Durham University

Computer Science

Summative Assignment

Module code and title	COMP1101 Programming (Black)
Academic year	2024-25
Coursework title	Web Development
Coursework credits	10 credits
% of module's final mark	50%
Lecturer	Steven Bradley
Submission date*	Thursday, 30 January 2025 at 14:00
Estimated hours of work	20
Submission method	Ultra/PeerScholar

Additional coursework files	*Please list all files provided for the coursework, e.g. datasets, template files*
Required submission items and formats	Source code (all zipped) - HTML and CSS and any media - Client and server side JavaScript - package.json including test and pretest scripts - .eslintrc - jest test cases e.g. app.test.js - documentation of API demonstration video Should not include node_modules in submission

* This is the deadline for all submissions except where an approved extension is in place. For benchtests taking place in practical sessions, the given date is the Monday of the week in which the benchtests will take place.

Late submissions received within 5 working days of the deadline will be capped at 40%.
Late submissions received later than 5 days after the deadline will receive a mark of 0.
It is your responsibility to check that your submission has uploaded successfully and obtain a submission receipt.

Your work must be done by yourself (or your group, if there is an assigned groupwork component) and comply with the university rules about plagiarism and collusion. Students suspected of plagiarism, either of published or unpublished sources, including the work of other students, or of collusion will be dealt with according to University guidelines (https://www.dur.ac.uk/learningandteaching.handbook/6/2/4/).

COMP1101 Programming (Black) Summative Assessment 1

Term 1 Programming Exercise Outline

- Submission of code and video by 14:00 30 January 2025
- Submission of peer reviews by 14:00 20 February 2025
- Return by 27 February 2025
- Contributes 50% of module marks
- Includes peer review feedback which you will be allocated
- This is an individual piece of work

Subject-specific Knowledge

- Interaction between JavaScript programs and the Document Object Model (DOM)
- Using control statements to loop and make decisions
- An understanding of the nature of imperative programming in the object-oriented style
- A knowledge and understanding of good programming practice (for example, reuse, documentation and style)
- Building collections of data within a program and using JavaScript Object Notation (JSON)
- Making programs robust through the use of exceptions and exception handling
- A knowledge and understanding of good programming practice (for example, reuse, documentation and style)

Subject-Specific Skills

- an ability to realise solutions to problems as working JavaScript programs
- an ability to apply reuse by exploiting predefined components
- an ability to use software tools related to programming (programming environments, code management, documentation tools, etc.)

Key Skills

- an ability to communicate technical information
- an ability to recognise and apply the principles of abstraction and modelling

Task summary

- Construct a dynamic web site for an application of your choosing

- Use static HTML pages loading dynamic JSON content from server via AJAX
- Server written in nodejs to provide JSON through REST API
- Prepare a 2 minute video demonstrating your code
- Do a code quality review of four other submissions

Dynamic web site

- Choose any application domain as long as it includes at least two kinds of entity e.g.
  - pictures
  - people
  - places
  - events
  - comments
- Could be e.g. club, social, health, gallery
- If you are not sure then ask me

Static HTML loading JSON via AJAX

- 'Single page app': page content loaded as JSON via AJAX
- Can have more than one page e.g. for user and admin
- Should provide clean and simple User Experience (UX)
- Should be responsive i.e. work well on desktop and mobile
- Recommend using front-end framework such as Bootstrap, Foundation
- Do not use non-standard language extensions e.g. React, TypeScript

Message sequence chart

Server provides JSON through a REST API

Each entity type (e.g. picture) has

- GET method to list/search (returns a list of ids and names)
- GET method for individual details (includes details of related entities)
- POST method to add new entity
- Document your API in the style of the ChatGPT API
- see other good API docs
- Response provided as JSON
- Content-type needs to be correct
- HTTP codes should be correct: use 200, 400 or 403 (if using authentication)

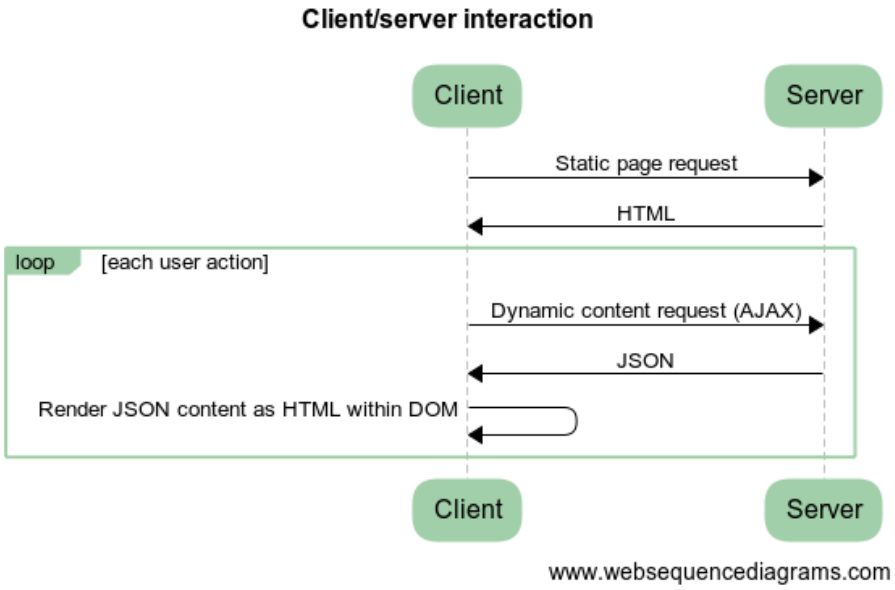

Figure 1: Message Sequence Chart showing Client server interaction with AJAX

Server written in nodejs

- Use npm for management
- Make sure you use –save or –save-dev option with packages you add
- Write jest test cases: run with `npm test`
- Use eslint: run with `npm run pretest`
- Recommend using express

Submission

Source code (all zipped)

- HTML and CSS and any media
- Client and server side JavaScript
- package.json including test and pretest scripts
- .eslintrc
- jest test cases e.g. app.test.js
- documentation of API
- demonstration video

Should not include `node_modules` in submission

Assessment Criteria

Equally weighted 9% each

- Client-side functionality
- Client-side quality
- Server-side functionality
- Server-side quality
- Video presentation

Client-side functionality criteria

- User Experience (UX): clean layout and minimal clicks/entry required
- App complexity: entities can be listed and edited
- 'Single page' style: asynchronous updates
- Staff reviewed

Client-side quality criteria

- Standards compliant (HTML5)
- Responsive to different viewport sizes
- Gracefully handles server disconnection
  - useful error messages
  - recommences on server restart
- Peer reviewed; staff moderated

Server-side functionality criteria

- More than one entity type, with relationships
- REST API provides each entity with appropriate GET/POST methods
- Installs with `npm install`
- Starts with `npm start`
- Staff reviewed

Server-side quality criteria

- Successful eslint (run with `npm run pretest`)
- Successful jest tests with good coverage (run with `npm test`)
- Testing includes content-type and HTTP code
- Completeness of API documentation
- Peer reviewed; staff moderated

Video Presentation

- Submit a 2 minute (max) video demonstrating your software
- Include demonstration of how to start the program

- All functionality will be assessed by what is demonstrated in the video
- If it is not demonstrated in the video, you will not get a mark for it
- Quality of video presentation will be marked separately from functionality:
  - Structure; Visual Presentation; Audio explanation
- Lose 10% of marks for every 10 seconds over 2 minutes
- Staff reviewed

Peer Review Marking

5% of the module marks are awarded for peer reviews

- Completion of all reviews on time
- Professional and helpful reviews
- The average student tends to get about 65%
- 65% is on the good/very good boundary of the marking conventions p15

How to do the assignment

- Design HTML
- Design web service
- Join with Fetch
- Read the FAQ