P 9 Two algorithms

```
public ArrayList<Card> dealPlayer1Card() {
    player1Hand = hand1.buildHand();
    Player player1  = new Player("Player1", hand1);
    player1Name = player1.getName();

    player1DealtCardSuit = hand1.getCardSuit();
    player1DealtCardRank = hand1.getCardRank();

    player1Hand = hand1.getCardsInHand();
    for (Card card:player1Hand) {
        Suit suit = card.getSuit();
        Rank rank = card.getRank();
        int cardValue = card.getValue(rank);
    }

    player1HandNewValue = player1HandOldValue + hand1.getCardValue();
    player1HandOldValue = player1HandNewValue;

    return player1Hand;
}
```

The algorithm above builds up a player's hand in the Android application by setting the player's name then assigning the rank and suit of the randomly generated card to the hand. The current value of the hand is calculated by iterating over the card objects currently in the hand adding the latest card to the previous value of the hand to give a new hand value. This is carried out for both players to calculate the winner with the highest hand.

```
private Suit suit;
private Rank rank;
private ArrayList<Card> deck;
private Random randomGenerator = new Random();

public Deck() {
    this.deck = new ArrayList<Card>();
    createDeck();
}

public int deckSize() { return deck.size(); }

public void createDeck(){
    for (Suit suit : Suit.values()) {
        for (Rank rank : Rank.values()) {
            deck.add( new Card(suit, rank));
        }
    }
}

public Card dealRandomCard() {
    int index = randomGenerator.nextInt(deck.size());
    Card card = deck.get(index);
    return card;
}
```

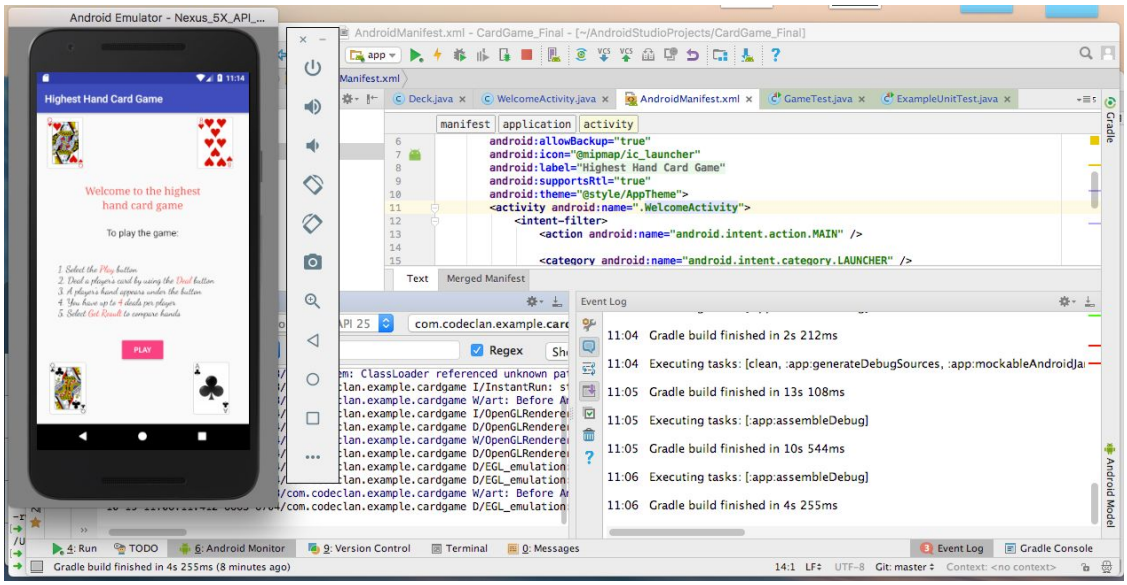
This algorithm creates a 52 card deck using enums for each of the 4 suits and for all the ranks. A random class is then used to randomly generate a card for a player once a card is dealt to them. This code is used by the first algorithm when displaying the cards on the screen and when adding up the card totals.

P10 Pseudocode

Pseudocode for basic working of Android application:

OUTPUT: Player shown welcome page and game rules
INPUT: Player selects ‘Play’ button
OUTPUT: Player navigates to deal page
FOR up to 4 cards
  IF ‘Deal Player 1’ button selected
    Generate random card and show on screen below ‘Deal Player 1’ button
  ENDIF
END FOR LOOP
FOR up to 4 cards
  IF ‘Deal Player 2’ button selected
    Generate random card and show on screen below ‘Deal Player 2’ button
  ENDIF
END FOR LOOP
INPUT: Player requests results of game - ‘Get Result’ button selected
OUTPUT: Relative running totals of both player’s hands are compared and a message appears showing player with the highest value hand.

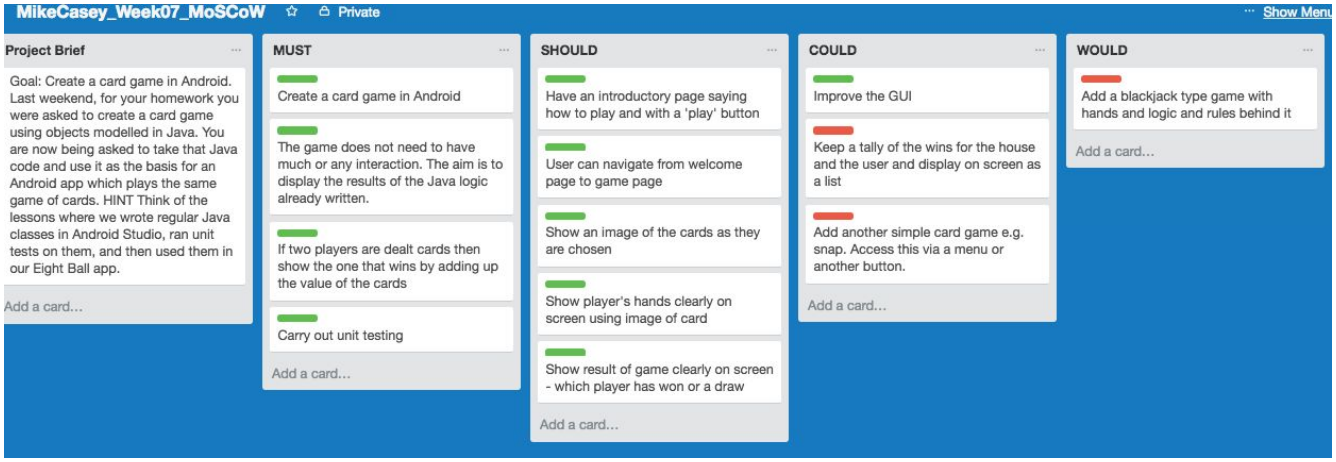
P 11 Project screenshot and github link



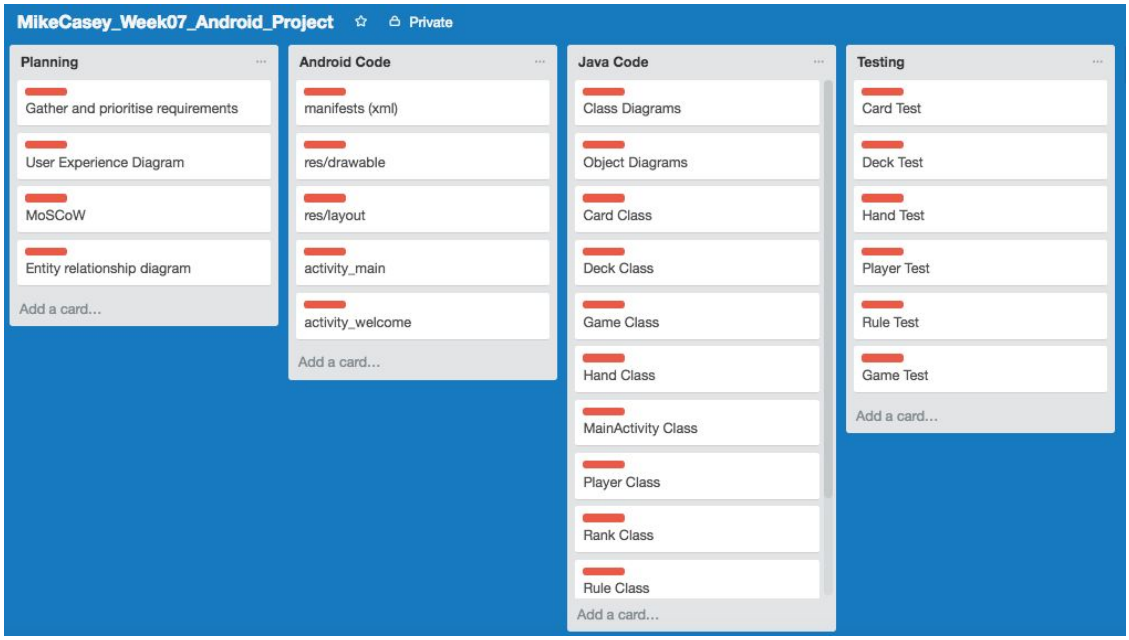
Github link:

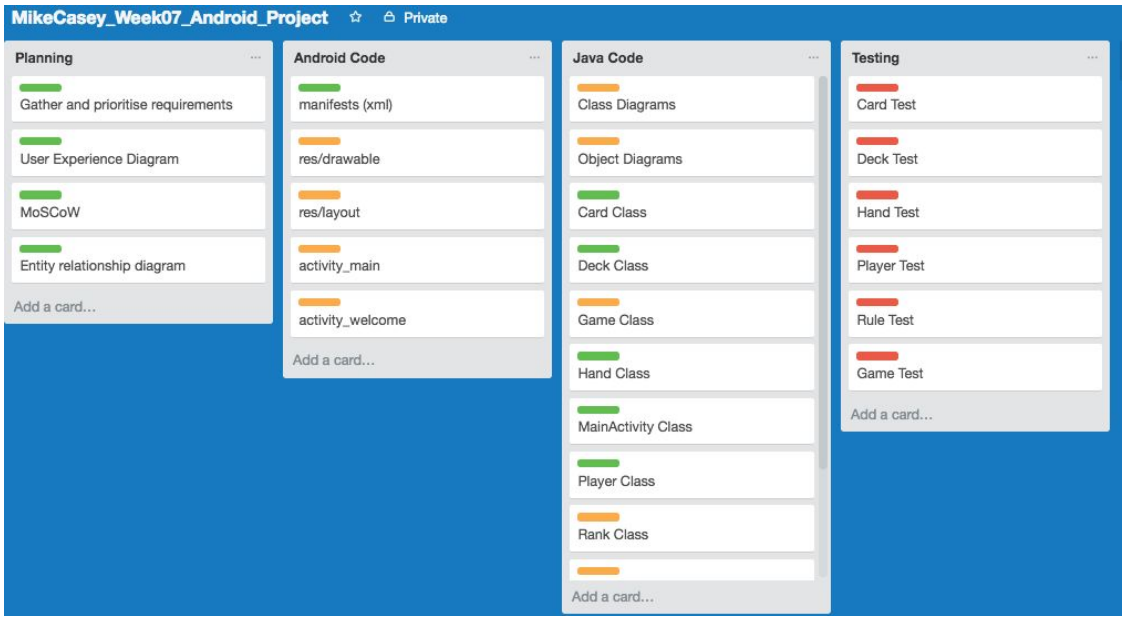
https://github.com/gulfstream15/Android_CardGame

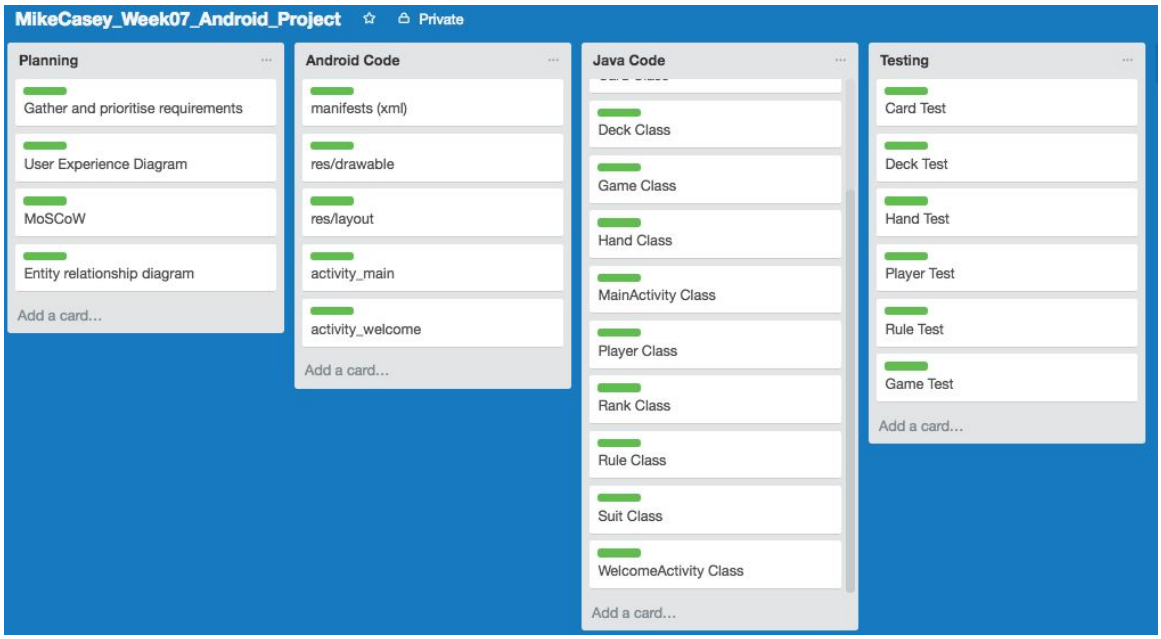
P 12 Project planning and development



Development stages

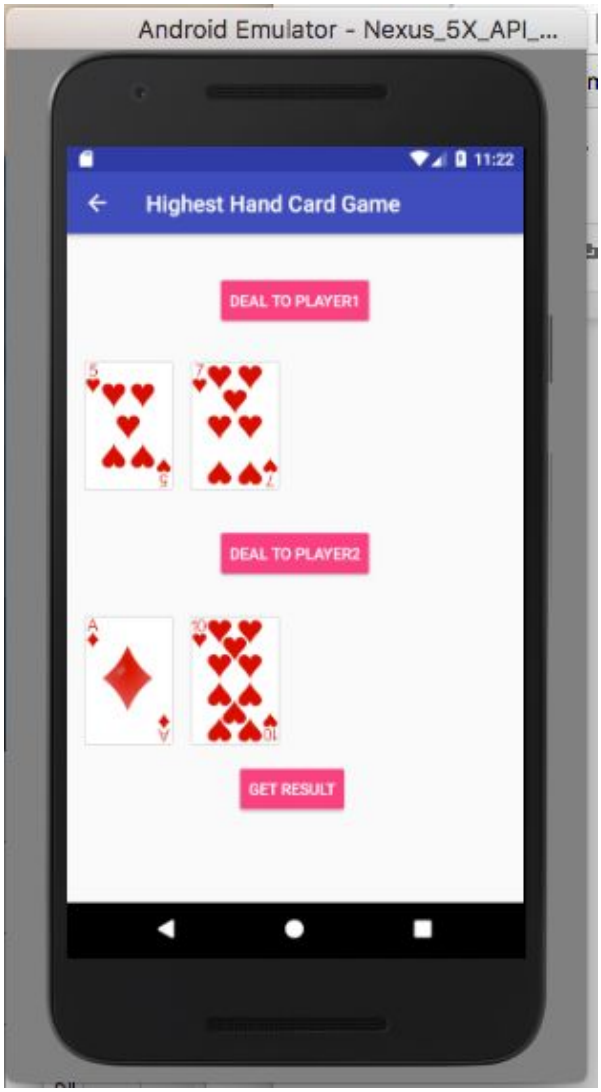






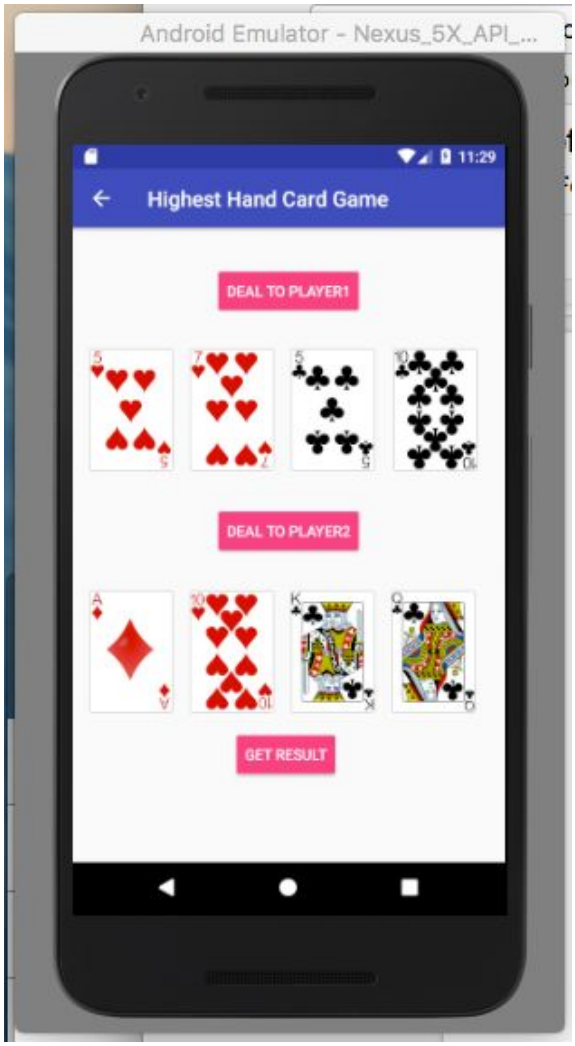
P 13 User Input according to design requirements

User inputs by clicking on each of the deal buttons (Deal to Player1 or Deal to Player2) to generate random cards in each hand. The cards appear in the screen.


Android Emulator - Nexus_5X_API_...
11:22
Highest Hand Card Game
DEAL TO PLAYER1
DEAL TO PLAYER2
GET RESULT

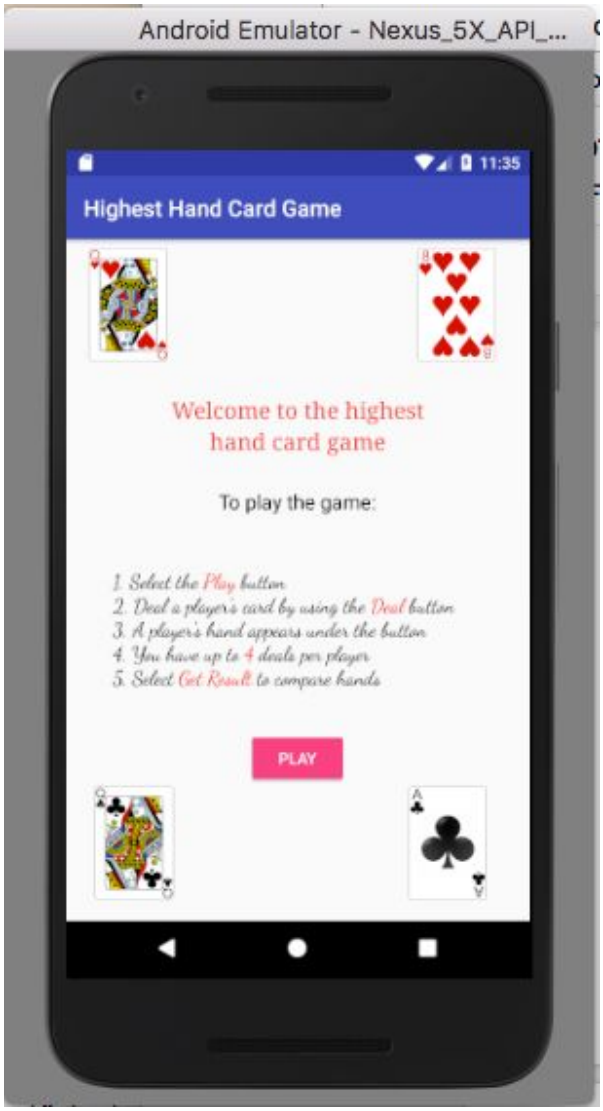
P 14 Interaction with data persistence

As each card is dealt via the deal buttons the previous cards remain on the screen - data is persisted in that the values of all the cards are used to get the total hand value.



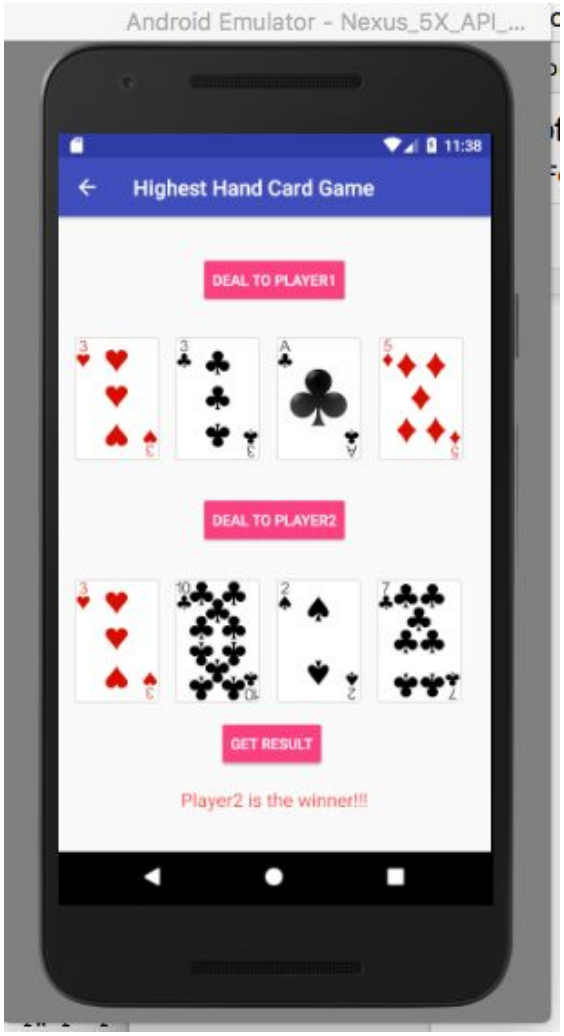
P 15 API usage

The emulator to represent the mobile application is an API, in this case a Nexus 5X Android emulator has been used. The front page of the application within the emulator is shown below.



P 16 Output results

In the image below the result of the interactive game is displayed. Once all the cards have been dealt for each hand the 'Get Result' button is selected. This tells the players who has won in terms of the highest value hand, the screen shot below shows Player 2 is the winner.



P 17 Bug tracking report

Task	**Pass/Fail**	**Resolution**	**Pass/Fail**
Clear layout of Android card game welcome page	Fail - text not lined up properly and resize of rules required	Edited activity_welcome.xml to tidy up layout.	Pass - layout now acceptable.
In main page present both hands that have been dealt lined on screen one above the other. 4 cards in each hand.	Fail - the cards were not aligned properly.	Edited activity_main.xml to tidy up layout.	Pass - layout now acceptable.
Add up the totals for the hands to	Fail - Total value of hands not being	Corrected a bug in the Java code (Hand	Pass - hands now have the correct total.

determine the winner based on the highest hand.	added up correctly.	class) that was causing incorrect total to be calculated.	
Use the ‘Get Result’ button to show which player has won.	Fail - Button not showing correct result.	Corrected the Android code behind the ‘Get Result’ button - Player1 was used twice.	Pass - correct winner displayed.

P 18 Testing

Example of test failure from JavaScript TDD heros and rats program.

```
> heroes_and_rats@1.0.0 test /Users/gulfstream/Desktop/CodeClan/HomeWork/week10/day05/heroes_and_rats
> mocha specs


  Food Tests
    ✓ Food should have a name
    ✓ Food should have a replenishment value

  Hero Tests
    ✓ Hero should have a name
    ✓ Hero should have health
    ✓ Hero has a favourite food
    ✓ Hero can say name
    ✓ Should start with an empty stomach
    ✓ Should be able to eat food
    ✓ Should be able to list stomach contents
    ✓ Hero should start with an empty task list
    ✓ Should be able to add a task to the task list
    ✓ Should be able to add multpile tasks to the task list
    ✓ Should be able to list the tasks
    ✓ Should be able to eat non-favourite food and health goes up
    ✓ Should be able to eat favourite food and health goes up
    ✓ Should be able to sort tasks by difficulty
    ✓ Should be able to sort tasks by urgency
    ✓ Should be able to sort tasks by reward
    1) Should be able to show tasks completed

  Task Tests
    ✓ Task should have a description
    ✓ Task should have a difficulty level
    ✓ Task should have a urgency level
    ✓ Task should have a reward
    ✓ Task should be marked as complete
    ✓ Prints out all task details as a string


  24 passing (23ms)
  1 failing

  1) Hero Tests Should be able to show tasks completed:
     TypeError: Cannot read property 'length' of undefined
      at Context.<anonymous> (specs/hero_spec.js:142:35)


npm ERR! Test failed.  See above for more details.
```

The bug in the previous page was due to the hero's completed task list not being built correctly. A fix for the above is given below:

New source code to build up list of completed tasks:

```
listCompletedTasks: function() {
  var completedTasksOnly = (this.tasks.filter(function(task) {
     return (task.complete === true);
  }));
  return completedTasksOnly;
}
```

Test code adding four hero tasks with three completed, one outstanding:

```
it("Should be able to count completed tasks ", function() {
  hero.addTask(task1);
  hero.addTask(task2);
  hero.addTask(task3);
  hero.addTask(task4);
  var totalCompletedTasks = hero.listCompletedTasks().length;
  assert.equal(3, totalCompletedTasks);
});
```

Output of final tests with the above code:

```
> heroes_and_rats@1.0.0 test /Users/gulfstream/Desktop/CodeClan/HomeWork/week10/day05/heroes_and_rats
> mocha specs


  Food Tests
    ✓ Food should have a name
    ✓ Food should have a replenishment value

  Hero Tests
    ✓ Hero should have a name
    ✓ Hero should have health
    ✓ Hero has a favourite food
    ✓ Hero can say name
    ✓ Should start with an empty stomach
    ✓ Should be able to eat food
    ✓ Should be able to list stomach contents
    ✓ Hero should start with an empty task list
    ✓ Should be able to add a task to the task list
    ✓ Should be able to add multpile tasks to the task list
    ✓ Should be able to list the tasks
    ✓ Should be able to eat non-favourite food and health goes up
    ✓ Should be able to eat favourite food and health goes up
    ✓ Should be able to sort tasks by difficulty
    ✓ Should be able to sort tasks by urgency
    ✓ Should be able to sort tasks by reward
    ✓ Should be able to count completed tasks

  Task Tests
    ✓ Task should have a description
    ✓ Task should have a difficulty level
    ✓ Task should have a urgency level
    ✓ Task should have a reward
    ✓ Task should be marked as complete
    ✓ Prints out all task details as a string


  25 passing (19ms)
```

P 19 Acceptance test plan

The following test plan refers to the Bookshop project.

Acceptance Criteria	**Expected Result**	**Pass/Fail**
User is able to see choice of entering author inventory or book inventory on intro page	Clear layout of buttons for navigating to chosen inventory.	Pass
User enters author inventory by selecting ‘author inventory’ button	User sees all the authors in in the author inventory.	Pass
User can add an author in the inventory list via ‘Add Author’ button above the author list.	User is taken to the ‘Add Author Details’ page, adds all the details then selects ‘Add’. The user is then taken to author inventory page and the new author appears in the inventory.	Pass
User can edit an author by selecting the ‘Edit’ button next to the author.	User is taken to the ‘Edit Author’ page, changes details then selects ‘Save Changes’. User to taken to author inventory page and the updated details are shown.	Pass
User can delete author in the inventory list via ‘Delete’ button next to author.	Author is removed from list.	Pass
User enters book inventory by selecting ‘book inventory’ button.	User sees all the books in the books inventory.	Pass
User can add a book in the inventory list via ‘Add Book button above the book list.	User is taken to the ‘Add book Details’ page, adds all the details then selects ‘Add’. The user is then taken to book inventory page and the new book appears in the inventory.	Pass
User can edit a book by	User is taken to the ‘Edit	Pass

selecting the ‘Edit’ button next to the book.	Book’ page, changes details then selects ‘Save Changes’. User to taken to book inventory page and the updated details are shown.	
User can delete book in the inventory list via ‘Delete’ button next to book.	Book is removed from list.	Pass