Evidence for Project Unit

Luis Farid Tejero Aoun E - 21

P. 1 Github Contributors page

P. 2 Project Brief

P. 3 Use of Trello

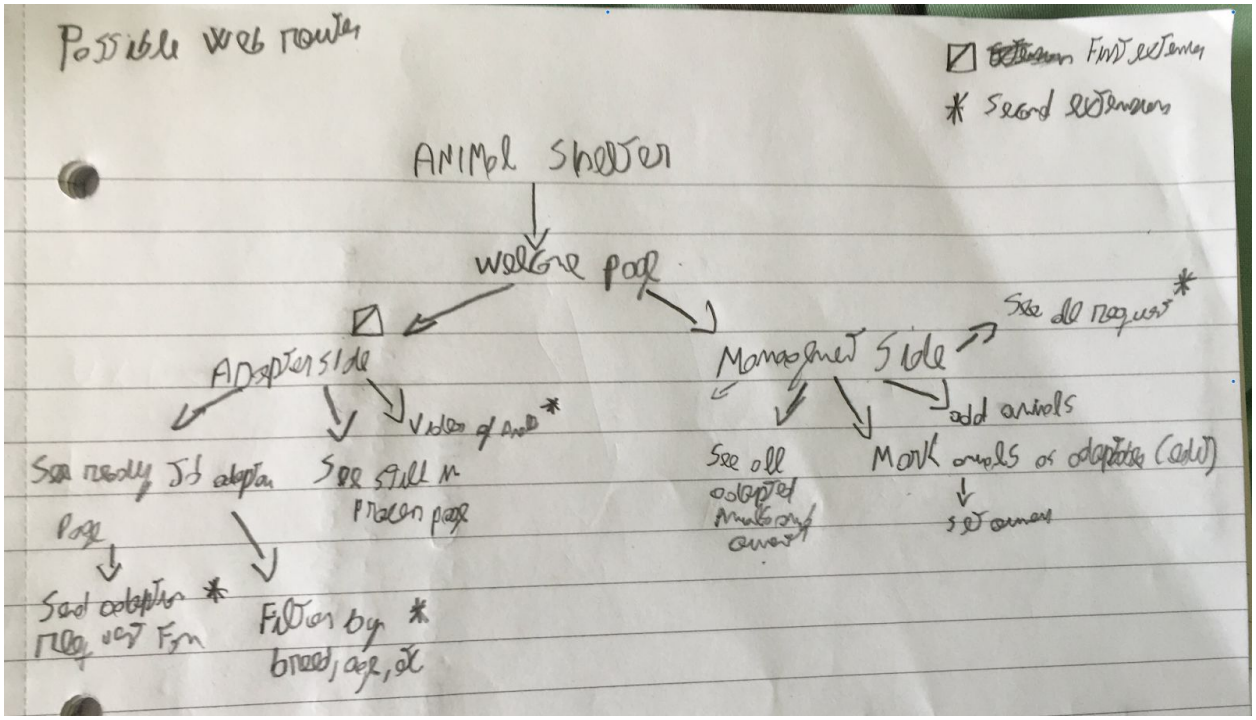
Evidence for unit

Evidence for unit

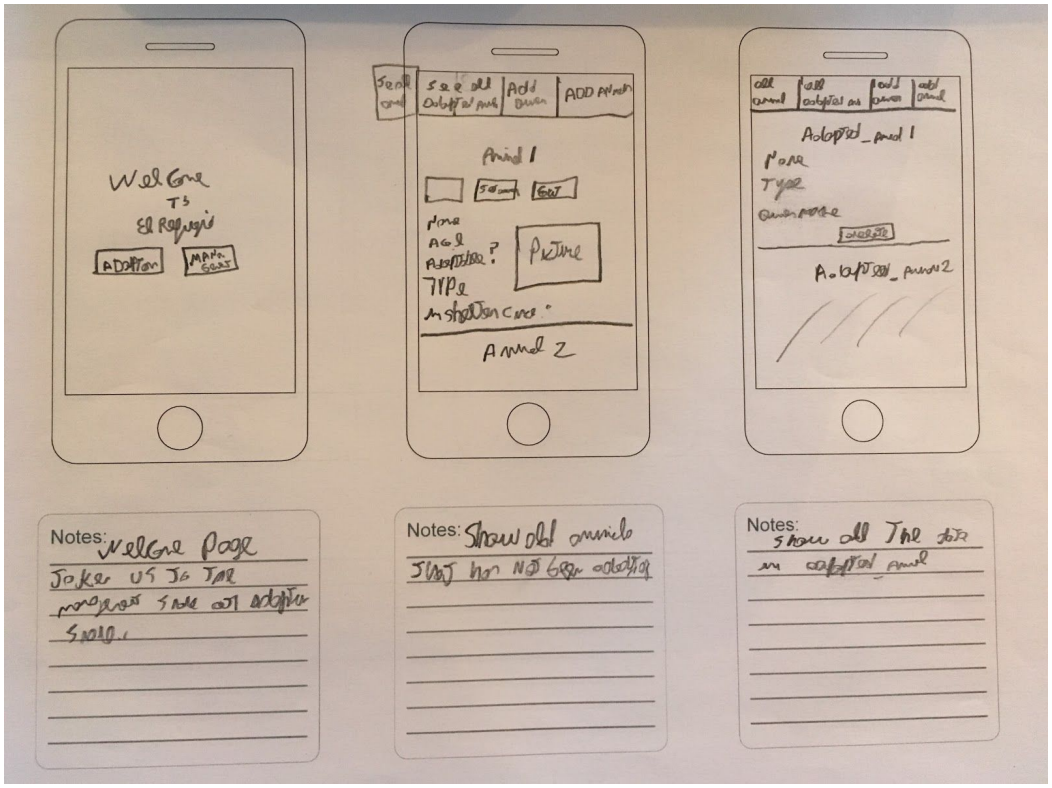
Evidence for unit

P. 4 Acceptance Criteria

P. 5 User sitemap



P. 6 Wireframes designs



P. 7 System interactions diagrams

Evidence for unit

P. 8 Two Object Diagrams

Evidence for unit

P. 9 Choice of two algorithms (find the algorithms on a program you might have written, show the code you have used.)

On this example please take a screenshot and write what it is doing and why you decided to use it.

Evidence for unit

Evidence for unit

P. 10 Example of Pseudocode

```
95
96  def adopted_by(owner)
97    #Check if the animal is adoptable
98    #Modify and update new animal
99    #create hash with the new data
100   #create the new object
101   #save new object in DB
102 end
```

P. 11 Github link to one of your projects

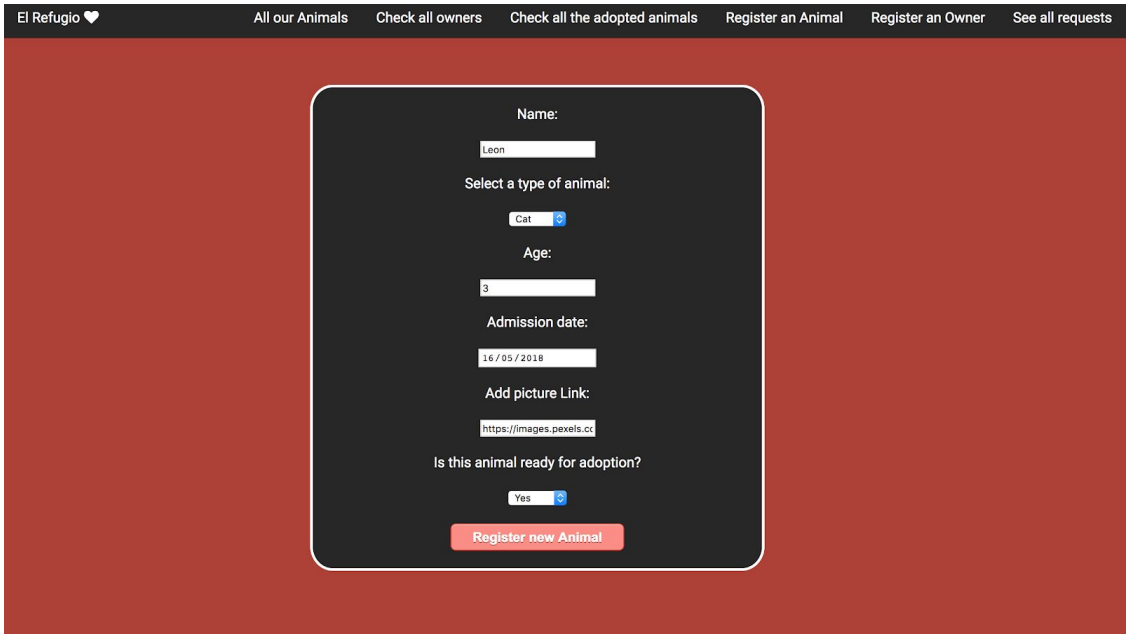
P. 12 Screenshot of your planning and the different stages of development to show changes.

Evidence for unit

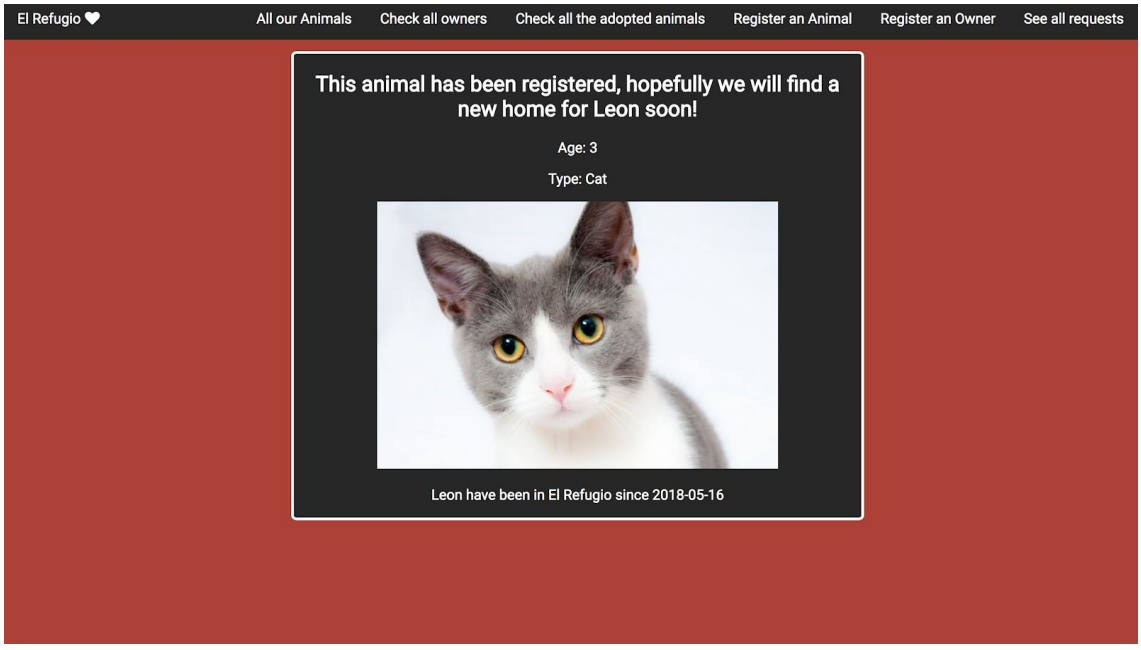
Evidence for unit

Evidence for unit

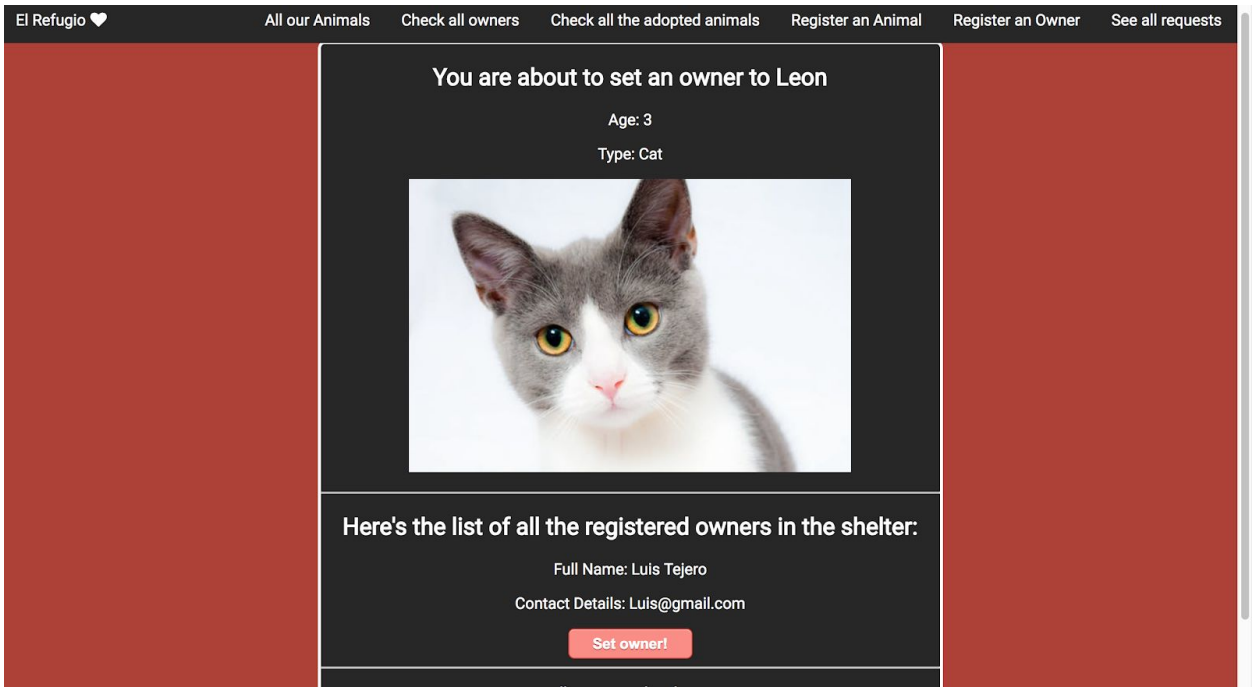
P. 13 User input

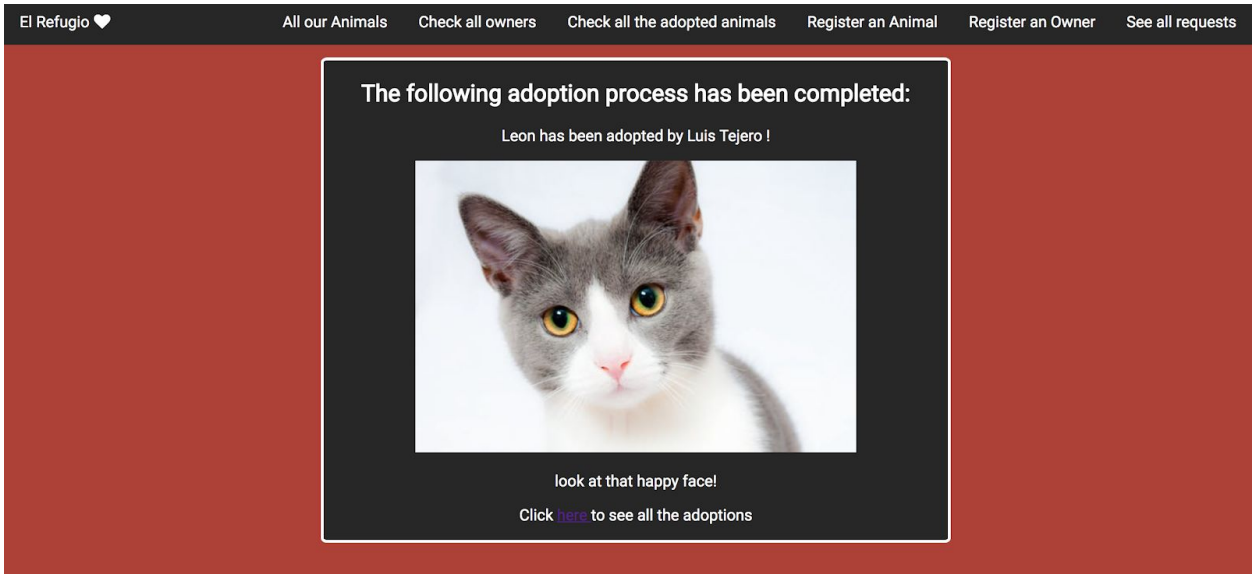


Input being added:

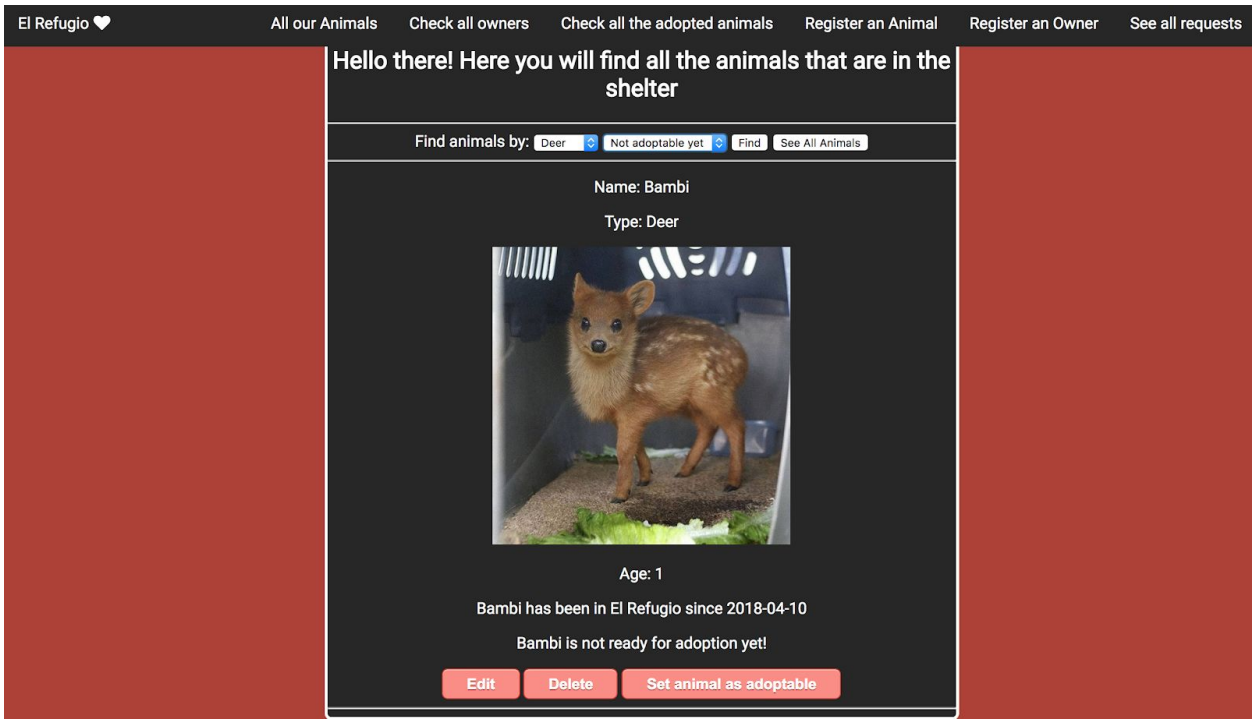


P. 14 Interaction with data persistence





P. 15 User output result



P. 16 Bug tracking report showing the errors diagnosed and corrected.

Evidence for unit

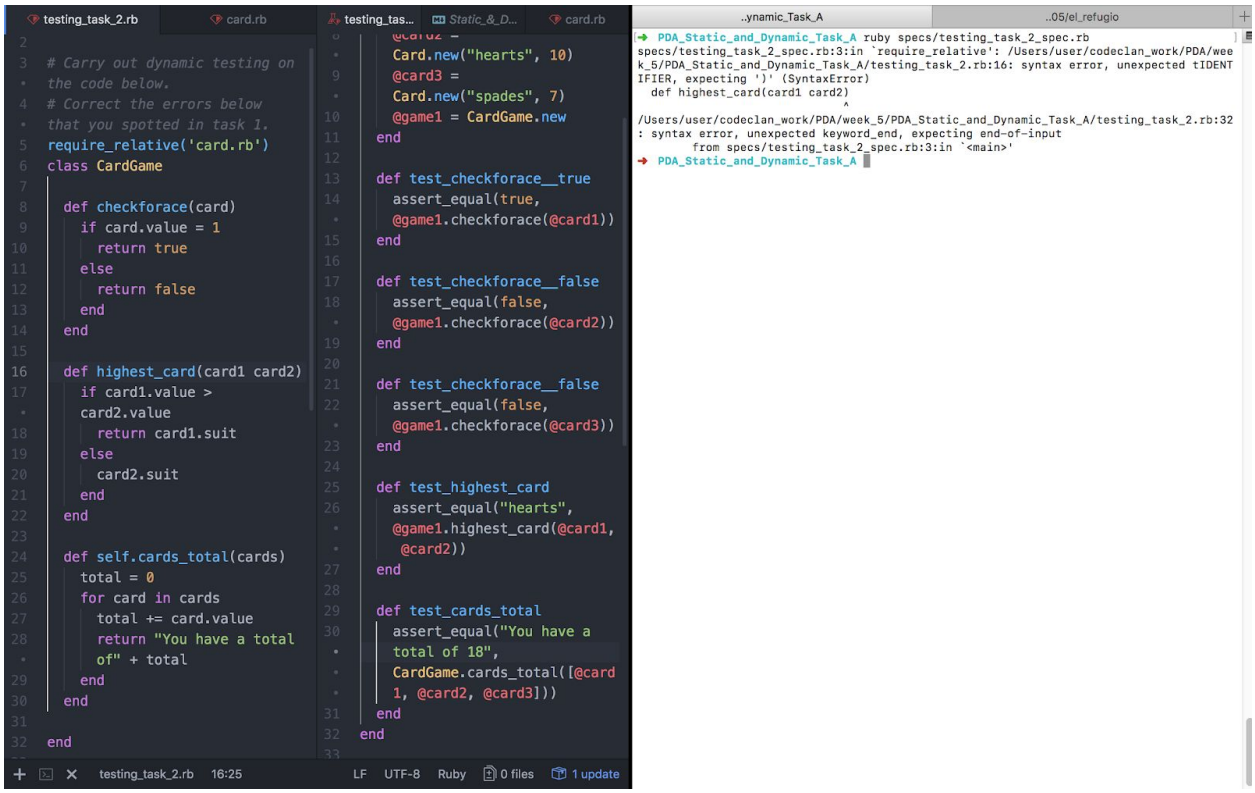
P. 17 Testing your program

Show the test code, the test not passing.....and then the test fixed.

Evidence for unit

P. 18 Demonstrating testing in the program

Test failing, Showing class file and test file.



Test File done

```ruby
require("minitest/autorun")
require("minitest/rg")
require_relative("../testing_task_2")

class CardTest < MiniTest::Test
  def setup
    @card1 = Card.new("ace", 1)
    @card2 = Card.new("hearts", 10)
    @card3 = Card.new("spades", 7)
    @card4 = Card.new("joker", 0)
    @game1 = CardGame.new
  end

  def test_check_for_ace__true
    assert_equal(true, @game1.check_for_ace(@card1))
  end

  def test_check_for_ace__false
    assert_equal(false, @game1.check_for_ace(@card2))
  end

  def test_check_for_ace__false2
    assert_equal(false, @game1.check_for_ace(@card3))
  end

  def test_highest_card__hearts
    assert_equal("hearts",  @game1.highest_card(@card1, @card2))
  end

  def test_highest_card__spades
    assert_equal("spades",  @game1.highest_card(@card1, @card3))
  end

  def test_highest_card__ace
    assert_equal("ace",  @game1.highest_card(@card4, @card1))
  end


  def test_cards_total__18
    assert_equal("You have a total of 18",  CardGame.cards_total([@card1, @card2, @card3]))
  end

  def test_cards_total__17
    assert_equal("You have a total of 17",  CardGame.cards_total([@card2, @card3]))
  end

  def test_cards_total__1
    assert_equal("You have a total of 1",  CardGame.cards_total([@card1]))
  end
end

```

Class file done

```ruby
### Testing task 2 code:

# Carry out dynamic testing on the code below.
# Correct the errors below that you spotted in task 1.
require_relative('card.rb')
class CardGame

  def check_for_ace(card)
    if card.value == 1
      return true
    else
      return false
    end
  end

  def highest_card(card1, card2)
    if card1.value > card2.value
      return card1.suit
    else
      card2.suit
    end
  end

  def self.cards_total(cards)
    total = 0
    for card in cards
      total += card.value
    end
    return "You have a total of #{total}"
  end

end
```

Test passing.

```
➜  PDA_Static_and_Dynamic_Task_A ruby specs/testing_task_2_spec.rb
Run options: --seed 25463

# Running:

.........

Finished in 0.001206s, 7462.6866 runs/s, 7462.6866 assertions/s.

9 runs, 9 assertions, 0 failures, 0 errors, 0 skips
➜  PDA_Static_and_Dynamic_Task_A
```