

COMP1101 Programming (Black) Summative Assessment 1

Term 1 Programming Exercise Outline

- Submission of code and video by 14:00 26 January 2023
 - Submission of peer reviews by 14:00 9 February 2023
 - Return by 23 February 2023
 - Contributes 50% of module marks
 - Includes peer review feedback which you will be allocated
-

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
-

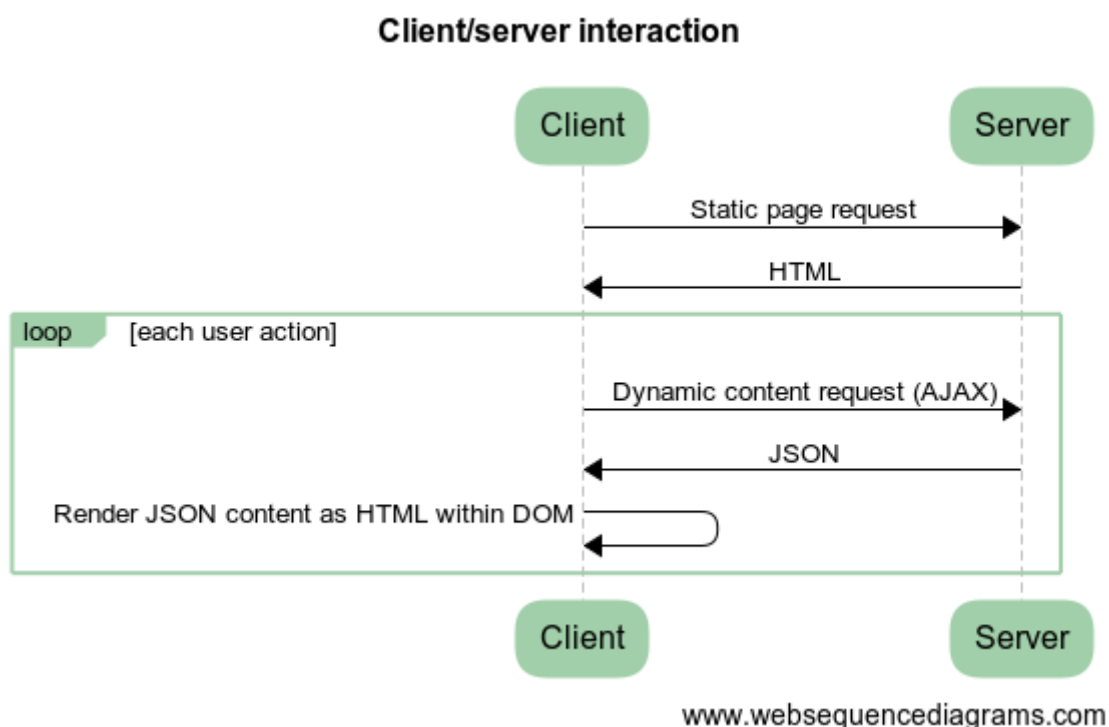
Dynamic web site

- 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 framework such as Bootstrap, semantic-ui

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 [Twitter API](#)
-

- Response provided as JSON
 - Content-type needs to be correct
 - HTTP codes should be correct: use 200, 400 or 403 (if using authentication)
-

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 assessment

- 2 marks for completion of four peer reviews
- 3 marks for the accuracy of the reviews
- Accuracy is based on agreement with peers

- Accuracy marks deducted for being over-harsh and over-generous
 - 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](#)