

LEANNE STRANKS

T1A3 - TERMINAL APP - QUIZ BITES

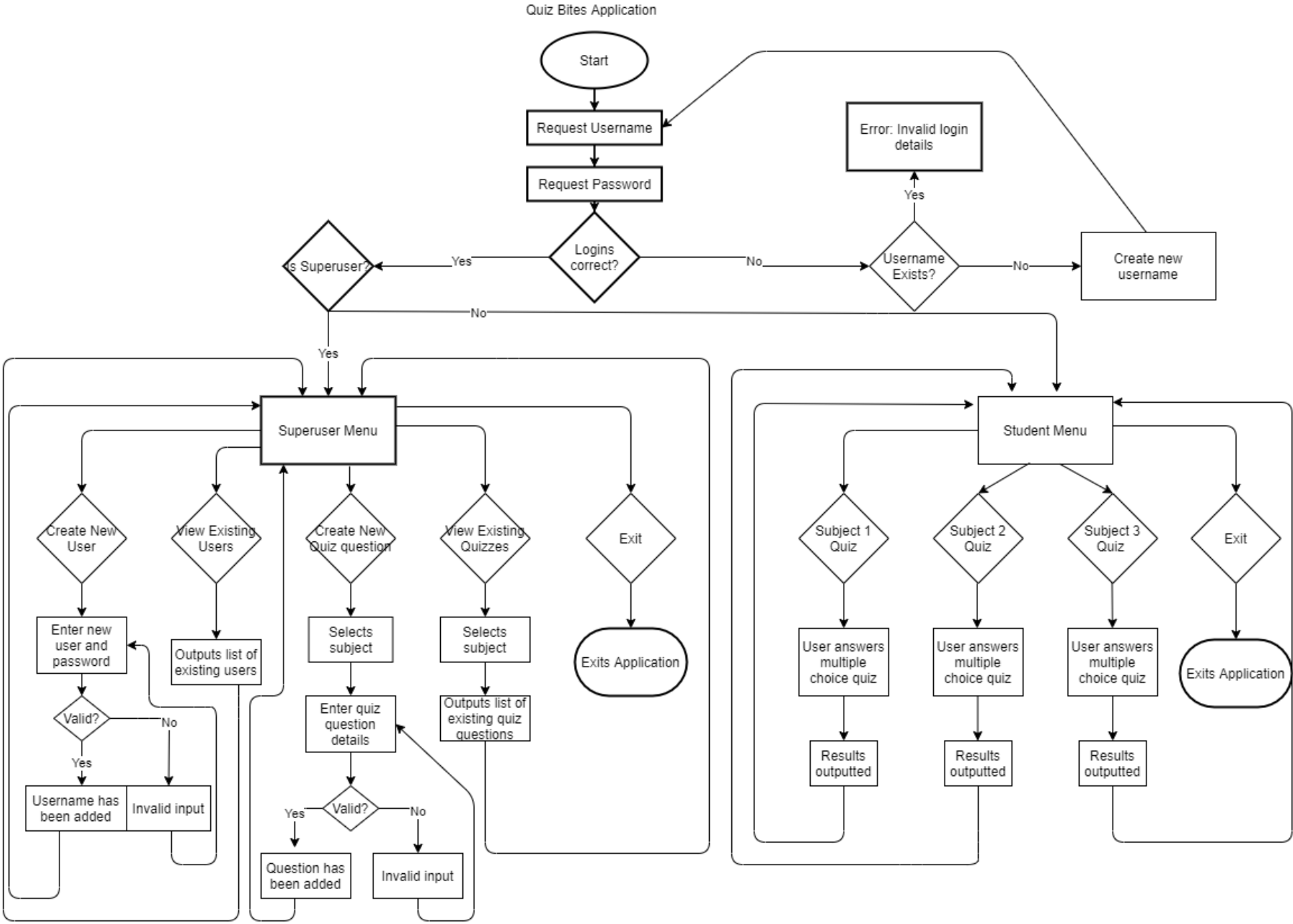
AGENDA

- Main features
- The logic of the Terminal application and code
- Review of the development/build process including challenges, ethical issues, favourite parts
- Overview of code – important parts of the code

MAIN FEATURES

- User login authentication – will parse through and read the user database stored in a JSON file
 - Check if there is an existing username and match the username to the inputted password
 - If the login is successful, it will check to see what the access level is granted for the user and display the appropriate menu
 - An error message will be raised if no username is found. User will be able to create a new account based on the lower access
- Superuser/Faciliator menu option
 - Create new user/view existing users – written and read via JSON file
 - Create new quiz questions/view existing quiz questions – written and read via CSV
 - Menu is generated using TTY Prompt to avoid errors in selecting options
- Student menu option
 - User selects a subject which will read the appropriate CSV quiz file, it will select a question at random where user will input answer. Once quiz is finished – output results to user

Flow Chart




LOGIC

- User can input username and password
- Program will determine their access level based on what permission is assigned to that username (stored in JSON file)
- Superuser/facilitator will have a higher access level with more options available to select from
- Application designed as a learning program for students

CHALLENGES

- THE INITIAL IDEA/MEETING CRITERIA
- INITIAL IDEA SIMPLICITY VS THE DEPTH REQUIRED WHILST PLANNING
- BREAKING CODE DOWN INTO SMALLER PIECES VS LOOKING AT OVERALL
- TRYING TO GET FROM START TO FINISH WITHOUT THINKING OF THE INBETWEEN
- TROUBLESHOOTING/CLASSES
- MAKING IT DRY

FAVOURITE PARTS

- WHEN THE CODE DID WHAT WAS EXPECTED OF IT
 - SEEING IT ALL COME TOGETHER
 - LEARNING WHAT WORKED/WHAT DIDN'T
 - THE CODING ITSELF
- 

```
list_of_users = JSON.parse(File.read("json/users.json"), symbolize_names: true)
user = list_of_users.find { |user| user[:username] == input_username}
if user[:password] == input_password
```

Reading and writing into JSON file.
Using prompt where appropriate to
ensure that input will be valid

```
if superuser_menu == "Create New User"
  puts "What is the new user's full name?"
  input_new_user = gets.chomp
  puts "What username would you like to assign to this user?"
  input_new_username = gets.chomp.capitalize.strip
  puts "Assign a password for this user:"
  input_new_password = gets.chomp.strip
  input_is_newuser_superuser = prompt.select("Is this new user a Facilitator or Student?") do |accesslevel|
    accesslevel.choice 'Facilitator'
    accesslevel.choice 'Student'
  end
end
```

```
newuser = {
  fullname: input_new_user,
  username: input_new_username,
  password: input_new_password,
  accesslevel: input_is_newuser_superuser
}
file = File.open('json/users.json', "r+").read
array = JSON.parse(file)
array.push(newuser)
json = JSON.generate(array)
File.write('json/users.json', json)
```

```

if student_menu == 'English'
  englishquizlist = CSV.read("csv/english_quiz_questions.csv")
  index = 0
  score = 0
  answer = ""
  while index < 10 do
    "Choose which is correct from the following options or type exit to leave the program:"
    quiz_question = englishquizlist.sample
    output_question = quiz_question.values_at(0..4)
    puts output_question
    answer = gets.chomp

    if answer == quiz_question[-1]
      score += 1
    elsif answer == "exit"
      puts pastel.cyan(font.write("Thank you!"))
      exit
    end

    index += 1
  end

  result = case score
  when 0..3 then puts "Your score was #{score}. There is room for improvement"
  when 4..7 then puts "Your score was #{score}. Well done"
  when 8..10 then puts "Your core was #{score}. Amazing work!"
  end
end

```

- Reading from a CSV file
- Picking a sample array to produce a random question from the list
- Outputting array indexes 0-4 only
- If student's answer input is equal to the last index (5) in array, it will increase the score
- Quiz will run through 10 questions
- Currently: Total score will output using a case statement dependent on what the score total is

Thank you