## Criterion A: Planning

### The scenario

The client and adviser, [REDACTED], is the owner and CEO of [REDACTED] Pty. Ltd, which is a housing construction company that operates in Sydney, New South Wales (NSW), Australia.

In NSW, contractors working on a building site must have appropriate and up-to-date certification. Certification includes a valid and up-to-date Australian Business Number (ABN) and Service NSW licence. Otherwise, the company hiring the contractor has to pay the contractor’s business taxes. This has substantial implications on the client, as he would have to pay “tens of thousands of dollars in contractor’s taxes every year”.

Currently, there is no easy way to check whether a contractor has appropriately valid licence to work. The current system involves the client’s partner logging each contractor in a spreadsheet weekly and confirming the contractor’s status’ weekly, which is stored on a cloud drive account once confirmed. “It is not ideal”, the client states, and if she did not have to do this task, she could allocate “time to do [...] productive things in the company.”

### Rationale for proposed solution

In “Interview 1”, we discussed possible solutions to reduce administrative overhead which reduces business costs. The primary objective we established was that it should store contractor certification on a project-by-project basis. Maintaining a strong reputation with building regulatory authorities and contractors is necessary for growth of the business. This is at a demise if projects and contractor records are managed poorly. Additionally, the software must be able to source data from reliable storage. Losing data would be a great detriment for operations and cause them to “receive [big fines]”. As records are currently managed “physically”, digitalising them will decrease the client’s stress level, and ensure that there is a suitable back-up at all times. Additionally, if the client goes on a holiday, the substitute manager would be able to manage necessary data. As indicated in the interview, the client stated that “updates” were not an immediate priority, as they “know the contractors well” and do not change often. Also, a “to-do list” for further items of action to be noted is desirable.

In “Interview 2” we discussed the implementation details of a solution. The use of HTML, CSS, and JavaScript allows for a responsive UI, that can be used in both office and building-site environments and on the client’s “phone” and “laptop”. JavaScript allows for the use of REST APIs and searching JSON databases, which enables data exchange between the frond-end and back-end of the solution. The use of Java and the SpringBoot framework in the back-end allows for class structures, and the use of open-source libraries (such as the JSON library), which makes software development and maintenance efficient. The MVC design paradigm between these software components simplifies future software changes. The client has an existing cloud storage solution which can be used to store data files in JSON.

### Success criteria

1. Client can manage building project records, consisting of:
   1. Title
   2. Address
   3. Project description
   4. Suburb
   5. Contractors
2. Client can manage contractors associated with projects, consisting of:
   1. Name
   2. Licence type
   3. Licence number
   4. Other appropriate pieces of information made publicly available about the contractors
3. Search the list of projects or contractors to find the desired one to view
4. Be able to download data as a JSON file format for storage in already-established cloud storage system consistently
5. Adhere to the following usability criteria:
   1. Intuitive, doesn’t require much training to use
   2. Instructions of usage are clearly displayed
   3. UI is consistent with client expectations
   4. UI is responsive to various screen sizes and functions fully
   5. Client device agnostic

Words: 429

## Criterion B: Solution overview

### Record of tasks

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **Planned action** | **Planned outcome** | **Time estimated** | **Target completion date** | **Criterion** | |
| 1 | Initial meeting with client, learning about situation and current system | Talk to the client about the project, learn about the requirements of the supervisor | 45 minutes | 06/09/19 | | A |
| 2 | Discussion with advisor about client, scope of project | Advisor approves the project in its current scope, and confirms my ability start prototyping | 45 minutes | 07/09/19 | | A |
| 3 | Prototyping user interface designs, researching underlying technologies | Start working through project requirements, start prototyping a browser-based UI that fits with requirements | 3 hours | 12/09/19 | | A & B |
| 4 | Second meeting with client, finalising prototype designs | Talk with supervisor about the first prototypes of browser-based UI for further prototyping | 45 minutes | 14/09/19 | | A & B |
| 5 | Create software design based on MVC paradigm (UI, UML, DFD, and ER diagrams) | Design how the software components of the application will work together in according to agreed requirements | 5 hours | 1/12/19 | | B & C |
| 6 | Learning of technologies, and initial creation of software components | Learning and development of the elements that need to be created for the development of the project and committing to GitHub. | 1 hour | 3/12/19 | | C |
| 7 | Development of the “View” software component | Work through developing the HTML, CSS, and JavaScript | 20 hours | 15/02/19 | | C |
| 8 | Development of the “Controller” software component | Work through developing the backend in Java using REST APIs | 15 hours | 01/05/20 | | C |
| 9 | Development of “Model” software component | Work through developing JSON file format | 5 hours | 07/05/20 | | C |
| 10 | Field test with client in a project environment; implementation of solution | Test if the application is fit for purpose as per agreed requirements | 2 hours | 10/05/20 | | D & E |
| 11 | Interview with client to confirm if developments are ok, and final adjustments to software components | Implement feedback from field test, adjust the software components accordingly | 45 minutes | 30/05/20 | | C & E |
| 12 | Creation of video with the project in use | Create a video to demonstrate how the application functions | 2 hours | 05/06/20 | | D |
| 13 | Final interview and reflection with client, as well as personal reflection | Review the final functionality of the application, and how application can be extended | 45 minutes | 10/06/20 | | E |

### Design overview

**Structure Chart**

**A close up of a piece of paper

Description automatically generated**

**Model-View-Controller (MVC) diagram (List of References, 1)**

**A picture containing screenshot

Description automatically generated**

**Software Component “View”**

Figure 1: Initial Main UI



* Expected behaviour upon clicking the search bar: The outline of the search bar will go blue, and upon typing will automatically start filtering out irrelevant projects
* Expected behaviour upon clicking the outline of a button/any element within it: The button will expand, to show more information, as is shown in Figure 2.
* Expected behaviour upon clicking the add new project button: The add new project input tabs will show up, as is shown in Figure 3
* Expected behaviour upon loading: Runs getAllContractors() API and getAllProjects() API to display previously stored data

Figure 2: Initial In-depth Project UI

A screenshot of a cell phone

Description automatically generated

* Expected behaviour of to-do section is clicked: The completion status will change to be completed
* When inputs for to-do items, project photo, description, name or contractor list overflow, the program will automatically generate scrollable sidebars, for which show all data

Figure 3: Initial Add New Project UI

A screenshot of a cell phone

Description automatically generated

* Expected behaviour upon clicking on any input field: Input box goes blue, and typing the query ending with an “enter key” will appropriately add the information to the project
* Expected behaviour upon clicking on the add project button: Data will be written on the front page, and will be written to an appropriate local file, through the createNewProject() API

Figure 4: Initial Mobile UI

A screenshot of a cell phone

Description automatically generated

* Website will function identically despite smaller screen size as annotated in Figure 1 and 5

Figure 5: Final Main UI

A picture containing computer

Description automatically generated

* Expected behaviour upon clicking the search bar: The outline of the search bar will go blue, and upon typing will automatically start filtering out irrelevant projects
* Expected behaviour upon clicking the outline of a button/any element within it: The button will expand, to show more information, as is shown in Figure 2.
* Expected behaviour upon clicking the add new project button: The add new project input tabs will show up, as is shown in Figure 3
* Expected behaviour upon loading: Runs getAllContractors() API and getAllProjects() API to display previously stored data

Figure 6: Final In-depth Project UI

A screenshot of a cell phone

Description automatically generated

* Expected behaviour of to-do section is clicked: The completion status will change to be completed
* When inputs for to-do items, project photo, description, name or contractor list overflow, the program will automatically generate scrollable sidebars, for which show all data

Figure 7: Final Add New Project UI

A screenshot of a cell phone

Description automatically generated

* Expected behaviour upon clicking on any input field: Input box goes blue, and typing the query ending with an “enter key” will appropriately add the information to the project
* Expected behaviour upon clicking on the add project button: Data will be written on the front page, and will be written to an appropriate local file, through the createNewProject() API

Figure 8: Final Mobile UI

A screenshot of a cell phone

Description automatically generated

* Website will function identically despite smaller screen size as annotated in Figure 1 and 5

**Software Component “Controller”**

UML Diagram

A screenshot of text

Description automatically generated

**Software Component “Model”**

Entity Relationship Diagram (ER)

### A screenshot of a cell phone Description automatically generated

**Software Component Integration**

Data Flow Diagram (DFD)

### A close up of text on a white background Description automatically generated

API Usage Diagram

A screenshot of a cell phone

Description automatically generated

### Function purposes

| **Function name** | **Function purpose** |
| --- | --- |
| + expandProject(projectNumber) | Use a designated project number in order to expand the front-end view of a building project |
| + search(link) | Search through list of all building projects to see if searched item appears in *any* places of a building project |
| + getLicenceID() | Retrieve the encapsulated licenceID variable |
| + getLicenceNumber() | Retrieve the encapsulated licenceNumber variable |
| + getBusinessNames() | Retrieve the encapsulated businessNames variable |
| + getCategories() | Retrieve the encapsulated categories variable |
| + getClasses() | Retrieve the encapsulated classes variable |
| + getLicenceName() | Retrieve the encapsulated licenceName variable |
| + getLicenceType() | Retrieve the encapsulated licenceType variable |
| + getlicensee() | Retrieve the encapsulated licensee variable |
| + getPostcode() | Retrieve the encapsulated postcode variable |
| + getStatus() | Retrieve the encapsulated status variable |
| + getSuburb() | Retrieve the encapsulated suburb variable |
| + getProjectID() | Retrieve the encapsulated projectID variable |
| + getClientName() | Retrieve the encapsulated clientName variable |
| + getStreetAddress() | Retrieve the encapsulated streetAddress variable |
| + getSuburb() | Retrieve the encapsulated suburb variable |
| + getDescription() | Retrieve the encapsulated description variable |
| + getContractorList() | Retrieve the encapsulated contractorList variable |
| + addContractor(Contractor contractor) | When given a Contractor object, add a contractor to an external database in JSON format |
| + getAllContractors() | Retrieve all Contractor objects stored on database in JSON format |
| + getContractorByID(String licenceID) | When given the licenceID of a Contractor object that has been stored, get the Contractor object and select it for below functions. If appropriate Contractor with licenceID searched has not been found, return nothing |
| + deleteContractorByID(String licenceID) | Given a Contractor object that has been previously selected using the above method, delete the object from database |
| + insertProject(Project project) | When given a Project object, add a contractor to a database in JSON format |
| + selectAllProjects() | Retrieve all Project objects stored on database in JSON format |
| + selectProjectByID(String projectID) | When given the licenceID of a Project object that has been stored, get the Project object and select it for below functions. If appropriate Contractor with licenceID searched has not been found, return nothing |
| + deleteProjectByID(String projectID) | Given a project object that has been previously selected using the above method, delete the object from database |

### Implementation plan to meet success criteria

|  |  |
| --- | --- |
| **Success criteria** | **Plan to implement** |
| Manage projects | Implement a back-end data base in order to store appropriate data that the client wants to store (such as a project name, photos, and to-do list items), and can be accessed from the client-side  Develop REST API for Project using Java and Springboot, using their appropriate methods (GET, POST, and DELETE) |
| Manage contractors | Implement a back-end data base in order to store appropriate data that the client wants to store (such as a contractor ID, business names, and classes), and can be accessed from the client-side  Develop REST API for Contractors using Java and Springboot, using their appropriate methods (GET, POST, and DELETE) |
| Search the list of projects and contained contractors to find a desirable one | Create a search bar in a JavaScript front-end which can search through everything that is contained within a particular project (such as the names of contractors, addresses of particular houses, etc.) |
| Conforms to aforementioned UI expectations | Throughout development, continually use desktop and mobile versions of the website to ensure that usage is fine. Make sure application is somewhat lightweight and can handle weaker phone hardware |
| Be able to download data into a JSON file format for storage in already-established cloud storage system | Ensure that the database, and the client, have access to a method in which the database is downloaded into a JSON file through a button on the application |

### Testing plan

| **Test scenario** | **Testing method** | **Expected result** | **Testing data** |
| --- | --- | --- | --- |
| Project view can be expanded | Click on project view when collapsed and expanded | Project view expands when already collapsed, and collapses when already expanded | \*User clicks on any section of button\* |
| Projects can be searched | Put through name of any certain project into search bar | Only the project with the searched project name shows up | First two characters of address - “14”  Middle word of description – “in” |
| Contractor can be added | Input sample contractor with random name, licenceID, and other points of information in JSON format | Contractor is inserted into database | “licenceID” = 5687  “licenceNumber” = 65  “businessNames” = Alex Building Inc  “categories” = Architect  “classes” = Trainee  “licenceName” = N/A  “licenceeType” = Trainee  “licensee” = NSW Apprenticeship  “postcode” = 2634  “status” = Not currently available  “suburb” = Pearce |
| Contractor can be retrieved from database | Input sample contractor licenceID | Contractor’s JSON data values are shown | “licenceID” = 5678 |
| Contractor that is selected can be deleted using their appropriate licenceID | Input sample contractor licenceID | Contractor with appropriate licenceID is deleted from database | “licenceID” = 5678 |
| Project can be added using their appropriate projectID | Input sample project projectID and all other appropriate fields | Project’s JSON data values are shown | “projectID” = 41675  “clientName” = John Sample  “streetAddress” = 18 Samplee Crescent  “suburb” = Mainstay  “description” = 2 bedroom two bathroom house  “contractorList” = John, Alex, Fish |
| Project can be retrieved from database | Input sample project projectID | Project’s JSON data values are shown | “projectID” = 41675 |
| Project that is selected can be deleted using their appropriate projectID | Input sample project projectID | Project with appropriate projectID is deleted from database | “projectID” = 41675 |
| User is attempting to operate the application | Observing and asking about the user’s attempt to use the application | User does not ask questions, refers to instructions on application, does not get confused, feels application is suitable | N/A |
| User is attempting to operate application on mobile phone, and laptop | Observing and asking about the user’s attempt to use the application on two different devices | User can perform the same functionality regardless of the devices that are being used, user feels as if the UX on mobile is similar to desktop counterpart, user can use application regardless of device manufacturer | N/A |
| Client has entered project or contractor data and wants to see written files | Insert real life project/contractor data (as above) and open contractorData.txt or projectData.txt accordingly | All of the projects that are inserted into the application can be read in JSON format within their appropriate file. |  |

## Criterion C: Development

### HTML, CSS, and JavaScript in the View

1. **Expanding buttons using JavaScript alterations of HTML + CSS elements**

### A screenshot of a cell phone Description automatically generated

* + JavaScript, HTML/CSS front-end (if/else statements + document.getElementById integration) function used to determine which button has been clicked on, and then whether it has to be expanded or contracted
    - Ingenuity: Different responses for when a button is already expanded, and when it needs to be expanded
    - Complexity: Gets the CSS values of HTML elements as parameters

1. **Search bar functionality to allow for filtering projects**

A screenshot of a cell phone

Description automatically generated

* + Search menu functionality to search through projects in View (List of References, 8)
  + JavaScript front-end (if/else statements + JavaScript array) to determine if searched item (link) has been found in searchedList, is shown if searched and hidden is not searched (through CSS)
    - Ingenuity: Uses for loop to hide elements that aren’t of the index entered in search bar
    - Complexity: Gets the CSS values of HTML elements as parameters

1. **ArrayLists used for storing Contractor objects**



* + Used to have a running list of all Contractor objects in Java backend, with an unlimited amount
    - Ingenuity: Imported data type used to allow for potential to increase the amount of Contractor objects that can be stored

1. **Creation of new HTML elements to display information on local file**

A screenshot of a cell phone

Description automatically generated

* + To programmatically create new buttons for each project, a HTML “div” element is created and other elements are appended to it (List of References, 5, 11)
  + Elements are cloned from a template project (project0), and are stored in an array elementsToDuplicate, which is looped through
  + Depending on what the id of the HTML element is, and the number of projects created, the program will change id to avoid duplicate HTML ids
    - Ingenuity: Creates a div element (the only element that can be appended to in HTML) for each new button, allowing for programmatic creation of new HTML elements
    - Complexity: Gets the CSS values of HTML elements as parameters, checks against id in if statements

### JavaScript REST API calls in the View

1. **Get contractor request from local HTTP server in JavaScript**

### A close up of a logo Description automatically generated

* + JavaScript used in front-end to create an XML HTTP Request to GET information about Contractors from a locally established server
  + Output is converted to a JavaScript object from JSON using JSON library
    - Ingenuity: API calls from JavaScript such that if there is an error in making the request, the user will be notified and no errors will occur in the application
    - Complexity: JSON library is used to parse information that comes through from local HTTP server

1. **Get project request from local HTTP server in JavaScript**

A picture containing drawing

Description automatically generated

* + JavaScript used in front-end to create an XML HTTP Request to GET information about Projects from a locally established server
  + Output is converted to a JavaScript object from JSON using JSON library
    - Ingenuity: API calls from JavaScript such that if there is an error in making the request, the user will be notified and no errors will occur in the application
    - Complexity: JSON library is used to parse information that comes through from local HTTP server

1. **Post contractor request to local HTTP server in JavaScript**

A screenshot of a cell phone

Description automatically generated

* + JavaScript used in front-end to create an XML HTTP Request to POST user-inputted Contractor data to local API
  + Converted from JavaScript object to JSON using JSON library (List of References, 4)
    - Ingenuity: API calls from JavaScript such that if there is an error in making the request, the user will be notified and no errors will occur in the application
    - Complexity: Variables are extracted from HTML elements, which are then turned into JSON objects through the JSON library, then sent to the local HTTP server

1. **Post project request to local HTTP server in JavaScript**

A screenshot of a cell phone

Description automatically generated

* + JavaScript used in front-end to create an XML HTTP Request to POST user-inputted Project data to local API
  + Converted from JavaScript object to JSON using JSON library
    - Ingenuity: API calls from JavaScript such that if there is an error in making the request, the user will be notified and no errors will occur in the application
    - Complexity: Variables are extracted from HTML elements, which are then turned into JSON objects through the JSON library, then sent to the local HTTP server

1. **Delete project request to local HTTP server in JavaScript**

A screenshot of a cell phone

Description automatically generated

* + JavaScript used in front-end to create an XML HTTP Request to DELETE user-inputted Contractor data to local API
  + Contractor to be deleted is the value defined in user input on website
    - Ingenuity: API calls from JavaScript such that if there is an error in making the request, the user will be notified and no errors will occur in the application
    - Complexity: Variables are extracted from HTML elements, then sent to the local HTTP server

1. **Delete contractor request to local HTTP server in JavaScript**

A picture containing drawing

Description automatically generated

* + JavaScript used in front-end to create an XML HTTP Request to DELETE user-inputted Project data to local API
  + Project to be deleted is the value defined in user input on website
    - Ingenuity: API calls from JavaScript such that if there is an error in making the request, the user will be notified and no errors will occur in the application
    - Complexity: Variables are extracted from HTML elements, then sent to the local HTTP server

### Java REST API calls in Controller

1. **Encapsulation of Java variables**

A close up of text on a black surface

Description automatically generated

* + Java back-end use of private final variables for Contractors in order to be encapsulated (List of References, 9), such that unintentional access/change does not occur
    - Ingenuity: Encapsulated variables fit the needs of the client well: protects against accidental alterations

1. **Getter functions for earlier encapsulated Java variables**

A close up of text on a black background

Description automatically generated

* + Java back-end getter methods for earlier private final variables for Contractors; setter variables not needed, as API always provides correct input data, and doesn’t need to change (List of References, 9)
    - Ingenuity: Encapsulated variables fit the needs of the client well: only a need to access information rather than update

1. **Use of SpringBoot JSONProperty and Java Constructor**

A screenshot of text

Description automatically generated

* + Java (List of References, 5 & 10), JSON (List of References, 4), and SpringBoot (List of References, 2) back-end Object Oriented Programming use, to initialise the earlier private final variables, and declare the variables as JSON such that they are sent to database for Contractors
    - Ingenuity: SpringBoot allows for annotations (in yellow) which decrease redundant code typically used in making JSON properties in SpringBoot
    - Complexity: Requires an understanding OOP, which is used to construct both contractor and project methods as is required

1. **Use of SpringBoot API functionality in Java**

A screenshot of a cell phone

Description automatically generated

* + Java + SpringBoot framework back end used in order to create REST API Controller for database connectivity, with appropriate POST, GET, and DELETE functionality to change database using JSON data for Contractors (List of References, 6)
  + POST ( addContractor() ) method sends a contractor to API Service in JSON
  + GET ( getAllContractors() ) method receives a list of all contractors in JSON
  + GET ( getContractorByID() ) method takes a given licenceID, and finds whether there is an associated Contractor JSON object in database
  + DELETE ( deleteContractorByID() ) method deletes any selected contractor, once given licenceID variable
  + *Similar methods were created for the “projects” class, with all appropriate private variables, encapsulation, constructors, JSON properties, and REST APIs developed* (List of References, 6)
    - Ingenuity: SpringBoot allows for the use of annotations (in yellow) decreases redundant code typically required in REST API development at the Controller stage
    - Complexity: Development of layers (API/Server Layer, Data Access Layer, Controller, and Model layers) that all connect to provide functional REST API

### Interactions with JSON Model

1. **Creating POST call to files for storing project/contractor data**

A screenshot of a cell phone

Description automatically generated

* When using previous stored data, a file is inputted into addWrittenContractors / addWrittenProjects
* A GET XML HTTP Request is made to the file to get its recorded data, and then split into the splitWritten array by new line
* Once this is completed, the array is looped through and a POST request is made for all of the splitWritten items such that they are on the external server (List of References, 12)
* *A similar method to addWrittenProjects has been developed* 
  + - Ingenuity: Ingenuity: API calls to JSON model such that if there is an error in making the request, the user will be notified and no errors will occur in the application
    - Complexity: Use of splitting array of characters from Java so can be written in the Java Model

## Criterion D: Flexibility and extensibility of product

<https://youtu.be/swdBnvYI8HI>

## Criterion E: Evaluation

Upon speaking to the client in Interviews 4 & 5, it was “clear” that all of the previously outlined success criteria had been met, and the needs of the client were fulfilled. Throughout the demonstrations of the functionalities of the application to the client in Interview 4, we addressed all of the test cases established in Criteria B. Already “being familiar” with applications allowed the client to read, create, and delete contractors and projects with ease. All of the data points for contractors and projects “were addressed”, and available to use. Similarly, the search bar was usable, and it took the client little time to get used to searching for projects. The output files “contractorData” and “projectData” worked “well enough” for the client, and the application worked well within the space and performance constraints of his phone.

Within the success criteria, the client could not find any “drastic” faults in the application and just suggested to “slightly tweak” the interface to make it more aesthetically pleasing, which would be ideal if he were to adopt the application for other people in his business.

In the questionnaire of Interview 5, the client stated that one of the possible improvements brought up was the use of external, “government-created” APIs to get information about contractors. In manually adding data, there is a chance that provided contractor data “could be wrong”, which could cause issues with taxation and licensing. Additionally, this would also allow for the client to use the application in other states where APIs are available, as the client does not know contractors from outside NSW, which was a “possibility” for him in the future.

Another possible improvement mentioned in the questionnaire of Interview 5 would be the “automated reminders” for licence expiries to a calendar or phone. The client already has an “online Google Calendar” solution and adding the functionality of being able to append a project to this solution could function to alert him more easily of when a particular contractor’s licence is going to expire. This would use the Google Calendar API from the “Controller” section to send POST calls.

Finally, another feature raised in the questionnaire of Interview 5 was to-do lists. Although the client did not need the functionality, having to-do lists for each project would allow users to be more proactive, and “not rely on other applications” to perform the same functionality. To-do lists could also be responsive to the statuses of contractors in a project and dynamically create to-do lists if the status of the contractor requires action (e.g. if their licence were to expire). This would be “especially useful” if the client were to adopt this application for the rest of his employees, which could decrease “costs” for other to-do applications.

Words: 454 words

## Appendix

#### List of References

1. Smith, S. (2020, February 12). Overview of ASP.NET Core MVC. Retrieved June 21, 2020, from <https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-3.1>
2. Building an Application with Spring Boot. (n.d.). Retrieved June 21, 2020, from <https://spring.io/guides/gs/spring-boot/>
3. JavaScript Tutorials. (2020). Retrieved 24 June 2020, from <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
4. JSON Tutorials. (2020). Retrieved 24 June 2020, from <https://www.w3schools.com/whatis/whatis_json.asp>
5. The Java™ Tutorials. (2020). Retrieved 24 June 2020, from <https://docs.oracle.com/javase/tutorial/>
6. Todd Fredrich, P. (2020). REST API Tutorial. Retrieved 24 June 2020, from <https://www.restapitutorial.com/>
7. The 4 Golden Rules of UI Design | Adobe XD Ideas. (2020). Retrieved 24 June 2020, from <https://xd.adobe.com/ideas/process/ui-design/4-golden-rules-ui-design/>
8. Search List. (2020). Retrieved 24 June 2020, from <https://www.w3schools.com/howto/howto_js_filter_lists.asp>
9. George, J. (2020). Encapsulation as a Concept. Retrieved 24 June 2020, from <https://stackoverflow.com/questions/4772956/encapsulation-concept>
10. Daru, N., Fernandez, P., & Sanchez, R. (2020). What's Object Oriented Programming?. Retrieved 24 June 2020, from <https://stackoverflow.com/questions/38467631/whats-object-oriented-programming>
11. Konieczny, J. (2020). How do I make a JSON file with JavaScript?. Retrieved 24 June 2020, from <https://www.quora.com/How-do-I-make-a-JSON-file-with-JavaScript>
12. Hill, J., Toumi, A., Wallwork, J., & Georgiev, O. (2020). Split String into Array. Retrieved 24 June 2020, from <https://stackoverflow.com/questions/7979727/split-string-into-array>

#### Interviews

#### Interview 1

Me: Good afternoon, in this interview, we should be able to establish a general idea of the problem and a solution that would fix it.

Client: Absolutely, good afternoon. Primarily, when working on a project, we hire a lot of contractors for every purpose, you know- tiling, electrical work, plumbing, and whatever else. Each one of those contractors has to have the appropriate certification to be able to work. Otherwise, we get charged for the taxes they have to pay. There’s no way currently to easily be able to check whether a contractor has valid certification, and that’s the problem.

Me: What certification do you have to manage for the contractors? What is your current method of managing these contracts?

Client: The certification includes an NSW Service licence for whatever job they’ve got to do. Currently, my wife inputs everything into a spreadsheet application after checking everything every week. It’s not ideal, because if she forgets to check everything for one week when we’re on holidays or something we could receive a significant fine – it would be like tens of thousands of dollars a year if at its worst. If she was free to not do this task, she could use that time to do productive things in the company.

Me: How do you use your current system? Would it have to be available to be used on the go, or would something like a stationary desktop application be more appropriate? How do you use your current system?

Client: I haven’t thought about that- we mainly use our current system on our laptops at home and bring them out to sites when we have to meet a new contractor. It would be quite useful to have a solution on the go, on our phones. We currently use our phones a great deal when calling contractors and clients for our company, so a way to use the solution on our mobile devices would be nice.

Me: A mobile system sounds doable and a nice solution to your problem. I imagine that you would also want it to run on your computer at home as well?

Client: That would be nice so that we could rely on both platforms, and it wouldn’t take that much time to learn over our current system.

Me: Do you have any other concerns with the operation of the solution? Anything else that you want me to consider when developing it?

Client: Yeah, a few things. I think it would have to be easy to use for people of our age, and something that we could learn to use in a short amount of time. We’re always getting new clients and building more houses, so the less time it takes to learn, the more time we can save and devote to building better houses. Also, it has to be secure and not lose its data or be compromised or something like that. One benefit of our current system is that it’s backed up on the cloud, and secure. We always know that it’s going to be there and that we’re not going to lose our information.

Me: Absolutely, I imagine that you might want a similar cloud-based system with the new solution as well. Anything else to add to the solution?

Client: ...Oh- like a small to-do list for each project so that we have actionable points to plan out our projects and any further action to do with contractors. Although we do know most of our contractors well, and probably wouldn’t need to update anything, we still might need a reminder to do stuff. ... And, yeah, I think we’re good.

Me: A to-do list sounds like a good idea. Thanks for your time. We will schedule a meeting after I’ve made some concept designs and mock-ups.

Client: Thanks, until next time.

#### Interview 2

Me: Thanks for making time again. I have come back with mock diagrams using all the information you gave me the last interview, and here they are.

\*Refer to Figure 1 in Mock Diagrams – Appendix\*

Me: Would something like this be appropriate for the main view on a laptop?

Client: Yeah, this looks nice. I think it contains everything that we would need to use. Is this just the only screen? Or does something happen if you click on one of the project boxes?

Me: So this is just the main screen that’s used to give you information about everything that’s stored on the website, and then you will be able to click on each of the projects to get a better view of them and edit whatever you need to.

Client: Brilliant. If we’re talking about looking at everything from a glance, would it be possible to get a box up the top or somewhere that rounds up all possible problems in each of the projects? Like if a contractor was to have an item of certification expire soon?

Me: Yes, that would be a great idea. I will put that in the final mock-up. I could probably implement upcoming to-do items as well if you want those throughout the project too- I believe I did not add those in the mock-ups.

Client: That would be great. Cheers.

\*Refer to changes in blue in Figure 5 in Mock Diagrams - Appendix\*

Me: Onto the next, this is the screen that you will get after clicking on any particular project.

\*Refer to Figure 2 in Mock Diagrams - Appendix\*

Me: This is what it would look like when you click on a specific project. If you want me to add timestamps for each to-do item that would be possible in the to-do items box.

Client: Yep, that would be good. Also, we need to go through what information to display in the contractor’s box. Do you know what you can find online and where to find it?

Me: Yes, I had a look through all of the websites and APIs that could lend data, and I think that getting the expiry date of their NSW Services licence and ABN. What else would be required?

Client: Something like the role or job that they’re registered to do, because if they do something else that they don’t have a proper license for the same tax issues come into play.

Me: Oh, ok. I will make a note of all of those for the final mock-ups. Anything else on this page?

Client: Nothing else, onto the next one.

\*Refer to changes in blue in Figure 6 in Mock Diagrams - Appendix\*

\*Refer to Figure 3 in Mock Diagrams - Appendix\*

Me: This is the screen that you would get when you add a new project from the main screen that we looked at earlier. As you said, the dates have to be added to the to-do list section. Anything else you can see that you would want?

Client: Again, with the contractor’s section, could we be a little more specific? Like, could you get it to give me a list of everyone with a specific name, Service NSW or ABN? Other than that, and the date added thing, it looks good.

Me: I do not think that we would easily be able to get API calls on the go for the ABN because of the way the government has structured the API, but I think that should be fine to do with the Service NSW and names.

Client: Ok, that’s all right.

\*Refer to changes in blue in Figure 7 in Mock Diagrams - Appendix\*

Me: The final mock-up is just of what I think it would look on your phone if you wanted to go mobile. There’s no real difference between this and the main view, just optimised to look better on a smaller screen.

Client: Yeah, that looks fine. I imagine that the rest of the functionality will do the same thing, like adding new projects and all that?

Me: Yes, that’s the aim.

Client: Too easy.

Me: As for the logistics with how to develop it, I’m planning on developing a website. That way, it will be accessible to you both on your laptop and your phone easily, and I have previous experience developing- so debugging and development will take less time.

Client: What about cyber-attacks and other things with compromising data? As I think I said in the first interview, if we lose our data, then we’re screwed.

Me: I was thinking about that. I’ll try my best to make the website as secure as possible, and I have had previous experience working on websites so that shouldn’t be too hard to cover. There are backup methods of security that we will have to use, however. Can you think of a way that you would use to do that?

Client: ... Could we download the data or something so that we could store it in the cloud?

Me: Yes, something like implementing a download as a spreadsheet system could be implemented. Would you prefer to use that over a system inbuilt into the app?

Client: We’ve got our current system that works well, managing cloud storage of data, so I don’t think that we need an implementation of it directly in the app. Other than that, ...I’ve got nothing.

Me: All right, thanks for that. We’ll schedule another interview when I’ve finished development.

Client: All right, cheers.

#### Interview 4

Me: Hi there, thanks for being patient and waiting for the final product. Here’s the product and how it functions.

\*Showed the functionality of the product by running it on laptop\*

Client: Oh, that’s pretty cool. I was now just thinking about it, and I probably wouldn’t need photos. You know, when constructing a new project it’s nothing more than a hole in the ground, so it probably isn’t necessary. And now that you mention it, we just bought a to-do list application last week so we probably wouldn’t need that feature in there anyways.

Me: Oh, ok. I can remove those things... that should be no worries. The data is all downloaded to a central file – those being called contractorData.txt and projectData.txt. The project would also function when used in mobile.

\*Demonstration of those two things\*

Client: Oh, that’s good. Could we sample some of the information that I would put in for one of the projects I’m doing?

Me: Sure.

\*Proceeded to complete successfully and end interview\*

#### Interview 5

Me: Hi again, thanks for being here for the final interview. I’m going to ask you a series of questions about the application and what you think of it. If I could also ask, please be as honest as possible when answering.

Client: That’s fine.

Me: Do you think that the functionality of the application matched the success criteria we established earlier, and your general use cases?

Client: Yeah, I do think so. It’s pretty clear that through our demonstration last interview that everything has been addressed from the success criteria. Let’s see... Well, I have been using applications like this, which has helped me in being more familiar with this application quickly. Everything that I could want was addressed, especially with the contractor and project data file storages – they work well enough for us. And... yeah, it would work for us.

Me: Do you think that the application is genuinely better than the system that you use now? And if so, how?

Client: For sure. Using physical records is entirely out of date, and your application allows us to be able to not be at risk of damaging our physical records when on building sites. Also, environmental reasons, being able to use the files on my phone and backing them up all make your solution a lot better than our current one.

Me: Do you think that, within the success criteria, any part of the project could be improved? If so, what, and how?

Client: ... There aren’t any *drastic* things that I can think of that I want, maybe just a slight tweak of the app interface so that it’s a bit cleaner? I’m thinking of using this for more of our projects, and that would mean involving more employees, who would probably appreciate that. Other than that, though... no, nothing.

Me: Outside of the success criteria we established a while ago, what do you think could be improved about the project?

Client: I had a few ideas, but after we completed our previous interview, but I didn’t bother speaking up about them because it was too late. Recently, the company has been thinking about expanding our building operations to other states; mainly Victoria. Now, I don’t know many contractors that are outside of NSW, so it would be nice to have an official... government-created list to find contractors from. Information that could be wrong, and not being able to find the right contractors for the job would have significant implications on the work that we do here, so that would be useful. Also, especially if we were to widely use this solution for all of our projects, having some kind of Google Calendar and to-do list functionality would be useful. We already have a Google Calendar solution for our office, and although I said that I don’t personally need to-do list functionality, it could decrease the entire office’s reliance on other applications and decrease costs for getting other software. As I said, I already have to-do list functionality which shows me what I have to address with regards to my contractors, but implementing something like that for all of our employees to use; something that would automate reminders for impending tasks would be beneficial I imagine.

Me: Thanks for that, and that concludes our interview. Thanks for being with me on the project.

Client: The pleasure is mine.