Digital Solutions 2019 v1.2

IA2 high-level annotated sample response

June 2018

Project — digital solution

This sample has been compiled by the QCAA to assist and support teachers to match evidence in student responses to the characteristics described in the instrument-specific marking guide (ISMG).

Assessment objectives

This assessment instrument is used to determine student achievement in the following objectives:

1. recognise and describe programming elements, user interface components and useability principles
2. symbolise and explain programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype
3. analyse the problem and information related to the technical proposal for a low-fidelity prototype digital solution
4. determine user interface, data, programmed and solution requirements of the digital solution and prescribed and self-determined criteria
5. synthesise information and ideas to determine data elements, user interface and programmed components for a digital solution
6. generate user interfaces and programmed components of the digital solution
7. evaluate impacts, components and the digital solution against prescribed and self-determined criteria to make refinements and justified recommendations
8. make decisions about and use mode-appropriate features, written language and conventions for a technical audience.

Instrument-specific marking guide (ISMG)

Criterion: Retrieving and comprehending

Assessment objectives

1. recognise and describe programming elements, user-interface components and useability principles
2. symbolise and explain programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype

The student work has the following characteristics:	Marks
• accurate and discriminating recognition and discerning description of relevant programming elements, user-interface components and useability principles • adept symbolisation and discerning explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.	7–8
• accurate recognition and effective description of relevant programming elements, user-interface components and useability principles • methodical symbolisation and effective explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.	5–6
• appropriate recognition and description of some programming elements, user-interface components and useability principles • competent symbolisation and appropriate explanation of algorithms and some information and ideas, and interrelationships between user experiences and data of the digital prototype.	3–4
• variable recognition and superficial description of programming elements, user-interface components or useability principles • variable symbolisation and superficial explanation of information, ideas or interrelationships.	1–2
• does not satisfy any of the descriptors above.	0

Criterion: Analysing

Assessment objectives

3. analyse the problem and information related to the technical proposal for a low-fidelity prototype digital solution
4. determine user interface, data, programmed and solution requirements of the digital solution and prescribed and self-determined criteria

The student work has the following characteristics:	Marks
• insightful analysis of the problem and relevant contextual information to identify the essential elements and features of user interface, data and programmed components and their relationships to the structure of the low-fidelity prototype digital solution • astute determination of the user interface, data, programmed and solution requirements of the digital solution and essential prescribed and self-determined criteria.	7–8
• considered analysis of the problem and relevant contextual information to identify the relevant elements and features of user interface, data and programmed components and their relationships to the structure of the low-fidelity prototype digital solution • logical determination of the user interface, data, programmed and solution requirements of the digital solution and effective prescribed and self-determined criteria.	5–6
• appropriate analysis of the problem and contextual information to identify some elements and features of user interface, data and programmed components and their relationships to the structure of the low-fidelity prototype digital solution • reasonable determination of the user interface, data, programmed and solution requirements of the digital solution and some prescribed and self-determined criteria.	3–4
• superficial analysis of the problem or partial information to identify aspects of elements or features of the low-fidelity prototype digital solution • vague determination of some solution requirements of the digital solution and some criteria.	1–2
• does not satisfy any of the descriptors above.	0

Criterion: Synthesising and evaluating

Assessment objectives

5. synthesise information and ideas to determine data elements, user interface and programmed components for a digital solution
6. generate user interfaces and programmed components of the digital solution
7. evaluate impacts, components and the digital solution against prescribed and self-determined criteria to make refinements and justified recommendations

The student work has the following characteristics:	Marks
• coherent and logical synthesis of relevant information and ideas to determine data elements, user interface and programmed components for a digital solution • purposeful generation of efficient user interface and programmed components of the digital solution • critical evaluation of impacts, user experience and coded components and the digital solution against essential prescribed and self-determined criteria to make discerning refinements and astute recommendations justified by data.	9–10
• logical synthesis of relevant information and ideas to determine data elements, user interface and programmed components for a digital solution • effective generation of user interface and programmed components of the digital solution • reasoned evaluation of impacts, user experience and coded components and the digital solution against effective prescribed and self-determined criteria to make effective refinements and considered recommendations justified by data.	7–8
• simple synthesis of information and ideas to determine data elements, user interface and programmed components for a digital solution • adequate generation of user interface and programmed components of the digital solution • feasible evaluation of impacts, user experience and coded components and the digital solution against some prescribed and self-determined criteria to make adequate refinements and fundamental recommendations justified by data.	5–6
• rudimentary synthesis of partial information or ideas to determine data elements, user interface or programmed components • partial generation of user interface and programmed components of the digital solution • superficial evaluation of impacts, user experience components or the solution against some criteria.	3–4
• unclear combination of information, ideas or solution components • identification of a change to an idea or a solution.	1–2
• does not satisfy any of the descriptors above.	0

Criterion: Communicating

Assessment objective

8. make decisions about and use mode-appropriate features, written language and conventions for a technical audience

The student work has the following characteristics:	Marks
• discerning decision-making about, and fluent use of - written and visual features to communicate about a solution - language for a technical audience - grammatically accurate language structures - referencing and project conventions.	3–4
• variable decision-making about, and inconsistent use of - written and visual features - suitable language - grammar and language structures - referencing or project conventions.	1–2
• does not satisfy any of the descriptors above.	0

Sample response

Criterion	Allocated marks	Marks awarded
Retrieving and comprehending Assessment objectives 1, 2	8	8
Analysing Assessment objectives 3, 4	8	8
Synthesising and evaluating Assessment objectives 5, 6, 7	10	10
Communicating Assessment objective 8	4	4
Total	**30**	**30**

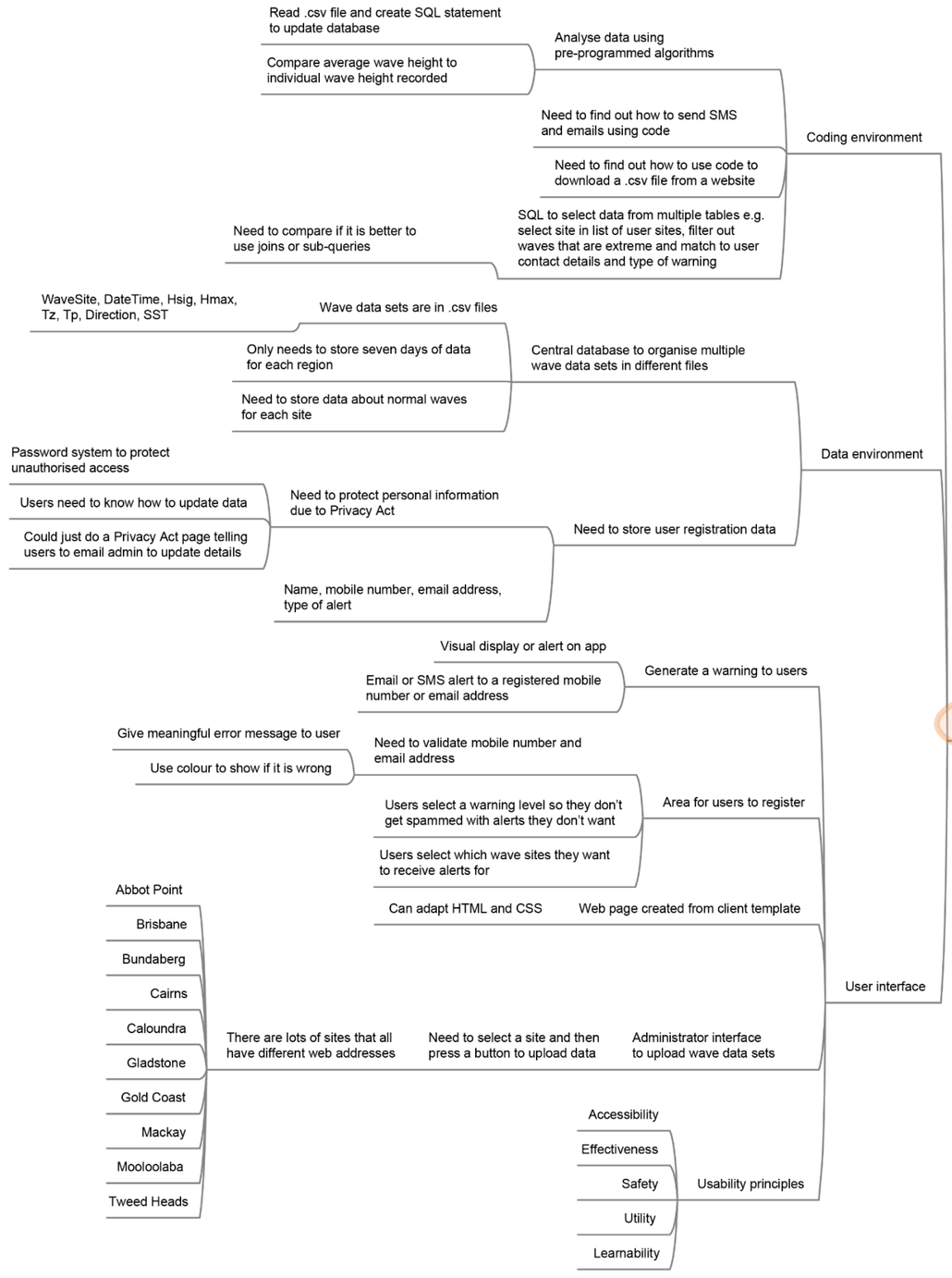
The annotations show the match to the instrument-specific marking guide performance-level descriptors.

Analysing [7–8]

Insightful analysis of the problem and relevant contextual information to identify the essential elements and features of user interface, data and programmed components and their relationships to the structure of the low-fidelity prototype digital solution.

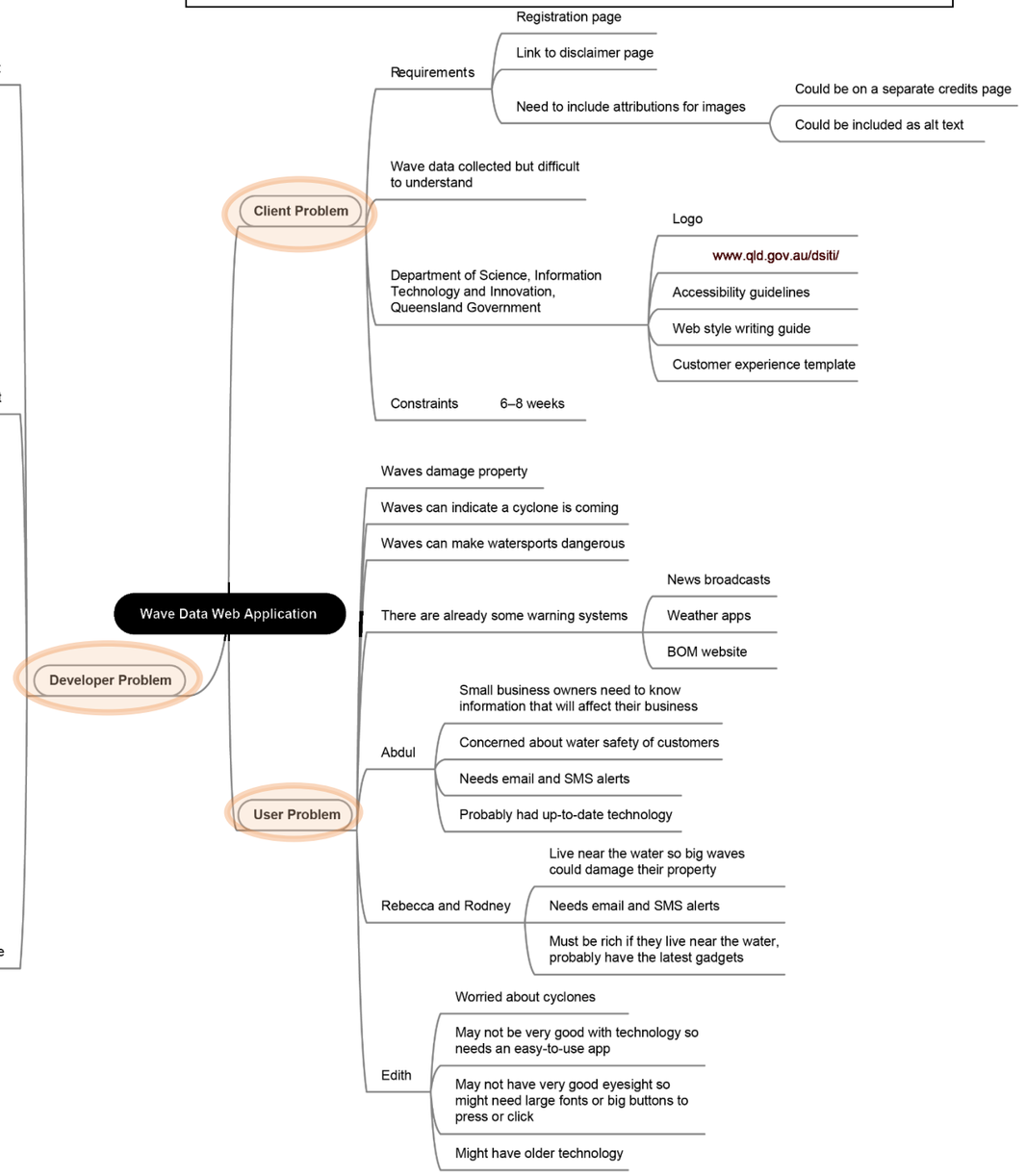
Astute determination of the user interface, data, programmed and solution requirements of the digital solution and essential prescribed and self-determined criteria.

Exploration of the problem to identify requirements and criteria



PLEASE NOTE: This response is completed within stipulated conditions.

- Pages 5–12 and 16–17 are the 8–10 A3 pages required. (Appendices on pages 18–19 are not included in the page count.)
- Pages 13–15 are the 4–6 A4 pages of code with annotations.



Retrieving and comprehending [7–8]

Accurate and discriminating recognition and discerning description of relevant programming elements, user-interface components and useability principles.

Adept symbolisation and discerning explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.

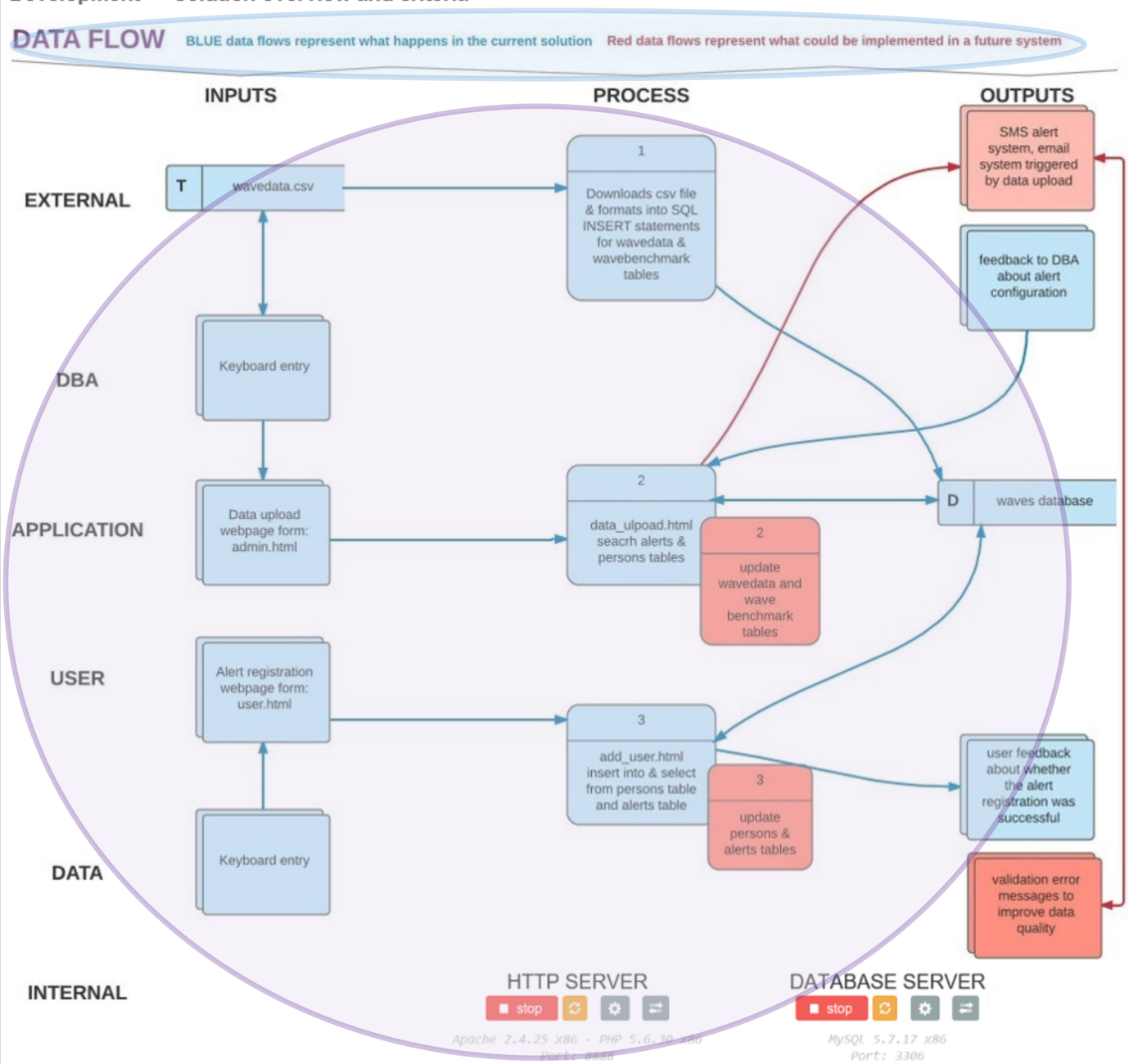
Analysing [7–8]

Astute determination of the user interface, data, programmed and solution requirements of the digital solution and essential prescribed and self-determined criteria.

Communicating [3–4]

Discerning decision-making about, and fluent use of written and visual features to communicate about a solution, language for a technical audience, grammatically accurate language structures and referencing and project conventions.

Development — Solution overview and criteria



Evaluation criteria - developed from the mind map and the technical proposal

PC (prescribed criteria)
SC (self-determined criteria)

- Must have 7 days of detailed wave data with fields: *Site, DateTime, Hsig (wave height), Hmax (max wave height), TZ, Tp, Direction, SST* (PC1)
- Must have normal wave height ranges for list of sites Abbot Point, Albatross Bay (Weipa), Brisbane, Bundaberg, Cairns, Caloundra, Emu Park, Gladstone, Gold Coast, Hay Point, Mackay, Mooloolaba, North Moreton Bay, Townsville, Tweed Heads (PC2)
- Must automate data upload for detailed wave data (PC3)
- An administrator can upload data to the site using a *.csv* file (PC4)
- A user can register details with the site (PC5)
  - … Must store contact details for users (PC5a)
  - … Must allow users to register for up to 3 specific wave sites (PC5b)
  - … Must allow users to register for a type of alert for each site (PC5c)
  - … An incorrect user registration is not stored in the database (PC5d)
- Appropriate alerts are emailed or texted to the user (PC6)
- Must validate user input for email address and/or mobile number (PC7)
- Must alert users based on user profile configuration (PC8)
- Impacts (evaluate) … personal, social, economic, legal, ethical (SC1)
  - … The website complies with the privacy act (PC9)
- Accuracy of code (SC2)
- Efficiency of code (SC3)
- Application useability from useability principles (accessibility, effectiveness, safety, utility, learnability) (SC4)
  - … The website complies with government website design standards (PC10)

Constraints

Completed within time budget
Compliant with useability principles
Password protection for user details and/or privacy policy
Disclaimer linked or added to user registration page

Prescribed criteria were constructed from analysis of the technical proposal and are indicated as PC. They are also numbered to allow indication of each throughout the response.
Self-determined criteria (SC) were constructed during problem analysis (mind mapping) and later throughout the life of the project.
Constraints were also determined from analysis of the task sheet and the technical proposal.
The data flow throughout the system is presented left. Inputs, data processes and outputs are indicated. The initial table *wavedata.csv* will be automatically downloaded from the government site and SQL statements will select data from this and add to a simple database. Data will also be added to the simple database from user and admin pages. A selection of user feedback messages are important to communicate with users about registration success, as well as the warning alerts themselves.

Synthesising and evaluating [9–10]

Coherent and logical synthesis of relevant information and ideas to determine data elements, user interface and programmed components for a digital solution.

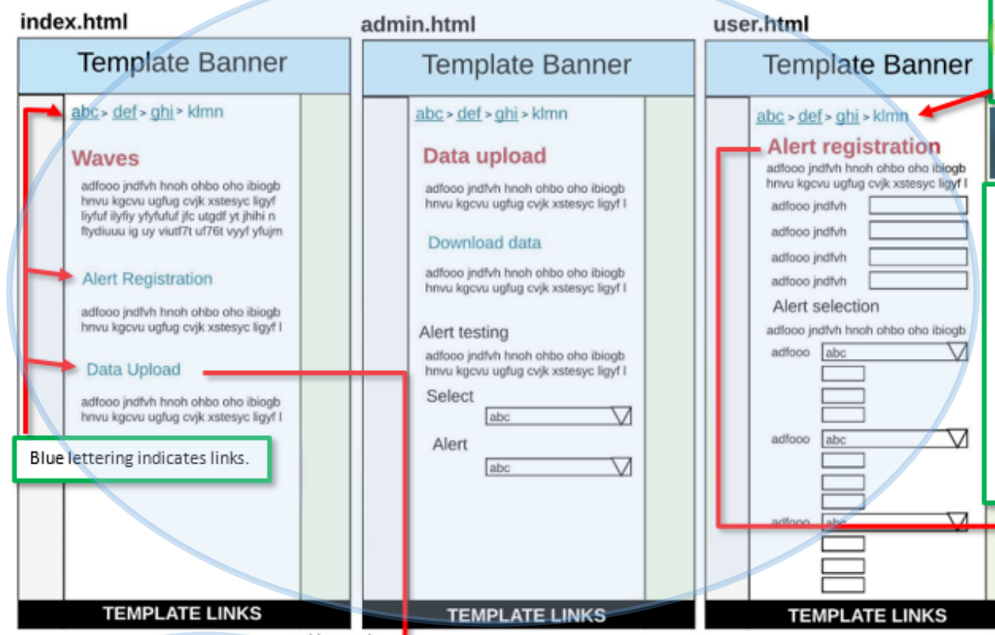
Purposeful generation of efficient user interface and programmed components of the digital solution

Retrieving and comprehending [7–8]

Accurate and discriminating recognition and discerning description of relevant programming elements, user-interface components and useability principles.

Adept symbolisation and discerning explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.

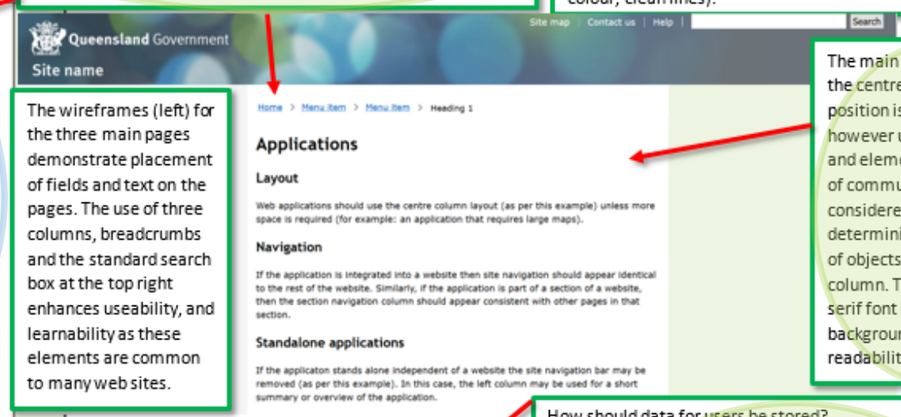
Development — User interface



Blue lettering indicates links.

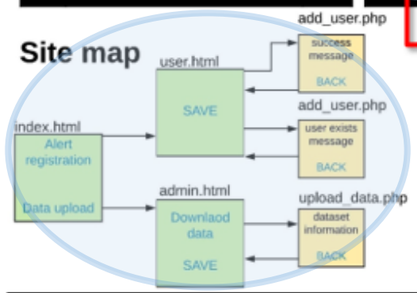
Breadcrumbs will link back to the launch site. No changes will be made to these, since they are consistent with many operating systems and web sites, users have seen this before. They support the useability principles of accessibility, learnability and utility (SC4).

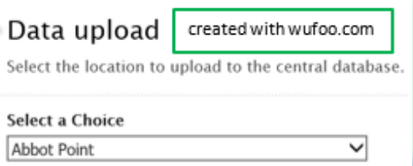
bluesteel xhtml-applications.html version was chosen as the user Interface template because this version would be best for the WaveData app as it is made for an application and includes elements that enhance useability (e.g. breadcrumbs, familiar, colour, clean lines).



The wireframes (left) for the three main pages demonstrate placement of fields and text on the pages. The use of three columns, breadcrumbs and the standard search box at the top right enhances useability, and learnability as these elements are common to many web sites.

The main webform will be in the centre column. The position is templated, however useability principles and elements and principles of communication must be considered when determining the placement of objects in the centre column. The use of clear serif font in black on a white background enhances readability.





created with wufoo.com

When admins click submit, there needs to be validation to check if the data has already been uploaded. Clicking submit will do three things:

1. upload the data to the database might be from the web address or from a .csv (file - yet to be determined)
2. trigger the function to work out which users need to be alerted
3. generate alerts for any users that are registered and need to receive an alert.

created with wufoo.com

How should data for users be stored?

Option 1:
Could be an email table and a mobile number table - the algorithm would go through all emails and then all the mobiles to send alerts.

Option 2:
Could be a table for each type of alert and then use wave height field to identify which table is used.

Option 3:
Put all the data in the one table.

When users submit the form needs to be validated:

- are any fields empty?
- does the email include an @ symbol?
- does the mobile number have the correct number of digits?
- does the user need to provide both email & mobile phone or is 1 optional?

Clicking submit will:

- check if the user already exists (2 people might have the same name but email and mobile phone will be unique)
- if the user doesn't exist add the record to the database (PC5d)
- if the user exists update the record in the database.

Disclaimer to include on home page i.e. Government design standards (PC10).

Wave monitoring disclaimer

These pages copyright ©2016 Queensland Government (Department of Science, Information Technology and Innovation).

The materials presented on this web site are distributed by the Queensland Government as an information source only.

While reasonable care and attention has been exercised in the collection and processing of this data it must be treated as unverified.

The State of Queensland makes no statements, representations or warranties about the accuracy or completeness of, and you should not rely on, any information contained on this web site. The State of Queensland disclaims all responsibility for information contained on this web site and all liability (including without limitation, liability in negligence) for all expenses, losses, damages and costs you may incur as a result of the information being inaccurate or incomplete in any way for any reason.

You should always check with the Bureau of Meteorology for the latest weather and coastal waters forecasts for your area before putting to sea.

From <environment.ehp.qld.gov.au/waves/>

The Site map indicates the projected relationship between each of the three HTML pages and PHP files, the code will be communicated on the code page later in the document. The wireframe shows the final placement of elements on each of the three pages within the chosen template (bluesteel xhtml-applications.html). In choosing this template useability principles accessibility, effectiveness, utility and learnability (SC4) were considered. The pastel colour choices of light green and grey and white point the user to the central column where the user must enter information. The banner at the top, which contains site map, contact, help and search links is consistent with many other pages and makes the page effective, with good utility (active links and search bar) and high learnability (common placement, colours and links). In the central column, a neutral colour white with black sans serif font was selected for readability and accessibility. The white background draws the user to the centre. The placement of breadcrumbs here was chosen once again to increase accessibility and utility and learnability. In the final version, the breadcrumbs should be changed to reflect more descriptive names to further aid the user experience.

The inclusion of the disclaimer on one of the pages is necessary (PC10) and placement of this on one of the pages (most likely the index page) will be decided later.

It will be important that user have the option to register for three alerts and the type of warning (cyclone, property damage or rough seas) and the type of alert (email or SMS) (PC5a-d), this will be determined later in the code and data storage components. The data upload section will most likely allow for the entire database (.csv) to be uploaded or a choice to upload individual site data this will be decided later. How data will be stored still has a few questions to be decided during data sheet development.

Synthesising and evaluating [9–10]

Critical evaluation of coded components and the digital solution against essential prescribed and self-determined criteria to make discerning refinements and astute recommendations justified by data.

Analysing [7–8]

Insightful analysis of the problem and relevant contextual information to identify the essential elements and features of user interface, data and programmed components and their relationships to the structure of the low-fidelity prototype digital solution.

Astute determination of the user interface, data, programmed and solution requirements of the digital solution and essential prescribed and self-determined criteria.

Retrieving and comprehending [7–8]

Adept symbolisation and discerning explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.

Development — User interface and data

Wave Data provided @ 14:35hrs on 11-06-2017

Sample *csv* file for sample data.

First row of every table can be ignored. Second row of table contains the column headings. Start importing from row 3.

Site	SiteNumber	Seconds	DateTime	Latitude	Longitude	Hsig	Hmax	Tp	Tz	SST	Direction
Tweed Heads	28	1.5E+09	2017-06-04T00:00:00	-28.1765	153.5759	1.4996	2.63	11.11	6.061	21.3	112.5
Tweed Heads	28	1.5E+09	2017-06-04T00:30:00	-28.1765	153.5759	1.5996	2.35	11.11	5.97	21.3	112.5
Tweed Heads	28	1.5E+09	2017-06-04T01:00:00	-28.1766	153.5759	1.5192	2.21	10.53	5.97	21.25	116.7
Tweed Heads	28	1.5E+09	2017-06-04T01:30:00	-28.1766	153.5759	1.3885	2.59	11.76	5.882	21.2	109.7
Tweed Heads	28	1.5E+09	2017-06-04T02:00:00	-28.1765	153.5759	1.5036	2.77	10	5.797	21.2	122.3

Why are seven (7) days of data required to be stored?

- If the latest record were used, this would be more efficient, unless the reading was incorrect
- Changes are likely to be gradual over several days
- There is a need to validate wave height of most recent wave against 7-day average to verify accuracy and ensure data integrity (data principle integrity)
- Impact on user could be critical if incorrect alerts are raised (useability principle-safety (SC4)). Recommend taking average of 7 days
- Solution is to take an average of the half hour readings over the last 24 hours, find the maximum height and compare to the 7-day average to raise alert.

Date and Time data is in one field separated by T. Need to convert into database field which separates date and time according to data format of yyyy/mm/dd hh:mm:ss i.e. replace T with space.

Latitude and Longitude are not required in the database since there is no need for a precise GPS location, site only can be used. Therefore, this is redundant data.

Data selection

Data types
Site: VARCHAR(200)
Hsig: FLOAT
Hmax: FLOAT
Tp: FLOAT
Tz: FLOAT
SST: FLOAT
Direction: FLOAT

SQL CREATE TABLE statement

The following SQL statement defines the "ID" column to be an auto-increment primary key field in the "persons" table. This code can be adapted to create a user registration table.

```
CREATE TABLE person (
  id integer NOT NULL UNIQUE
AUTO_INCREMENT,
  given_name varchar(50) NOT NULL,
  family_name varchar(50) DEFAULT NULL,
  mobile_number varchar(10) DEFAULT NULL,
  email_address varchar(254) DEFAULT NULL,
  primary key (id)
);
```

INSERT INTO *table_name*
VALUES (*value1, value2, value3, ...*);
Use this syntax to insert values into tables
From <www.w3schools.com/sql/sql_insert.asp

Developer environment for php/msql

- download DevServer install from www.easyphp.org
- use port 8888 for webserver and port 1111 for phpserver
- test php code with localhost:1111\ or localhost:8888\
- Phpmyadmin can be started from EasyPHP dashboard for initial setup.

MySQL uses the AUTO_INCREMENT keyword to perform an auto-increment feature. By default, the starting value for AUTO_INCREMENT is 1, and it will increment by 1 for each new record.

SQL AUTO INCREMENT a Field
www.w3schools.com/sql/sql_autoincrement.asp

The *wavedata.csv* file downloaded from the government site has several fields that are not required as well as a date & time field. Data importing should occur from row 3 as rows 1 and 2 are header rows only. The required fields can be extracted from *wavedata.csv* and using SQL, placed in a new database (waves) and a new table (*wave_data*) using the Site id (*site_id*) and the date & time (date_time) as a primary key. User can register their information (PC5a) and this can be stored in a table (person) using the *Auto Increment* command to assign a sequential identifier (*id*) as a primary key. Incorrect users cannot be stored in the database (PC5d) and so data validation must occur using email address or phone number (PC7 - which are unique identifiers for users). There are 16 fields required for a user to successfully register themselves for three sites, as well as another Auto increment field.

A decision has been made to calculate the long term average for each site based on the historical data provided on the website. Functionality will be implemented to read a historical file from the website, calculate the average and insert this value into the site (average_height) table. This historical data will be filtered to exclude values where *Hsig* is -99.9.

Data selection has been demonstrated by identifying data field types above (centre) and also in the entity relationship (ER) diagrams on the following pages (data).

Foreign key constraints have been used in the database to ensure data integrity.

- An alert record can only be inserted if the person_id value exists in the person table's id column and the site_id value exists in the sit table's id column.
- A wave_data record can only be inserted if the site_id value exists in the sit table's id column.

Synthesising and evaluating [9–10]

Critical evaluation of user experience against effective prescribed and self-determined criteria to make effective refinements and considered recommendations justified by data.

Analysing [7–8]

Astute determination of the user interface, data, programmed and solution requirements of the digital solution and essential prescribed and self-determined criteria.

Retrieving and comprehending [7–8]

Adept symbolisation and discerning explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.

Development — Data (early trial stage not continued)

SQL to create database

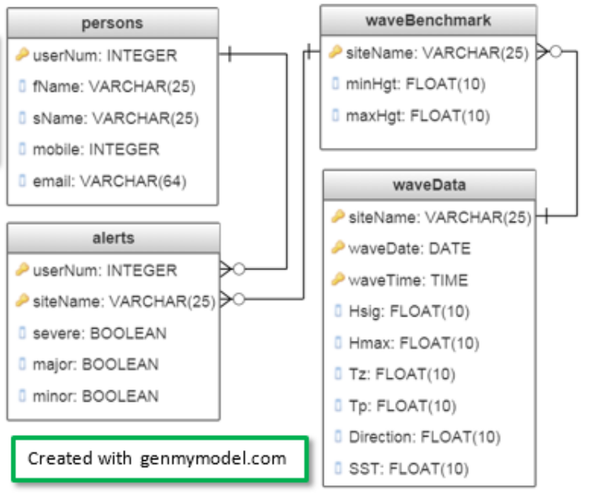
```
CREATEDB waves;

CREATE TABLE persons (
userNum: INT NOT NULL AUTO_INCREMENT,
fName: VARCHAR(25) NOT NULL,
sName: VARCHAR(25),
mobile: int,
Email: VARCHAR(255)
PRIMARY KEY userNum
)
```

```
INSERT INTO persons
(fname,sname,mobile,email) VALUES
('Abdul','Abdullah',0404123123,'abdul
@gmail.com.au');
```

Example insert statement
INSERT INTO wavedata VALUES
('Tweed Heads','2017-06-19','00:00:00',1.4566,2.4,9.09,5.556, 116.7,20.8);

SQL to select alerts to generate:

```
SELECT * FROM alerts a, persons p
WHERE a.userNum = p.personID
AND siteName="Brisbane"
AND severe=True;
```



Home > DSITI > Waves > Alert registration

Alert registration

Register to receive alerts when waves are unsafe.

First name — Field Number 1
Last name — Field Number 2
Email — Field Number 3
Mobile Number — Field Number 4

Alert selection

You can choose to receive warnings from up to 3 sites. What type of warnings do you want to receive?

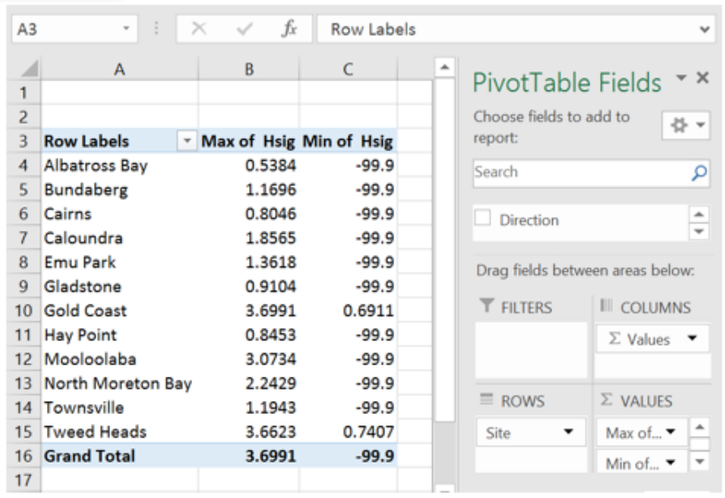
Select a site: Not selected — Field Number 5
Check All that Apply
Cyclone — Field Number 6
Property damage — Field Number 7
Rough seas — Field Number 8

Select a site: Not selected — Field Number 9
Check All that Apply
Cyclone — Field Number 10
Property damage — Field Number 11
Rough seas — Field Number 12

Select a site: Not selected — Field Number 13
Check All that Apply
Cyclone — Field Number 14
Property damage — Field Number 15
Rough seas — Field Number 16

Submit

Rate this page



Row Labels	Max of Hsig	Min of Hsig
Albatross Bay	0.5384	-99.9
Bundaberg	1.1696	-99.9
Cairns	0.8046	-99.9
Caloundra	1.8565	-99.9
Emu Park	1.3618	-99.9
Gladstone	0.9104	-99.9
Gold Coast	3.6991	0.6911
Hay Point	0.8453	-99.9
Mooloolaba	3.0734	-99.9
North Moreton Bay	2.2429	-99.9
Townsville	1.1943	-99.9
Tweed Heads	3.6623	0.7407
Grand Total	**3.6991**	**-99.9**

Originally, I could not determine a best method for arranging the data and so I had pursued a solution that required a user to manipulate the downloaded data in excel using pivot tables to take 7-day wave height averages and then save this data into the spreadsheet. This method is not automated and would require a person to manage the database on a daily basis. This would not then align to PC3 which requires automation of the updates. This idea represented on this page was then discarded and a new approach using SQL and php pages refining and simplifying code to automate this aspect of the database.

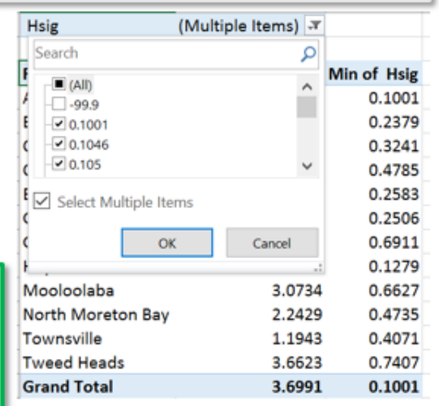
Code was implemented in PHP to process the csv files, validate the data and insert it into the database. This does not require any user intervention, apart from triggering the process. In future, this could be fully automated so that new data was automatically downloaded and processed at specified times or at a regular interval.

DEVELOPMENT OF EXCEL DATA COMPONENT CEASED.

Pivot table used in Excel to calculate initial Min and Max Wave heights for each site.
Some values are -99.9 showing the data needs to be cleaned.
Filter added for Hsig value to exclude -99.9 from aggregated min and max values.
Excel CONCATENATE function used to generate SQL INSERT commands:

```
INSERT INTO wavebenchmark VALUES ('Albatross Bay',0.1001,0.5384);
INSERT INTO wavebenchmark VALUES ('Bundaberg',0.2379,1.1696);
INSERT INTO wavebenchmark VALUES ('Cairns',0.3241,0.8046);
INSERT INTO wavebenchmark VALUES ('Caloundra',0.4785,1.8565);
INSERT INTO wavebenchmark VALUES ('Emu Park',0.2583,1.3618);
INSERT INTO wavebenchmark VALUES ('Gladstone',0.2506,0.9104);
INSERT INTO wavebenchmark VALUES ('Gold Coast',0.6911,3.6991);
INSERT INTO wavebenchmark VALUES ('Hay Point',0.1279,0.8453);
INSERT INTO wavebenchmark VALUES ('Mooloolaba',0.6627,3.0734);
INSERT INTO wavebenchmark VALUES ('North Moreton Bay',0.4735,2.2429);
INSERT INTO wavebenchmark VALUES ('Townsville',0.4071,1.1943);
INSERT INTO wavebenchmark VALUES ('Tweed Heads',0.7407,3.6623);
```

Note: min and max values in table are different order to pivot table so columns switched around in concatenate function.



	Min of Hsig
	0.1001
	0.2379
	0.3241
	0.4785
	0.2583
	0.2506
	0.6911
	0.1279
Mooloolaba	0.6627
North Moreton Bay	0.4735
Townsville	0.4071
Tweed Heads	0.7407
Grand Total	**0.1001**

Synthesising and evaluating [9–10]

Critical evaluation of user experience against effective prescribed and self-determined criteria to make effective refinements and considered recommendations justified by data.

Analysing [7–8]

Astute determination of the user interface, data, programmed and solution requirements of the digital solution and essential prescribed and self-determined criteria.

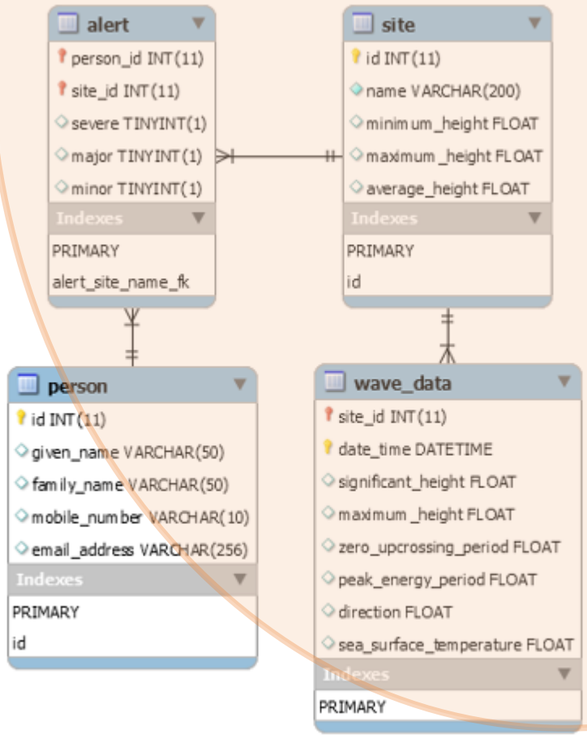
Development — Data

SQL to create database

```
create database 'waves';
create user waves identified by 'waves';
grant all on waves.* to waves;
```

SQL to insert data into site table

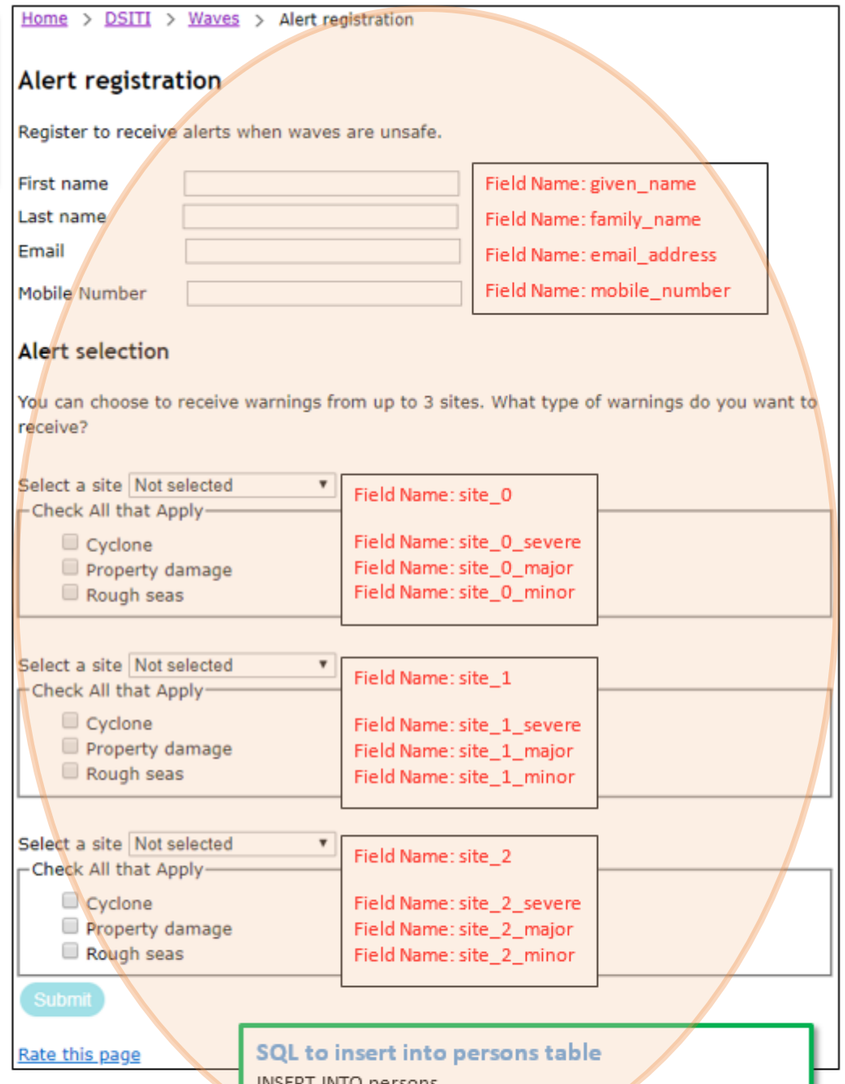
```
insert into site (name) values('Abbot Point');
insert into site (name) values('Albatross Bay (Weipa)');
insert into site (name) values('Brisbane');
insert into site (name) values('Bundaberg');
insert into site (name) values('Cairns');
insert into site (name) values('Caloundra');
insert into site (name) values('Emu Park');
insert into site (name) values('Gladstone');
insert into site (name) values('Gold Coast');
insert into site (name) values('Hay Point');
insert into site (name) values('Mackay');
insert into site (name) values('Mooloolaba');
insert into site (name) values('North Moreton Bay');
insert into site (name) values('Townsville');
insert into site (name) values('Tweed Heads');
```



SQL to create tables

```
CREATE TABLE alert (
  person_id integer NOT NULL,
  site_id integer NOT NULL,
  severe boolean DEFAULT NULL,
  major boolean DEFAULT NULL,
  minor boolean DEFAULT NULL,
  PRIMARY KEY (person_id, site_id)
);

CREATE TABLE person (
  id integer NOT NULL UNIQUE AUTO_INCREMENT,
  given_name varchar(50) NOT NULL,
  family_name varchar(50) DEFAULT NULL,
  mobile_number varchar(10) DEFAULT NULL,
  email_address varchar(254) DEFAULT NULL,
  primary key (id)
);

CREATE TABLE site (
  id integer NOT NULL UNIQUE AUTO_INCREMENT,
  name varchar(200) NOT NULL,
  average_height float DEFAULT NULL,
  PRIMARY KEY (id)
);

CREATE TABLE wave_data (
  site_id integer NOT NULL,
  date_time datetime NOT NULL,
  significant_height float DEFAULT NULL,
  maximum_height float DEFAULT NULL,
  zero_upcrossing_period float DEFAULT NULL,
  peak_energy_period float DEFAULT NULL,
  direction float DEFAULT NULL,
  sea_surface_temperature float DEFAULT NULL,
  PRIMARY KEY (site_id, date_time)
);

ALTER TABLE alert
  ADD CONSTRAINT alert_person_id_fk FOREIGN KEY
(person_id) REFERENCES person(id);

ALTER TABLE alert
  ADD CONSTRAINT alert_site_name_fk FOREIGN KEY
(site_id) REFERENCES site(id);

ALTER TABLE wave_data
  ADD CONSTRAINT wave_data_site_name_fk FOREIGN KEY
(site_id) REFERENCES site(id);
```

Version 1 only allowed alert type, not site type. Have updated user form to allow registration of three (3) sites as required by prescribed criteria 5b.



The SQL sets an alert when the user attempts to register twice.

SQL to insert into persons table

```
INSERT INTO persons
(given_name,family_email_address,mobile_number) VALUES
('Abdul','Abdullah','abdul@gmail.com.au',0404123123);
```

The ER diagram on the left shows the new database design to accommodate more useable fields and more descriptive naming conventions. SQL has been written to create the database and required tables and to insert data into the tables and add foreign key constraints.

Where possible consistent field names have been used in both the code and the database table columns.

Communicating [3–4]

Discerning decision-making about, and fluent use of written and visual features to communicate about a solution and language for a technical audience.

Retrieving and comprehending [7–8]

Adept symbolisation and discerning explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.

Development — Algorithms

Algorithm to Process Wave Alerts

SELECT from the wave_data and site tables, the maximum difference between the recorded wave height and the long-term average wave height for each site.
FOR each record:
    SET the warning level to none.
    IF the difference value is > 5 metres
        SET the warning level to severe.
    ELSE IF the difference value is > 3 metres
        SET the warning level to major.
    ELSE IF the difference value is >= 1 metre
        SET the warning level to minor.
    IF the warning level is not none
        SELECT from the alert table all records for this site
        FOR each alert record
            IF this warning level has been selected
                SEND a notification to the person

Databases are very efficient at querying large datasets. We should aim to return as small a record set as possible from our queries.

The provided algorithm was adapted to only request smaller datasets from the database. After review; it appeared to better suit this solution design.

Algorithm to Process Wave Alerts

SELECT the person's record from the person table
SELECT the site's record from the site table
IF the person record contains an email address
    FORMAT and send a notification email
IF the person record contains mobile number
    FORMAT and send a notification SMS

Different message formats should be used for email and SMS notifications. SMS notifications need to be concise to suit the SMS size restrictions. Email messages are not size restricted and could contain additional information.

The algorithms presented describe the required logic in a technology independent manner. When these algorithms were implemented in PHP it was possible to utilise some functionality provided by PHP to simplify the required code. For example, PHP provides the fgetcsv() function which will read each line from a file and parse the comma separated fields. When writing code we should always use system proved functions were possible instead of writing code.

Algorithm to Add Users

GET all fields FROM user form
VALIDATE the data
IF Invalid
    REDISPLAY the form with the entered data and validation messages
ELSE
    Check if Persons with Mobile or Email exists in DATABASE
    IF more than one Person exists
        DISPLAY "A informational message to the user"
    ELSE IF one Person exists
        UPDATE the person record with the entered details
        DELETE the user's existing alerts
        INSERT the entered alerts
        SEND a Registration Updated notification to the user
    ELSE
        INSERT a Person record
        INSERT the entered alerts
        SEND a Registration Created notification to the user

All the user entered data should be validated initially so that all validation errors are displayed to the user.

Algorithm to Validate Form Data

Given name field must be provided
Family name must be provided
A valid email address or a valid mobile number must be provided
At least one valid alert must be provided

Algorithm to Validate Alert

If a site is selected and no alerts are selected, it is invalid
If alerts are selected and no site is selected it is invalid.

Algorithm to Validate Mobile Number

COUNT the number of numeric characters
COUNT the number of non-numeric characters (exclude spaces)
If the numeric character count is not 10, it is invalid
If the non-numeric count is non-zero, it is invalid

Phone numbers need to be processed as a string of numeric characters not as a number.

Algorithm to read records from waves.csv file and store them in the database table

The csv file may be uploaded from the user or downloaded from the website.
If selected file exists:
    If selected file is of type 'text':
        If selected file contains > 2 line of data:
            Skip initial 2 lines at beginning of text file
        While not end of file
            Read next line of file
            Split line based on commas
            If split line contains 12 parts
                Find site record where name equals first field
                If not found
                    Record site name
                    Skip record
                Validate field 4 as a date time value
                If not valid
                    Record error
                    Skip record
                Validate fields 7 to 12 as numeric values
                If any not valid
                    Record error
                    Skip record
                Insert record into wave data table
                If rejected as duplicate
                    Record error
                    Skip record
                Increment site record count
        Display any errors to user
        Display site record counts to user

Data is first validated against recognised features: number of fields and field data types.

Research into comma-separated file input showed that the split function was commonly used in most languages to separate each line into elements of data. Other methods such as line parsing were considered but were deemed to be inefficient.

It is important to inform the user of any errors found and how many records have been added. The user can then take any necessary action to deal with data quality issues.

Communicating [3–4]

Discerning decision-making about, and fluent use of written and visual features to communicate about a solution, language for a technical audience, grammatically accurate language structures and referencing and project conventions.

Analysing [7–8]

Insightful analysis of the problem and relevant contextual information to identify the essential elements and features of user interface, data and programmed components and their relationships to the structure of the low-fidelity prototype digital solution.

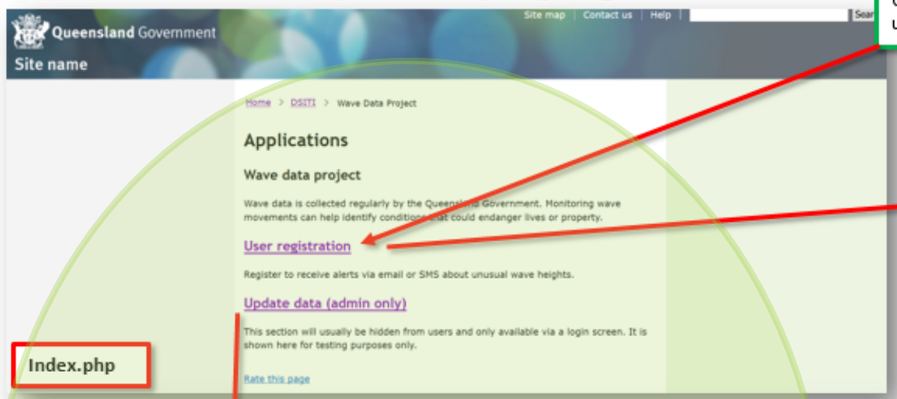
Synthesising and evaluating [9–10]

Coherent and logical synthesis of relevant information and ideas to determine data elements, user interface and programmed components for a digital solution.

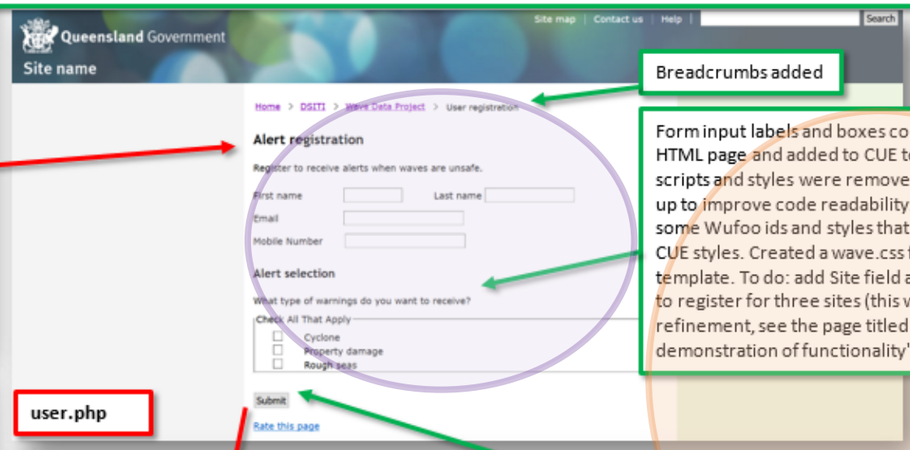
Critical evaluation of coded components and the digital solution against essential prescribed and self-determined criteria to make discerning refinements and astute recommendations justified by data.

Communicating [3–4]

Generation — User interface (HTML code) early stages to be refined

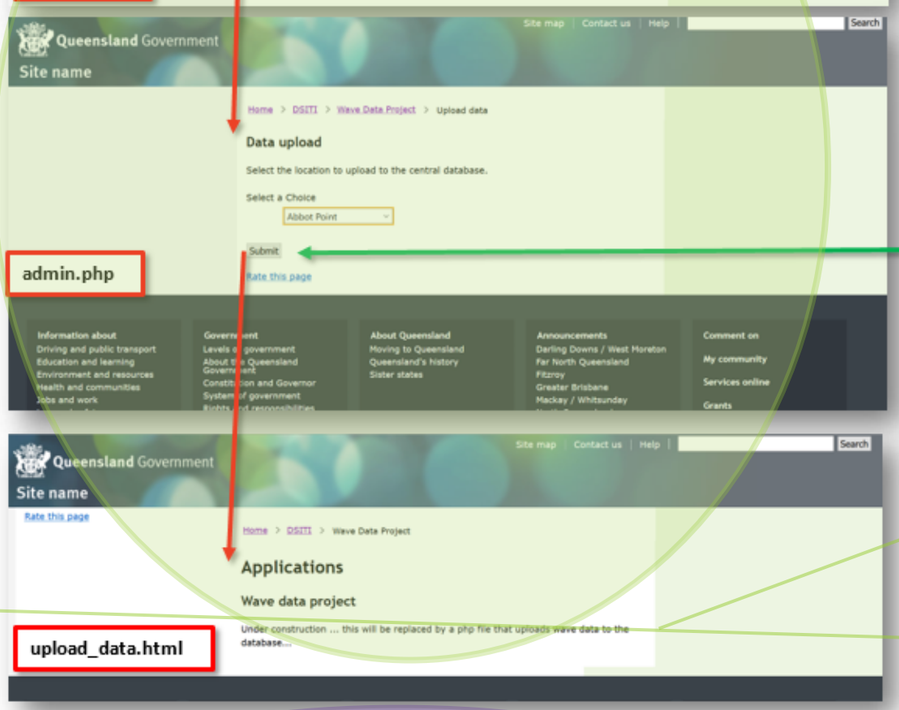


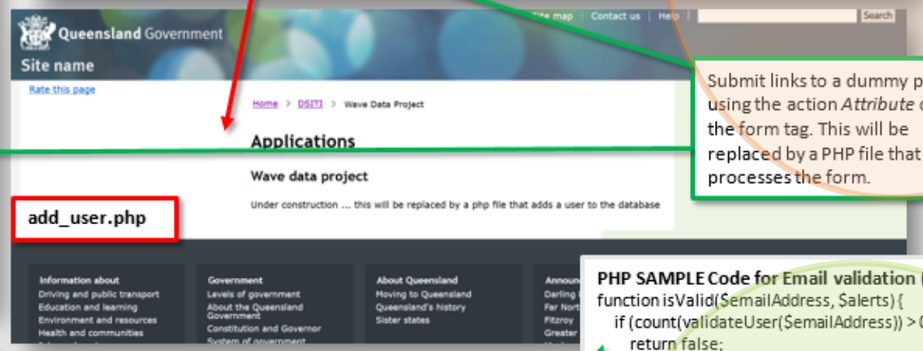
User registration will be changed to Alert registration. This will direct a user to a page where they can either register as a new user using their email as a username and nominate a password or be directed to the alert page after entering email and password.



Breadcrumbs added

Form input labels and boxes copied from wufoo, HTML page and added to CUE template. Wufoo scripts and styles were removed and form code tidied up to improve code readability. Needed to change some Wufoo ids and styles that caused conflicts with CUE styles. Created a wave.css file and linked to the template. To do: add Site field and offer opportunity to register for three sites (this was added later during refinement, see the page titled "Generation - demonstration of functionality").





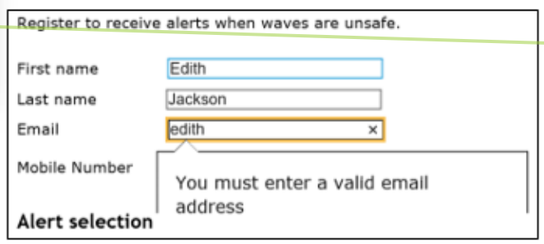
Submit links to a dummy page using the action *Attribute* of the form tag. This will be replaced by a PHP file that processes the form.

This is a working prototype with all breadcrumbs, form inputs and links active. Email validation is the only validation working (see below). The PHP code (right) will be adjusted to validate other fields in the final solution.

Sample HTML code for breadcrumbs

```
<div id="breadcrumbs">
  <h2>You are here:</h2>
  <ol>
    <li><a href="index.php">Home</a></li>
    <li><a href="index.php">DSITI</a></li>
    <li><a href="index.php">Waves</a></li>
    <li class="last-child">Alert registration</li>
  </ol>
```

Although it appears as a vertical list in HTML it is styled with the breadcrumbs id descriptor within the CUE CSS style sheet to appear in a horizontal layout as shown on the screen clippings.



Useability efficiency: tabindex attribute used to control tab order through the fields.

PHP SAMPLE Code for Email validation (to be further refined)

```
function isValid($emailAddress, $alerts) {
  if (count(validateUser($emailAddress)) > 0) {
    return false;
  }
  if (count(validateAlerts($alerts)) > 0) {
    return false;
  }
  foreach ($alerts as $alert) {
    if (count(validateAlert($alert)) > 0) {
      return false;
    }
  }
  return true;
}
function validateUser($emailAddress) {
  // Given name and family name are mandatory and either email address or mobile number must be supplied
  $errors = array();
  if (($emailAddress !== null && trim($emailAddress) !== '')) {
    $errors = array_merge($errors, validateEmailAddress($emailAddress));
  }
  if (($emailAddress === null || trim($emailAddress) === '')) {
    $errors[] = 'Please enter your email address';
  }
  return $errors;
}
```

Discerning decision-making about, and fluent use of written and visual features to communicate about a solution, grammatically accurate language structures and referencing and project conventions.

Retrieving and comprehending [7–8]

Adept symbolisation and discerning explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.

Synthesising and evaluating [9–10]

Purposeful generation of efficient user interface and programmed components of the digital solution

Generation — Code

Code for user registration (user.php)

```
<!DOCTYPE html>
<?php
  $pageTitle = "Waves: Alert Registration";

  require_once('inc/header.inc');

  require_once('inc/database.php');
  require_once('inc/validation.php');
  require_once('inc/notification.php');
?>
<div id="page-container">
  <div class="max-width">
    <div id="content-container">
      <div id="breadcrumbs">
        <h2>You are here:</h2>
        <ol>
          <li><a href="index.php">Home</a></li>
          <li><a href="index.php">DSITI</a></li>
          <li><a href="index.php">Waves</a></li>
          <li class="last-child">Alert registration</li>
        </ol>
      </div>
      <div id="content">
        <div class="article">
          <div class="box-sizing">
          <?php
  // Flag used to check if the user has successfully registered
            $registrationOutcome = null;
            if ($_SERVER['REQUEST_METHOD'] === 'GET') {
        // Set up our initial data
            $givenName = "";
            $familyName = "";
            $emailAddress = "";
            $mobileNumber = "";
            $alerts = array();
            for ($s = 1; $s <= $maxSites; $s++) {
              $alerts[] = array('site_id' => 'ns', 'severe' =>
false, 'major' => false, "minor" => false);
            }
          }
          if ($_SERVER['REQUEST_METHOD'] === 'POST')
{
        // Get the form data
            $givenName = $_POST['given_name'];
            $familyName = $_POST['family_name'];
            $emailAddress = $_POST['email_address'];
            $mobileNumber =
$_POST['mobile_number'];
            for ($s = 0; $s < $maxSites; $s++) {
              $alerts[$s]['site_id'] = $_POST["site_${s}"];
              $alerts[$s]['severe'] =
isset($_POST["site_${s}_severe"]);
              $alerts[$s]['major'] =
isset($_POST["site_${s}_major"]);
              $alerts[$s]['minor'] =
isset($_POST["site_${s}_minor"]);
            }
            $validationErrors = array();
            // Validate the form data
            if (isValid($givenName, $familyName,
$emailAddress, $mobileNumber, $alerts)) {
        // Form data is valid, see if the user exists
            if ($mobileNumber !== null) {
        // Remove any entered spaces from the mobile number
              $mobileNumber = str_replace(' ', '',
$mobileNumber);
            }
            $user = null;
            try {
              $user =
getUserWithEmailOrMobile($emailAddress,
$mobileNumber);
            } catch (Exception $e) {
              $validationErrors[] = $e->getMessage();
            }
            try {
              if ($user !== null) {
        // Is an existing user, update the person record
                updateUser($user['id'], $givenName,
$familyName, $emailAddress, $mobileNumber);
        // Delete the existing alerts
                deleteAlerts($user['id']);
        // Add the new alerts
                foreach ($alerts as $alert) {
                  if ($alert['site_id'] != 'ns') {
                    addAlert($user['id'],
$alert['site_id'], $alert['severe'], $alert['major'],
$alert['minor']);
                  }
                }

sendUpdatedRegistrationNotification($user['id']);
                $registrationOutcome = 'updated';
              } else {
        // Is a new user, create the user and alert records
and send registration email
                $personId = addUser($givenName,
$familyName, $emailAddress, $mobileNumber);
                foreach ($alerts as $alert) {
                  if ($alert['site_id'] != 'ns') {
                    addAlert($personId,
$alert['site_id'], $alert['severe'], $alert['major'],
$alert['minor']);
                  }
                }

sendInitialRegistrationNotification($personId);
                $registrationOutcome = 'created';
              }
            } catch (Exception $e) {
              $validationErrors[] = $e->getMessage();
            }
          }
        ?>
        <?php
          if ($registrationOutcome == 'created') {
        ?>
          <section class="successful-registration">
            <p>Your registration has been successful and
a confirmation message has been sent to you.</p>
            <h2><a href="index.php">Return to the
home page</a></h2>
          </section>
        <?php } else if ($registrationOutcome ==
'updated') { ?>
          <section class="successful-registration">
            <p>Your registration has been updated and
confirmation message has been sent to you.</p>
            <h2><a href="index.php">Return to the
home page</a></h2>
          </section>
        <?php } else { ?>
          <!-- Show user registration details -->
          <form name="registration" action="user.php"
method="post">
            <header>
              <h2>Alert registration</h2>
              <p>Register to receive alerts when waves
are unsafe. </p>
            </header>
            <section class="user-details">
              <label for="given_name">Given
name</label>
              <input id="given_name"
name="given_name" tabindex="1" class="field text fn"
type="text" size="25" maxlength="100" required
value="<?php echo $givenName; ?>">
              <br>
              <label for="family_name">Family
name</label>
              <input id="family_name"
name="family_name" tabindex="2" class="field text ln"
type="text" size="25" maxlength="100" required
value="<?php echo $familyName; ?>">
              <br>
              <label for="email_address">Email
address</label>
              <input id="email_address" tabindex="3"
class="field text medium" spellcheck="false" type="email"
size="25" maxlength="2000" value="<?php echo
$emailAddress; ?>">
              <br>
              <label for="mobile_number">Mobile
Number</label>
              <input id="mobile_number" tabindex="4"
class="field text medium" type="tel" size="25"
maxlength="10" value="<?php echo $mobileNumber; ?>">
            </section>
            <?php
              if ($_SERVER['REQUEST_METHOD'] ===
'POST') {
        // Display the user validation errors
              $errors = validateUser($givenName,
$familyName, $emailAddress, $mobileNumber);
              showAnyErrors($errors);
            }
          ?>
          <section>
            <h3>Alert selection </h3>
            <?php
              echo "<div>You can choose to receive
warnings from up to $maxSites sites. What type of warnings
do you want to receive?</div>";
              $sites = getSites();
              $tabIndex = 5;
              for($s = 0; $s < $maxSites; $s++) {
              ?>
              <div class="siteselect">
                <?php
                $tabIndex++;
                echo "<label for=\"site_$s\">Select a
site </label>";

                echo "<select id=\"site_$s\"
name=\"site_$s\" tabindex=\"$tabIndex\">";
                $selected = $alerts[$s]['site_id'] ===
'ns' ? 'selected' : '';
                echo "<option $selected
value=\"ns\">Not selected</option>";
                foreach ($sites as $site) {
                  $selected = $alerts[$s]['site_id'] ==
$site['id'] ? 'selected' : '';
                  $value = $site['id'];
                  $label = $site['name'];
                  echo "<option $selected
value=\"$value\">$label</option>";
                }
                ?>
              </select>
            </div>
            <fieldset>
              <legend>Check All that Apply</legend>
              <span class="subfield">
              <?php
                $id = "site_{$s}_severe";
                $tabIndex++;
                echo "<input id=\"$id\" name=\"$id\"
tabindex=\"$tabIndex\" type=\"checkbox\"";
                if ($s < count($alerts) &&
$alerts[$s]['severe']) {echo "checked";}
                echo ">";
                    echo "<label
for=\"$id\">Cyclone</label>";
              ?>
              </span>
              <br>
              <span class="subfield">
              <?php
                $id = "site_{$s}_major";
                $tabIndex++;
                echo "<input id=\"$id\" name=\"$id\"
tabindex=\"$tabIndex\" type=\"checkbox\"";
                if ($s < count($alerts) &&
$alerts[$s]['major']) {echo "checked";}
                echo ">";
                echo "<label for=\"$id\">Property
damage</label>";
              ?>
              </span>
              <br>
              <span class="subfield">
              <?php
                $id = "site_{$s}_minor";
                $tabIndex++;
```

The same page is used for data entry and results. If any validation errors are detected the page and entered data are re-displayed so the user does not need to re-enter the data.

POST is always used to submit form data. It is also more secure than GET as POST requests do not remain in the browser history (w3schools.com, 2017).

The $registrationOutcome flag is used to determine what should be displayed to the user.

The alert selection html is created in a for loop. It only needs to be written once.

The list of sites for the <select> is retrieved from the database. If an additional site is added we do not need to change the code.

Database access, validation and notification functionality has been implemented as separate functions. This allows for reuse and improved maintainability.

Analysing [7–8]

Insightful analysis of the problem and relevant contextual information to identify the essential elements and features of user interface, data and programmed components and their relationships to the structure of the low-fidelity prototype digital solution.

```
                    echo "<input id=\"$id\" name=\"$id\"
tabindex=\"$tabIndex\" type=\"checkbox\"";
                    if ($s < count($alerts) &&
$alerts[$s]['minor']) {echo "checked";}
                    echo ">";
                    echo "<label for=\"$id\">Rough
seas</label>";
                ?>
                        </span>
                <?php
                    if ($_SERVER['REQUEST_METHOD']
=== 'POST') {
            // If a site has been selected, validate that at least
one alert type has been specified
                    $errors = validateAlert($alerts[$s]);
                    showAnyErrors($errors);
                }
                ?>
            </fieldset>
            <?php
            }
            if ($_SERVER['REQUEST_METHOD'] ===
'POST') {
        // Validate that at least one site has been selected
                $errors = validateAlerts($alerts);
                showAnyErrors($errors);
        // Display any other validation errors
                showAnyErrors($validationErrors);
            }
            ?>
        </section>
        <div>
        <?php
            $tabIndex++;
```

Display any validation errors for this alert in the <fieldset>.

Show any other validation errors after the last alert <fieldset>

```
                echo "<input id=\"$id\" name=\"$id\"
tabindex=\"$tabIndex\" class=\"action-button\"
id=\"saveForm\" type=\"submit\" value=\"Submit\">";
                ?>
                </div>
            </form>
        <?php } ?>
        </div>
    </div>
</div>
<div id="page-feedback">
    <form action="" method="post">
        <h2>Rate this page</h2>
        <ol class="questions">
            <li class="select1">
                <fieldset>
                    <strong>How useful was the information
on this page?</strong>
                    <ol class="options">
                        <li><label for="rating-high"><input
type="radio" name="rating" id="rating-high" value="high">
Very useful</label></li>
                        <li><label for="rating-medium"><input
type="radio" name="rating" id="rating-medium"
value="medium"> Somewhat useful</label></li>
                        <li><label for="rating-low"><input
type="radio" name="rating" id="rating-low" value="low">
Not very useful</label></li>
                    </ol>
                </fieldset>
            </li>
            <li class="textarea">
                <label>Other feedback</label>
                <textarea name="feedback" id="feedback"
cols="50" rows="7"></textarea>
            </li>
        </ol>
        <div class="actions">
            <input class="primary-action" type="submit"
value="Submit feedback">
        </div>
    </form>
</div>
</div>
</div>

<?php
require_once('inc/footer.inc');
?>
```

PHP Code for process data function (process.php)

```
<?php
require_once('inc/config.php');

// establish a link to the waves database
$link = mysqli_connect($dbHost, $dbUser, $dbPassword,
$dbDatabase);

/**
 * Get the sites
 * @return array of sites
 */
function getSites() {
    $stmt = $GLOBALS['link']->prepare("SELECT * FROM site
ORDER BY name");
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $result = $stmt->get_result();
    $sites = array();
    while ($row = $result->fetch_assoc()) {
        $sites[] = array('id' => $row['id'], 'name' =>
$row['name']);
    }
    return $sites;
}

/**
 * @param $siteName
 * @return null
 * @throws Exception
 */
function getSiteWithName($siteName) {
    $stmt = $GLOBALS['link']->prepare("SELECT * FROM site
where name = ?");
    $stmt->bind_param('s', $siteName);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $result = $stmt->get_result();
    $stmt->close();

    if ($result->num_rows > 1) {
        throw new Exception("Multiple site records found for
site name \"$siteName\"");
    }

    if ($result->num_rows == 1) {
        while ($row = $result->fetch_assoc()) {
            return $row;
        }
    }
    return null;
}

/**
 * @param $siteId
 * @return null
 * @throws Exception
 */
function getSiteWithId($siteId) {
    $stmt = $GLOBALS['link']->prepare("SELECT * FROM site
where id = ?");
    $stmt->bind_param('i', $siteId);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $result = $stmt->get_result();
    $stmt->close();

    if ($result->num_rows > 1) {
        throw new Exception("Multiple site records found for
site id $siteId");
    }

    if ($result->num_rows == 1) {
        while ($row = $result->fetch_assoc()) {
            return $row;
        }
    }
    return null;
}
```

The database operations are implemented as independent functions.

The SQL uses bound variables instead of concatenating the SQL and variables into a string. This improves security by preventing SQL injection attacks. https://www.owasp.org/index.php/SQL_Injection

```
/**
 * Get the person record from the database
 * @param $emailAddress the email address of the user
(optional)
 * @param $mobileNumber the mobile number of the
person (optional)
 * @return the person record or null if not found
 * @throws Exception if email address and mobile number
belong to different people
 */
function getUserWithEmailOrMobile($emailAddress,
$mobileNumber) {
    $emailUser = !empty($emailAddress) ?
getUserWithEmail($emailAddress) : null;
    $mobileUser = !empty($mobileNumber) ?
getUserWithMobile($mobileNumber) : null;

    if ($emailUser !== null && $mobileUser !== null) {
        // Has matched on both email address and mobile
number
        if ($emailUser['id'] !== $mobileUser['id']) {
            // These are different people
            throw new Exception("Email address and mobile
number are used by different people");
        }
        return $emailUser;
    }

    // Has matched on email address
    if ($emailUser !== null) {
        return $emailUser;
    }

    // Has matched on mobile number
    if ($mobileUser !== null) {
        return $mobileUser;
    }

    // No match
    return null;
}

/**
 * Get the person record with the specified email
 * @param $emailAddress email_address of the required
person
 * @return the person record or null if not found
 * @throws Exception if more than one person is found
 */
function getUserWithEmail($emailAddress) {
    $stmt = $GLOBALS['link']->prepare("SELECT * FROM
person WHERE email_address = ?");
    $stmt->bind_param('s', $emailAddress);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $result = $stmt->get_result();
    $stmt->close();

    if ($result->num_rows > 1) {
        throw new Exception("This email address is associated
with multiple people");
    }

    if ($result->num_rows == 1) {
        while ($row = $result->fetch_assoc()) {
            return $row;
        }
    }
    return null;
}

/**
 * Get the person record with the specified mobile number
 * @param $mobileNumber mobile_number of the required
person
```

Retrieving and comprehending [7–8]

Adept symbolisation and discerning explanation of algorithms and relevant programming information and ideas, data structures and interrelationships between user experiences and data of the digital prototype.

```
 * @return the person record or null if not found
 * @throws Exception if more than one person is found
 */
function getUserWithMobile($mobileNumber) {
    $stmt = $GLOBALS['link']->prepare("SELECT * FROM
person WHERE mobile_number = ?");
    $stmt->bind_param('s', $mobileNumber);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $result = $stmt->get_result();
    $stmt->close();

    if ($result->num_rows > 1) {
        throw new Exception("This mobile number is associated
with multiple people");
    }

    if ($result->num_rows == 1) {
        while ($row = $result->fetch_assoc()) {
            return $row;
        }
    }
    return null;
}

/**
 * Get the user record for the specified id
 * @param $personId
 * @return null
 * @throws Exception
 */
function getUserWithId($personId) {
    $stmt = $GLOBALS['link']->prepare("SELECT * FROM
person WHERE id = ?");
    $stmt->bind_param('i', $personId);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $result = $stmt->get_result();
    $stmt->close();

    if ($result->num_rows > 1) {
        throw new Exception("This id is associated with multiple
people");
    }

    if ($result->num_rows == 1) {
        while ($row = $result->fetch_assoc()) {
            return $row;
        }
    }
    return null;
}

/**
 * Get the alerts for the specified user
 *
 * @param $personId
 * @return array
 * @throws Exception
 */
function getUserAlerts($personId) {
    $stmt = $GLOBALS['link']->prepare("SELECT a.* FROM
alert a WHERE person_id = ?");
    $stmt->bind_param('i', $personId);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $result = $stmt->get_result();
    $alerts = array();
    while ($row = $result->fetch_assoc()) {
        $alerts[] = $row;
    }
    $stmt->close();
    return $alerts;
}

/**
 * Get the alerts for the specified user
 *
 * @param $siteId
 * @return array
 * @throws Exception
 */
function getSiteAlerts($siteId) {
    $stmt = $GLOBALS['link']->prepare("SELECT a.* FROM
alert a WHERE site_id = ?");
    $stmt->bind_param('i', $siteId);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $result = $stmt->get_result();
    $alerts = array();
    while ($row = $result->fetch_assoc()) {
        $alerts[] = $row;
    }
    $stmt->close();
    return $alerts;
}

/**
 * Create a new user record
 * @param $givenName
 * @param $familyName
 * @param $emailAddress
 * @param $mobileNumber
 * @return the id of the inserted record
 * @throws Exception
 */
function addUser($givenName, $familyName,
$emailAddress, $mobileNumber) {
    $stmt = $GLOBALS['link']->prepare("INSERT INTO
person(given_name, family_name, email_address,
mobile_number) VALUES(?,?,?,?)");
    $stmt->bind_param('ssss', $givenName, $familyName,
$emailAddress, $mobileNumber);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $personId = $stmt->insert_id;
    $stmt->close();
    return $personId;
}

/**
 * Update the person_record
 * @param $personId id of the record to be updated
 * @param $givenName
 * @param $familyName
 * @param $emailAddress
 * @param $mobileNumber
 * @throws Exception
 */
function updateUser($personId, $givenName, $familyName,
$emailAddress, $mobileNumber) {
    $stmt = $GLOBALS['link']->prepare("UPDATE person SET
given_name=?, family_name=?, email_address=?,
mobile_number=? WHERE id = ?");
    $stmt->bind_param('ssssi', $givenName, $familyName,
$emailAddress, $mobileNumber, $personId);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $stmt->close();
}

/**
 * Add an alert record
 * @param $personId the id of the person
 * @param $siteId the id of the selected site
 * @param $severe
 * @param $major
 * @param $minor
 * @throws Exception
 */
function addAlert($personId, $siteId, $severe, $major,
$minor) {
    $stmt = $GLOBALS['link']->prepare("INSERT INTO
alert(person_id, site_id, severe, major, minor)
VALUES(?,?,?,?,?)");
    $stmt->bind_param('iiiii', $personId, $siteId, $severe,
$major, $minor);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $stmt->close();
}

/**
 * Delete all alerts for the specified user
 * @param $personId
 * @throws Exception
 */
function deleteAlerts($personId) {
    $stmt = $GLOBALS['link']->prepare("DELETE FROM alert
WHERE person_id = ?");
    $stmt->bind_param('i', $personId);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $stmt->close();
}

/**
 * Insert a wave_data record into the database
 * @param $siteId
 * @param $dateTime
 * @param $significantHeight
 * @param $maximumHeight
 * @param $peakEnergyPeriod
 * @param $zeroUpcrossingPeriod
 * @param $seaSurfaceTemperature
 * @param $direction
 * @throws Exception if an error occurs
 */
function addData($siteId, $dateTime, $significantHeight,
$maximumHeight, $peakEnergyPeriod,
$zeroUpcrossingPeriod, $seaSurfaceTemperature,
$direction) {
    $stmt = $GLOBALS['link']->prepare(
        "INSERT INTO " .
        "wave_data(site_id, date_time, significant_height,
maximum_height, zero_upcrossing_period,
peak_energy_period, sea_surface_temperature, direction)
" .
        "VALUES(?,?,?,?,?,?,?,?)");

    $stmt->bind_param('isddddddd', $siteId, $dateTime,
$significantHeight, $maximumHeight, $peakEnergyPeriod,
$zeroUpcrossingPeriod, $seaSurfaceTemperature,
$direction);
    if (!$stmt->execute()) {
        error_log($stmt->error);
        throw new Exception($stmt->error);
    }
    $stmt->close();
}

/**
 * Get the maximum difference between the
significant_height value and the average for each site since
the specified date
 * @param $sinceDateTime datetime
 * @return array
 * @throws Exception
 */
function getSiteMaxHeightDifference($sinceDateTime) {
    $stmt = $GLOBALS['link']->prepare(
        "select w.site_id, max(w.significant_height -
s.average_height) diff " .
        "from wave_data w " .
        "join site s on s.id = w.site_id " .
        "where w.date_time > ? " .
        "and s.average_height is not null " .
        "group by w.site_id");

    // The datetime needs to be a string
    $dateTimeStr = $sinceDateTime->format("Y-m-d H:i:s");
    $stmt->bind_param('s', $dateTimeStr);
```

CODE HAS BEEN CLIPPED HERE BUT CAN BE SEEN IN THE VIDEO

Communicating [3–4]

Discerning decision-making about, and fluent use of written and visual features to communicate about a solution, language for a technical audience, grammatically accurate language structures and referencing and project conventions.

Synthesising and evaluating [9–10]

Critical evaluation of impacts and coded components and the digital solution against essential prescribed and self-determined criteria to make discerning refinements and astute recommendations justified by data

Evaluation and References— Ongoing and impacts

Includes accessibility checklist
PC (prescribed criteria)
SC (self-determined criteria)

☒ Must have 7 days of detailed wave data Site, DateTime, Hsig (wave height), Hmax (max wave height), TZ, Tp, Direction, SST (PC)
☒ Must have normal wave height ranges for list of sites for list of sites Abbot Point, Albatross Bay (Weipa), Brisbane, Bundaberg, Cairns, Caloundra, Emu Park, Gladstone, Gold Coast, Hay Point, Mackay, Mooloolaba, North Moreton Bay, Townsville, Tweed Heads (PC)
☒ Must automate data upload for detailed wave data (PC)
The administrator can download the csv file to the computer if required
☒ An administrator can upload data to the site using a .csv file (PC)
☒ A user can register details with the site (PC)
☒ + Must store contact details for users (PC)
☒ + Must allow users to register for up to 3 specific wave sites (PC)
☒ + Must allow users to register for a type of alert for each site (PC)
☒ + An incorrect user registration is not stored in the database (PC)
Duplicates are not stored but blank registrations are possible.
☒ Appropriate alerts are emailed or texted to the user (PC)
Email and SMS interface not developed. A testing web page shows which alerts would be generated.
☒ + Must validate user input for email address and/or mobile number (PC)
☒ + Must alert users based on user profile configuration (PC
List of user contact details are generated showing which users should be notified for a specific site name, alert combination but alerts are not generated.
☒ Appropriate alerts are emailed or texted to the user (PC)
☒ + Must validate user input for email address and/or mobile number (PC)
☒ + Must alert users based on user profile configuration (PC)
☒ Impacts - personal, social, economic, legal, ethical (SC)
☒ + Impact: The website complies with the privacy act (PC)
According to the Privacy Act users have the right to remove or modify their personal details. Although users are invited to contact admin if changes are required this should be an automated process.
☒ Accuracy of code (SC)
☒ Efficiency of code (SC)
☒ Application useability from useability principles (accessibility, effectiveness, safety, utility, learnability) (SC)
☒ +The website complies with government website design standards (PC)

Constraints
☒ Completed within time budget
☒ Compliant with accessibility standards
☐ Password protection for user details and/or privacy policy
Implementation issues, however recommendations have been included.
☒ Disclaimer linked or added to user registration page
User registration page was already quite long so the disclaimer was added to the index page instead.

☒ Impacts (SC) - personal, social, economic, legal.
Personal (accessibility, useability and privacy): Most of the accessibility, useability and digital copyright criteria have been adequately met, however a number of important privacy considerations need to be addressed before the application would be suitable for public release. The accessibility checklist completed after an early progress evaluation recommended a number of changes to comply with the government accessibility guidelines and these recommendations were addressed to a high standard. Useability criteria such as complying with the web style guide and ensuring fields and buttons are placed in a logical order, alignment and colour choices in the layout, all comply fully with the web style guide provided. Customer personas like Rebecca, Rodney and Edith will be better prepared for severe weather events by registering with this site. They will feel more secure in the knowledge that they will be warned about these events. Site security of personal details is the biggest risk. The web site does not use encryption or any other security methods other than username and password for data security. This should be addressed in future versions. However, the only details recorded are name, phone and email which most people share with many businesses.

Social: The site helps meteorologists identify conditions that could endanger lives or damage property and so is a socially responsible site.

Economic: Abdul (customer persona) as a business owner will be more informed about severe weather events to better manage his business. Business owners will be more aware of alerts that may physically damage their business or affect their customer base or supply chain.

Legal (digital copyright): All the data used was open data and publicly available. No images were used other than template images provided by the Queensland Government. Therefore, digital copyright issues with the application are not anticipated. The main weakness of the solution is the lack of a login screen to prevent unauthorised access to personal details. The functionality to modify user details is also missing making it difficult to comply with the Australian Privacy Act which requires that users have the right to update and modify any personal identifying information stored about them. At the moment, this right could only be fulfilled through manual processes. This also needs to be addressed in future versions.

References

Anderson, S, 2017, How Fast Should A Website Load in 2017? Accessed from www.hobo-web.co.uk/your-website-design-should-load-in-4-seconds/

EasyPHP, 2017, Develop with DevServer and Host with WebServer, accessed from www.easyphp.org

GenMyModel, 2017, Online Modelling Platform, accessed from www.genmymodel.com

Oracle corporation, 2018, MySQL 5.7 Reference Manual, accessed from dev.mysql.com/doc/refman/5.7/en/

Queensland Government, 2016, Customer User Experience Template, accessed from www.forgov.qld.gov.au/cue-template-downloads

Security innovation Europe, 2016, What is the difference between hashing and encrypting, accessed from www.securityinnovationeurope.com/blog/page/whats-the-difference-between-hashing-and-encrypting

W3.CSS, 2017, w3schools web reference, accessed from www.w3schools.com

W3schools.com, 2017, HTTP Methods: GET vs. POST, accessed from www.w3schools.com/tags/ref_httpmethods.asp

Wufoo, 2018, Wufoo Online Form Builder, accessed from www.wufoo.com

Useability recommendations

✓ Change index.html page template heading "Applications" to a more appropriate heading
✓ Make sure headings match the breadcrumbs
✓ Change tab order on user registration page so that submit button is after the checkboxes
✓ Indicate whether mobile number and email are mandatory

Progress Evaluation - Ongoing evaluation TO DO list
- ☐ Create login screen that protects user data and can be used to decide whether or not to show admin option.
- ☐ Create an admin interface to insert manual wave data records for testing purposes
- ☑ Create a way to list alerts generated for testing purposes (eg.temporary table , logfile or webpage)
- ☐ Allow users to modify their preferences

Progress Evaluation ongoing
- ☑ Update user interface webpages
  - ☑ Recommendations from early development
  - ☑ Additional fields for 3 sites
- ☑ Setup developer environment
  webserver port: 8888 php port: 1111
- ☑ Database name: waves
  Database access for conn.php file:
  Userid: dba001
  Password: js0h0SMYjbaulrVV
- ☑ Check php works (e.g. echo date to webpage)
- ☑ Create waves db in phpmyadmin
- ☑ Create tables using SQL
- ☑ Populate tables with initial data for testing using SQL statements created with CONCATENATE function in Excel
  - ☑ Wavedata
  - ☑ Persons (needed to rename from users)
  - ☑ Wavebenchmark
  - ☑ alerts
- ☑ Write code to display test data on user interface
- ☑ Write and test code to register user details
- ☑ Write and test code to register alert preferences of users

Synthesising and evaluating [9–10]

Critical evaluation of impacts and coded components and the digital solution against essential prescribed and self-determined criteria to make discerning refinements and astute recommendations justified by data

Evaluation — Testing and recommendations

Accessibility checklist

- ☒ All pages have a title that appears in the browser tab some titles are too long to appear in the tab
- ☒ Page titles are appropriate for the pages - mostly but on main page template wording "Application" is used as the heading instead of Wave Data project which should be h1 not h2. Some headings don't match the breadcrumbs e.g. Upload data is the page heading and data upload is the breadcrumb heading
- ☒ A different title is used for each page
- ☐ Alt text is used for all content images (excluding decorative images) no images used
- ☐ Alt text appropriately describes the content image it relates to no images used
- ☐ Alt text is not used for decorative images no images used
- ☐ The alt text attribute is set to null for decorative images
- ☒ Every web page has at least one heading
- ☒ Heading levels on each web page have a meaningful hierarchy
- ☒ Contrast ratio between text and background is appropriate (colour contrast)
- ☒ Web page displays correctly for page zoom settings with no horizontal scrolling (some people need to enlarge web content in order to read it)
- ☒ All text gets larger when page is zoomed
- ☒ All buttons are visible when page is zoomed
- ☒ Web page can be navigated in a logical order with keystrokes or tabs (no mouse) form fields navigate in a logical order but instead of going to submit button at the end the tab key selects the links in the template and then finally selects the submit button last
- ☒ It is possible to tab to all web page elements
- ☒ Form fields and other form controls have a visible label to allow interaction with voice input and increase clickable area
- ☒ Mandatory fields are clearly indicated and do not rely on colour alone mandatory fields are indicated by error message when not completed
- ☒ General instructions for user input are at the top of the form or section they relate to
- ☐ Required formats, such as dates (year, month, day) are clearly indicated no format guidelines given
- ☒ Error messages or validation messages are clear and specific incorrect email address gives a meaningful error
- ☒ Error messages do not cause the form to be completely reset
- ☐ Any moving or flashing content that lasts for 5 seconds or more can be disabled or controlled by the user n/a
- ☐ Text transcripts are provided for audio and video elements n/a

Test plan and recommendations

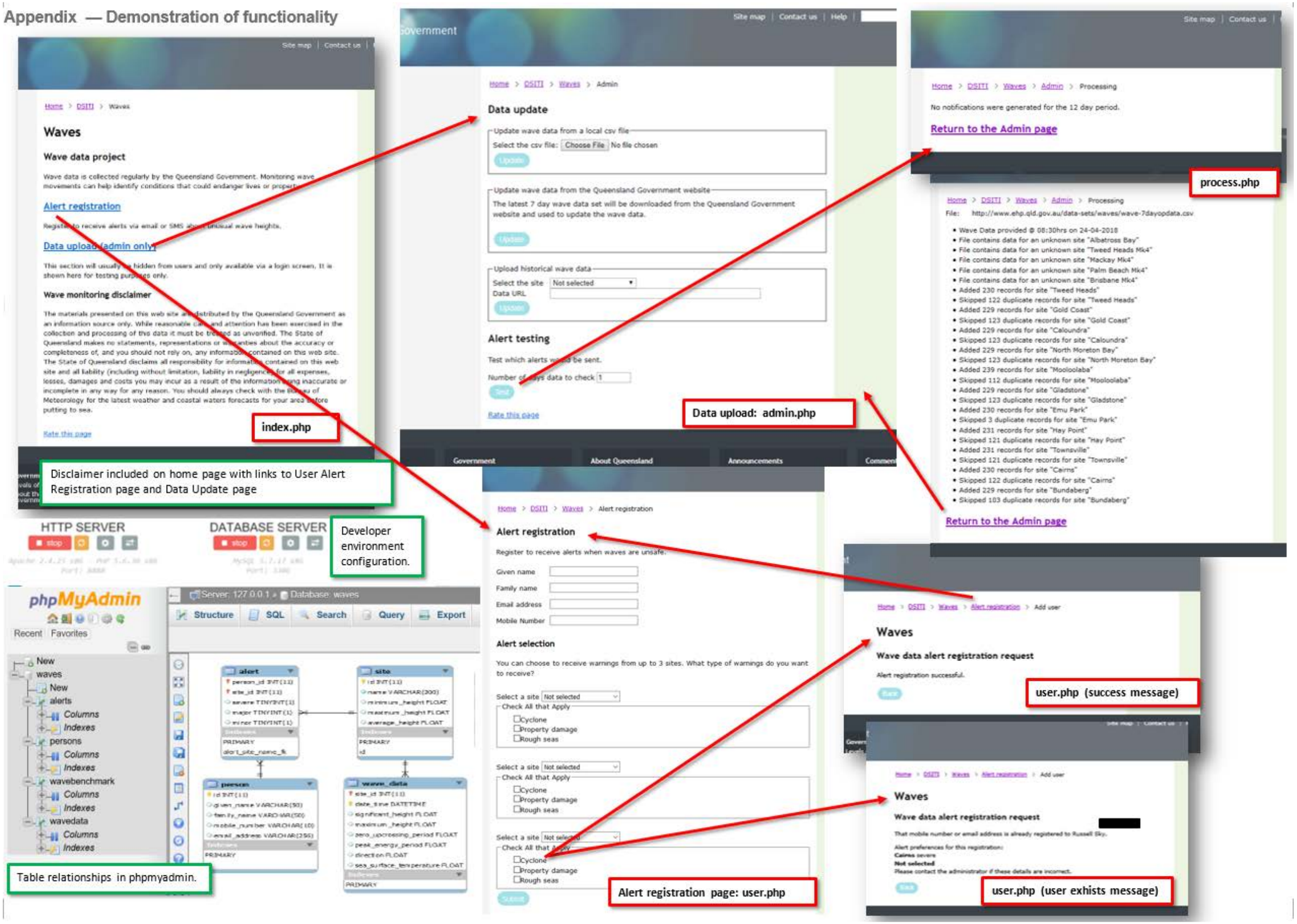
Action	Expected outcome	Actual outcome	Recommendations
Test links	Links go to correct page, correct page title appears in tab, correct page heading shows, breadcrumbs change	As expected	
Test adding a user that exists	Appropriate message displayed for user but user is not added	As expected	
Test adding a user with all fields blank	Appropriate message displayed for user but user is not added	User was added with an autogenerated userNum and null or 0 values in all other field	Add field validation to prevent all blanks but allow blank mobile or email address.
Test adding a user that does not exist with three alert selections	Appropriate records added to persons table and alerts table. Success message displayed to user	As expected	
Test adding a user that does not exist with 1 alert selection and the other site names set to "Not selected"	Appropriate records added to persons table and alerts table. Not selected sites not added to alerts table. Success message displayed to user	User added correctly but one alert generated for site_Name "Not Selected". No error generated for breach of foreign key constraints or uniqueness constraint	Investigate failure of referential integrity db constraints and resolve problem. Update code so that Not selected records are skipped and not used to update database. Needed to change "Not Selected" to "Not selected"
Test uploading data	Pop-up appears asking if you want to save or open the csv file	As expected	Add code to automate this process by reading the records in the csv file and adding them to the local server database.
Show alerts that would need to be sent	Clicking test displays a contact list for alerts that match the site and severity selected	As expected	Add code to raise alerts and keep contact list private.
Test privacy of user data	Database is password protected with users having access to their own data only	Not included yet	Create login page. Add personType column to persons table to track whether person logged in is user level or admin level. Add an admin registration page for admin eyes only. Hide admin content on index page when user is logged in.

Additional evaluation and recommendations

During testing, it was noted that even though no validation code has yet been added, the email validation is in place because the form element type email has been used. A pink colour outline and error message is shown if an incorrect email is entered. This is an efficient way of validating the email address entered so no additional email validation will be required. Initially the checkboxes were not working during the evaluation even though they had worked in the past. This problem needed to be investigated and fixed otherwise the user wouldn't have been able to use the checkboxes to show the alert type they are interested in. This checkbox problem was caused by the padding setting preventing a click in checkbox from registering. This was fixed in version 2. The webpage template does not have any images so it is quite boring and therefore does not align with good visual design principles (balance - page looks unbalanced due to the and contrast - lack of contrast because there is a lack of colour). This might be suitable for adults but children that want to use this service would not use it because of the uninteresting layout. There is a moving water image in the creative commons library that would look great as a background image for the webpage but then the website would not comply with the template requirements or the accessibility requirement to be able to turn off moving backgrounds. Therefore, it is recommended that the current layout is acceptable to the client and shouldn't be changed. According to the user stories not many teenagers will need to use this website anyway so the client and user stories needs have a higher priority. The page load times were tested using Google Development tools (see appendix). The load time for the admin, alert and index pages were 4.16, 4.22 and 8.45 seconds respectively. According to Anderson 2017, these page load times are too high and would result in a page abandonment rate of between 25 and 37%. Googles' PageSpeed Insights developer tool makes recommendations for fixes to increase page load times by eliminating render-blocking JavaScript and CSS, optimizing images, and minifying JavaScript. This tool was used to analyse each of the pages in the waves solution and the following recommendations were offered:

1. Optimize CSS delivery by in lining small CSS resources rather than using them as above-the-fold content, this allows the browser to proceed with rendering the page.
2. Optimising images: the developer tool has optimised these images and these have been downloaded and included in the final version, this resulted in a 1 second improvement in load time.
3. Minifying JavaScript: Minification refers to the process of removing unnecessary or redundant data without affecting how the resource is processed by the browser - e.g. code comments and formatting, removing unused code, using shorter variable and function names, and so on. These fixes will be employed in the final version.

Appendix — Demonstration of functionality



Appendix — load time statistics accessed from Google Developer page