max-width: 300px;">
		<input type="text" class="form-control form-control-sm" placeholder="Search...">
		<button class="btn btn-outline-secondary btn-sm" type="button">
```

```
                <i class="fas fa-search"></i>
            </button>
            </div>
            </div>
            <div class="dropdown">
            <a href="#" class="d-flex align-items-center text-decoration-none
dropdown-toggle" data-bs-toggle="dropdown">
            <img src="https://via.placeholder.com/40" class="rounded-circle me-2"
width="32" height="32">
            <span class="d-none d-sm-inline">Admin User</span>
            </a>
            <ul class="dropdown-menu dropdown-menu-end">
            <li><a class="dropdown-item" href="#"><i class="fas fa-user me-2"></i>
Profile</a></li>
            <li><a class="dropdown-item" href="#"><i class="fas fa-cog me-2"></i>
Settings</a></li>
            <li><hr class="dropdown-divider"></li>
            <li><a class="dropdown-item" href="#"><i class="fas fa-sign-out-alt me-2"></i>
Logout</a></li>
            </ul>
            </div>
        </div>

        <!-- Stats Cards -->
        <div class="row g-3 mb-4">
            <div class="col-xl-3 col-lg-6 col-md-6 col-sm-6">
            <div class="stat-card text-white bg-primary">
            <div class="card-body">
                <h6>Total Revenue</h6>
                <h3>$24,500</h3>
                <i class="fas fa-dollar-sign icon"></i>
            </div>
            </div>
            </div>

            <div class="col-xl-3 col-lg-6 col-md-6 col-sm-6">
            <div class="stat-card text-white bg-success">
            <div class="card-body">
                <h6>Total Students</h6>
                <h3>1,450</h3>
                <i class="fas fa-users icon"></i>
            </div>
            </div>
            </div>
```

```
		<div class="col-xl-3 col-lg-6 col-md-6 col-sm-6">
		<div class="stat-card text-white bg-warning">
		<div class="card-body">
			<h6>Pending Orders</h6>
			<h3>18</h3>
			<i class="fas fa-shopping-basket icon"></i>
		</div>
		</div>
		</div>

		<div class="col-xl-3 col-lg-6 col-md-6 col-sm-6">
		<div class="stat-card text-white bg-danger">
		<div class="card-body">
			<h6>Upcoming Exams</h6>
			<h3>12</h3>
			<i class="fas fa-calendar-alt icon"></i>
		</div>
		</div>
		</div>
	</div>

	<!-- Data Tables -->
	<div class="row g-3">
		<div class="col-lg-6">
		<div class="data-card card">
		<div class="card-header d-flex justify-content-between align-items-center">
			<h5 class="mb-0">Recent Orders</h5>
			<a href="#" class="btn btn-sm btn-outline-primary">View All</a>
		</div>
		<div class="card-body p-0">
			<div class="table-responsive">
			<table class="table table-hover mb-0">
			<thead>
				<tr>
				<th>Order ID</th>
				<th>Customer</th>
				<th>Amount</th>
				<th>Status</th>
				</tr>
			</thead>
			<tbody>
				<tr>
				<td>#ORD-1001</td>
```

```
                                        <td>John Smith</td>
                                        <td>$120.00</td>
                                        <td><span class="badge bg-warning
text-dark">Pending</span></td>
                                        </tr>
                                        <tr>
                                        <td>#ORD-1002</td>
                                        <td>Sarah Johnson</td>
                                        <td>$85.50</td>
                                        <td><span class="badge bg-success">Completed</span></td>
                                        </tr>
                                        <tr>
                                        <td>#ORD-1003</td>
                                        <td>Michael Brown</td>
                                        <td>$210.00</td>
                                        <td><span class="badge bg-primary">Shipped</span></td>
                                        </tr>
                              </tbody>
                              </table>
                              </div>
                    </div>
                    </div>
                    </div>

                    <div class="col-lg-6">
                    <div class="data-card card">
                    <div class="card-header d-flex justify-content-between align-items-center">
                              <h5 class="mb-0">Exam Schedule</h5>
                              <a href="#" class="btn btn-sm btn-outline-primary">View All</a>
                    </div>
                    <div class="card-body p-0">
                              <div class="table-responsive">
                              <table class="table table-hover mb-0">
                              <thead>
                                        <tr>
                                        <th>Exam</th>
                                        <th>Date</th>
                                        <th>Course</th>
                                        <th>Status</th>
                                        </tr>
                              </thead>
                              <tbody>
                                        <tr>
                                        <td>Midterm Exam</td>
```

```
                                <td>Oct 15</td>
                                <td>Mathematics</td>
                                <td><span class="badge bg-primary">Scheduled</span></td>
                                </tr>
                                <tr>
                                <td>Final Exam</td>
                                <td>Nov 25</td>
                                <td>Computer Science</td>
                                <td><span class="badge bg-primary">Scheduled</span></td>
                                </tr>
                                <tr>
                                <td>Quiz 1</td>
                                <td>Oct 5</td>
                                <td>Physics</td>
                                <td><span class="badge bg-warning
text-dark">Pending</span></td>
                                </tr>
                        </tbody>
                        </table>
                        </div>
                </div>
                </div>
                </div>
        </div>
        </div>
        </div>

        <!-- Bootstrap JS Bundle with Popper -->
        <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

        <script>
        // Toggle sidebar on mobile
        document.getElementById('sidebarToggle').addEventListener('click', function() {
        document.querySelector('.sidebar').classList.toggle('active');
        });

        // Close sidebar when clicking outside on mobile
        document.addEventListener('click', function(e) {
        const sidebar = document.querySelector('.sidebar');
        const toggleBtn = document.getElementById('sidebarToggle');

        if (window.innerWidth < 992 &&
                !sidebar.contains(e.target) &&
                e.target !== toggleBtn &&
                !toggleBtn.contains(e.target)) {
                sidebar.classList.remove('active');
        }
        });

        // Close sidebar when resizing to larger screens
        window.addEventListener('resize', function() {
        if (window.innerWidth >= 992) {
                document.querySelector('.sidebar').classList.remove('active');
        }
        });
        </script>
</body>
</html>
```

2. Git

```
# 1. Configure Git (only once)
git config --global user.name "Your Name"
git config --global user.email "your-email@example.com"

# 2. Initialize a Git repository in your project folder
cd your-project-folder/
git init

# 3. Add your files to the staging area
git add .

# 4. Commit the files
git commit -m "Initial commit"

# 5. Create a new repository on GitHub (do this from the GitHub website)

# 6. Add the remote origin (replace the URL with your repo's URL)
git remote add origin https://github.com/your-username/your-repo-name.git

# 7. Push the code to GitHub
git push -u origin master  # or use 'main' if your branch is named main
```

2.CC

Here is a detailed step-by-step guide to implement your assignment from the beginning using the Google App Engine (GAE) Launcher, focusing on Python since GAE Launcher primarily supports Python 2.7-based apps (note: GAE Launcher is now deprecated, but for academic/historical/assignment purposes, you can still use it locally).

⚠️ IMPORTANT NOTE (Before Starting)

- **GAE Launcher only works for Python 2.7 (Standard Environment).**
- **It is part of the Google App Engine SDK for Python (Legacy).**
- **This method is not for modern Python 3.x apps.**

✅ OBJECTIVE

Install Google App Engine Launcher and use it to deploy "Hello World" and another simple web app using Python 2.7.

🛠️ STEP 1: Install Required Tools

1.1 Install Python 2.7

- **Download: https://www.python.org/downloads/release/python-2718/**
- **During installation, add Python to PATH.**

1.2 Install Google App Engine SDK for Python

- **Download GAE SDK for Python (includes Launcher):**
  📥 [**https://storage.googleapis.com/appengine-sdks/featured/google_appengine_1.9.40.zip**](https://storage.googleapis.com/appengine-sdks/featured/google_appengine_1.9.40.zip)

1.3 Extract and Setup

- **Extract the zip to a location like `C:\gae-python-sdk`**
- **Inside the folder, find `GoogleAppEngineLauncher.exe` and run it.**

✅ STEP 2: Create a Hello World App using GAE Launcher

2.1 Open GAE Launcher → File → New Application

- **Application Name: hello-app**
- **Application Directory: Choose a folder (e.g., `C:\gae-projects\hello-app`)**
- **Runtime: Python**
- **Click Create**

2.2 In the Project Folder, You'll See:

- **`app.yaml` – configuration file**
- **`main.py` – your Python code**

2.3 Edit `main.py`

```python
import webapp2


class MainPage(webapp2.RequestHandler):
    def get(self):
        self.response.write("Hello, World from Google App Engine Launcher!")


app = webapp2.WSGIApplication([
    ('/', MainPage),
], debug=True)
```

2.4 Edit `app.yaml`

```yaml
application: your-app-id   # (change this after deploying)
version: 1
runtime: python27
api_version: 1
threadsafe: true


handlers:
- url: /.*
  script: main.app
```

✅ STEP 3: Run App Locally Using GAE Launcher

- **In GAE Launcher, select your app**
- **Click Run**
- **Open browser and go to: http://localhost:8080**

🚀 STEP 4: Deploy App to Google App Engine

4.1 Create a Project in Google Cloud Console

- **Go to https://console.cloud.google.com**
- **Create a new project (e.g., `my-gae-legacy-app`)**
- **Enable App Engine and billing**

4.2 Update `app.yaml`

- **Replace the `application:` value with your actual project ID**

4.3 Upload App

In GAE Launcher:

- **Select your app**
- **Click Deploy**

- **Sign in when prompted**
- **Follow the steps and wait for upload**

4.4 View Live App

- **Visit:** `https://<your-project-id>.appspot.com`

💡 STEP 5: Create Another Simple App

Example: Greet User by Name

`main.py`

```
import webapp2


class MainPage(webapp2.RequestHandler):
    def get(self):
        name = self.request.get("name")
        if name:
            self.response.write("Hello, " + name + "!")
        else:
            self.response.write("<form>Enter your name: <input
name='name'><input type='submit'></form>")


app = webapp2.WSGIApplication([
```

('/', MainPage),

], debug=True)

- **Save the file**
- **Run again from GAE Launcher**
- **Test on http://localhost:8080**
- **Deploy it similarly**

Sure! Here's a **simple Node.js + Express + MongoDB (Mongoose)** backend with **4 API endpoints (Create, Read, Update, Delete)** that supports the Angular assignment **2.C**: *Register User, Login User, Show User Data on Profile Component*.

📦 Technologies Used

- Node.js
- Express.js
- MongoDB (Mongoose)
- bcrypt (for password hashing)
- JSON Web Token (JWT) for login sessions (optional, basic version)

✅ Step 1: Install Dependencies

```
npm init -y
npm install express mongoose body-parser cors bcryptjs
```

✅ Step 2: File Structure

```
assignment-2c-api/
├── models/
│   └── User.js
├── routes/
│   └── userRoutes.js
├── server.js
```

✅ 3. `models/User.js`

```
const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({
  name: String,
  email: { type: String, unique: true },
  password: String
});

module.exports = mongoose.model('User', userSchema);
```

✅ 4. `routes/userRoutes.js`

```
const express = require('express');
const router = express.Router();
const User = require('../models/User');
const bcrypt = require('bcryptjs');

// REGISTER (CREATE)
router.post('/register', async (req, res) => {
  try {
    const { name, email, password } = req.body;
    const hashedPassword = await bcrypt.hash(password, 10);
    const newUser = new User({ name, email, password: hashedPassword });
    await newUser.save();
    res.status(201).json({ message: 'User registered' });
  } catch (err) {
    res.status(400).json({ error: 'Email already exists or invalid input' });
  }
});
```

```
// LOGIN (READ)
router.post('/login', async (req, res) => {
  const { email, password } = req.body;
  try {
    const user = await User.findOne({ email });
    if (!user) return res.status(404).json({ error: 'User not found' });

    const isMatch = await bcrypt.compare(password, user.password);
    if (!isMatch) return res.status(401).json({ error: 'Invalid password' });

    res.json({ message: 'Login successful', user });
  } catch (err) {
    res.status(500).json({ error: 'Something went wrong' });
  }
});

// GET PROFILE (SHOW USER DATA)
router.get('/profile/:email', async (req, res) => {
  try {
    const user = await User.findOne({ email: req.params.email }, '-password');
    if (!user) return res.status(404).json({ error: 'User not found' });
    res.json(user);
  } catch (err) {
    res.status(500).json({ error: 'Error fetching user data' });
  }
});

// UPDATE USER
router.put('/update/:email', async (req, res) => {
  try {
    const updated = await User.findOneAndUpdate({ email: req.params.email }, req.body, {
new: true });
    res.json({ message: 'User updated', user: updated });
  } catch (err) {
    res.status(400).json({ error: 'Update failed' });
  }
});

// DELETE USER
router.delete('/delete/:email', async (req, res) => {
  try {
    await User.findOneAndDelete({ email: req.params.email });
    res.json({ message: 'User deleted' });
  } catch (err) {
    res.status(500).json({ error: 'Delete failed' });
  }
});

module.exports = router;
```

✅ 5. `server.js`

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');
const userRoutes = require('./routes/userRoutes');

const app = express();
app.use(cors());
app.use(bodyParser.json());

app.use('/api', userRoutes);

mongoose.connect('mongodb://localhost:27017/angular-auth', {
  useNewUrlParser: true,
  useUnifiedTopology: true
}).then(() => {
  console.log('MongoDB connected');
  app.listen(3000, () => console.log('Server running on http://localhost:3000'));
}).catch(err => console.error('MongoDB error:', err));
```

✅ API Summary

Method	Endpoint	Purpose
POST	`/api/register`	Register user
POST	`/api/login`	Login user
GET	`/api/profile/:email`	Get user profile

PUT	`/api/update/:email`	Update user
DELETE	`/api/delete/:email`	Delete user

Would you like a basic **Angular frontend** example that connects to these APIs?

Great! Here's a simple **Angular frontend** with **3 components**:

- `RegisterComponent`
- `LoginComponent`
- `ProfileComponent`

that connects to the Node.js API we created earlier.

✅ Step 1: Create Angular Project

```
ng new user-auth-app
cd user-auth-app
ng generate component register
ng generate component login
ng generate component profile
```

✅ Step 2: Add HTTP Client

In `app.module.ts`:

```
import { HttpClientModule } from '@angular/common/http';
@NgModule({
  imports: [
    HttpClientModule,
    // ...
  ]
})
```

✅ Step 3: Create a Simple User Service

`src/app/user.service.ts`

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';

@Injectable({ providedIn: 'root' })
export class UserService {
  private api = 'http://localhost:3000/api';

  constructor(private http: HttpClient) {}

  register(data: any) {
    return this.http.post(`${this.api}/register`, data);
  }

  login(data: any) {
    return this.http.post(`${this.api}/login`, data);
  }

  getProfile(email: string) {
    return this.http.get(`${this.api}/profile/${email}`);
  }
}
```

✅ Step 4: Register Component

`src/app/register/register.component.ts`

```
import { Component } from '@angular/core';
import { UserService } from '../user.service';

@Component({
  selector: 'app-register',
  template: `
    <h2>Register</h2>
    <form (ngSubmit)="registerUser()">
      <input [(ngModel)]="user.name" name="name" placeholder="Name" required>
      <input [(ngModel)]="user.email" name="email" placeholder="Email" required>
      <input [(ngModel)]="user.password" name="password" type="password"
placeholder="Password" required>
      <button type="submit">Register</button>
    </form>
    <p>{{ message }}</p>
  `
})
export class RegisterComponent {
  user = { name: '', email: '', password: '' };
  message = '';

  constructor(private userService: UserService) {}

  registerUser() {
    this.userService.register(this.user).subscribe({
      next: () => this.message = 'User registered!',
      error: err => this.message = err.error.error
    });
  }
}
```

✅ Step 5: Login Component

`src/app/login/login.component.ts`

```
import { Component } from '@angular/core';
import { Router } from '@angular/router';
import { UserService } from '../user.service';

@Component({
  selector: 'app-login',
  template: `
    <h2>Login</h2>
    <form (ngSubmit)="loginUser()">
      <input [(ngModel)]="credentials.email" name="email" placeholder="Email" required>
      <input [(ngModel)]="credentials.password" name="password" type="password"
placeholder="Password" required>
      <button type="submit">Login</button>
    </form>
    <p>{{ message }}</p>
  `
})
export class LoginComponent {
  credentials = { email: '', password: '' };
  message = '';

  constructor(private userService: UserService, private router: Router) {}

  loginUser() {
    this.userService.login(this.credentials).subscribe({
      next: (res: any) => {
        localStorage.setItem('email', res.user.email);
        this.router.navigate(['/profile']);
      },
      error: err => this.message = err.error.error
    });
  }
}
```

✅ Step 6: Profile Component

`src/app/profile/profile.component.ts`

```
import { Component, OnInit } from '@angular/core';
import { UserService } from '../user.service';

@Component({
  selector: 'app-profile',
  template: `
    <h2>User Profile</h2>
    <div *ngIf="user">
      <p><strong>Name:</strong> {{ user.name }}</p>
      <p><strong>Email:</strong> {{ user.email }}</p>
    </div>
  `
})
export class ProfileComponent implements OnInit {
  user: any;

  constructor(private userService: UserService) {}
  ngOnInit() {
    const email = localStorage.getItem('email');
    if (email) {
      this.userService.getProfile(email).subscribe(data => this.user = data);
    }
  }
}
```

✅ Step 7: Set Up Routing

`src/app/app-routing.module.ts`

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { RegisterComponent } from './register/register.component';
import { LoginComponent } from './login/login.component';
import { ProfileComponent } from './profile/profile.component';

const routes: Routes = [
  { path: 'register', component: RegisterComponent },
  { path: 'login', component: LoginComponent },
  { path: 'profile', component: ProfileComponent },
  { path: '', redirectTo: 'register', pathMatch: 'full' }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule {}
```

✅ Start Your Project

Run backend:

```
node server.js
```

-

Run Angular app:

```
ng serve
```

-

Would you like Bootstrap added to the Angular UI for better styling?