Final Lab Assignment Instructions v3

For Dynamic Web Apps

Changes from v2 highlighted in green.

Overview

The largest element of your coursework assessment for this module will be for a project that you will need to complete by the end of term. This will make up 20% of your final marks for this module and as a guideline you should spend around 20 hours on it.

The aim of the project is to consolidate your learnings from the labs into a single piece of work. You will need to demonstrate knowledge of all the skills you have learnt in the labs.

For the project, you must build an app on the theme of **health and fitness**. You can build any app and features you want, but here are some ideas:

- A fitness tracking app that allows users to enter their fitness achievements and report on them
- A clinic appointment booking app, that allows patients to book appointments and clinic staff to manage the appointments

Technology Stack

The application must be built using:

- Node.js
- Express
- EJS
- MySql

In addition:

- The application must be deployed to the Goldsmiths virtual server environment or another public-facing hosting platform. You will need to submit your application URL.
- The code must be shared through Github. You will need to submit a link to your code repo and add me (lfern002) as a developer

Do not use any other frameworks, such as React or Angular.

Your application should look good, but make sure you spend most of your time on the back end, which is where the majority of marks will be awarded.

Features

The following features are compulsory:

- There must be a **home** page, with links to other pages
- There should be an **about** page
- There should be some **search** functionality with searches against the database
- The application must store some data in a **database** (MySQL)
- There should be a **form** that allows users to enter data and store it in the database
- If your app requires a login, it should work with username:**gold** and password:**smiths** or '**smiths123ABC$**'
- The app should be installable on the marker's computer:
    - Running **npm install** should install all required modules
    - There should be a database script called **create_db.sql** that creates the data model from scratch in the database
    - There should be a database script called **insert_test_data.sql** that inserts required data in the data model, including the default login credentials if required
- Once installed, the app should be runnable on the marker's computer:
    - Running **node index.js** should run the app and it should listen on **port 8000**
- You should provide **documentation**, as a file called **report.docx** or **report.pdf** in the root of your Github repo
- You should provide a file called **links.txt** in the root of your Github repo, with links to all the pages of your deployed app that you want us to look at. One of these should be the **home** link. Each link should be given a unique name in the file, e.g:
  home=https://www.doc.gold.ac.uk/usr/123/
  about=https://www.doc.gold.ac.uk/usr/123/about
  search=home=https://www.doc.gold.ac.uk/usr/123/find-patient
  add_patient=https://www.doc.gold.ac.uk/usr/123/add-patient

You can add whatever other features you want. Marks will be awarded for demonstrating a range of techniques from those shown on the module. Marks will not be awarded for repeating the same techniques in this assignment (e.g. 5 similar forms).

Documentation

Your documentation must include:

- An **outline** describing the application you have built (max 200 words)
- A high-level **architecture** including a diagram and description (max 100 words) describing what technologies and components you have used in your application tier and data tier
- A **data model** including a diagram and description (max 100 words)
- A description of the user-facing **functionality** of your application, adding screenshots to help explain (max 500 words)
- An optional description of any **advanced techniques** that you want us to consider as a demonstration of your development skills. You must include code snippets to illustrate how you have used these techniques and refer to the files that contain that code. (max 500 words)
- An **AI declaration**, stating how you have used AI in this assignment

Your documentation must follow the structure described above, with headings exactly as follows:

- Outline
- Architecture
- Data Model
- User Functionality
- Advanced Techniques
- AI Declaration

Final Submission Details

Your application should be running on the Goldsmiths virtual server and also be installable from your Github repository. Ensure you follow these instructions *exactly* so that we can mark your application.

Github

- Your submission should consist of a single github repo named **10_health_12345678** where **12345678** is your 8-digit student id. Your repo will be cloned immediately after the deadline.
- You can create a public or private repo.
- Add me (**lfern002**) as a developer to your repo.

Users

- Register a user called '**gold**' with password '**smiths**' or '**smiths123ABC$**'. We will use the user when we mark your deployed application. One way to set up this user is to add it to the `insert_test_data.sql` script, so it is inserted when you deploy your application, but of course you wouldn't normally add login credentials to a source controlled script like this.

- Ensure any password validation for new users **is no more restrictive** than requiring 8 characters, at least one lowercase, at least one uppercase, at least one number and at least one special character. E.g it should allow the password **aaAA123!**,.

Database Configuration

- Your database should be called **health**

- If you are using dotenv, use the following environment variables:

  HEALTH_HOST='localhost'
  HEALTH_USER='health_app'
  HEALTH_PASSWORD='qwertyuiop'
  HEALTH_DATABASE='health'
  HEALTH_BASE_PATH='http://localhost:8000'

- When we install your application, we will create a database user matching the above credentials.
- The HEALTH_BASE_PATH can be used if you want to use absolute paths for URL redirection, but you can of course use relative paths instead.

Assessment Criteria

In this coursework you will apply a range of techniques, which can be classified as follows:

- **Compulsory functionality** – detailed in the features section of this document
- **Basic techniques** – Techniques taught in guided steps in the lab
- **Additional techniques** – Techniques taught in the course beyond the guided steps
- **Advanced techniques** – Techniques beyond those covered in the course material

Submissions will be graded according to the following criteria:

Grade range	Criteria
0-39%	Application misses some compulsory functionality or fails to implement a significant proportion of basic techniques covered in the labs
40-49%	Application implements all compulsory functionality and basic techniques
50-59%	Application implements all compulsory functionality and basic techniques as well as some additional techniques
60-69%	Application implements all compulsory functionality and basic techniques a good range of additional techniques
70%-100%	Application implements all compulsory functionality and basic techniques, a good range of additional techniques as well as some advanced techniques

Marks will be awarded out of 20 (this assignment is worth 20% of your overall grade).

Penalties will apply for late submissions unless a valid EC is obtained.

END