

An abstract painting with vibrant colors including red, yellow, blue, and green. It features thick, expressive brushstrokes and a central blue, swirling shape that resembles a stylized figure or a dynamic flow. The overall texture is rich and layered.

# BRISBANE GIRLS GRAMMAR SCHOOL: Digital Art Gallery

RACHEL CHIONG  
JOSEPHINE CLOUGH

## VIDEO LINK:

[https://youtu.be/sqP\\_447mHg4](https://youtu.be/sqP_447mHg4)

## CONTENTS:

- 1 – EXPLORE
- 4 – DEVELOP
  - 4 – Wireframes
  - 6 – Algorithms
- 9 – GENERATE
- 14 – EVALUATE
- 16 – Appendix



# EXPLORE

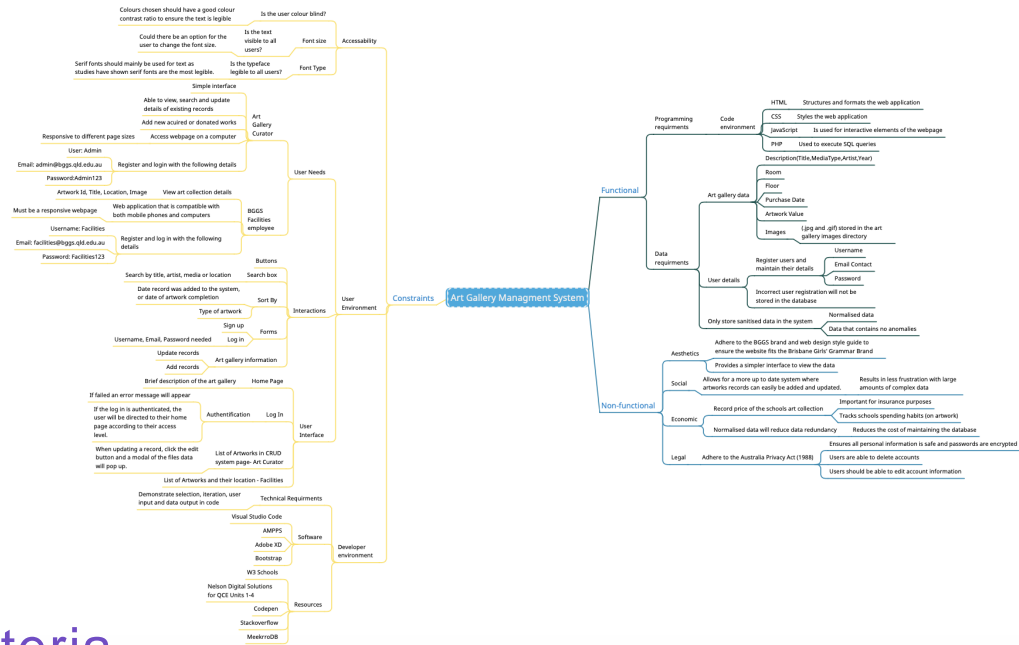
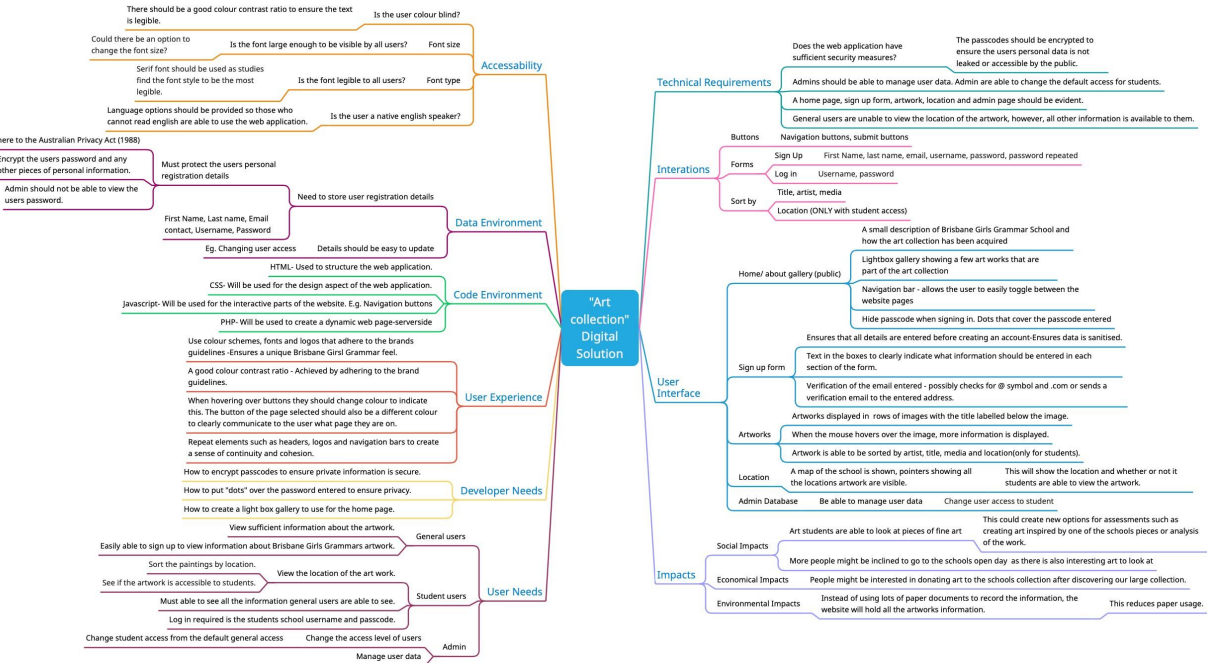
## Introduction

Brisbane Girls’ Grammar School has an extensive fine art collection which is managed by the appointed art curator the facilities staff. The school would like to have a dedicated dynamic online gallery where both the school community and general public can view the school’s art work and their related information. To better manage the art collections information, the appointed curator has also requested an art collection management system web application. This web application would provide the appointed art curator with a computer compatible, simple web application interface to view, search update existing art records as well and create new records. The web application would also allow Facilities staff to view the art collection details, in particular, the current display location of the artworks in the school which can be viewed on either a computer or more conveniently, on a mobile phone.

Component of Digital Solution	Coding language/s constraints	Resources used
Front-end web design (what the user sees and interacts with directly)	HTML (structural), CSS (styling), Javascript (animations and toggling hide and show) JQuery Saved as PHP file	<ul style="list-style-type: none"><li>Visual studio code (coding environment)</li><li>Google Chrome (testing environment)</li><li>Mozilla Firefox for developers (testing environment)</li><li>Codepen.io</li><li>BGGGS website</li><li>Bootstrap</li></ul>
Back-end database management	PHP (submission of forms and error handling) MySQLi (connection to database) SQL (querying the database, inserting new data)	<ul style="list-style-type: none"><li>AMPPS (local host)</li><li>myPhpAdmin (database management)</li><li>MeekroDB</li></ul>

## Mind Maps

Larger versions of the mind maps can be viewed in the appendix.



## Project Criteria

The following prescribed criteria (directly from the art curator and the school) and self-determined (self-developed parameters we wished to follow to enhance the user experience) criteria were adhered to.

Prescribed Criteria	Self determined Criteria
<ul style="list-style-type: none"><li>• <b>Generate a dynamic online gallery that is facilitated by a relational database</b> and associated interface, live to the web that will be accessible to school community and the general public to use.</li><li>• Users must be able to <b>filter artworks by title, artist or media</b> (and location for student access level users).</li><li>• Code a <b>fully functioning registration form</b> which records the users details and stores them in a database.</li><li>• <b>Encrypt personal data</b></li><li>• <b>Allow users to sign up and log in</b></li><li>• <b>Login required for access to gallery search</b></li><li>• <b>The admin will be able manage user data and change user’s access levels.</b></li><li>• <b>Develop a web-based art collection management system.</b></li><li>• <b>Represent BGGGS as a brand</b> by complying to the BGGGS style guide.</li><li>• Have <b>responsive web design</b> (not including admin)</li><li>• Include an <b>administrator's dashboard</b> that:<ul style="list-style-type: none"><li>• <b>Allows the use of CRUD operations</b> (create, read, update and delete records)</li><li>• <b>Display artworks records in a table format</b></li></ul></li><li>• Adhere to the Australian Privacy Act (1988)</li><li>• Comply with web accessibility guidelines.</li><li>• Must comply with copyright laws.</li></ul>	<ul style="list-style-type: none"><li>• <b>Protect from SQL injection</b> to ensure the database is not tampered with.</li><li>• <b>Use closed-entry form input</b> where possible</li><li>• Allow the admin to <b>sort records</b> by ID, title, artist, media or location</li><li>• Create a webpage which only uses <b>vertical scrolling to display the data table</b></li><li>• <b>A map of the school on the locations page</b> locating areas where artwork can be found. This map will show locations students are able to visit.</li><li>• <b>Use placeholders in forms</b> so the user can clearly understand what information they should fill in the input box.</li><li>• Use a <b>flexible search method</b> that is non-case sensitive and allow for placeholders</li><li>• <b>Dynamic loading of pages</b></li><li>• <b>Modularisation of pages</b></li></ul>

# EXPLORE

## SWOT Analysis

	Opportunities/Strengths	Weaknesses/Threats
Individual / Personal	<ul style="list-style-type: none"><li>Students are able to view a massive part of Brisbane Girls' Grammar's culture.</li><li>Students are able to view the location of the artwork at the school and can see the artwork.</li><li>Improved knowledge and understanding of the artworks</li><li>Reduces anxiety and stress levels for administrator due to a well-organized system</li><li>Protect personal data</li><li>Facilities and administrator staff will work better together as the new system will allow administrators to pass information to facilities staff in an organized and efficient manner.</li></ul>	<ul style="list-style-type: none"><li>Users are not able to view descriptions of the artworks.</li><li>Users' private information could be accessed by the public if their information is not encrypted.</li></ul>
Organisation / Business	<ul style="list-style-type: none"><li>Viewers might wish to donate or loan art to the school after discovering its extensive collection.</li><li>People might donate money to the school's art curator to purchase new pieces of fine art.</li><li>Better time efficiency</li><li>Can find the total value of the artwork collection, which is useful for updating the collections insurance policy.</li><li>Removal of data redundancy in the system decreases the cost of maintaining the database and the amount of data storage needed.</li></ul>	<ul style="list-style-type: none"><li>If the administrator's password is lost, they would lose access to the database.</li><li>Locations and prices of artworks could accidentally be exposed to the public leaving the schools artworks vulnerable. It needs to be ensured that SQL injection prevention is used as well as validation methods to ensure this does not occur.</li><li>Users could possibly hack into the database and find where the school's artworks are displayed, and their price leaving the artworks vulnerable.</li><li>People could hack into the database and tamper with records.</li></ul>
Government	<ul style="list-style-type: none"><li>The website should adhere to the Australian Privacy Act (1988)</li><li>Passwords should be encrypted to ensure user privacy.</li></ul>	
Society	<ul style="list-style-type: none"><li>Users researching an artist could look at the Art collection website to discover more about the artist work. This would be particularly useful if the artist was a Brisbane Girls' Grammar alumni.</li></ul>	<ul style="list-style-type: none"><li>Users could possibly hack into the database and find where the school's artworks are displayed, leaving the artworks vulnerable.</li></ul>

## Original Database Structure

This was the original dataset provided by the school's Art curator. The original database included 7 fields: asset no, description asset category, building room, floor, department owning cost centre, purchase date, purchase value, and status. The provided dataset has more than one piece of information in each cell. To ensure the database's data would be sanitized, the flat-file will be normalised with relational tables.

Brisbane Girls Grammar School							
Asset List by Asset No							
Asset No	Description Asset Category	Building Room	Floor	Department Cost Centre	Owning	Purchase Date Purchase Value	Status
91	Painting Portrait 'Women in Black' - Ella Fry Fine Arts Collection	M M118	1	FAC 00-5300-00-01		1/01/1984 \$1,500.00	Active
58	Painting - The Landscape - Ella Fry Fine Arts Collection	M M110	1	FAC 00-5300-00-01		1/01/1984 \$2,000.00	Active
32	Painting 'Shipwreck' - J W Jenner - 1884 Fine Arts Collection	M M110	1	FAC 00-5300-00-01		1/01/1984 \$10,500.00	Active
32	Print - 'The Two Sisters, On The Terrace' Pierre-Auguste Renoir Fine Arts Collection	C C301	3	FAC 00-5300-00-01		1/01/1990 \$100.00	Disposed
23	Oil Painting 'Evolution/Irene Amos Fine Arts Collection	M M202	5	FAC 00-5300-00-01		1/05/2013 \$0.00	Active
36	Oil on Linen 'Primitive' - Irene Amos Fine Arts Collection	C C1	5	FAC 00-5300-00-01		1/05/2013 \$50.00	Active

Data redundancy – Repeat of building code, department, owing cost centre and asset category.

Unsanitised data-more than 1 type of data stored in a field

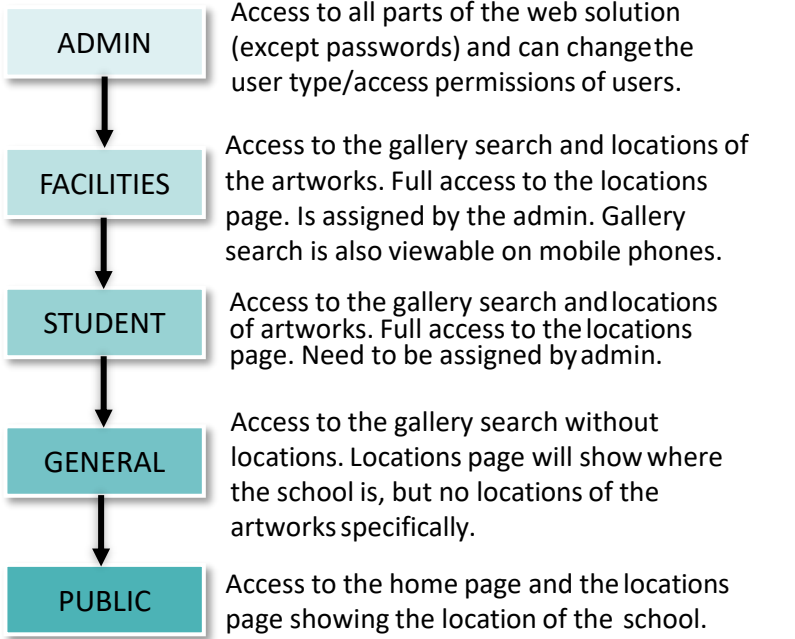
Overall, the issues evident are:

- Non-atomic data: Name of painting, media, name of artist and year are in the same field. These need to be split up.
- Redundant data: department owing cost center are not required
- Purchase date and purchase value are in the same field and need to be split up.
- Currently is difficult to search up artworks in the collection: specific phrases must be entered.

## Data Management Security Solutions

As the people using the art collection website may not be from the school, certain pages and information, such as the locations of artwork need to be hidden from such users. The art collection solution will have four tiers of level of access corresponding with the type of user.

## User Types





# EXPLORE

## Existing Solutions

The Virtual Art Gallery needed to be coherent with the BGGGS website. Possible designs for the administrator database management system were also explored

## Mobile View & Desktop View



Large header navigation with collapsed hamburger menu for compactness.

Splash screen still covers the entire screen

Good contrast: use of white boxes for actionable buttons



2-column grid has become 1-column to fill the screen.

Text is center-aligned to reinforce the reading pattern.



The use of white space enhances the usability as it makes all the contrasting buttons and logos on the header clearly visible. The white font against the coloured background ensures legibility and therefore increases the websites accessibility.

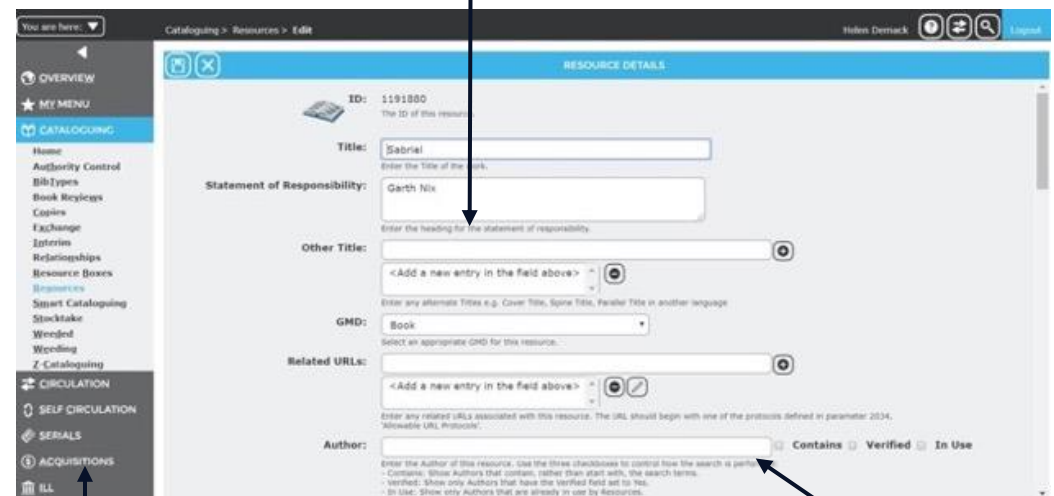


All important pieces of font and titles have all been coloured in blue to distinguish them from large bodies of text drawing the users eye to the key pieces of information on the website.

## Brisbane Girls' Grammar Brand Guidelines



Users are informed of what they should enter in the form through comments underneath the input boxes.



The menu bar located on the side of the page allows for the user to easily navigate the library admin pages.

The use of dropdown selections enhances user efficiency as they can quickly select the data, they wish to input into the form without having to type out long text which they could possibly make mistakes with. Instructions below each user input box enhances user learnability as they can easily identify exactly what information each field requires.

User input boxes are where data is then parsed into a SQL query where the data is then stored in the database.

Simple relation			
#	Firstname	Lastname	Office in
1	Andy	Fitter	Sydney
2	Anthony	Bow	San Francisco
3	Barry	Jones	London
4	Diane	Murphy	Tokyo
5	Foon Yue	Tseng	NYC
6	George	Vanauf	NYC
7	Gerard	Hernandez	Paris
8	Gerard	Bondur	Paris
9	Jeff	Firrelli	San Francisco
10	Julie	Firrelli	Boston

Buttons where the user can either view, edit or delete records from the database. This is communicated through a use of standardised icons and colours.

Selected rows, pages and number of records per page are indicated as they are highlighted in purple.

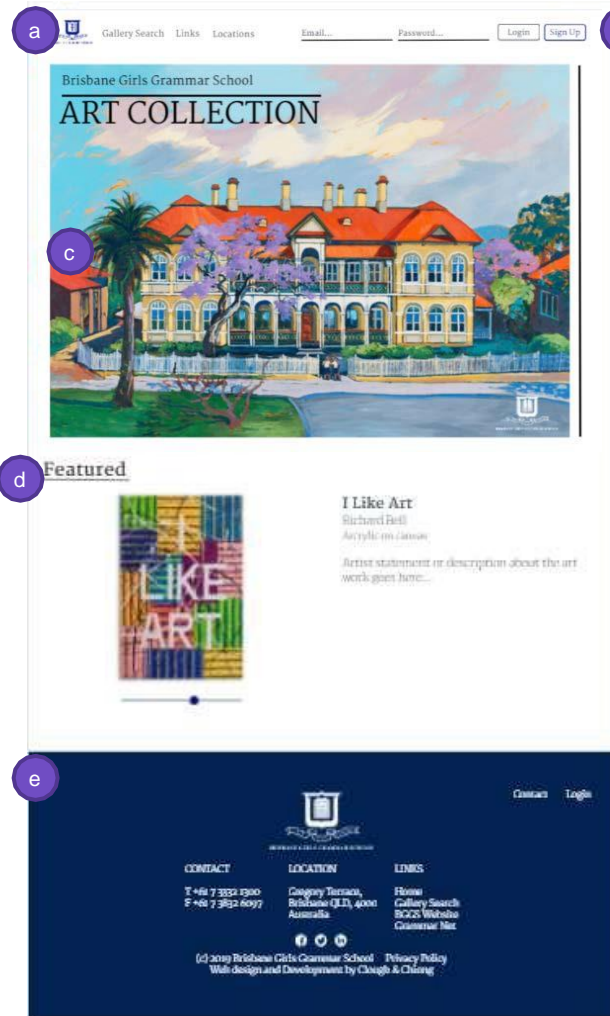
The primary colour palette uses dark neutral colour tones to create a sense of sophistication and class. The royale blue colour is often used for titles. The colour charcoal is used for bodies of text as it has a good colour contrast ration against a white background and is not as harsh as solid black font. Reflex blue is used sparingly in the main colour palette and is only featured in the school's logo.

The secondary colour palette is used sparingly throughout the Brisbane Girls' Grammar School website and is used only to highlight key words, pieces of information and buttons. These colours could possibly be used in the art gallery management system to emphasise the delete and update buttons.

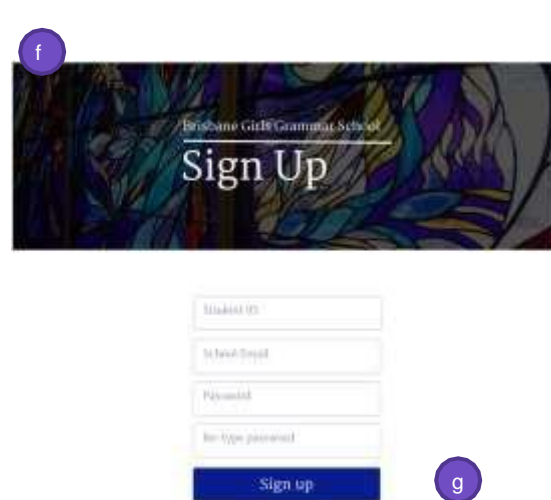
Brisbane Girls Grammar Schools primary typeface, ITC New Baskerville Roman is used for all formal school communications, including the school's public website. The serif font was chosen for its timeless elegance and its legibility (serif fonts are easier to read than sans serif fonts).



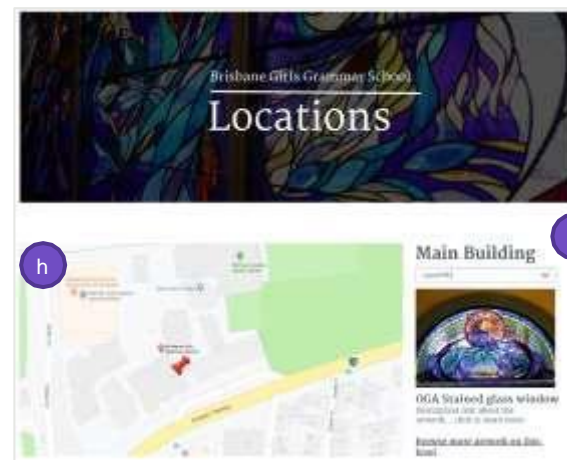
# WIREFRAMES



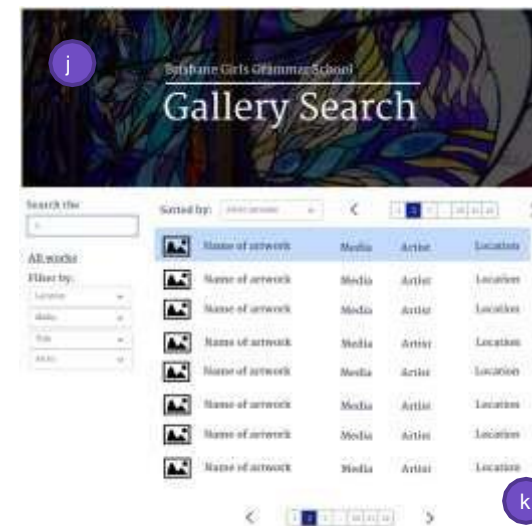
Index.php



Sign\_Up.php



Locations.php



Gal\_search.php

- i) The **side bar** on the locations page will appear when the user has logged in, and wishes to know which artworks are in the building they click on in the map. A short description of the area, and a link to the gallery search (for art in that building) will be visible.
- j) The **side search bar** allows users to search the database and toggle different filters using drop down menus. This side search bar will be static so that as the user scrolls down, it stays at the same position on the page for efficiency when changing searches.
- k) The **search results table** will have basic information about the artwork and a picture. If users wish to learn more about a specific artwork, they will need to click on the row. The use of the compact table saves space and improves the efficiency of loading the code.

**Learnability:** The web app is very learnable as there are a lot of familiar parts, such as drop down menus, search and navigation bars and google maps.

**Usability:** Users will not get 'stuck' or 'lost' in the web app as they are able to navigate back to any page using either the navigation bar, hamburger menu or footer.

**Accessibility:** Good colour contrast so that users are able to distinguish between background and foreground text. Buttons have outlines to show the clicking area of the buttons.

## Mobile View

Collapsed 'hamburger' menu with easy access login and logout buttons and gallery search. This is to declutter the top navigation. Featured section will be stacked instead of inline, so that users can view images and text better.

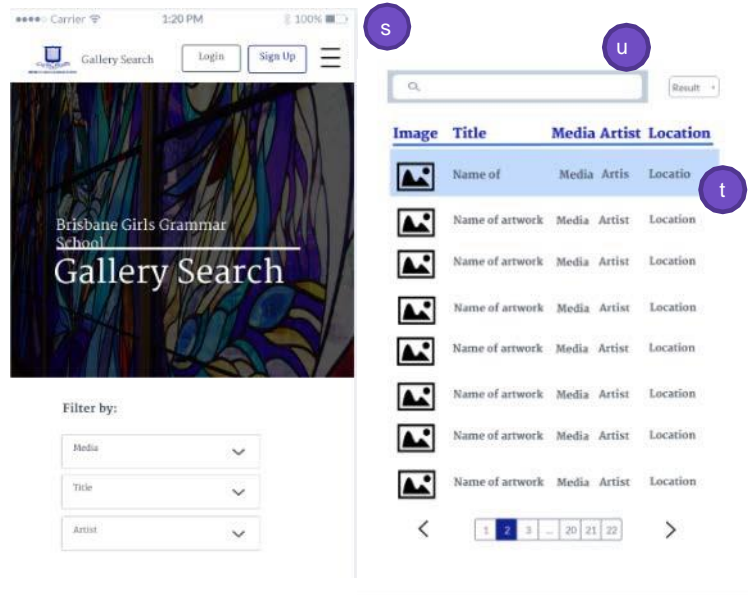


- a) The **navigation bar** provides quick access to various parts of website. The use of the white background makes the design more open, increasing understandability and engagement. When each tab is hovered over, a blue, animated line will appear beneath to highlight the page the user is about to click on, making the navigation bar more actionable and engaging.
- b) The **sign up button** is outlined blue instead of grey to draw the attention of viewers to alert them to sign up. Users who have an account can log in via the navigation bar to improve efficiency.
- c) **Large splash image** of alumni artwork makes the page more engaging. It stands out against the white background, and matches the style of the BGGGS website.
- d) **Featured section** will contain the 'about the collection' text as well as featured artwork from different alumni. The slideshow will play automatically, and users can use the slider to switch between displays. This makes the page more engaging for the user.

- e) The **Footer** is adapted from the footer from the BGGGS website and provides links within the website and external, so that users can learn more/contact the school.
- f) The **header banner** that is used across all pages (except the index page) engages the attention viewer. The wide image makes the site more contemporary and is inline with the styles on the BGGGS website.
- g) The simple **sign up form** is easy to use and understandable, with placeholder text so that users know what to enter. The 'submit' button in royal blue, making it more actionable as it contrasts with the transparent form fields. The centered text improves the efficiency of the page.
- h) The **locations page** features an integrated google maps object so that users can either find the school if they are a general user, or view which artworks are in each building using the drop down menu and the map. This engages the user with a familiar and understandable map.

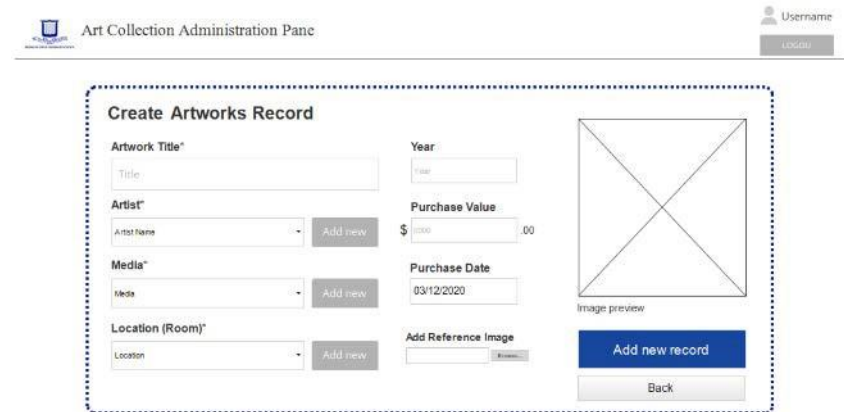
# WIREFRAMES

## Mobile View (gallery search for facilities)



Many pages are interrelated, with many being integrated (e.g. the dashboard file is the same across all tables)

## Create/Update Page



## Read Page



Once a record is successfully created or updated, the user is automatically redirected to the read page for that record to alert the user that the record has been successfully created.

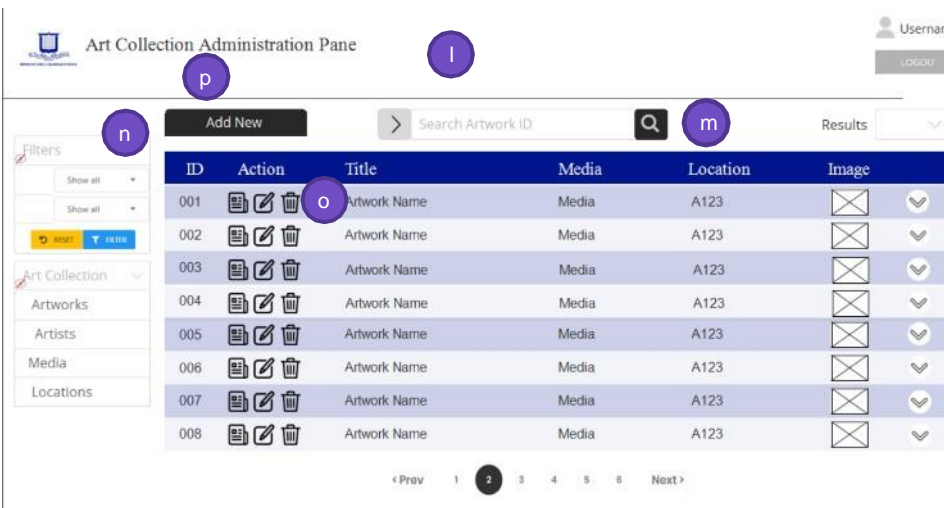
## Administrator's Dashboard

- Once administrators are logged in, they are automatically re-directed to the administrator dashboard panel, to increase accessibility. The same style of navigation bar increases understandability and cohesion as it matches the landing page.
- Users can search for specific records via ID, Title, Media and Location using the search bar and selecting which field they would like to search. The multiple functions increase its utility. It's location at the top of the page allows for increased accessibility as it is easily located.
- The static side-navigation bar is used so the other tables in the database, and the filters that can be applied are easily accessible when scrolling down the page.
- The actions field allows users to utilize the CRUD system: to read, update and delete certain records in the database. Icons are used to increase understandability with alt-text for accessibility.
- The 'Add New' button allows users to create new records in the table that they are currently viewing. It is located at the top of each table and is part of the static secondary navigation bar. It is in an accessible position.

## CRUD Pages (ADMIN)

- All CRUD pages (create, update, read and delete pages) follow the same structure and aesthetic features, with the image on the right, and other information/fields displayed beside it in two columns. This increases the effectiveness of the page as it minimizes unnecessary white space, making the page more usable.
- The Create and Update pages (represented using the 'create' page) follow the same structure, fields with labels and placeholders, to increase understandability. Searchable dropdown lists are used for artists, media and location to increase the security of the data (preventing SQL injection). If users wish to add new records to these respective tables, a button is provided that will open a new window so that they can do this.
- The gallery search will be responsive. A hamburger menu icon is used for learnability as its use-case is familiar to users: as a menu option.
- The gallery records will be displayed in a similar way to the artworks table on the dashboard, but without the actions and additional information. The clean table design increases usability and follows the BGGStyle guide.
- Users can search using the top search navigation, similar to the dashboard with artworks table to increase learnability.

## Dashboard with Artworks Table



### SQL query for Read page (and artworks table without the received ID condition)

```
SELECT a.a_id AS 'ID', a.a_title AS 'Title', CONCAT(r.a_fname,' ', r.a_lname) AS 'Artist_Name', m.m_name AS 'Media', l.room AS 'Location', a.img AS 'Image', a.year AS 'Year', a.purchase_date AS 'Purchase_Date', a.artwork_value AS 'Value' FROM artworks AS a, artists AS r, locations AS l, media AS m WHERE a.artist_id = r.artist_id AND a.l_id = l.l_id AND a.m_id = m.m_id AND a.a_id = " received artwork-id
```



# Algorithms and Dataflow Diagrams

Create Artworks Record Algorithm

BEGIN Program

RETRIEVE form input:

SET *title* = InputTitle

SET *m\_id* = InputMediaName

SET *l\_id* = InputRoomCode

SET *artist\_id* = InputArtistName

SET *year* = InputYear

SET *purchase\_date* = InputDate

SET *artwork\_value* = InputValue

SET *img\_data* = GET ImageDataFromFile

CONNECT to art\_database

SET *table* = GET table name

IF *imgdata* is not empty

SET *allowedFiles* = array[‘gif’, ‘jpeg’, ‘jpg’]

SET *filename* = GET ImageNameFromFile

SET *filesize* = GET ImageSizeFromFile

SET *fileExtensionType* = pathInfo of *filename*

IF *fileExtensionType* NOT IN *allowedFiles*

RETURN to create page for *table*

RETURN invalid imge type error

EXIT program

ELSE IF *filesize* > 200 KB

RETURN to create page for *table*

RETURN file size too big error

EXIT program

ELSE

IF *title* OR *m\_id* OR *l\_id* OR *artist\_id* OR *purchase\_date* OR *artwork\_value* is empty

RETURN to create page for *table*

RETURN Empty Fields Error

EXIT program

ELSE

SET *m\_iderror* = *l\_iderror* = *artist\_iderror* = NULL

QUERY database (“SELECT \* FROM artists WHERE CONCAT(a\_fname, “ “, a\_lname)=“*. artist\_id .’*””)

STORE result as *fetchartist\_id*

IF *fetchartist\_id* is NULL

*artist\_iderror* = TRUE

QUERY database (“SELECT \* FROM media WHERE m\_name =“*. m\_id .’*””)

STORE result as *fetchmedia\_id*

IF *fetchm\_id* is NULL

*m\_iderror* = TRUE

QUERY database (“SELECT \* FROM locations WHERE room =“*. l\_id .’*””)

STORE result as *fetchloc\_id*

IF *fetchloc\_id* is NULL

*l\_iderror* = TRUE

IF *m\_id* OR *l\_id* OR *artist\_id* is NULL

RETURN to create page

WRITE *artist\_iderror* AND *l\_iderror* AND *m\_iderror*

EXIT program

ELSE

QUERY database (“INSERT a\_title, m\_id, artist\_id, year, l\_id, purchase\_date, artwork\_value, img INTO artworks VALUES (*title, m\_id, artist\_id, year, l\_id, purchase\_date, artwork\_value, img*)”)

SET *new\_record\_id*

QUERY database ('SELECT a\_id FROM artworks WHERE a\_title="*title*" AND m\_id="*m\_id*" AND artist\_id="*artist\_id*" AND l\_id="*l\_id*" ')

STORE result as *new\_record\_id*

RETURN to read page for a\_id == *new\_record\_id*

EXIT Program

DISCONNECT from the database

END program

Read Artworks Record Algorithm

BEGIN Program

IF received table-input == ‘Artworks’

CONNECT to database

SET sql to “SELECT a.a\_id AS 'ID', a.a\_title AS 'Title', CONCAT(r.a\_fname, ' ', r.a\_lname) AS 'Artist\_Name', m.m\_name AS 'Media', l.room AS 'Location', a.img AS 'Image', a.year AS 'Year', a.purchase\_date AS 'Purchase\_Date', a.artwork\_value AS 'Value' FROM artworks AS a, artists AS r, locations AS l, media AS m WHERE a.artist\_id = r.artist\_id AND a.l\_id = l.l\_id AND a.m\_id = m.m\_id AND a.a\_id = " received artwork-id

SET mysqli prepared statement to include connection to database and *sql*

IF mysqli prepared statement is not executed successfully

RETURN error

EXIT Program

ELSE

SET result to mysqli statement result

IF number of rows in result == 1

FETCH associative array

FOREACH *row* retrieved from *results*

WRITE row label

WRITE row-element

IF *row* contains image THEN

ENCODE *row* with base64 as image is jpeg

DISPLAY image

CLOSE sql statement

CLOSE program

EXIT Program

User Login algorithm

BEGIN program

IF User has clicked login button

CONNECT to database

SET uname to INPUT username OR email

SET pword to INPUT password

IF uname is empty OR pword is empty

RETURN Empty fields error

EXIT program

ELSE

SET sql = "SELECT \* FROM users WHERE username= '\$uname' OR email= '\$uname'";

SET result = connection to database, execution of SQL query

SET resultsCheck = number of rows in database

IF resultsCheck < 1

RETURN login error

EXIT Program

ELSE

IF row matches uname

SET pwdCheck to verification of password

IF pwdCheck is incorrrect

RETURN wrong password error

EXIT program

ELSE IF pwdCheck is correct

START Session

RETRIEVE user ID, First Name and the user type

RETURN Login success

EXIT Program

User registration algorithm

BEGIN program

IF User has submitted signup form

CONNECT to database

SET *Stu\_ID* to INPUT student ID

SET *F\_Name* to INPUT First Name

SET *L\_Name* to INPUT Last Name

SET *email* to INPUT email

SET *pwd* to INPUT password

SET *pwdRpt* to INPUT password repeat

IF *Stu\_ID* is empty OR *F\_name* is empty OR *L\_name* is empty OR *email* is empty OR *pwd* is empty OR *pwdRpt* is empty

RETURN Empty fields error

EXIT program

ELSE IF *Stu\_ID* and *email* are invalid

RETURN Invalid Student ID and email error

EXIT program

ELSE IF *email* is invalid

RETURN Invalid email error

*Stu\_ID* to sign up page

EXIT program

ELSE IF *pwd* does not match *pwdRpt*

RETURN Password not matching error

EXIT Program

ELSE

SET *sql\_code* to “SELECT Student\_ID FROM Users WHERE Student\_ID=?,”

SET *stmt* to MySQLi connection initiation

IF MySQLi statement preparation FAILED

RETURN Sql error

ELSE

BIND single parameter *Stu\_ID* to *stmt*

EXECUTE *stmt* (to check if *Stu\_ID* exists in database)

STORE result in *stmt*

SET *result\_check* to Number of rows the executed statement returned stored in *stmt*

IF *result\_check* > 0

RETURN Student\_ID taken

ELSE

SET *sql\_code* to “INSERT INTO Users (Student\_ID, F\_name, L\_name, Email, P\_Word) VALUES (?, ?, ?, ?, ?);”

IF MySQLi statement preparation FAILED

RETURN Sql error

ELSE

SET *Pwd\_Hash* to hashed password using Bcrypt hashing method.

BIND five parameters *Stu\_ID, F\_name, L\_Name, Email, Pwd\_hash* to *stmt*

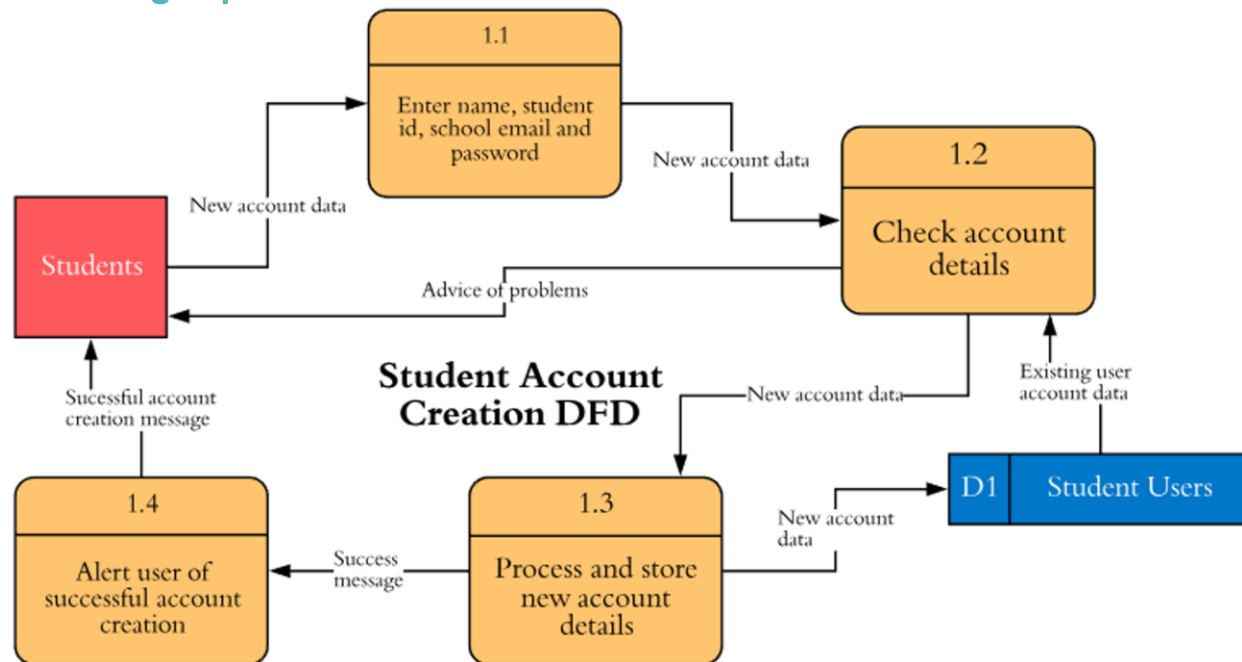
EXECUTE *stmt* to insert new user to database

RETURN Sign Up success

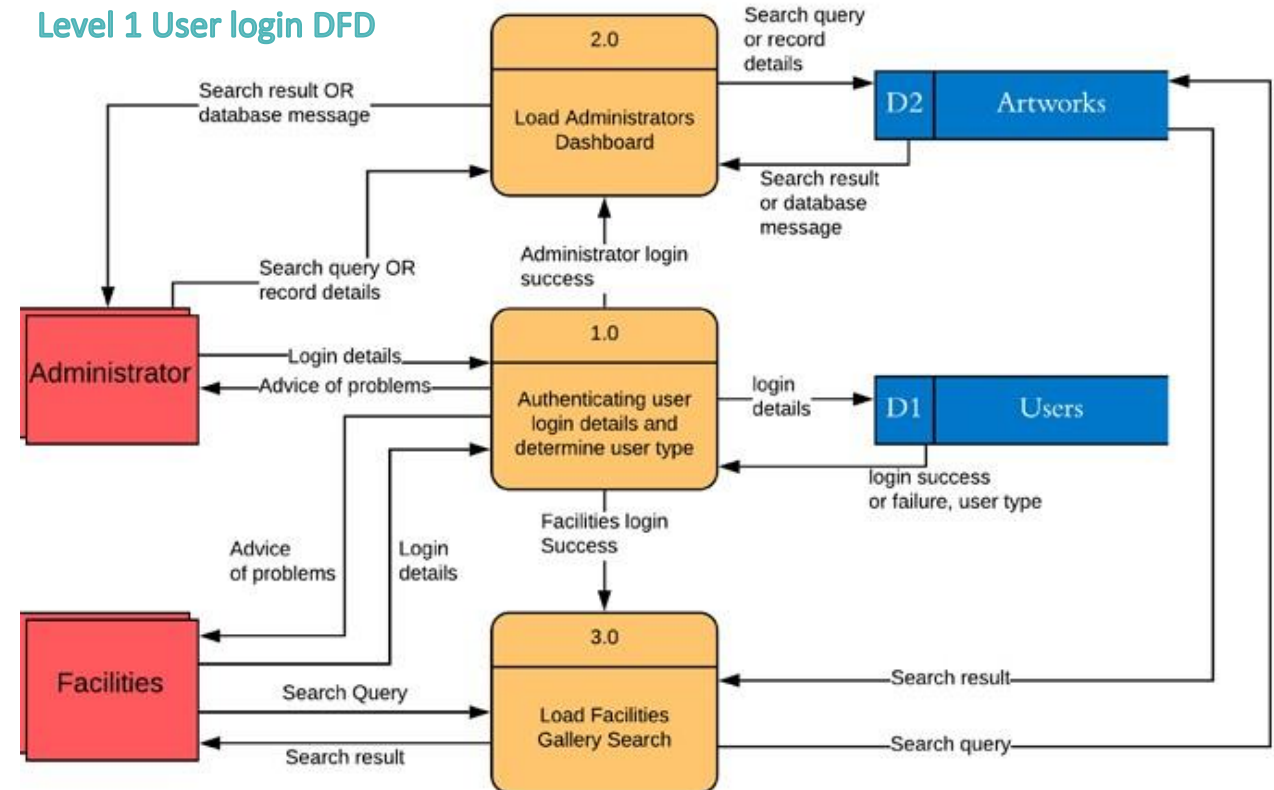
CLOSE database connection using SQLi

# DATA FLOW DIAGRAMS

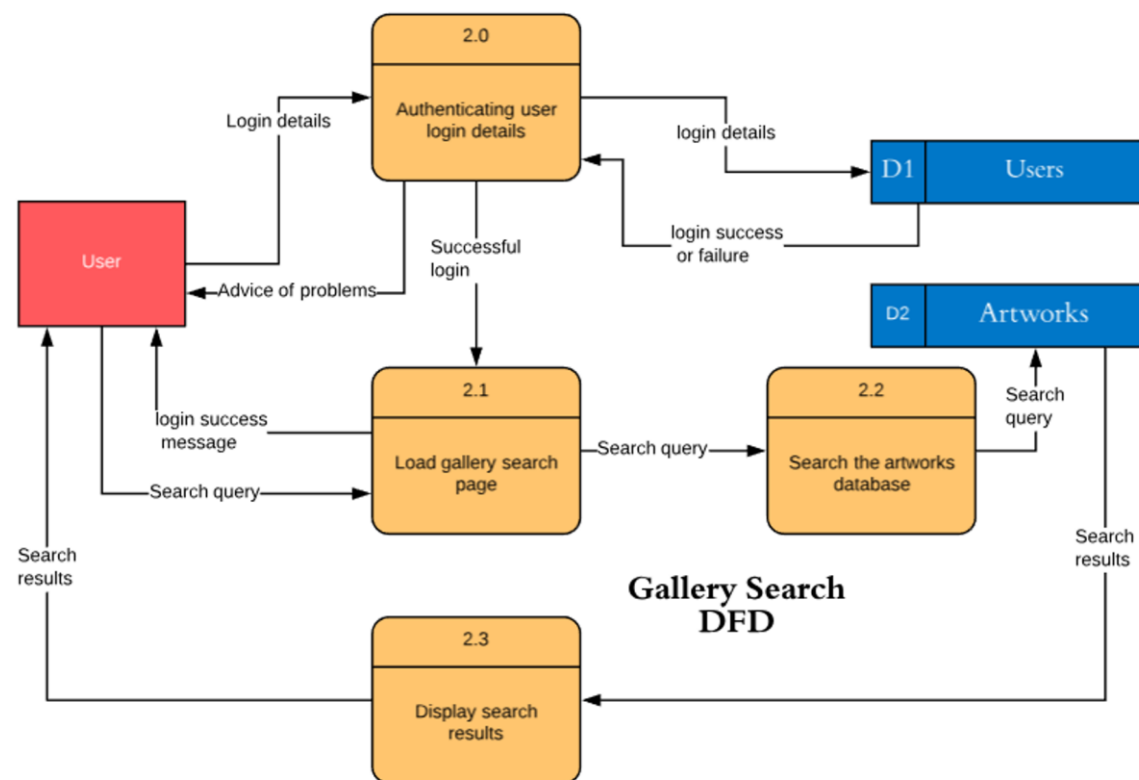
## Level 1 Sign Up DFD



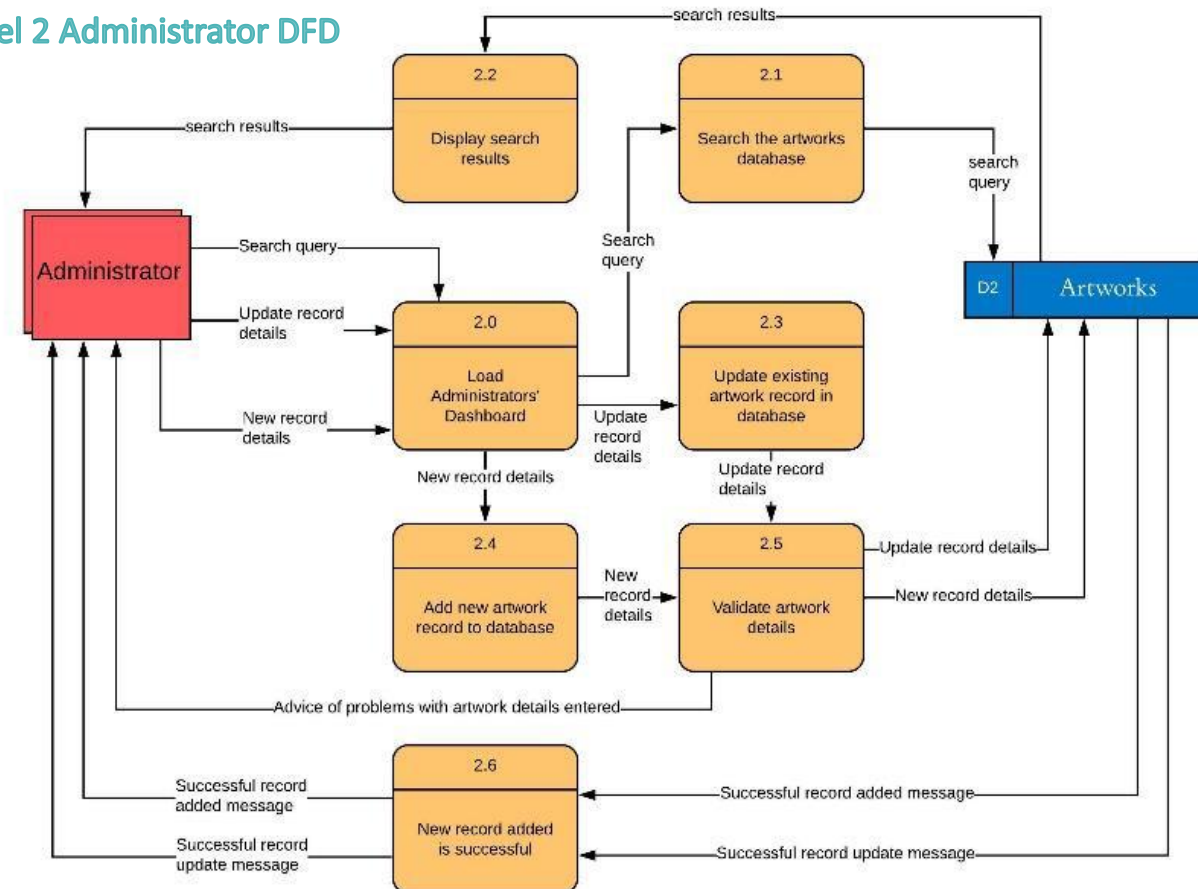
## Level 1 User login DFD



## Level 2 Gallery Search DFD



## Level 2 Administrator DFD





# DEVELOP

## Data Dictionary

Table	Field	Data Type	Nullable	Description
Users	u_id (PK)	INT	No	Auto-increment as each account is created
Users	Fname	varchar(255)	No	User First name
Users	Lname	varchar(255)	No	User Last name
Users	username	varchar(20)	No	Username used to sign into the database goes here
Users	emailUsers	varchar(255)	No	User's email
Users	pwordUsers	longtext	No	Password is encrypted using Bcrypt before being stored in the database
Users	UserType	varchar(255)	No	Default is set to 'General'. User type is changed by admin to 'ADMIN', 'STUDENT' or 'FACILITIES'
Artworks	a_id (PK)	INT	No	Auto-increment unique Id for each artwork added into the database
Artworks	Title	VarChar(255)	No	Name of the artwork
Artworks	M_id (FK)	INT(3)	Yes	Materials used to make the artwork
Artworks	Artist_id (FK)	VarChar(255)	No	Name of the artist who made the artwork
Artworks	Year	INT(4)	Yes	Date artwork was created
Artworks	L_id (FK)	INT(4)	Yes	ID of the location where the artwork is placed
Artworks	Purchase_Date	Date	No	When the artwork was purchased
Artworks	Artwork_value	INTI(10)	No	Estimated re-sell price of artwork
Artworks	img	longblob	No	Data for artwork image uploaded to database
Artist	Artist_id (PK)	INT(3)	No	Auto-increment unique id for each artist
Artist	A_fname	VarChar(255)	No	First name of artist
Artist	A_lname	VarChar(255)	No	Last name of artist

Media	M_id (PK)	INT(3)	No	Auto-increment unique id for each type of media
Media	Collection	VarChar(255)	No	Type of collection (e.g. Fine arts collection)
Media	M_name	VarChar(255)	No	Name of the media used (e.g. Painting)

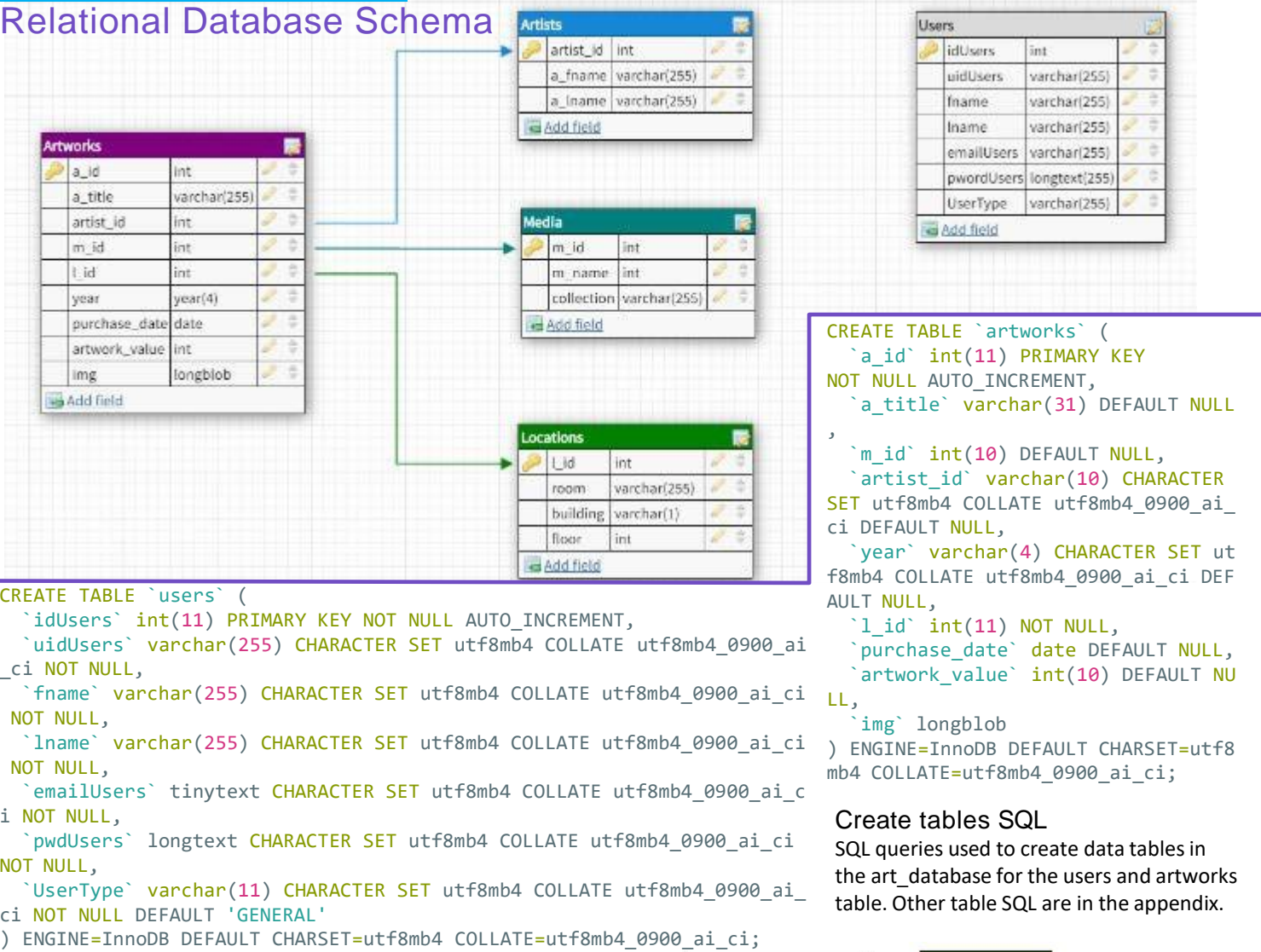
Location	L_id (PK)	INT(4)	No	Auto-increment unique ID for each location
Location	Room	VarChar(255)	No	Room name
Location	Building	VarChar(1)	No	Building symbol/code of the room
Location	Floor	INT(1)	Yes	Floor number where artwork is displayed

Separate tables for artist, media and location so that these can be changed efficiently. Also improves data integrity as it will allow for drop-down menu selection categories. (so that database can be searched easily and efficiently)

### Flat File Tables vs Relational Tables

It was decided to use relational database tables over flat file database tables for several reasons. Unlike flat file database tables, relational database tables are normalized removing data anomalies and data redundancy. Normalised data is less expensive to maintain as there is less repeated data to manage in the database. Unlike flat file tables, SQL queries can be used to retrieve data from tables in the database, allowing for easy data retrieval and data upload.

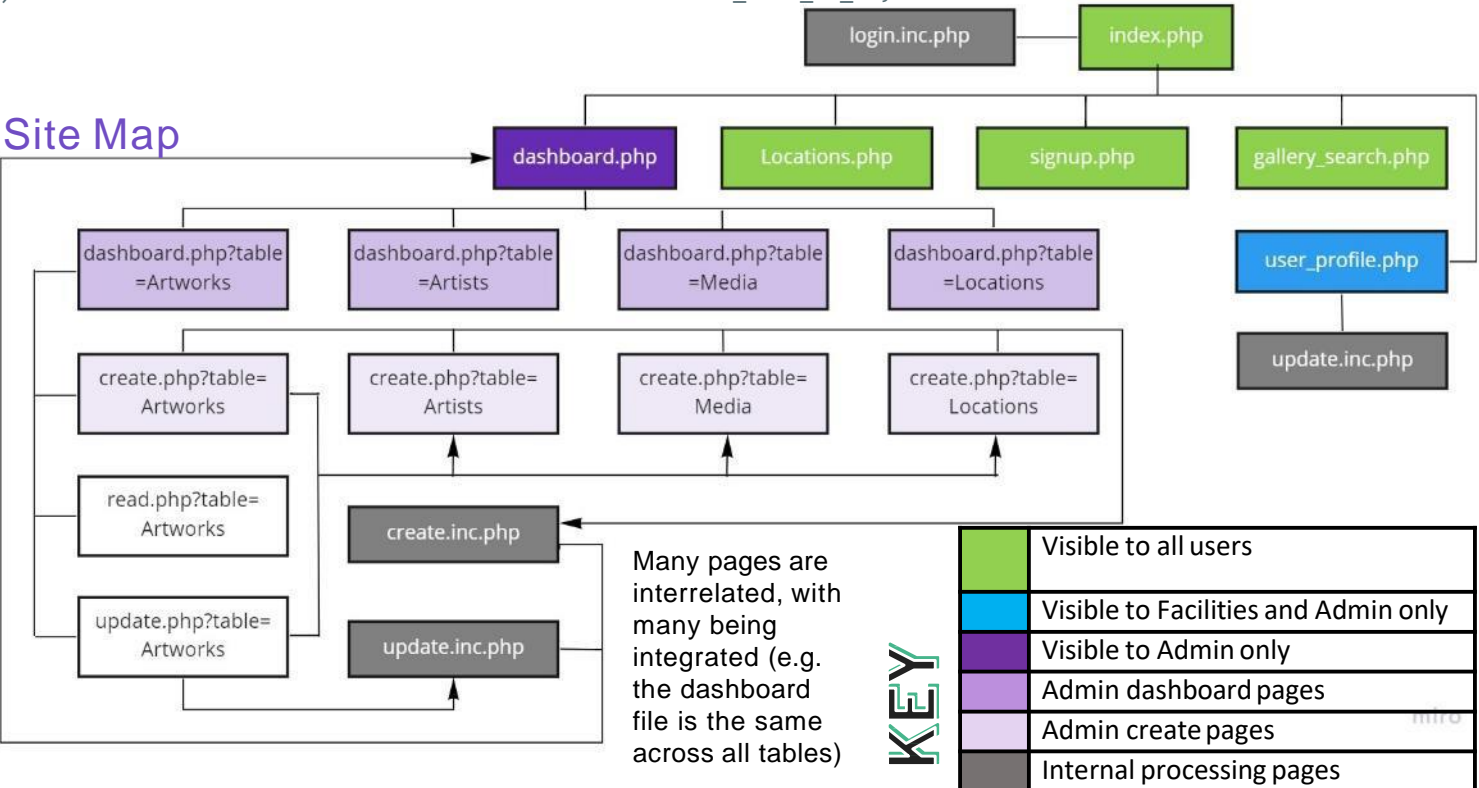
## Relational Database Schema



### Create tables SQL

SQL queries used to create data tables in the art\_database for the users and artworks table. Other table SQL are in the appendix.

## Site Map





# GENERATE

## SIGN UP

### Form fill-out

Form allows users to enter details easily and efficiently into the fields. The strong, blue underlines beneath each input field will be animated when the user clicks on it. This is to improve understandability as the user can easily tell where in the form they are, as well as improve reading clarity for users with myopia.

### Error communication

The user is alerted of an error in the filled out form just above the first input field. When fully coded, the fields where the error is located will be highlighted. This will improve understandability as users can easily identify what the issues are and fix them accordingly.

### Alternative login

Although not fully styled, users are able to login from the sign up page, by accessing a login popup. This is especially useful for users on small-medium devices, where the signup button is hidden in the hamburger menu. This is to give users flexibility and make their experience more efficient.

### Large font and buttons

The large font sizes are to help who are partially vision impaired. The large reflex-blue sign up button creates great contrast with the background, making it very actionable as users know to click it when they have filled out the form.

### Difference between \$\_GET and \$\_POST:

\$\_POST and \$GET are both methods used to transfer data, however, \$\_GET carries the request parameter in the URL, \$\_POST carries the request parameter in the message body.



### GET method

Information received using the 'get' method is displayed in the header. This is used during error handling to return information to the user when they have incorrectly filled out the form.

### Signup.php

```
<form class="form-signup" action="includes/signup.inc.php" method="post">
```

### Signup.inc.php

```
2 // Check if this page was accessed from
3 ~ if (isset($_POST['signup-submit'])) {
4
5     include "dbh.inc.php";
6     // Assigning variables to data collected from the sign up
7     form ("POST" is case sensitive)
8     $fname = $_POST['f_name'];
9     $lname = $_POST['l_name'];
10    $username = $_POST['uid'];
11    $email = $_POST['email'];
12    $password = $_POST['pwd'];
13    $passwordrepeat = $_POST['pwd-repeat'];
```

### POST method

The 'post' method is used to transfer and store user data entered from the sign up page to the database securely.

## ERROR HANDLERS

There are four error handlers in this prototype signup form. The empty fields error handler (checks whether any input fields are empty) and the password check error handler (where the password repeat does not match the first password) have not been stepped through in this section as their methods and reasoning for errors are the most straightforward.

```
/// Check if there are any underscores in the username
else if (preg_match("/^[a-z]+_[a-z]+$/i", $username)) {
    header("Location: ../signup.php?error=uidunderscores&f_name=".$fname."&l_name=".$lname."&mail=".$email);
    exit();
}
```

### Underscores in Username Error (prevention)

As underscores are commonly used in php, underscores in strings in variables may become an issue as in future development. E.g. displaying the username on a users' dashboard, posting comments etc. Therefore, using a pregmatch code (where an underscore is present in the middle of a string with two blocks of text on either side), the user will be unable to create an account with a username containing an underscore. Other data that has been entered will be present in the form when the user receives the error message for efficiency.

```
39 // Check if the username is already taken
40 else {
41     // Create a prepared SQL statement
42     $sql = "SELECT uidUsers FROM users WHERE uidUsers=?";
43     $stmt = mysqli_stmt_init($conn);
44     // Check if there is a connection error
45     if (!mysqli_stmt_prepare($stmt, $sql)) {
46         header("Location: ../signup.php?error=sqlerror");
47         exit();
48     }
49     // search the database for the same username and store
50     // into the 'stmt' variable
51     else {
52         mysqli_stmt_bind_param($stmt, "s", $username);
53         mysqli_stmt_execute($stmt);
54         mysqli_stmt_store_result($stmt);
55
56         // Check that there are no rows containing the
57         // username
58         $resultCheck = mysqli_stmt_num_rows($stmt);
59         // result check is greater than zero, then username
60         // is taken
61         if ($resultCheck > 0){
62             header("Location: ../signup.php?error=usertaken&mail=".$email);
63             exit();
64         }
65     }
66 }
```

### Check if username is already taken

To do this, an SQL search query into the database is used to find any matching usernames. This is completed by creating a mysqli prepared statement, binding the username to the binding parameter (these are explained in depth in the 'mysqli, prepared statements, security' section), and counting the number of rows returned from the query using the 'mysqli\_stmt\_num\_rows' function. Then, using an if-statement and a Boolean, if the search returned any rows, the username has been taken, and the user is alerted of this on the sign up page.

## USING MYSQLI, PREPARED STATEMENTS, SECURITY

**Using MySQLi to query the database** MySQLi is used for database queries instead of MySQL or PDO. This is because it is faster, and allow for the use of prepared statements, making it more secure.

### What is SQL Injection?

SQL injection occurs when SQL code is entered into an input field, which is entered directly into the SQL query. The system is unable to differentiate the SQL insertion and executes it as part of the SQL.

### Inserting into the database using prepared statements

Prepared statements were used to process SQL results because they reduce parsing time as the query preparation is only done once, they also prevent against SQL injection. Prepared statements are statements that contain an SQL query, but with changing variables which are called bound parameters, using the placeholder: '?' An example of this from the signup.inc.php page:

```
$sql = "INSERT INTO users (uidusers, fname, lname, emailusers, pudusers) VALUES (?, ?, ?, ?, ?)";
```

The SQL query remains untampered with and is sent to the server first, without any data in it.

```
$stmt = mysqli_stmt_init($conn);
if (!mysqli_stmt_prepare($stmt, $sql)) {
    header("Location: ../signup.php?error=sqlerror");
    exit();
}
```

The statement is 'prepared', sent to the server using the connection initiation. An error handler is used to check whether this step has been completed correctly.

```
$hashedPwd = password_hash($password, PASSWORD_DEFAULT);
```

Before the data can be submitted to the server, the password needs to be hashed (encrypted) using bCrypt. This is so that if hackers enter the database, the passwords are encrypted and cannot be seen. Furthermore, it provides privacy for the users from the admin.

```
mysqli_stmt_bind_param($stmt, "sssss",
$username, $fname, $lname, $email,
$hashedPwd);
// Insert user data into the database
mysqli_stmt_execute($stmt);
```

Then, the data is 'binded' to the parameters from the SQL query. The 'sssss' tells that the datatype of all the data entered are strings, and therefore cannot be merged with the SQL query. Then the statement (connection to the database) is executed, and the bound parameters replace the placeholders, completing the query and inserting the data securely into the database.



GALLERY SEARCH

As more artworks are added, the search/filter fields will remain static on the page as the user scrolls for actionability and ease of use.

As the user enters search terms, the results will load real-time  
Information about the artwork (including locations if logged in as a student) appear on load, and can be filtered using the search

Search the collection		Image	Artwork Title	Artist	Media	Location
Painting			Rescue the Peak	David Allen	Acrylic Painting	G14
Filters			Women in Black	Elia Fry	Painting Portrait	M113
Artists			The Landscapes	Elia Fry	Painting	M110
Media			Shipwreck	T W Jenner	Painting	M110
C Block Building			Evolution	Irene Amos	Oil Painting	M202
J1 Block			Primitive	Irene Amos	Oil Painting	C301
C Block						
O Block						

Drop down filters will work in a fully developed version but could not be completed due to time constraints. Adapted from codepen.io (reference in code)

A pop-up box containing an enlarged image and description of the artwork should show when the artwork is clicked.

To search the collection, jQuery filter search was used. This is explained on the coding pages.

```
107 <?php
108 require_once 'connect.php';
109 /// SQL required to access the data in the Artworks database using meekroDB
110 $allartworks = DB::query('SELECT aw_title, aw_artist, aw_pic, aw_media, aw_location
111 > FROM Artworks ORDER BY aw_artist');
112 > /// Construct the table...
113 > /// Only display the 'Locations' header if the user is a student...
124 > /// Check if the user has signed in using a student account:
125 if ($uType == 'STUDENT') {
126     /// Uses iteration to print the rows into the search table
127     foreach($allartworks as $row) {
```

The above php code was used to query the database to display the artworks. Using the ‘foreach’ function, each record in the artworks table was displayed. The ‘locations’ field was only displayed if the user was a student (this was tested using an if-Boolean statement)



Grid to Block(responsiveness)

Gallery layout display is set to ‘grid’, using percentages to define the size of the search and results table sections  
The grid layout is changed to ‘block’ once the screen’s width is below 800px. This is so that the elements stack like they would without the grid formatting.

```
.gal_layout {
    display: grid;
    grid-template-columns: 30% 68%;
    column-gap: 2%;
}

/* Responsive code for the Gallery */
@media only screen and (max-width: 800px) {
    .gal_layout {
        display: block;
    }
}

/* ... */
```

This method is also used for the locations and signup pages.

LOCATIONS (STUDENT VIEW)

The School Map uses the ‘image map’ html abilities. Each building is clickable using ‘area’ tags. As these area tags do not have stylistic features, David Lynch’s jQuery plugin MapHighlight is added to improve understandability and accessibility, as when each building is ‘hovered’ over, the building will be shaded to alert the user that they are able to click on it.

Artwork display  
An in-element scrollable card layout is used to display the artworks in the selected building. Just the name of the artwork and its location is displayed in ascending room number.  
Eventually when each card is clicked, more information about the artwork will be displayed in a lightbox. However due to coding and time constraints this was not achieved.



Future Development

Currently, separate pages containing different SQL queries for artworks in different buildings are used. In future development, only one additional page should be required, with different queries being used when the user clicks on building. This is to improve the efficiency of the code. Php code segments are only included when used/required, decreasing the loading time of the locations page



### Find Us

Brisbane Girls Grammar School address:  
Gregory Terrace,  
Brisbane QLD, 4000  
Australia

## Google Maps API

### Integration of Google Maps

For general users, their view of the locations page will have an integrated google map showing the location of the school. This is so that prospecting artists and donors can find the school. The google maps integration is built on top of an ‘iframe’ and uses a google maps APK to load the map.

```
<div class="gen_loc_cont">
  <div class="mapouter"><div class="gmap_canvas"><iframe
    width="523" height="351" id="gmap_canvas" src="https://
    maps.google.com/maps?q=Brisbane%20Girls%20Grammar%20School&t=&
    z=15&ie=UTF8&iwloc=&output=embed" frameborder="0"
    scrolling="no" marginheight="0" marginwidth="0"></iframe></
    div><style>.gmap_canvas {overflow:hidden;
    background:none!important;}</style></div>

  <div class="side_text">
    <h2>Find Us</h2>
    <p><i>Brisbane Girls Grammar School address: </i><br>
      Gregory Terrace, <br>
      Brisbane QLD, 4000 <br>
      Australia
    </div>
</div>
```





# DATA SECURITY AND FACILITIES VIEW

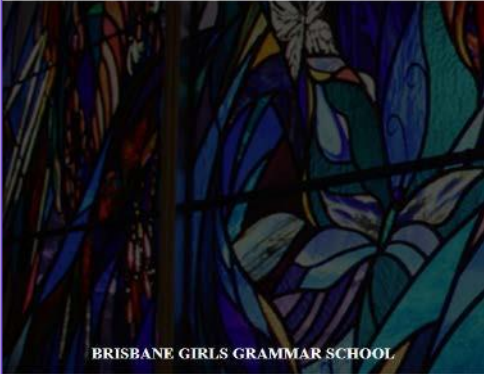
## Improvements: Filters

A series of filter dropdowns are included for the gallery search for both the administrators and facilities users. These dropdowns currently do not work, but can be technically implemented in future iterations of the digital solution. These filters would further improve usability and utility of the webpage.

Hamburger menu icon to symbolize the menu

 GALLERY SEARCH





BRISBANE GIRLS GRAMMAR SCHOOL

### Gallery Search

Filters





Artist

Media

Building

General search

Result

Title	Artist	Media	Location	Image
Women in Black	Ella Fry	Painting Portrait	M118	
The Landscape	Ella Fry	Painting	M110	
Shipwreck	I. W. Jenner	Painting	M110	
The Two Sisters, On the Terrace	Pierre-Auguste Renoir	Print	C301	
Evolution	Irene Amos	Oil Painting	M202	

Home

Gallery Search

Locations

User Profile

Logged in as:FACILITIES

LOGOUT

## Responsive web interfaces

The gallery search and responsive interface UI were built with the mobile platform being the primary access method. While the facilities staff may work mostly on laptop, centering the visual communication of elements on the gallery search and header on mobile design made the pages cleaner and more user-friendly, thus improving learnability. The gallery search was organized using a grid format, so that when the page was expanded, the artworks table and filters would be beside each other. This makes the page adaptable as its components are flexible.

## Pages from the previous project

This digital solution has been built on top of an existing digital solution, which had included a sign up and locations page. These pages were not explored or optimized in this task. Although BGGGS issues credentials the to students, facilities and administrators, general users will need to sign up to access the database using the signup form. As this user group was not investigated, the signup page will not be explored. The locations page was unnecessary for the task but was kept.

## mysql Prepared statements

When meekroDB is not used, mysql prepared statements were written manually to protect against SQL injection. This is done by using placeholders for the data to be inserted, to indicate that they are a value only – bound to parameters (e.g. ‘s’ for string).

```
if($table == 'Artists'){
    $fname = $_POST['f_name'];
    $lname = $_POST['l_name'];
    // Check whether all the fields are filled out
    if (empty($fname) || empty($lname)){
        header("Location: ../create.php?table=Artists&error=emptyfields&f_name=".$_f_name."&l_name=".$_lname);
        exit(); } else {
        $sql = "INSERT INTO ".$table." (a_fname, a_lname) VALUES (?, ?)";
        $stmt = mysqli_stmt_init($Link);
        if (!mysqli_stmt_prepare($stmt, $sql)) {
            header("Location: ../create.php?error=sqlerror");
            exit();
            /// All error handlers are cleared
        } else {
            // Bind parameters to insert into the database
            mysqli_stmt_bind_param($stmt, "ss", $fname, $lname)
        ;
        /// Insert user data into the database
        mysqli_stmt_execute($stmt);
        if(isset($return)) {
            echo "<script>>window.close();</script>";
        } else {
            header("Location: ../../dashboard.php?table=Artists")
        ;
            exit(); } } }
```

## The login form (and its protection)

The login form input is in the header of the website and can be accessed on the index and gallery search pages. This improves accessibility as the user will find it easy to login. If left unprotected, input fields connected to the database may make the database and its data vulnerable to SQL injection attacks. A few defense methods are shown below.

Username/email

Password

LOGIN

Mysqli\_escape\_string changes all ‘special’ characters in the input field into regular characters. This is to prevent SQL injection.

```
// retrieve and declare variables from the submitted form
$mailuid = mysqli_escape_string($conn, $_POST['mailuid']);
$password = mysqli_escape_string($conn, $_POST['pwd']);

$passwordCheck = password_verify($password, $row['pwdUsers'] );
```

Passwords in the database are scrambled with Bcrypt and will require the ‘password verify’ function to unscramble.

## Protecting artwork data from non-administrators

To Protect the data, the administrator dashboard and CRUD pages are protected so that they may only be loaded when the user is signed in as an admin. To achieve this, once the user is logged in, the ‘UserType’ value is retrieved from the database and is assigned as a session variable. If the user is not an administrator, they are redirected to the index page.

## ../Login.inc.php

```
if ($row['UserType'] == 'ADMIN'){
    $_SESSION['userType'] = $row['UserType'];
    $_SESSION['idUsers'] = $row['idUsers'];
    header("Location: ../dashboard.php?table=Artworks");

} else {
    header("Location: ../index.php??login=success"); }
exit();
```

## ../header\_admin.php

```
<?php
session_start();
if ($_SESSION['userType'] != 'ADMIN') {
    header('Location: ../index.php?NotSignedIn');
    exit();
} ?>
```

## Using the MeekroDB php mysql Library

The MeekroDB library has been used to increase efficiency in coding, while maintaining good data security against mysql injection among other hacks. It simplifies the code and reduces the likelihood for errors and bugs in the code. This proved to be especially useful when inserting or updating artwork records, which handled dates and blobs efficiently and with no issues. Traditional mysql queries and functions were used for search queries as it allowed for greater flexibility e.g. creating and using temporary tables.

## User Privacy

BGGGS issues credentials and login details for the facilities staff, which cannot be altered. However, to adhere to the Australian Privacy act, users can view their user details stored on the database, and change their password. In future development, a ‘forgot password’ feature will be implemented to further comply with the Privacy Act.

User ID	4
Username	Facilities
First Name	Peter
Last Name	Smith
Email	Facilities@facilities.b
Current Password	
New Password	
Confirm Password	

Update Details

View Privacy Statement



# DYNAMIC LOAD & TEMPORARY TABLES

## Using Temporary Tables

A temporary table was created to execute the search query. The Artworks table that is displayed to the user contains the artist, media and location names which are retrieved from their respective tables and joined with the artwork ID, title and image from the artworks table stored in the database. This can be made into a temporary aggregate table that is queried and displayed. As the number of artworks added increases, this method of searching is preferred over a toggle hide-show method (e.g. jQuery). Furthermore, the searched phrase/word could be identified in all fields, and not just one field, as without a temporary table being created, there would be conflicting operators(AND, OR and LIKE). Due to time constraints, the temporary table search method was only used for the Artworks admin search, with jQuery used for all other tables.

### Search Algorithm (for Artworks Table)

```
BEGIN Program
SET UserSearch = INPUTsearchQuery
IF UserSearch IS NOT NULL
SET connection = CONNECT to database
SET MakeTempTable = “CREATE TEMPORARY TABLE temptable(
    `temp_id` INT PRIMARY KEY AUTO_INCREMENT, `ID` INT(11),
    `Title` varchar(255), `Artist_Name` varchar(255),
    `Media` varchar(255), `Location` varchar(255),
    `Image` longblob”
SET InsertTempTableData = “INSERT INTO temptable(`ID`, `Title`,
`Artist_Name`, `Media`, `Location`, `Image`)
    SELECT a.a_id, a.a_title, CONCAT(r.a_fname,' ', r.a_lname),
m.m_name, l.room, a.img
    FROM artworks AS a, artists AS r, locations AS l, media AS m
    WHERE a.artist_id = r.artist_id AND a.l_id = l.l_id AND a.m_id =
m.m_id”
EXECUTE MySQLi query(connection, MakeTempTable)
EXECUTE MySQLi query(connection, InsertTempTableData)
SET SearchSql= “SELECT ID, Title, Artist_Name, Media, Location, Image
FROM temptable WHERE
    Location LIKE ‘%” UserSearch “%’ OR Media LIKE ‘%” UserSearch “%’
OR
    Artist_Name LIKE ‘%” UserSearch “%’ OR Title LIKE ‘%” UserSearch
“%””
SET SearchResults = EXECUTE MySQLi query(connection, SearchSql)
IF NumberOfRows IN SearchResults > 0
    WHILE row = fetch associative row in SearchResults
        WRITE row[‘ID’]
        WRITE row[‘Title’]
        WRITE row[‘Artist_Name’]
        WRITE row[‘Media’]
        WRITE row[‘Location’]
        SET encodelmg = ENCODE TO BASE 64 row[‘Image’]
        WRITE encodelmg
    ENDWHILE
ELSE WRITE “0 Results Found”
CLOSE connection
END Program
```

ART GALLERY ADMINISTRATION PANEL

USER PROFILE

LOGOUT

Tables

Artworks

Artists

Media

Locations

Filters

Artist

Media

Building

Artworks

Add New

General Search...

Q

C

Results

ID ↕	Title	Artist	Media	Location ↕	Image	Actions
1	Women in Black	Ella Fry	Painting Portrait	M118		
2	The Landscape	Ella Fry	Painting	M110		
3	Shipwreck	I. W. Jenner	Painting	M110		
4	The Two Sisters, On the Terrace	Pierre-Auguste Renoir	Print	C301		
5	Evolution	Irene Amos	Oil Painting	M202		
6	Primitive	Irene Amos	Oil on Linen	C301		
7	The Race	Mary Norrie	Watercolour	C301		

First

<

1

2

3

4

>

Last

## Decreasing Load Times with Pagination

A dynamic table loading system was used, where only a set number of rows were retrieved from the database for the results that were displayed on the page (e.g. 6 like above). To do this, an ‘offset’ variable is created to determine which record to start retrieving to display. These are retrieved using the intermediate temporary table.

```
$offset = ($pageno - 1) * $no_of_records_per_page;
```

First < 1 2 3 > Last						‘First’ and ‘Last’ tabs allow users to easily switch pages. The ‘active’ page was highlighted to increase usability.
----------------------	--	--	--	--	--	--

## Flexible searching

By using an sql-search method, placeholder ‘%’ can be used in the search, which can be used to search specific words and not just concatenated characters, or to search for two similar words. E.g. To find paintings and prints, the search query “P%int” can be used. The find artworks starting with ‘The’, the search query ‘The %’ is used. Search is non-case sensitive to retrieve as many results that satisfy the search query.









Add New

P%InT

Q

C

Results

ID ↓ <sub>1</sub>	Actions	Title	↓ <sub>2</sub>	Artist	↓ <sub>2</sub>	Media	↓ <sub>2</sub>	Location ↓ <sub>2</sub>	Image
3	  	Shipwreck		I. W. Jenner		Painting		M110	
4	  	The Two Sisters, On the Terrace		Pierre-Auguste Renoir		Print		C301	













Add New

The %

Q

C

Results

ID ↓ <sub>1</sub>	Actions	Title	↓ <sub>2</sub>	Artist	↓ <sub>2</sub>	Media	↓ <sub>2</sub>	Location ↓ <sub>2</sub>	Image
2	  	The Landscape		Ella Fry		Painting		M110	
4	  	The Two Sisters, On the Terrace		Pierre-Auguste Renoir		Print		C301	
7	  	The Race		Mary Norrie		Watercolour		C301	

## Sorting the retrieved records

To further optimize the search results and improve

When order button is clicked in each field, the entire table will be sorted alphabetically by that field. A tooltip is visible on hover for each button to improve accessibility so that users can identify which field they are sorting by. This is defined by passing the variable ‘selOrder’.

Sort by Artist		
Artist	Media	Location
Bob Ross	Oil F	
Daphne Mayo	Sculp	
Davida Allen	Inkjc	

```
// Search query executed on the temporary table us
ing the escaped string search query
$searchSql = "SELECT ID, Title, Artist_Name, Medi
a, Location, Image FROM temptable WHERE
    Location LIKE ‘%” $searchQuery .“%’ OR
    Media LIKE ‘%” $searchQuery .“%’ OR
    Artist_Name LIKE ‘%” $searchQuery .“%’ OR
    Title LIKE ‘%” $searchQuery .“%’ ORDER BY
$selOrder LIMIT $offset, $no_of_records_per_page";
```

```
$searchtable = mysqli_query($conn, $searchSql) or
trigger_error(mysqli_error($conn));
```

## Secure search

Usage of ‘mysqli\_real\_escape\_string’ to protect against sql injection, as it ‘escapes’ special characters

```
$searchQuery = mysqli_real_escape
_string($conn, $_GET[‘aInput’]);
```

By searching the temporary table, if SQL injection is attempted in the search bar, any ‘damage’ caused will only affect the temporary table, leaving the real tables untouched. This is a further protection measure to protect the data if the other security measures are by-passed. As shown below, special symbols are converted to generic numbers and symbols:

localhost/bggs\_arts/dashboard.php?alnut=%27"%3B%29%2C&table=Artworks

Assigning a session variable to the inputted search query, so that the results retrieved can be returned to easily from different CRUD pages, improving efficiency and utility.


```
$_SESSION[‘aInput’] = $_GET[‘aInput’];
```



# CRUD SYSTEM

## View Artworks Record

Record successfully updated!

ID	Year	Image
52	1985	
Title	Purchase Date	
Meadow Stream	2015-04-27	
Artist	Artwork Value	
Bob Ross	\$1000.00	
Media		
Oil Painting		
Location		
S101		

### Sample code: displaying images

To convert images to a viewable format from the database, they are encoded using 'base64' before being displayed.

```
<label>Image</label>
<p class="form-control-static"><img class="read_img" style="max-width: 400px; max-height: 400px;" src="data:image/jpeg;base64,'.base64_encode($row['Image'])'></p>;
```

### Verifying record details

When a record is created or updated successfully, they are automatically redirected to the read page, to show that their form request was successful. The use of the bootstrap alerts further emphasized this, improving the accessibility and safety, reassuring the users that they have filled out the form correctly and have been successful.

### Always able to backtrack

To improve safety, and accessibility, all pages have a 'back' button, so that the user can return to their search results/default table if they accidentally clicked something.

### Drag and Drop & Error improvements

The 'choose file' option has drag and drop capabilities. In future development, a larger 'drop zone' can be implemented to improve accessibility and efficiency. Furthermore, a future improvement could be to test the validity of the images that are uploaded before the entire form is submitted to maximise safety and learnability.

## Create Artists Record

First name

Last name

ADD RECORD

CLOSE

## Adding new Artists, Media and Locations

Users have the option to add new artists, media and locations into the database. This can be done on their respective table pages or directly from an artworks update or create form, which will open a new window with their respective forms. If these pages are accessed from the artworks update/create forms, a 'close' button appears, and the add record button automatically closes the window if the creation is successful. This is because users cannot add new data directly into these fields on the artworks page, as they are linked to other tables. This decreases data redundancy and repetition, for easier categorization, improving efficiency, effectiveness and usability.

## Update Artworks Record

Artwork Title

Meadow Stream

Year

1985

Artist

Bob Ross

Add New

Date

04/27/2015

Media

Oil Painting

Add New

Artwork Value (to nearest dollar)

\$ 1000

.00

Location

S101

Add New

Choose file

Blue...r.jpg

Image must be under 200 KB  
Image must be a .gif, .jpg or .jpeg

New Image



Existing Image



UPDATE RECORD

Back

### Autofill suggestions

Autofill suggestions enhance users efficiency, making it faster to enter information. It also minimises typos, enhances safety and ensures data entered into the database is sanitised. User efficiency is enhanced by the user of a dropdown bar, making it faster to enter information. User efficiency as also enhanced through the use of a date picker for the purchase date, so the user can select the date on the calendar rather than typing it out, and possibly making mistakes.

```
function previewFile() {
    var preview = document.querySelector('img[name="preview"]');
    var file = document.querySelector('input[type=file]').files[0];
    var reader = new FileReader();
    reader.onloadend = function () {
        preview.src = reader.result;
    }
    if (file) {
        reader.readAsDataURL(file);
    } else {
        preview.src = "";
    }
}
```



# TESTING TABLE

Function Test inputs, outputs, and processes	Example questions	Yes/ No	Notes: What was the expected result vs the actual result of the test? How can you fix the errors if relevant?
<b>Media</b> Images	<ul style="list-style-type: none"><li>Do pictures show as intended?</li><li>If the images are unable to load, does the alt text appear?</li></ul>	Yes	All images showed as intended in search browsers Safari, Google Chrome, and Firefox. All images that were given alt text (with the exception of artwork photos in the database) were displayed as intended in different search browsers.
<b>Loading speed of webpages</b> To see if the loading times are fast enough, the Google Chrome extension Lighthouse will be used. <a href="#">The full lighthouse assessment can be found in the Appendix – Lighthouse Assessment</a>	<ul style="list-style-type: none"><li>Check the page loading speed if more than a few seconds page content needs to be optimized (e.g. smaller images)</li></ul>	Yes	The art gallery management system was said to have a performance of 100% and had a speed index of 0.9s indicating the website pages load quickly. However, the pages load time could be further decreased by using properly sized images.
<b>Formatting of the pages (CSS)</b> What will be tested: Usability, accessibility, appearance, also check readability and clarity. These should all adhere to the prescribed criteria (by following the BGGs style guide) and the self determined criteria. *CARP Principles (Contrast, alignment, repetition, proximity)	<ul style="list-style-type: none"><li>Are the fonts readable by different users?</li><li>Does the website design adhere to the BGGs style guide?</li><li>Are headings consistently styled and effective in enhancing communication?</li><li>Does the website follow CARP design principles?</li><li>Is the layout appealing and working as intended?</li><li>Is there continuity between pages?</li></ul>	Yes	A sense of continuity throughout the website is created using the same header and footer on each page as well as adhering to the BGGs style guide to ensure the website represented the BGGs brand. Adhering to the BGGs brand included using the Times New Roman serif font. The use of a serif font also enhanced user accessibility as it has been proven that serif fonts are easier to read. To further ensure text legibility, it was ensured that each text and its corresponding background at least had a colour contrast ratio of 5:1. This allows the website to easily be viewed by users with vision impairments such as colorblindness and poor vision. This adherence to accessibility guidelines is evident in the Google Chrome extension Lighthouses assessment that the website was 100% accessible. The pages layout was then tested using 3 different search browsers Safari, Google Chrome and Firefox. Upon testing, it was evident that the websites CSS behaved the same across different platforms.
<b>Buttons</b> To see that if the buttons link to the correct pages and the correct functions.	<ul style="list-style-type: none"><li>Do all the buttons link to the correct pages?</li><li>Do all the buttons have the correct CSS styling</li></ul>	Yes	When testing the buttons, it was found that they all linked to their intended page. It was also checked and confirmed that each button on the website was formatted with the correct CSS code.
<b>Breakpoints</b> Do all the elements adjust in size and layout with the chosen breakpoint of 480px in width for the facilities page.	<ul style="list-style-type: none"><li>Does the page allow for vertical scrolling?</li><li>Are all images appropriately sized?</li><li>Are fonts readable to different users?</li><li>Does the layout look appealing and work as intended?</li></ul>	Yes/ No	The index.php and gallery_dashboard_facilities.php are both responsive to a width of 480px ensuring Facilities staff can use the art gallery management system on their mobile phones. The layout looks appealing and is presented as intended and the fonts are large enough to be legible to a wide variety of users. However, the images in the table are very small and cannot be enlarged, leaving them useless to the facilities staff.
<b>Forms</b> To see if forms validate user entries and only store sanitized data.	<ul style="list-style-type: none"><li>Does the login form work if the user enters their username and password or email and password?</li><li>Does the create and update form validate data to ensure all data entered into the database is sanitised?</li></ul>	Yes	The login form located in the header of the website works if the user enters either their username or emails as well as their password. The create form only allows sanitized data into the database, displaying error messages if required input boxes are left blank, the wrong data type is inputted, the image size is too large, or if the image is in the wrong file format (JPG and gif allowed). The update page also ensures only sanitised data enters the database by only allowing the user to change the 3 fields that possibly would have to be edited overtime. Similarly to the create form, if the user leaves an input box blank or enters unallowed characters, error messages are displayed on the screen alerting them to this.
<b>Error messages</b> To see if all errors are displayed on the webpage or in the URL.	<ul style="list-style-type: none"><li>Do error messages in the URL and within the website appear?</li><li>Do they explain what caused the error?</li></ul>	Yes	All error messages displayed in form help blocks and the websites URL telling the user what error has occurred and how it can be mitigated (e.g. enter a positive integer value).
<b>Access level</b> To check if each user is viewing the permitted information on the artworks.	<ul style="list-style-type: none"><li>Do users have to log in before viewing the gallery page?</li><li>Are admin and facilities users sent to the correct dashboard pages</li></ul>	Yes	When the user is not logged in and clicks on the gallery page, they will be prompted to log in so they can gain access to the gallery. Once the user has logged in there are sent to their correct page views, where the admin is directed to a CRUD system (excluding the delete component), and the Facilities staff are sent to the facilities dashboard where they can view key information about the artwork.
<b>Search bar text input</b> To see if users can search through the data table without errors.	<ul style="list-style-type: none"><li>Does the table format stay the same?</li><li>Does it show all data including the terms that are searched by the user?</li></ul>	Yes	Both Admin and Facilities Users are able to search all the table fields for information. Tests also confirmed that the search bar is also not case sensitive.



# EVALUATE

The final prototype generated for the art gallery management system would have an overall positive impact. To ensure that the website would not be socially negatively impacted, methods of SQL injection including prepared statements were used. The webpage also provided a positive economic impact for the school as the administrator can easily access art pieces value for insurance purposes and can modify the value of the artwork if it was reappraised at a different value. The use of relational tables reduced the negative economic impact of the website as it ensured that there was no data redundancy or anomalies, minimising the cost of maintaining the database.

The final prototype also provides easier access to artwork files for both the admin and facilities user, adheres with copyright laws, but does not adhere to the Australian Privacy Act (1988). The lack of adherence to the Australian Privacy Act has a negative personal impact as users' passwords are not encrypted, leaving users private information vulnerable to hackers on the school's artGallery database. Both the admin and facilities users also are not presented with the option to change their password, so if their password is forgotten, they will lose access to the whole system. In generating the prototype there were also many errors and action was taken to resolve them.

The main issue faced was inserting the create record form data into the two different tables (artist and artworks tables). This was mitigated by creating different create record pages, one for the artist and another for artwork records. The second problem faced was that the artwork's artist could not be inputted into the form as when inserting the information into the database, the artist's first name and last name needed to be inserted into a different table than all the other artwork information. This was fixed by allowing the administrator to only reference the artist in the form by their artist id. To find an artist's id. A searchable table was placed next to the form so the administrator could find the artist's corresponding artist id.

The final issue faced was ensuring the purchase date was displayed in the format dd/mm/yyyy as in the database, the data was stored in the American date format. This issue was resolved by using the SQL DATE\_FORMAT() function. After resolving these issues, it was evident that the art gallery management system adhered to the majority of prescribed and self-determined criteria were fulfilled including ensuring the website is accessible and adhered to the BGGs style guide. The set prescribed criteria to adhere to the Australian Data Privacy Act, however, was not adhered to as both admin and facilities users are unable to change their password or any other account details. The self-determined criteria to encrypt passwords and use modal images in the artGallery table were also not fulfilled due to coding constraints.

## ADHERENCE TO THE AUSTRALIAN PRIVACY ACT

As users are required to input personal data to create an account, the web app must adhere to the Australian privacy principles (APPs). The Australian government's Office of the Australian Information Commissioner categorises these principles into five main sections:

- **Consideration of personal information privacy (APPs 1 and 2)**

Users can view the privacy statement on their user profile page. "User data that is filled into this form is handled in a secure way and is stored on a database so that users may log in again", as well as informing the user that the art collection adheres to the privacy act. Members of the BGGs staff faculties and students are required to use their allocated login with credentials issued by BGGs.

- **Collection of personal information (APPs 3, 4, 5, 6)**

Users agree to the following statement: "By submitting this form and using this website when logged in, you solicit the using of this information within the BGGs Art Collection that is moderated by administrators" when updating their passwords or creating a new account (for general users). Passwords stored on the server are encrypted.

- **(APPs 7-10 are not relevant).**

- **Integrity of personal information (APP 11)**

All user data will be stored safely on the databases, which are protected from SQL injection (APP 11), the user data is not used on any pages.

- **Access to, and correction of, personal information (APPs 12 and 13)**

Users are given the opportunity to access their personal data: to view what data is stored on the database (APP 12). They are given the opportunity to view their own details and credentials: including their first and last name, email and username. Users are also able to change their password (APP 13). Due to coding constraints, 'forgot password' feature is currently unavailable.

## ACCESSIBILITY EVALUATION



Silktide was used to evaluate the accessibility of the web app, as it is able to test for blindness, myopia, colour blindness, dyslexia and tunnel vision.

## FUTURE IMPROVEMENTS AND RECOMMENDATIONS

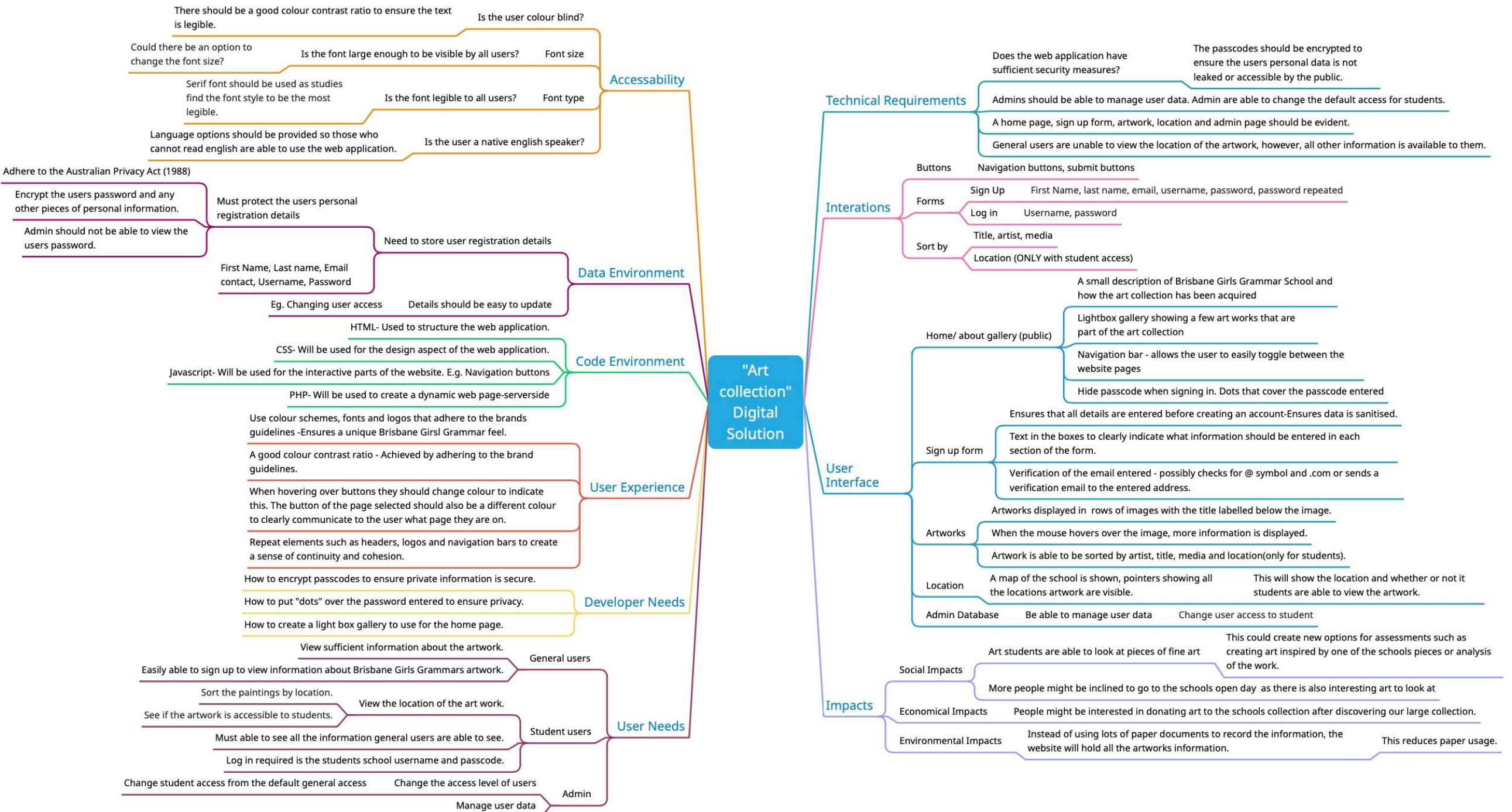
- The dropdown additional artwork information on the artworks table, as shown in the wireframe was not achieved. The idea was deemed impractical given the coding and time constraints but could be implemented in the future to enhance usability and utility.
- Ability to change the number of results displayed per page. This would increase utility and efficiency if users are wanting to view more rows at a time as the database grows.
- Change image upload to using a file directory method as it would be more efficient in the long run, as well as being able to accept larger and higher quality images, improving effectiveness.
- To improve accessibility, the ability to change the font sizes could be implemented, as they are currently quite small.
- Implement the ability to export search results in a .CSV file for external use from the database. This would improve utility.
- Be able to upload a CSV file directly into the database to increase efficiency and utility, as the user would not need to fill in a form individually for each artwork. They can just 'dump' the data into the database.
- A delete page has been constructed but was not explored in this task. Possibly a further implementation of deletion, with the option to keep the artwork but just delete the artist/media being possible.
- Having a fixed and uniform sized container for all artworks forms. Once an image was uploaded, size of the box changes. This may confuse users. To increase safety and effectiveness the box should be uniform regardless of the image being uploaded.

- **Non-mouse navigation:** All pages could be navigated using keys in an orderly and structured fashion.
- **Fields:** Have placeholder text so users know what the field requires. Labels above form input fields were placed for extra clarity.
- **Error messages (or validation messages):** The use of red and green message boxes, for errors and success respectively were used. Advice on issues were written.
- **Silktide tests:** Good colour contrasts -- web app was still understandable in black and white, as well as other colour-blindness tests. Pages were still understandable for users with myopia up to 20% blurriness. Page was still understandable for dyslexic users. Users with tunnel vision had regular access to the web app.
- **Page titles:** All pages have unique and relevant page titles that relate to the table being involved. E.g. Create: Artworks
- **Alt text:** All images have an alternative text for screen readers. This was tested using silktide.
- **Headings:** Each page has the heading displayed in large font at the top of the page, so users know which page they've clicked on immediately. The name of the table being used is also visible.
- **Zooming of pages:** While the gallery search is optimized for smaller screen sizes, the administration pages are not, and therefore there was side-scrolling when zoomed in on. Elements would bunch together and overlap



# Appendix

## MINDMAP 1: Gallery





# MINDMAP 2: Management System

