The final report content should minimally contain the following:

 Title Page The title page should contain the following information:

Dublin Business School

Higher Diploma in Science in Computing

 The course stream

 The title of project

 “Final Report”

 The name of the author

 The author’s e-‐mail address

 The author’s student id

 The name of the supervisor

 The submission date

**Abstract** This is, at maximum, a quarter page summary of the project.

**Acknowledgments** A brief paragraph or two acknowledging professional advice and help in submitting the report.

**Contents** List of contents, including tables and figures, with page numbers.

**Chapter 1: Introduction**

Further refinement and additions to the introduction in the interim report.

**Chapter 2: Background/Literature Review**

Further refinement and additions to the background section in the interim report.

**Chapter 3: Specification and Design**

Design of the artefact should be discussed in this chapter which is augmented by diagrams.

**Chapter 4: Implementation**

The student should discuss the implementation details here. It includes software and version of software used. Any algorithm developed during this process may be discussed. Any coding snippets can be discussed. *Please note, full code should be included in appendix not in this*

*chapter.*

**Chapter 5: Testing and Results**

This chapter contains a clear description of testing and the results of the project. The chapter also contains a description of the results of the final tests carried out on the product.

An important section in this chapter is the *critical evaluation* of the final project, where the student demonstrates the ability to critically evaluate the work done, the shortcomings in the project and so on. Objectivity is important when writing this chapter.

**Chapter 6: Conclusions and Future Work**

A review of what the project achieved, a final review of the project in terms of the proposed goals and project plan.

● Any changes from the interim report should be discussed and justified.

● The student should reflect on the learning experiences gained in doing the project and its relevance to on-going progress as a learner and future practising IT professional.

● This section should also provide a starting point for another student to continue the work.

**References and Bibliography**

Harvard referencing style.

**Appendices**

Note: Well-documented code listings should be included in an appendix, not in the main body of the report.

PROPS = PROPERTIES!

: <https://expressjs.com/>

<https://www.pagecloud.com/blog/how-to-add-custom-fonts-to-any-website>

<https://techtrim.tech/express-vs-flask/>

<https://www.w3schools.com/css/css3_buttons.asp>

<https://www.youtube.com/watch?v=Y_yw5L-D3IQ&ab_channel=RizwanKhan>

Use DOM in Node: <https://github.com/jsdom/jsdom>

<https://codingsans.com/blog/node-config-best-practices>

To Do:

Drop-Down Login & Logo with Coloration

<https://stackoverflow.com/questions/28863097/template-inheritance-with-node-js-handlebars-and-express>

<https://www.youtube.com/watch?v=WqMbzVWIAjY>

PATCH not PUT for amending prescription!

Surgeries should have prescribe (POST), amend (PATCH), and Delete (plus autodelete when expired)

Pharmas should have retrieve (GET) and mark prescription as dispensed?!?

Prescribe, Register both Doc & Pharmas forms with rendering of record into UI ( <div id=”output”> )with confirm buttons and clear buttons w/iterators (or uuid for random IDs?)

Put: Updating records ‘Working with thee edit state’?

New JSON entity shemas?

Form validations – REGEX?

PSI & MCI Nos validator?

Disappearing / Timeouts of warning/error notices

Push and pull pharma, doctor regs , and prescription JSONs to and from local storage

Easy HTTP library API Calls? / fetch API?

Arrow functions\*

<https://kinsta.com/blog/nodejs-vs-python/>

CSS Grid: [https://en.wikipedia.org/wiki/CSS\_grid\_layout /](https://en.wikipedia.org/wiki/CSS_grid_layout%20/) <https://www.youtube.com/watch?v=0xMQfnTU6oo>

MongoDB: <https://www.youtube.com/watch?v=2QQGWYe7IDU>

Bibliography:

Abeln , M. (2019) *What is the purpose of having two running ports when we working with reactjs and nodejs?*, *Stack Overflow*. Available at: https://stackoverflow.com/questions/57362813/what-is-the-purpose-of-having-two-running-ports-when-we-working-with-reactjs-and (Accessed: December 19, 2022).

Admin (2022) *Redux in ReactJS*, *W3schools*. Available at: https://www.w3schools.blog/redux-reactjs (Accessed: December 20, 2022).

Bosler, F. (2020) *Set up an express.Js app with passport.js and mongodb for password authentication*, *Medium*. The Startup. Available at: https://medium.com/swlh/set-up-an-express-js-app-with-passport-js-and-mongodb-for-password-authentication-6ea05d95335c#94df (Accessed: January 4, 2023).

Codes, C. (2022) *Google Identity Services Login with react (2022 react Google Login)*, *YouTube*. YouTube. Available at: https://www.youtube.com/watch?v=roxC8SMs7HU&t=944s&ab\_channel=CooperCodes (Accessed: January 5, 2023).

*Conditional Rendering* (no date) *ReactJS*. Available at: https://reactjs.org/docs/conditional-rendering.html#gatsby-focus-wrapper (Accessed: January 3, 2023).

*CORS* (no date) *Node Package Manager*. Available at: https://www.npmjs.com/package/cors (Accessed: December 22, 2022).

Eich, B. and Friedman, L. (2021) “Brendan Eich: JavaScript, Firefox, Mozilla, and Brave | Lex Fridman Podcast #160,” *The Lex Friedman Podcast*. Available at: https://www.youtube.com/watch?v=krB0enBeSiE&ab\_channel=LexFridman (Accessed: November 30, 2022).

Erikson, M. (2022) *Redux createstore() is deprecated - cannot get state from getState() in Redux Action*, *Stack Overflow*. Available at: https://stackoverflow.com/a/71947129/19219155 (Accessed: December 20, 2022).

Fatunmbi, T. (2022) *A comparison of cookies and tokens for secure authentication*, *Okta Developer*. Okta Inc. Available at: https://developer.okta.com/blog/2022/02/08/cookies-vs-tokens#:~:text=Cookies%20and%20tokens%20are%20two,characters%20created%20by%20the%20server. (Accessed: October 4th, 2022).

*JavaScript Async* (no date) *W3 Schools*. Available at: https://www.w3schools.com/js/js\_async.asp#:~:text=async%20makes%20a%20function%20return,function%20wait%20for%20a%20Promise (Accessed: October 19, 2022).

*JSX in depth* (no date) *React*. ReactJS. Available at: https://reactjs.org/docs/jsx-in-depth.htm (Accessed: December 13, 2022).

\*\*On accepting passwords at login\*\*:

Maaajomaaajo 74955 silver badges1010 bronze badges, K.E. (2020) *What is the Difference Between handlechange vs onchange in which is used in react?*, *Stack Overflow*. Available at: https://stackoverflow.com/questions/62197917/what-is-the-difference-between-handlechange-vs-onchange-in-which-is-used-in-reac (Accessed: December 31, 2022).

Node.js (no date) *Overview of blocking vs Non-Blocking*, *Node.js*. Node.js. Available at: https://nodejs.org/en/docs/guides/blocking-vs-non-blocking/ (Accessed: December 13, 2022).

Olawanle, J. (2022) *Post HTTP Request in React*, *Stack Abuse*. Stack Abuse. Available at: https://stackabuse.com/post-http-request-in-react/ (Accessed: December 21, 2022).

*Openid Connect* (2022) *IBM.com*. International Business Machines Inc. Available at: https://www.ibm.com/docs/en/was-liberty/base?topic=liberty-openid-connect (Accessed: December 19, 2022).

Patadiya, J. (2022) *React vs react native - key difference, features, advantages*, *Radixweb*. Radixweb. Available at: https://radixweb.com/blog/react-vs-react-native (Accessed: December 19, 2022).

Raj, V. (2019) *Running react and node.js in one shot with concurrently!*, *DEV Community*. DEV Community. Available at: https://dev.to/numtostr/running-react-and-node-js-in-one-shot-with-concurrently-2oac (Accessed: December 19, 2022).

*React ES6 arrow functions* (no date) *W3 Schools*. Available at: https://www.w3schools.com/react/react\_es6\_arrow.asp (Accessed: December 19, 2022).

*React ES6 spread operator* (no date) *W3 Schools*. Available at: https://www.w3schools.com/react/react\_es6\_spread.asp (Accessed: December 19, 2022).

Sakimura, N. *et al.* (2022) *OpenID Connect Core 1.0 incorporating errata set 1*, *OpenID Connect Core 1.0* . OpenID Connect. Available at: https://openid.net/specs/openid-connect-core-1\_0.html (Accessed: December 22, 2022).

*Start using your openid* (2012) *OpenID*. Available at: https://openid.net/start-using-your-openid/ (Accessed: November 8, 2022).

*Throw, and Try...Catch...Finally* (no date) *JavaScript Errors*. W3 Schools. Available at: https://www.w3schools.com/js/js\_errors.asp#:~:text=JavaScript%20try%20and%20catch,occurs%20in%20the%20try%20block. (Accessed: November 3, 2023).

Tiwari, V. (2022) *How to fix cors error "no 'access-control-allow-origin' header is present on the requested resource"?*, *codedamn*. Available at: https://codedamn.com/news/backend/how-to-fix-cors-error (Accessed: December 22, 2022).

*Understanding and Resolving Cors Error* (no date) *Contentstack RSS*. Available at: https://www.contentstack.com/docs/developers/how-to-guides/understanding-and-resolving-cors-error/ (Accessed: December 22, 2022).

*Using the Effect Hook* (no date) *ReactJS*. Available at: https://reactjs.org/docs/hooks-effect.html (Accessed: December 24, 2022).

*Using the State Hook* (no date) *ReactJS*. Available at: https://reactjs.org/docs/hooks-state.html (Accessed: December 19, 2022).

*Why redux toolkit is how to use Redux Today* (2022) *ReduxJS*. Available at: https://redux.js.org/introduction/why-rtk-is-redux-today (Accessed: December 20, 2022).