

Evidence Gathering Document for SQA Level 8 Professional Developer Award.

This document is designed for you to present your screenshots and diagrams relevant to the PDA and to also give a short description of what you are showing to clarify understanding for the assessor.

Please fill in each point with screenshot or diagram and description of what you are showing.

Each point requires details that cover each element of the Assessment Criteria, along with a brief description of the kind of things you should be showing.

Week 2

Unit	Ref	Evidence	
I&T	I.T.5	Demonstrate the use of an array in a program. Take screenshots of: *An array in a program *A function that uses the array *The result of the function running	
		Description:	

Paste Screenshot here

```
def initialize(name, type)
  @name = name
  @type = type
  @fish_stored = []
  @stomach = []
end

def remove_fish_from_river(river)
  return "No fish to remove!" if river.contents.empty? == true
  @fish_stored << river.contents.last
  river.remove_fish
end
```



```
def test_bear_take_fish_from_river
  @bear.remove_fish_from_river(@river)
  assert_equal([@fish_2], @bear.fish_stored)
  assert_equal([@fish_1], @river.contents)
end

def test_bear_take_fish_from_river__no_fish
  river_2 = River.new("Amazon")
  assert_equal("No fish to remove!",
    @bear.remove_fish_from_river(river_2))
end
```

```
[→ bear_river_fish git:(master) ruby specs/bear_spec.rb
Run options: --seed 35033
```

Running:

.....

Finished in 0.001531s, 5878.5108 runs/s, 6531.6787 assertions/s.

Description here

For this program there is a Bear class which has an attribute named fish_stored which is an array. The method remove_fish_from_river manipulates this array. The method takes a parameter 'river'. In the example above an instance of a River class is passed into the method. The method checks if the river attribute 'contents', which is also an array, is empty and if so returns a string. If it is not empty then the last element in rivers 'contents' is pushed/shovelled into the fish_stored array. The method then asks the river to call upon its own method remove_fish to remove the element from it's own 'contents' array. The tests above check the outcome of passing a river with elements in the 'contents' array and the outcome of an empty 'contents' array. Both tests are passed.

Unit	Ref	Evidence	
I&T	I.T.6	Demonstrate the use of a hash in a program. Take screenshots of: *A hash in a program *A function that uses the hash *The result of the function running	
		Description:	



Paste Screenshot here

```
class GenreTest < MiniTest::Test

  def setup
    @genre = Genre.new({'id' => '1', 'genre_name' => 'Test_Genre'})
  end

  def test_genre_exists
    assert_equal(Genre, @genre.class)
  end

  def test_genre_can_have_id
    assert_equal(1, @genre.id)
  end

  def test_genre_has_name
    assert_equal('Test_Genre', @genre.genre_name)
  end

end
```

```
def initialize(options)
  @id = options['id'].to_i if options['id']
  @genre_name = options['genre_name']
end
```

→ `record_shop_inventory git:(master) ruby specs/genre_spec.rb`
Run options: --seed 20123

Running:

...

Finished in 0.000864s, 3472.2222 runs/s, 3472.2222 assertions/s.

3 runs, 3 assertions, 0 failures, 0 errors, 0 skips

Description here

The above example of a hash is used to pass information into the new method for a class. The hash contains two keys (id and genre_name). The initializer for the Genre class takes in an 'options' parameter. The hash is passed in and the values of the keys are used to set the attributes of the Genre instance. The tests check that these attributes are set successfully and all the tests pass.



Week 3

Unit	Ref	Evidence	
I&T	I.T.3	Demonstrate searching data in a program. Take screenshots of: *Function that searches data *The result of the function running	
		Description:	

Paste Screenshot here

```
Park.prototype.findByMostVisitors = function () {  
  const mostVisitors = [];  
  for (let dinosaur of this.dinosaurs) {  
    if (mostVisitors.length === 0) {  
      mostVisitors.push(dinosaur);  
    }  
    else if (dinosaur.guestsAttractedPerDay > mostVisitors[0].guestsAttractedPerDay) {  
      mostVisitors.splice(0,1);  
      mostVisitors.push(dinosaur);  
    }  
  };  
};  
return mostVisitors;  
};|
```

```
it('should be able to find the dinosaur that attracts the most visitors', function () {  
  park.addDinosaur(dinosaur1);  
  park.addDinosaur(dinosaur2);  
  assert.deepStrictEqual([dinosaur2], park.findByMostVisitors());  
});
```

Park

- ✓ should have a name
- ✓ should have a ticket price
- ✓ should have a collection of dinosaurs
- ✓ should be able to add a dinosaur to its collection
- ✓ should be able to remove a dinosaur from its collection
- ✓ should be able to find the dinosaur that attracts the most visitors
- ✓ should be able to find all dinosaurs of a particular species
- ✓ should be able to remove all dinosaurs of a particular species
- ✓ should be able to calculate the total number of visitors per day
- ✓ should be able to calculate the total number of visitors per year
- ✓ should be able to calculate the total revenue from ticket sales for one year

14 passing (12ms)



Description here

The above code is a method belonging to a class named Park. An instance of a park object contains an array of instances of Dinosaur objects. Dinosaur objects contain an attribute named guestsAttractedPerDay. The method searches for and returns the Dinosaur object with the highest value of this attribute. The screenshots show the expected outcome and passing of the test.

Unit	Ref	Evidence	
I&T	I.T.4	Demonstrate sorting data in a program. Take screenshots of: *Function that sorts data *The result of the function running	
		Description:	

Paste Screenshot here

```
numbers = [3,7,3,82,4,2,8,9,55,4,3,5,7,8,4,39,20]

def sort_data(data)
  return data.sort()
end

puts sort_data(numbers)
```

```
[→ PDA Code ruby sort.rb
2
3
3
3
4
4
4
5
7
7
8
8
9
20
39
55
82
—
```

Description here

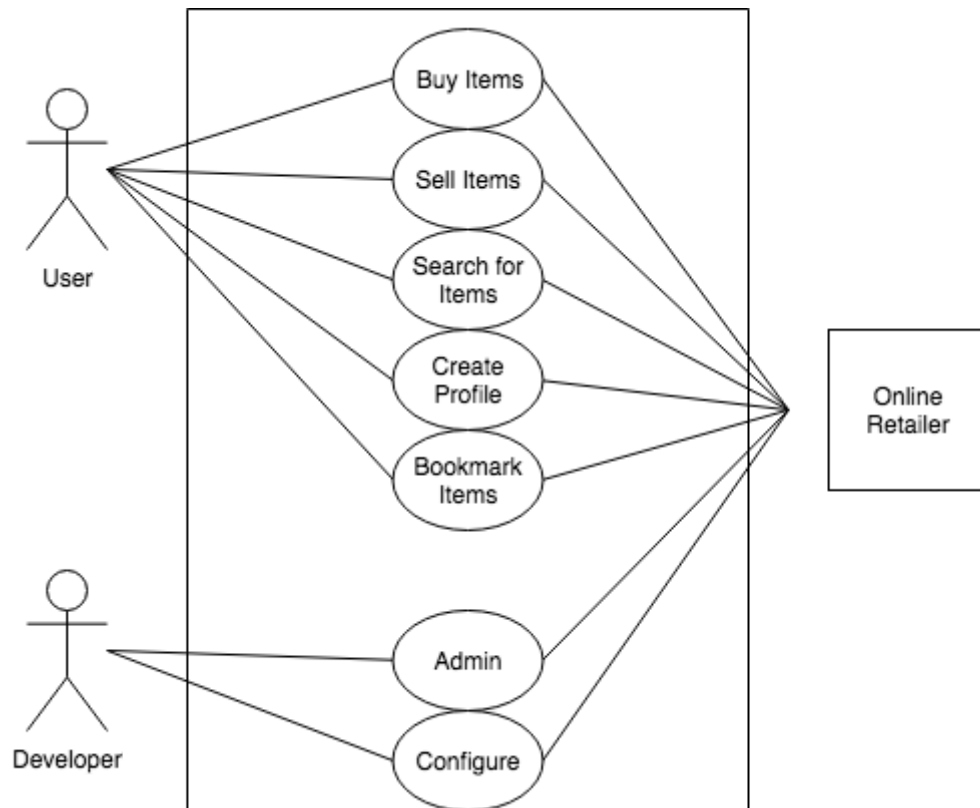
The sort_data function in the first screenshot above takes in an array of objects and returns the sorted array. The second screenshot shows the result of the method.



Week 5 and 6

Unit	Ref	Evidence	
A&D	A.D.1	A Use Case Diagram	
		Description:	

Paste Screenshot here

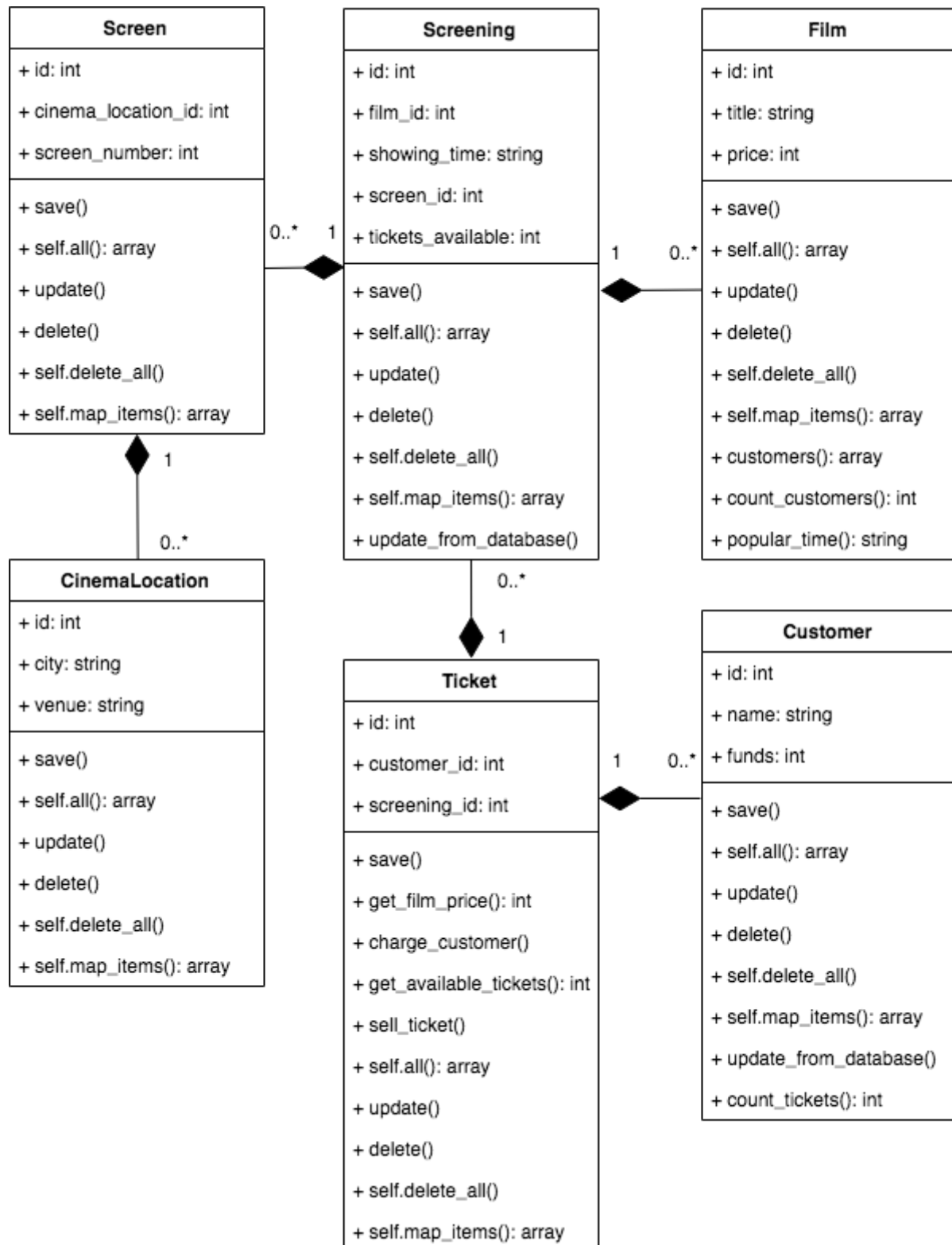


Description here

The above diagram displays the actions that both a user and a developer would need to perform using a system designed for an online retailer.

Unit	Ref	Evidence	
A&D	A.D.2	A Class Diagram	
		Description:	

Paste Screenshot here



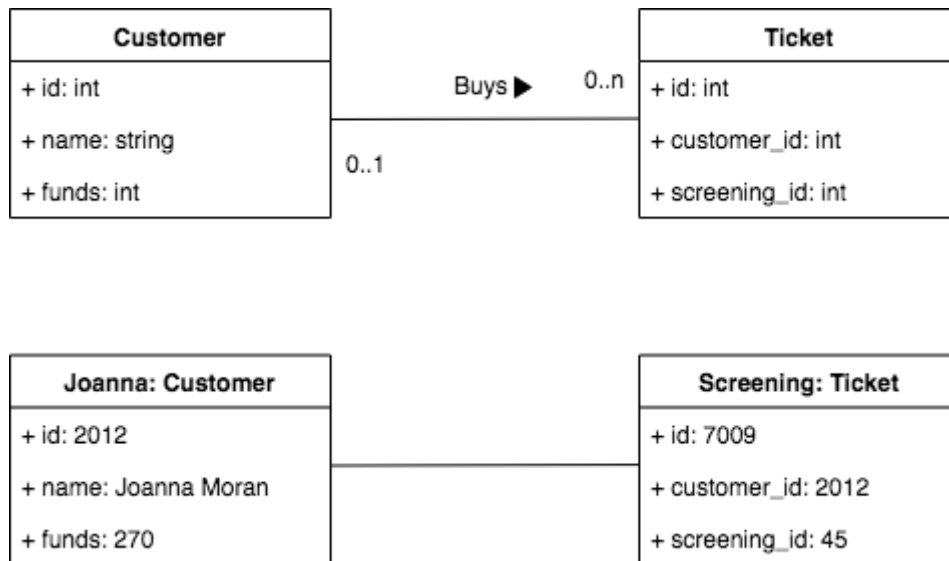
Description here

The above class diagram displays the classes for a cinema ticket ordering system including the methods and attributes of each.



Unit	Ref	Evidence	
A&D	A.D.3	An Object Diagram	
		Description:	

Paste Screenshot here

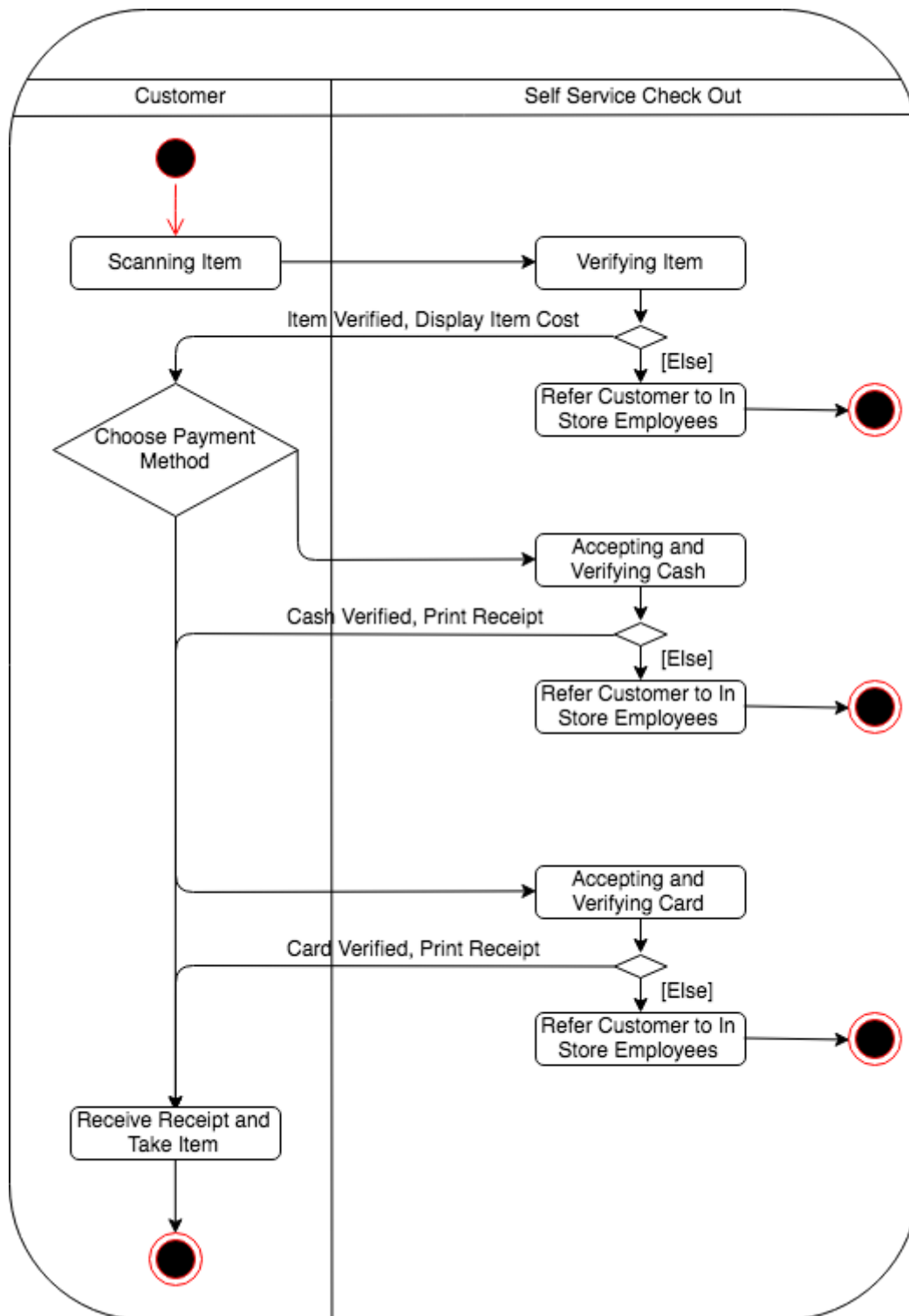


Description here

The above diagram displays object examples of a customer and a ticket for a cinema film ordering system.

Unit	Ref	Evidence	
A&D	A.D.4	An Activity Diagram	
		Description:	

Paste Screenshot here



Description here

The above diagram displays the actions available for a self service checkout in a supermarket.



Unit	Ref	Evidence	
A&D	A.D.6	Produce an Implementations Constraints plan detailing the following factors: *Hardware and software platforms *Performance requirements *Persistent storage and transactions *Usability *Budgets *Time	
		Description:	

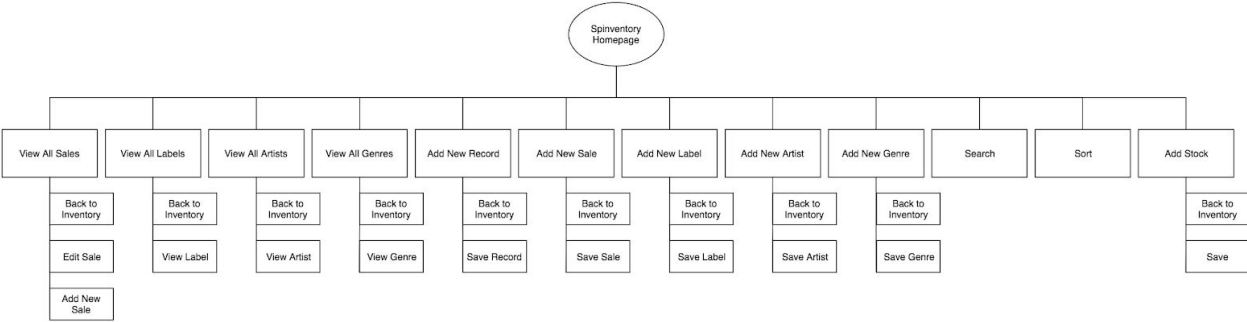
Paste Screenshot here

<u>Constraint Category</u>	<u>Implementation Constraint</u>	<u>Solution</u>
Hardware and software platforms		
Performance requirements		
Persistent storage and transactions		
Usability		
Budgets		
Time		

Description here

Unit	Ref	Evidence	
P	P.5	User Site Map	
		Description:	

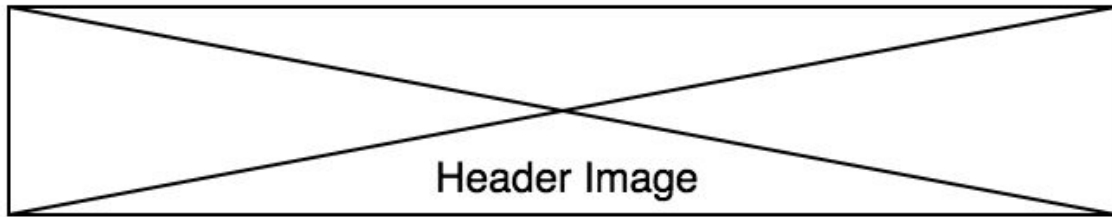
Paste Screenshot here



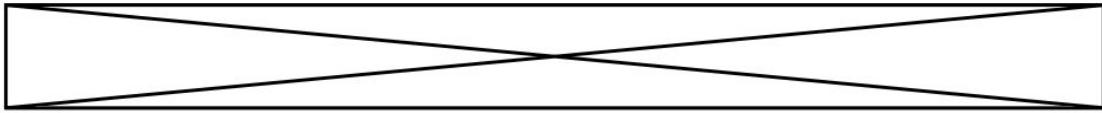
Description here

Unit	Ref	Evidence	
P	P.6	2 Wireframe Diagrams	
		Description:	

Paste Screenshot here



Album Art	Artist	Album Name	Stock	Price	
					Add Stock Edit Delete
					Add Stock Edit Delete
					Add Stock



Detail:

Detail:

Detail:

Detail:

Detail:

Detail:

Detail:

Detail:

Match commentary

Description here

Unit	Ref	Evidence	
P	P.10	Example of Pseudocode used for a method	
		Description:	

comment on a function and delete the function

Paste Screenshot here



Pseudocode

```
Take in an array of numbers as an argument and a number to remove from the array
For every number in the numbers array
Check if the number item is equal to the number to remove
If true then remove the current item from the array
return the altered array
```

Description here

Unit	Ref	Evidence	
P	P.13	Show user input being processed according to design requirements. Take a screenshot of: * The user inputting something into your program * The user input being saved or used in some way	
		Description:	

Paste Screenshot here



New Record

Label:

Label Name:

Label Location:

Title:

Artist:

Artist Name:

Genre(s):

- ☒ Electronica
- ☐ Downtempo
 - ☐ Trip Hop
 - ☐ Electronic
- ☐ Plunderphonics
 - ☐ Vaportrap
- ☐ Industrial Hip Hop
 - ☐ Indie Rock
- ☐ Ambient Techno
 - ☐ Shoegaze
- ☐ Psychedelic Pop

Genre Name:

Release Date:

Stock Quantity:

Unit Price:


Retail Price:

Upload Album Artwork: boc-g.jpg



[Back to Inventory](#)

Geogaddi

Album Art	Artist	Genre	Release Date	Stock	Unit Cost	Retail Cost	Markup	Label
	Boards of Canada	Electronica IDM	2002-02-13	5	£16.0	£25.0	£9.0	Warp

[EDIT RECORD](#)
[DELETE RECORD](#)
[ADD STOCK](#)
[ADD SALE](#)

Description here

The first screenshot above shows the user inputting details for a record to be stored in the program. The second screenshot shows the result of the inputted data being used in the program.

Unit	Ref	Evidence	
P	P.14	Show an interaction with data persistence. Take a screenshot of: * Data being inputted into your program * Confirmation of the data being saved	
		Description:	

Paste Screenshot here

Description here

Unit	Ref	Evidence	
P	P.15	Show the correct output of results and feedback to user. Take a screenshot of: * The user requesting information or an action to be performed * The user request being processed correctly and demonstrated in the program	
		Description:	

Paste Screenshot here





[View All Sales](#) [View All Labels](#) [View All Artists](#) [View All Genres](#)

Spinventory

Search by Title: [Sort](#)

Sort by: [-- Choose Sort Method --](#) [Sort](#)

Album Artwork	Title	Artist	Genre(s)	Release Date	Label	Stock		
	Music Has the Right to Children	Boards of Canada	Electronica	1998-04-20	Warp	10	ADD STOCK	ADD SALE
								

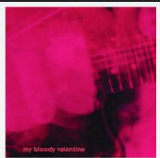
[Add New Record](#) [Add New Sale](#) [Add New Label](#) [Add New Artist](#) [Add New Genre](#)

[View All Sales](#) [View All Labels](#) [View All Artists](#) [View All Genres](#)

Spinventory

Search by Title: [Sort](#)

Sort by: [-- Choose Sort Method --](#) [Sort](#)

Album Artwork	Title	Artist	Genre(s)	Release Date	Label	Stock		
	Loveless	My Bloody Valentine	Shoegaze	1991-11-04	Creation Records	2	ADD STOCK	ADD SALE

[Add New Record](#) [Add New Sale](#) [Add New Label](#) [Add New Artist](#) [Add New Genre](#)

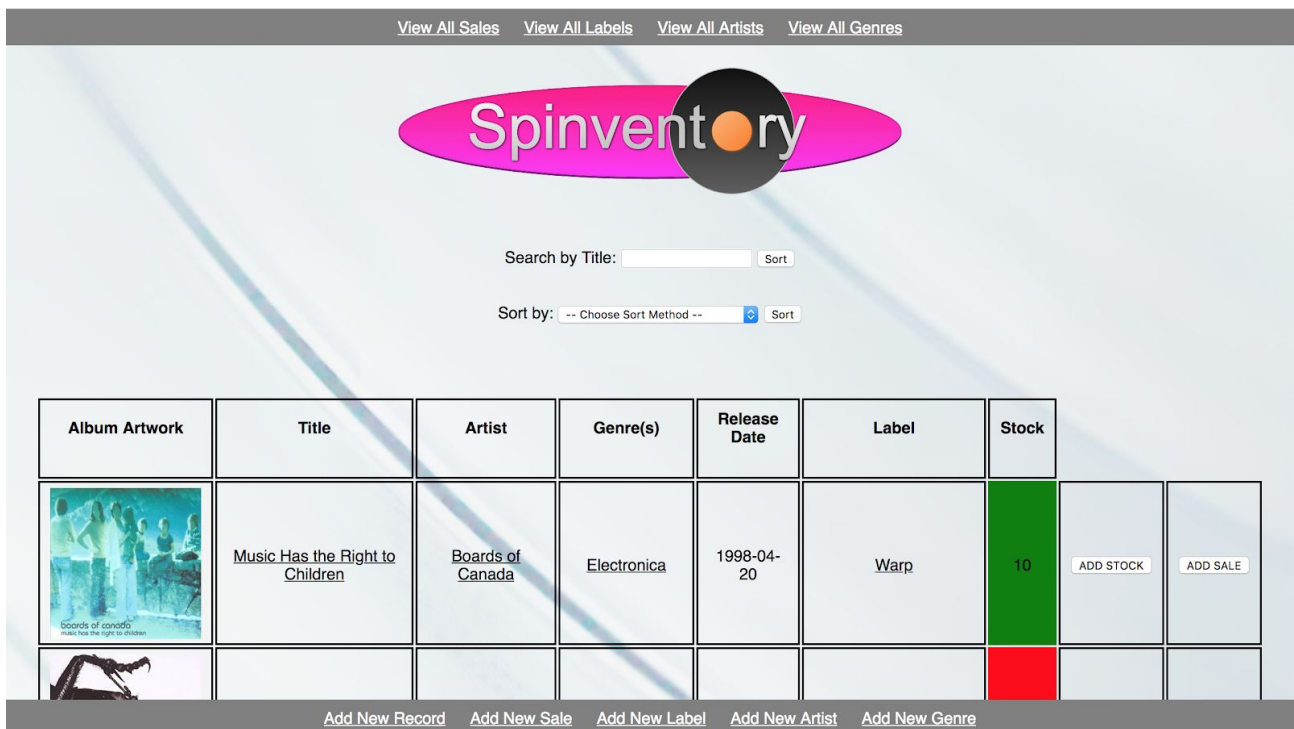
Description here

The first screenshot shows the user typing a word into a search box. The second screenshot shows the result of the search, displaying the items that include the searched word.



Unit	Ref	Evidence	
P	P.11	Take a screenshot of one of your projects where you have worked alone and attach the Github link.	
		Description:	

Paste Screenshot here

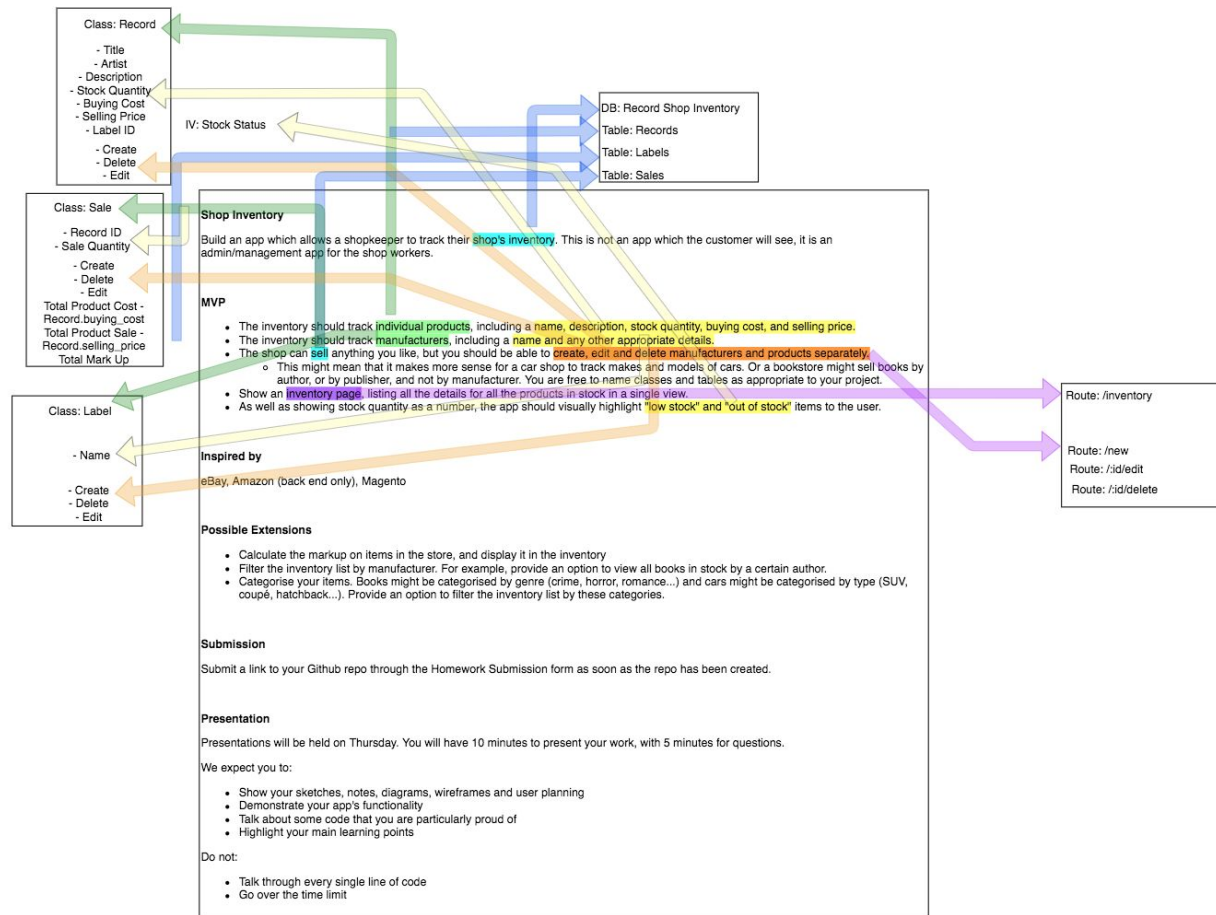


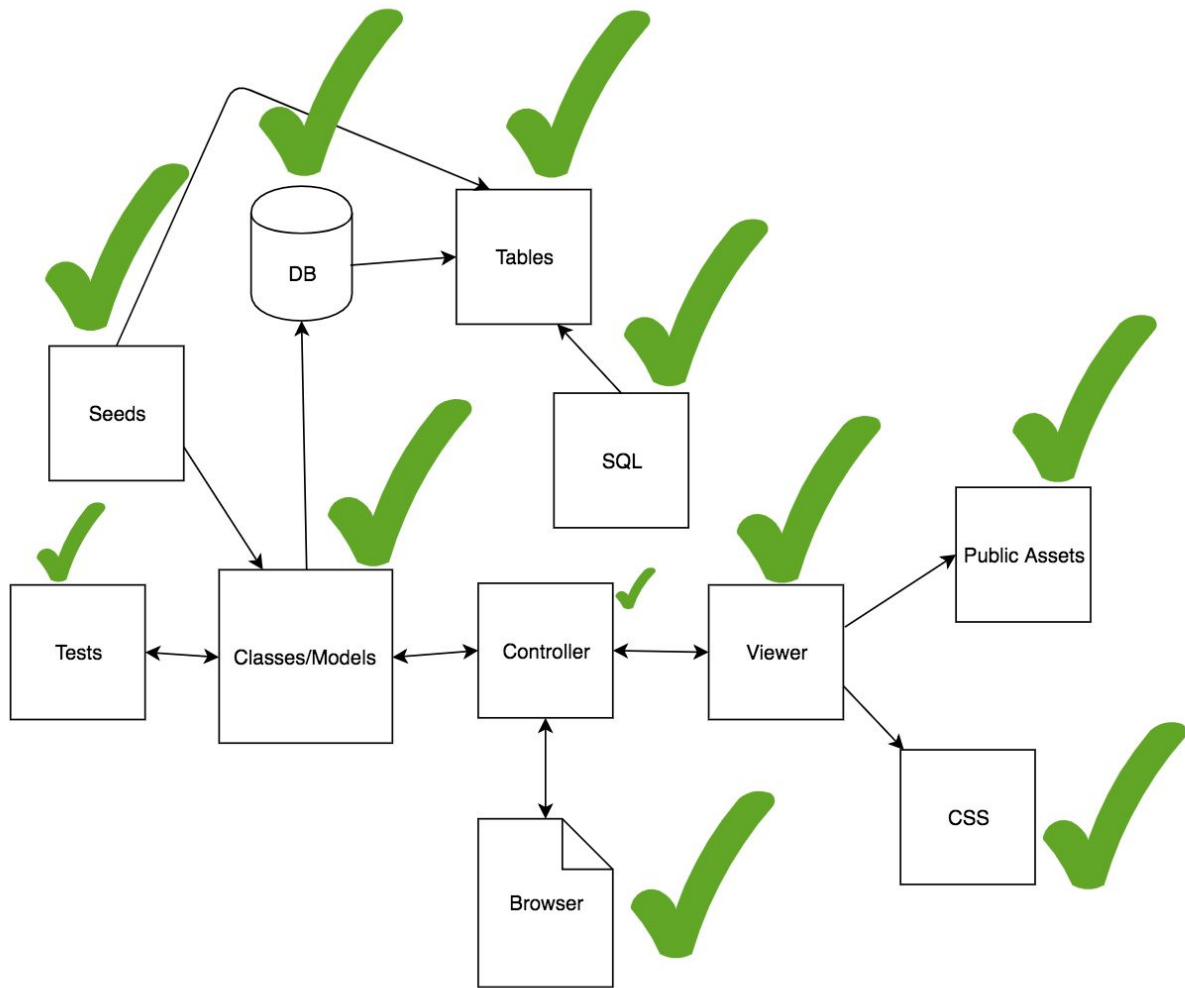
Description here

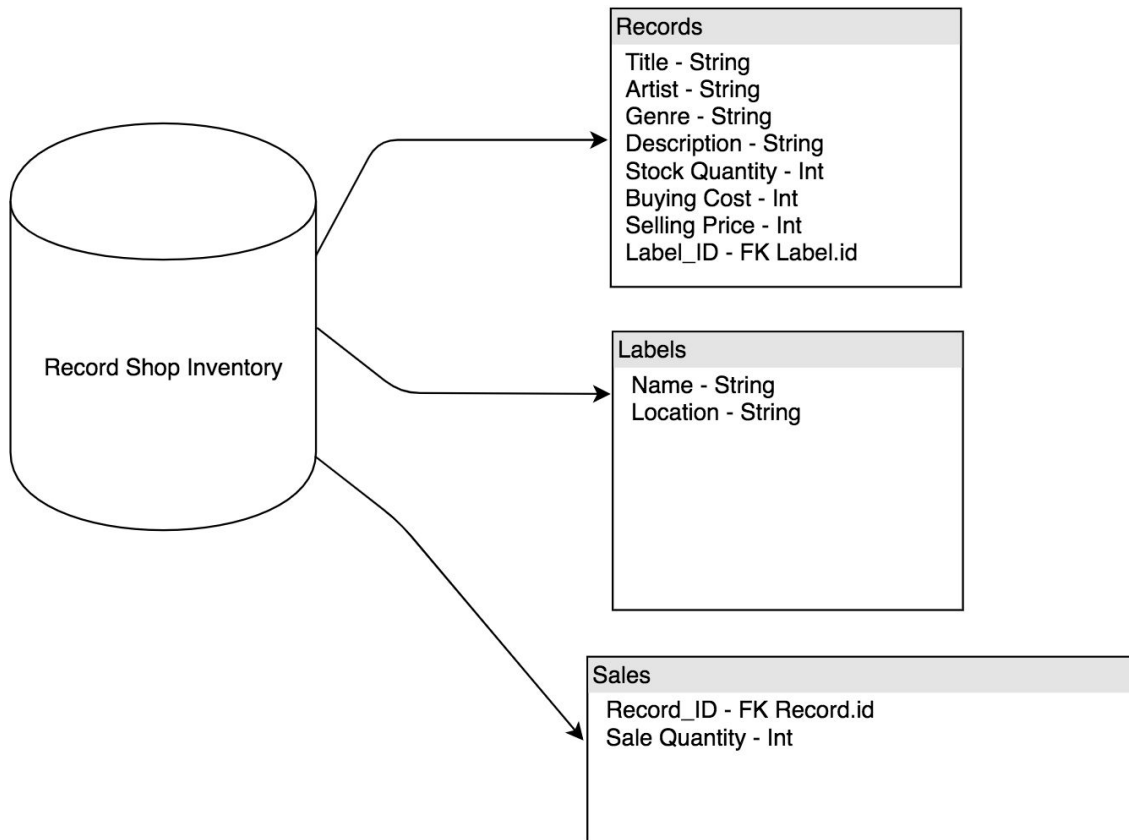
https://github.com/CMilligan26/project_record_shop_inventory

Unit	Ref	Evidence	
P	P.12	Take screenshots or photos of your planning and the different stages of development to show changes.	
		Description:	

Paste Screenshot here










Record
@title @artist @genre @description @stock_quantity @buying_cost @selling_price @label_id
self.all select save update delete self.delete_all markup

Label
@name @location
self.all select save update delete self.delete_all

Sale
@record_id @sale_quantity
self.all select save update delete self.delete_all sale_record_buying_cost sale_record_selling_price sale_markup sale_profit total_record_buying_cost total_record_selling_price total_markup total_profit



<div></div> <div>Name Rob Gordon</div>	Behaviours Listens to music everyday. Likes to categorise his own collection of music dependant on his mood. Interested in many different genres of music. Takes care of his records, he likes to have them all accounted for. Likes to view album covers when looking for new records. Regularly uses music streaming services. Is very familiar with Discogs, eBay and Amazon.
Demographics 20 - 40 years old. Has disposable income. Business owner.	Needs and goals Needs to store which records he sells. Needs to store the labels that the records have been bought from. Needs to record how many of a particular record he sells. Needs to know the current stock of his record shop and be notified when he is low on stock for individual records. Needs to sort the stored records by different categories. His goal is to easily manage and record his stock.

User needs

As a...	I want to...	So that...
record store owner who needs to know which records I have in stock	add new records i get in stock	I know which records I have in stock
record store owner who needs to know which records I have in stock	see all the records i have in stock	I know which records I have in stock
record store owner who needs to know which records I have in stock	update record details if they change or are incorrect	I know which records I have in stock
record store owner who needs to know which records I have in stock	delete records no longer being sold	I know which records I have in stock
record store owner who needs to keep track of sales	add sales	I can view the sales later
record store owner who needs to know which records I have in stock after sales	see all the records I have in stock after sales	I know which records I have still have in stock
record store owner who needs to know which labels I bought records from	see all the labels i bought records from	I know which labels i bought records from

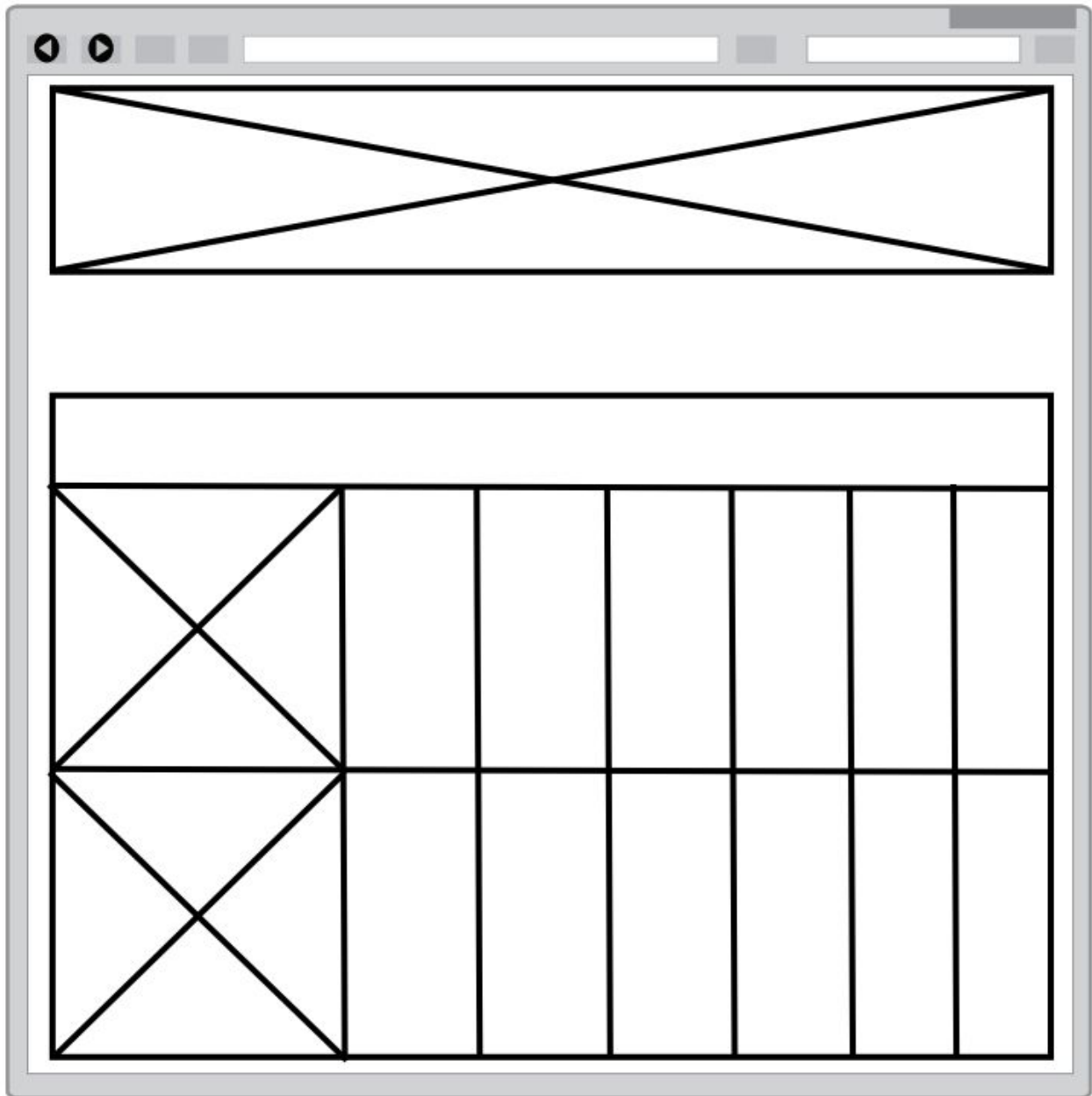


User journey

Your name _____ Date _____



User action	User action	User action	User action
1 Start Rob opens the app	2 Rob wants to add a new record	3 Rob enters the record details	4 Rob is happy with the details he entered and wants to save it
System response get request for '/' connect with db select all records from the records table pull html respond with '/'	System response get request for '/new' pull html respond with '/new'	System response	System response post request for '/' create record connect to db save to db redirect to '/:id'
User action	User action	User action	User action
5 Rob wants to know that he added the record	6	7	8 End
System response get request for '/:id' connect to db select record pull html respond '/:id'	System response	System response	System response



Description here

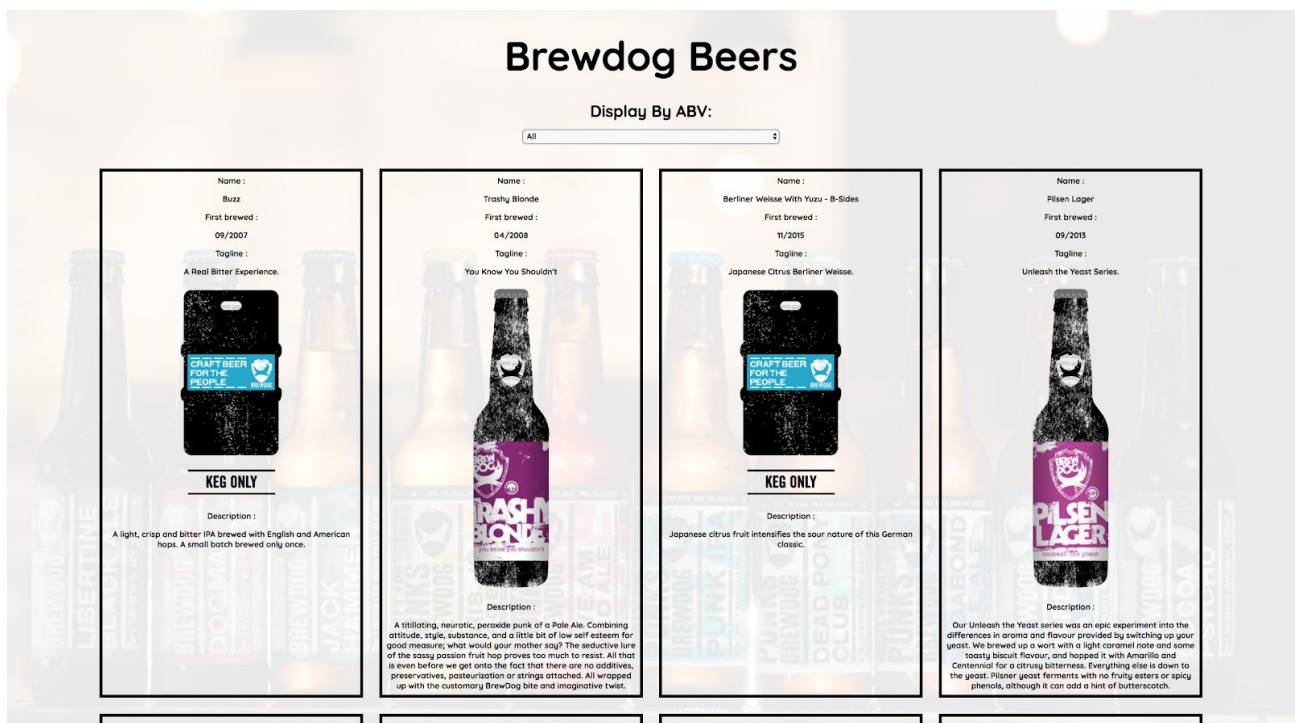


Week 7

Unit	Ref	Evidence	
P	P.16	Show an API being used within your program. Take a screenshot of: * The code that uses or implements the API * The API being used by the program whilst running	
		Description:	

Paste Screenshot here

```
page.setPageDetails();
const dataModel = new DataModel("https://api.punkapi.com/v2/beers", ['name', 'first_brewed',
'tagline', 'image_url', 'description'], 'abv');
dataModel.getData();
const dataList = new DataList('data_container');
dataList.bindEvents();
const selectView = new SelectView('data_select');
selectView.bindEvents();
});
```



Description here



Unit	Ref	Evidence	
P	P.18	Demonstrate testing in your program. Take screenshots of: <ul style="list-style-type: none">* Example of test code* The test code failing to pass* Example of the test code once errors have been corrected* The test code passing	
		Description:	

Paste Screenshot here

```
require('minitest/autorun')
require('minitest/rails')

require_relative('../models/artist')

class ArtistTest < MiniTest::Test

  def setup
    @artist = Artist.new({'id' => '1', 'artist_name' => 'Test_Artist'})
  end

  def test_artist_exists
    assert_equal(Artists, @artist.class)
  end

  def test_artist_can_have_id
    assert_equal(2, @artist.id)
  end

  def test_artist_has_name
    assert_equal('Test_artist', @artist.artist_name)
  end

end
```



```
[→ record_shop_inventory git:(master) ✖ ruby specs/artist_spec.rb  
Run options: --seed 1843
```

```
# Running:
```

```
FEF
```

```
Finished in 0.001107s, 2710.0270 runs/s, 1806.6847 assertions/s.
```

```
1) Failure:  
ArtistTest#test_artist_can_have_id [specs/artist_spec.rb:17]:  
Expected: 2  
Actual: 1
```

```
2) Error:  
ArtistTest#test_artist_exists:  
NameError: uninitialized constant ArtistTest::Artists  
Did you mean? Artist  
specs/artist_spec.rb:13:in `test_artist_exists'
```

```
3) Failure:  
ArtistTest#test_artist_has_name [specs/artist_spec.rb:21]:  
Expected: "Test_rtist"  
Actual: "Test_Artist"
```

```
3 runs, 2 assertions, 2 failures, 1 errors, 0 skips
```

```
require('minitest/autorun')  
require('minitest/rng')  
  
require_relative('../models/artist')  
  
class ArtistTest < Minitest::Test  
  
  def setup  
    @artist = Artist.new({'id' => '1', 'artist_name' => 'Test_Artist'})  
  end  
  
  def test_artist_exists  
    assert_equal(Artist, @artist.class)  
  end  
  
  def test_artist_can_have_id  
    assert_equal(1, @artist.id)  
  end  
  
  def test_artist_has_name  
    assert_equal('Test_Artist', @artist.artist_name)  
  end  
  
end
```

```
[→ record_shop_inventory git:(master) ✖ ruby specs/artist_spec.rb  
Run options: --seed 28739
```

```
# Running:
```

```
...
```

```
Finished in 0.000987s, 3039.5135 runs/s, 3039.5135 assertions/s.
```

```
3 runs, 3 assertions, 0 failures, 0 errors, 0 skips
```

Description here



Week 9

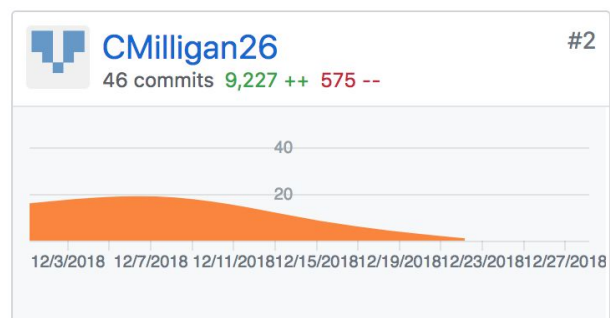
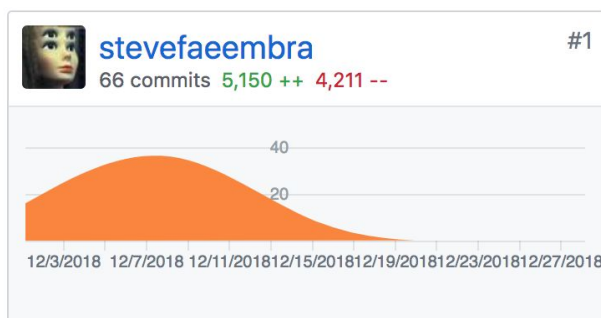
Unit	Ref	Evidence	
P	P.1	Take a screenshot of the contributor's page on Github from your group project to show the team you worked with.	
		Description:	

Paste Screenshot here

Dec 2, 2018 – Dec 29, 2018

Contributions: Commits ▾

Contributions to master, excluding merge commits



Description here



Unit	Ref	Evidence	
P	P.2	Take a screenshot of the project brief from your group project.	
		Description:	

Paste Screenshot here

Browser Game

Create a browser game based on an existing card or dice game. Model and test the game logic and then display it in the browser for a user to interact with.

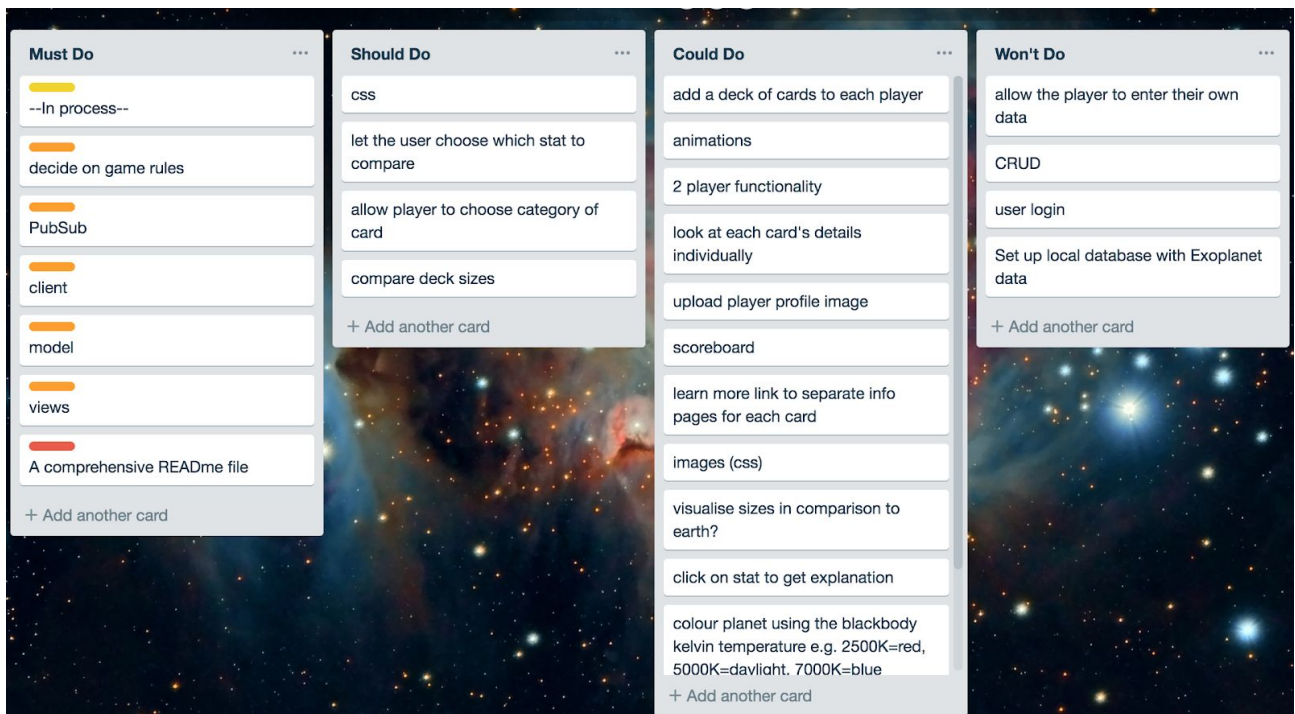
Write your own MVP with some specific goals to be achieved based on the game you choose to model.

You might use persistence to keep track of the state of the game or track scores/wins. Other extended features will depend on the game you choose.

Description here

Unit	Ref	Evidence	
P	P.3	Provide a screenshot of the planning you completed during your group project, e.g. Trello MOSCOW board.	
		Description:	

Paste Screenshot here



Description here

Unit	Ref	Evidence	
P	P.4	Write an acceptance criteria and test plan.	

Paste Screenshot here

Description here

Unit	Ref	Evidence	
P	P.7	Produce two system interaction diagrams (sequence and/or collaboration diagrams).	
		Description:	

Paste Screenshot here

Description here



Unit	Ref	Evidence	
P	P.8	Produce two object diagrams.	
		Description:	

Paste Screenshot here

Description here

Unit	Ref	Evidence	
P	P.17	Produce a bug tracking report	
		Description:	

Paste Screenshot here

Description here

Week 12

Unit	Ref	Evidence	
I&T	I.T.7	The use of Polymorphism in a program and what it is doing.	
		Description:	

Paste Screenshot here

Description here

Unit	Ref	Evidence	
A&D	A.D.5	An Inheritance Diagram	
		Description:	

Paste Screenshot here



Description here

Unit	Ref	Evidence	
I&T	I.T.1	The use of Encapsulation in a program and what it is doing.	
		Description:	

Paste Screenshot here

Description here

Unit	Ref	Evidence	
I&T	I.T.2	Take a screenshot of the use of Inheritance in a program. Take screenshots of: *A Class *A Class that inherits from the previous class *An Object in the inherited class *A Method that uses the information inherited from another class.	
		Description:	

Paste Screenshot here

Description here

Unit	Ref	Evidence	
P	P.9	Select two algorithms you have written (NOT the group project). Take a screenshot of each and write a short statement on why you have chosen to use those algorithms.	
		Description:	

Paste Screenshot here

Description here