Centre of Professional Development and Community Outreach
ICT Upskilling Program – Assignment Brief Description

TECHNICAL TRACK NAME	FULL-STACK WEB DEVELOPMENT
Technical Instructor Name	Hussam ElBawwab
Soft Skills Instructor Name	Sawsan Alsayeh
English Language for Business	Basheer Mufleh

OVERVIEW

This assignment provides the requirement for the capstone project for trainees enrolled in the DigiSkills ICT Upskilling Programme. The assignment provides trainees with the opportunity to present evidence of their learning in the technical, soft skills, and English language components.

The trainees' work will be assessed by the instructors and industry technical experts according to the criteria specified herein. ***The trainees who don't meet the pass criteria will not be qualified to receive the completion certificate.***

SUBMISSION FORMAT

Report (produced using MS-Word) that includes relevant diagrams, plots, justification, and discuss results for the decisions you made. The following format shall be considered when submitting the draft report or the final version:

1. The font size should be 12 point and should be in the style of Times New Roman.
2. Set line spacing to 1.5 and justify all paragraphs.
3. The report should be properly cited (using IEEE citation format1). All sources used in the production of your report should be cited in text and listed at the end of report.
4. Ensure that all headings are consistent in terms of size and font style.
5. Use footer function in MS-Word to insert your name, course, project name, and page number on each page.
6. Use the spell check and grammar check function of MS-Word to review the use of language on your assignment.
7. The caption for a figure appears below the graphic, for a table, above.
8. Minimum Number of pages 7 up to 30.
9. The caption should be brief and explanatory. All Figures and Tables should be numbered. Capitalize only the first letter of the first word of all headings. Notes are placed below the table.

DELIVERABLES

You will work remotely on this project and use your own computer to provide the following deliverables containing the expected results. You are advised to take notes during the class to help you with the assignment.

Component	Deliverable	Deadline
Technical	Github link	20 / June / 2025
English	Technical Report P(7-35) Export the Test cases, reports and attach as appendixes	20 / June / 2025
Soft Skills	Presentation	23 / June / 2025

LEARNING OUTCOMES

Technical Skills

- Front-End Foundations: Mastery of HTML, CSS, JavaScript, Bootstrap, and React.js for building dynamic and responsive web interfaces.
- Back-End Foundations: Proficiency in Node.js and Express.js for server-side programming and API development.
- Database Foundations: Expertise in SQL and PostgreSQL for data management and integration with applications.
- Full-Stack Integration: Ability to connect front-end and back-end systems, ensuring seamless data flow and functionality.
- Deployment & Version Control: Skills to deploy applications to cloud platforms and manage version control with Git/GitHub.

Language Skills

- Participants show structure and organization
- Participants show coherence and cohesion
- Participants show strong content
- Participants show grammatical accuracy
- Participants use well prepared visual aids written in
- Participants speak clearly with proper pronunciation
- Participants use range of vocabulary

Presentation Skills

- Speak clearly and at an understandable pace.
- Maintain goof eye contact with the audience.
- Be well-rehearsed, and well prepared
- Demonstrate a limited use of filler words (“umm”, “Like”, etc.)
- Use body language appropriately.
- Present within time limits.
- Be able to answer questions professionally.
- Be dressed appropriately

ASSIGNMENT BRIED ND GUIDANCE

Scenario #1 Online Learning Management System (LMS)
This project is an Online Learning Management System (LMS) that allows students, instructors, and administrators to interact in an e-learning environment. The system includes:

User authentication (OAuth 2.0 via Google)
Role-based access control (Student, Instructor, Admin)
Course management (CRUD operations)
Enrollment system
Progress tracking
Interactive quizzes & assignments
PostgreSQL database
RESTful API with Express.js
React.js frontend with responsive design

Assignment Details
1. Project Scenarios & Use Cases
Scenario 1: Student Registration & Course Enrollment
A student signs up using OAuth 2.0 (Google).
Browses available courses, enrolls in one, and tracks progress.
Completes quizzes and submits assignments.

Scenario 2: Instructor Dashboard
An instructor logs in and creates a new course.
Uploads videos, adds quizzes, and grades assignments.
Views student performance analytics.

Scenario 3: Admin Management
An admin manages users (students/instructors).
Approves/rejects courses.
Generates reports on system usage.

2. Database Design (PostgreSQL)
Entity-Relationship Diagram (ERD)
Users (id, name, email, password_hash, role, oauth_provider, oauth_id)
Courses (id, title, description, instructor_id, category, created_at, updated_at)
Enrollments (id, user_id, course_id, enrolled_at, completed_at, progress)
Modules (id, course_id, title, description, order)
Lessons (id, module_id, title, content_type (video/quiz/text), content_url, duration)
Quizzes (id, lesson_id, question, options, correct_answer)

Assignments (id, lesson_id, title, description, deadline)
Submissions (id, assignment_id, user_id, submission_url, submitted_at, grade)
Notifications (id, user_id, message, is_read, created_at)
Categories (id, name)

Relationships:
Users → Enrollments (One-to-Many)
Courses → Enrollments (One-to-Many)
Courses → Modules (One-to-Many)
Modules → Lessons (One-to-Many)
Lessons → Quizzes (One-to-One)
Lessons → Assignments (One-to-One)
Assignments → Submissions (One-to-Many)
Users → Notifications (One-to-Many)

FINAL REPORT

Student Name: [Your Name]
Course: Full Stack Web Development
Submission Date: [Date]

1. Introduction
Project Goals

2. Project Plan & Methodology
Planning & Design
Backend Development
Frontend Development
Testing & Debugging
Deployment

TECHNICAL COMPONENT PASSING REQUIREMENTS

To pass the project, the following minimum functional and technical deliverables must be completed:
1. Frontend (React.js + HTML/CSS/JS)
User Interface (UI):
At least 3 responsive pages (e.g., Home, Courses, Dashboard) using React components.
Basic styling with CSS (e.g., Flexbox/Grid for layout).
Navigation:
Functional routing (React Router) between pages.

Dynamic Content:

Fetch and display course data from the backend API (e.g., list of courses).

2. Backend (Node.js + Express.js)
RESTful API:
Minimum 5 API endpoints (e.g., GET /courses, POST /enroll, GET /users/:id).
Use of HTTP methods (GET, POST, PUT/PATCH, DELETE).

Authentication:
OAuth 2.0 (Google) login implemented.
Role-based access (e.g., /admin routes protected).

Error Handling:
Basic status codes (e.g., 404 for not found, 500 for server errors).

3. Database (PostgreSQL)
Database Schema:
At least 3 core tables (e.g., Users, Courses, Enrollments).
Proper primary/foreign keys (e.g., user_id in Enrollments links to Users).

Queries:
Basic CRUD operations (e.g., INSERT INTO Users, SELECT * FROM Courses).

4. Integration (Frontend + Backend + Database)
API Consumption:
Frontend fetches data from backend APIs (e.g., Axios/fetch to display courses).

Data Flow:
User registration → Saved to PostgreSQL → Reflects in React UI.

5. Core Features (Minimum Viable Product)
Student Use Case:
Sign up/login via Google OAuth.
Enroll in a course (POST request to /enroll).

Instructor Use Case:
Create a course (POST request to /courses).

Admin Use Case:
View list of users (GET /users).

6. Code Quality & Submission
Project Structure:
Separate folders for frontend (/client) and backend (/server).

Submission:
Code pushed to GitHub (with README explaining setup).
No critical errors (application runs without crashing).

ASSIGNEMENT LEARING AND ASSESSMENT CRITEIRA

Assignment Learning Outcomes and Assessment Criteria (Technical)		
Pass %50	**Merit %25**	**Distinction %25**
LO1: Frontend		
Basic React components with state management Simple UI using HTML/CSS (may use Bootstrap) Minimal responsive design	Advanced React features (hooks, context API) Responsive design with media queries Form validation and error handling	Advanced state management (Redux), optimizations, reusable component
LO2: Backend		
Functional Express.js server with basic routing Simple REST API endpoints (CRUD operations) Basic error handling	Middleware implementation Advanced routing with parameters API versioning	caching, rate limiting Transactions, analytics
LO3: Database		
PostgreSQL tables matching provided ERD Basic SQL queries (SELECT, INSERT, UPDATE)	Optimized queries with JOINs Transactions for data integrity Basic indexing	Advanced query optimization
LO4: Authentication		
Google OAuth login implemented Role-based access control (basic implementation)	Session management JWT implementation Password hashing	Multi-factor authentication OAuth token refresh implementation
LO5: Deployment & Operations		
Basic Git usage with commit history	Application deployed to a platform like Firebase	Application deployed to a platform like Firebase
LO6: Code Quality		
Working application with	Modular code structure	Comprehensive test coverage

some bugs Minimal documentation	API documentation (Swagger/Postman) Meaningful commit messages	