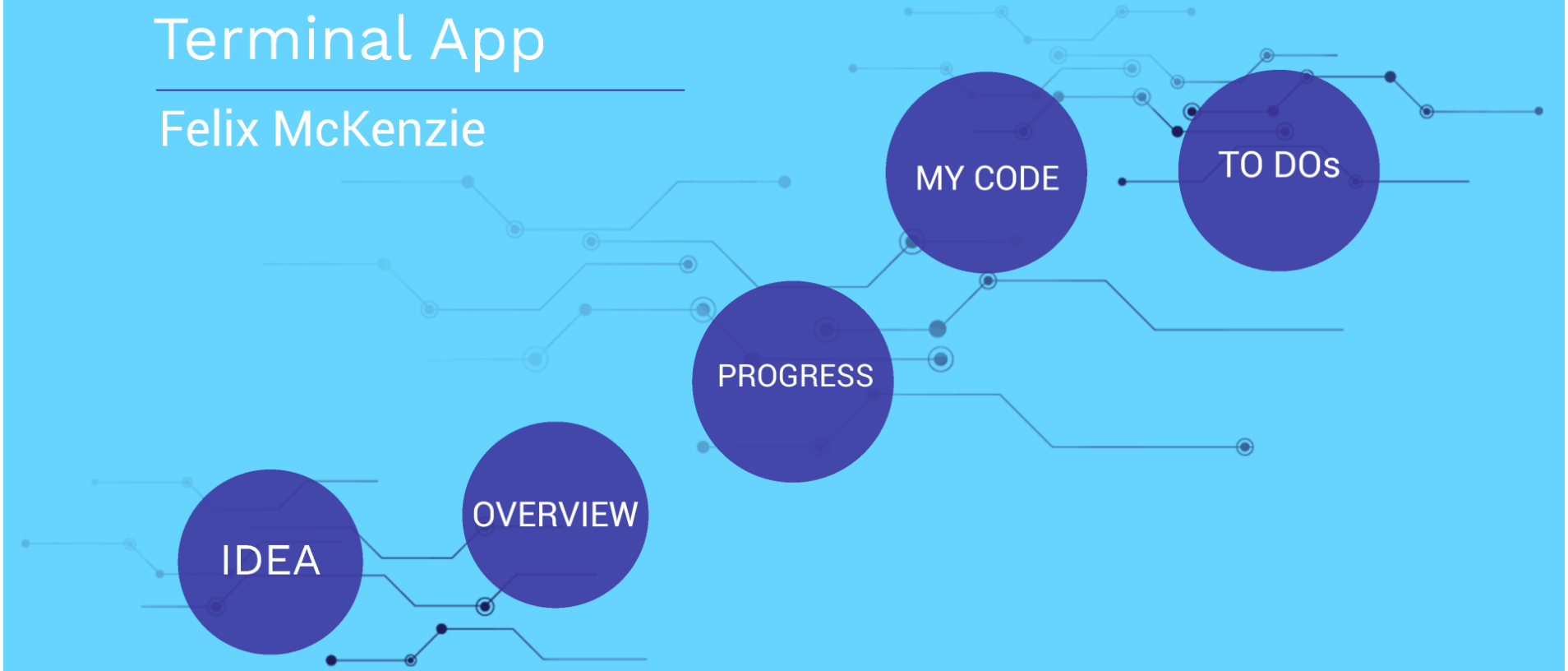
Terminal App

Felix McKenzie

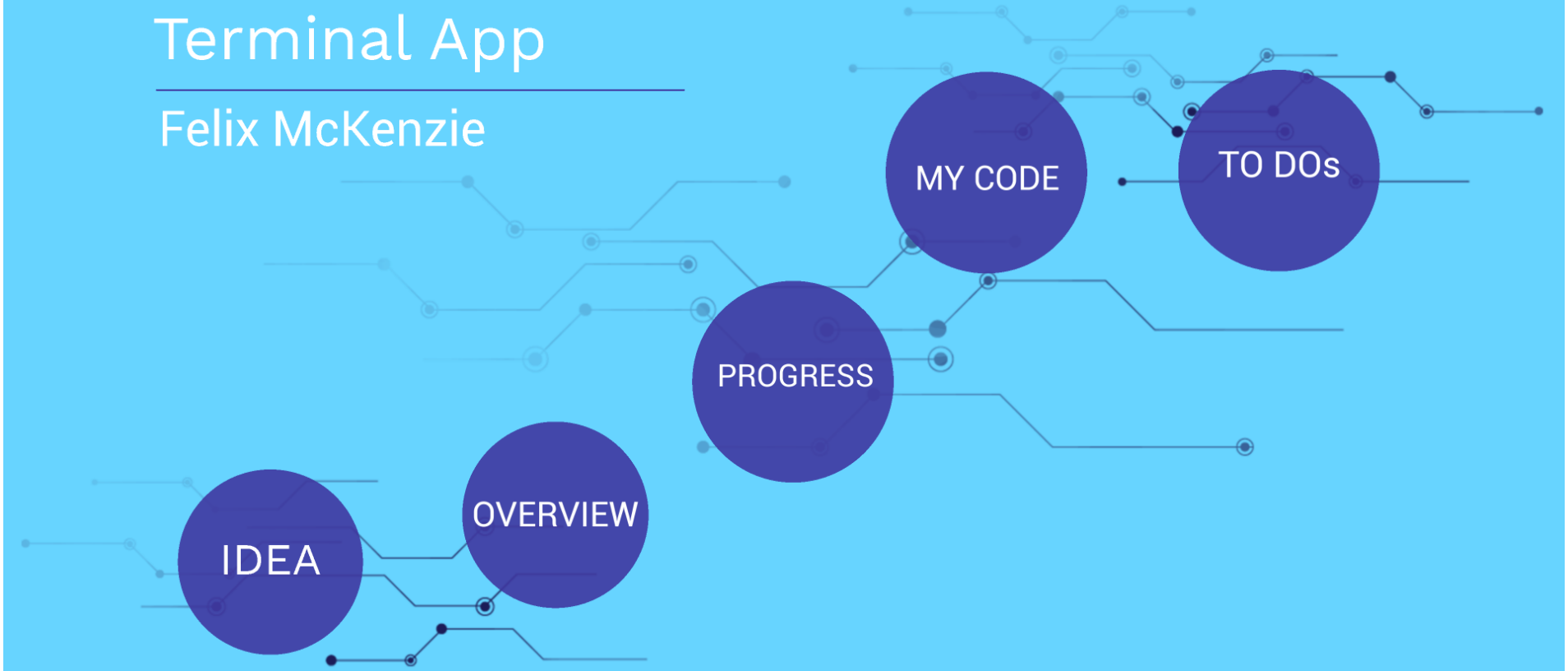


THE IDEA

Quiz app for the terminal

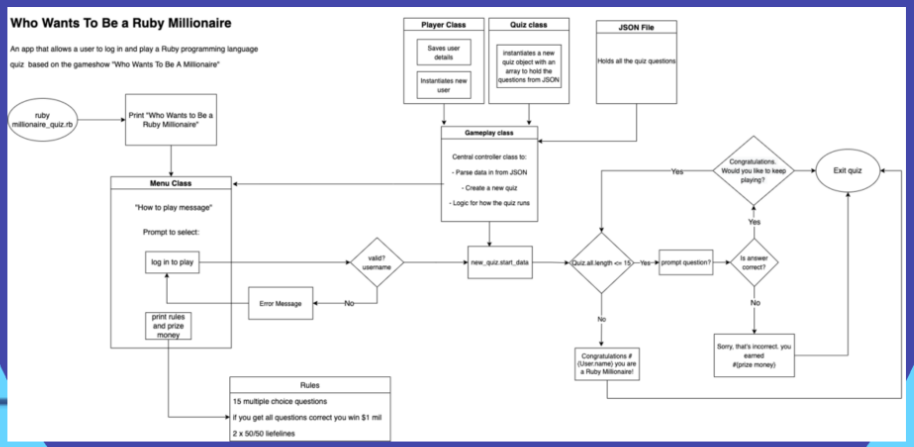
Terminal App

Felix McKenzie



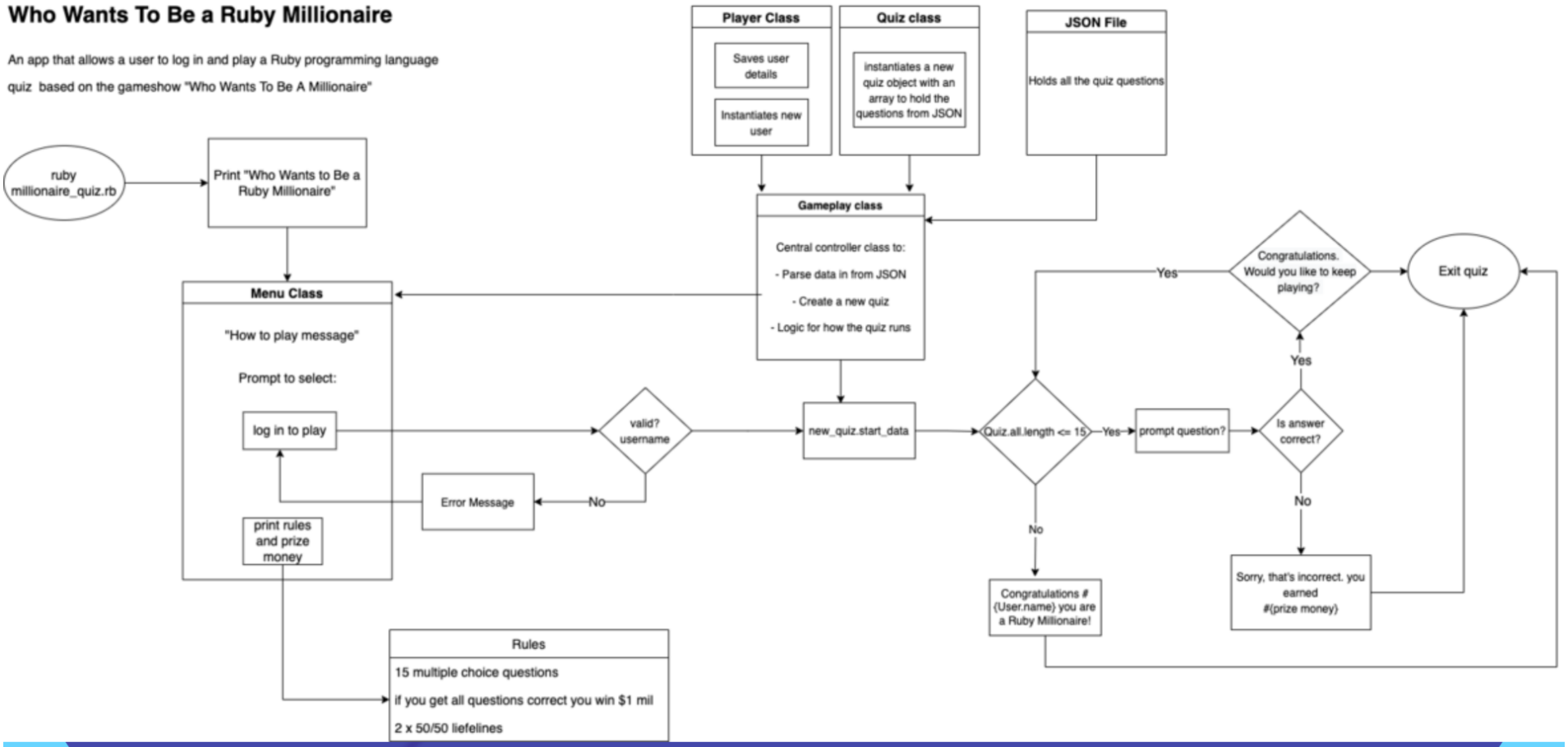
FLOW CHART

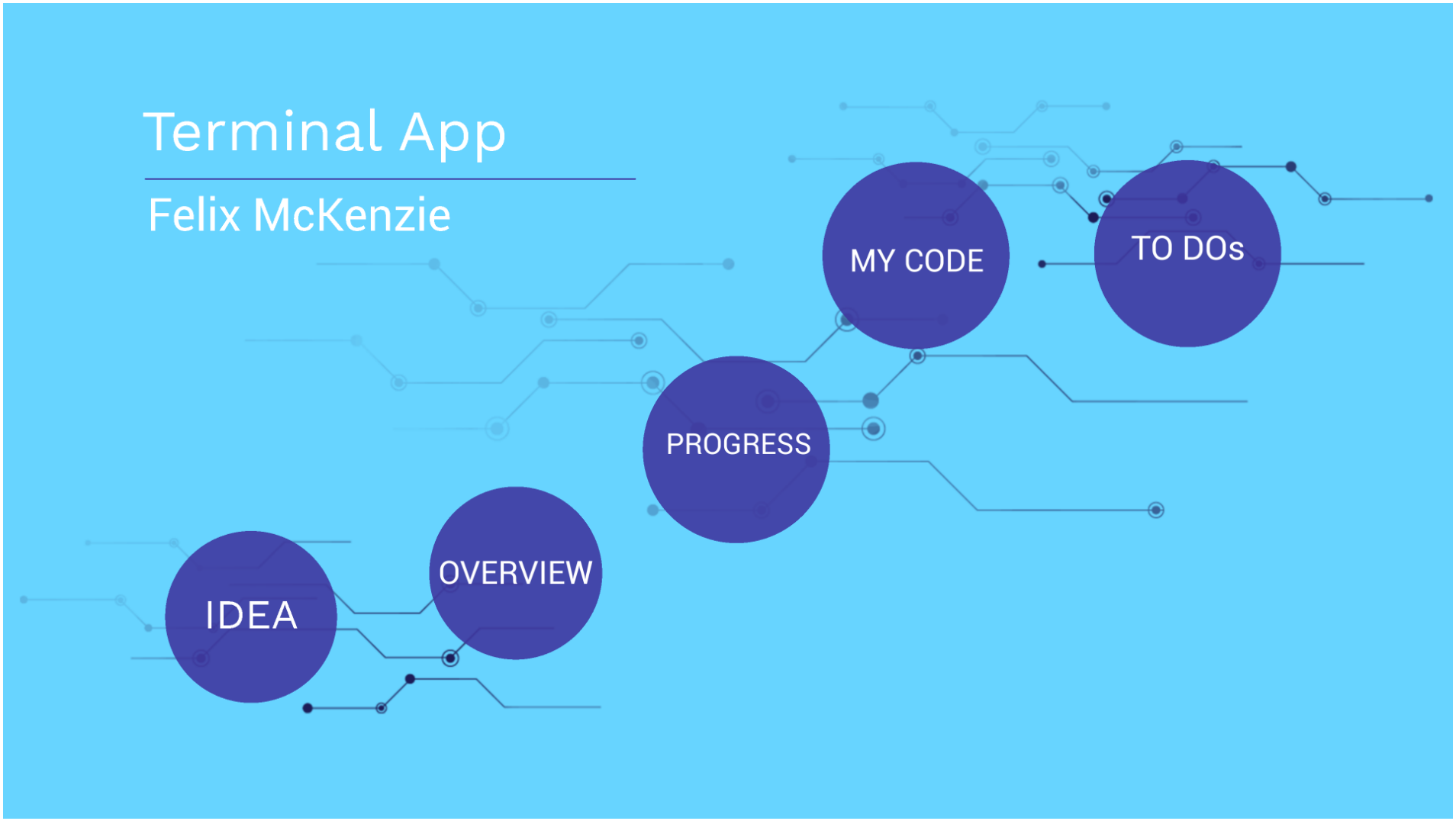
overview



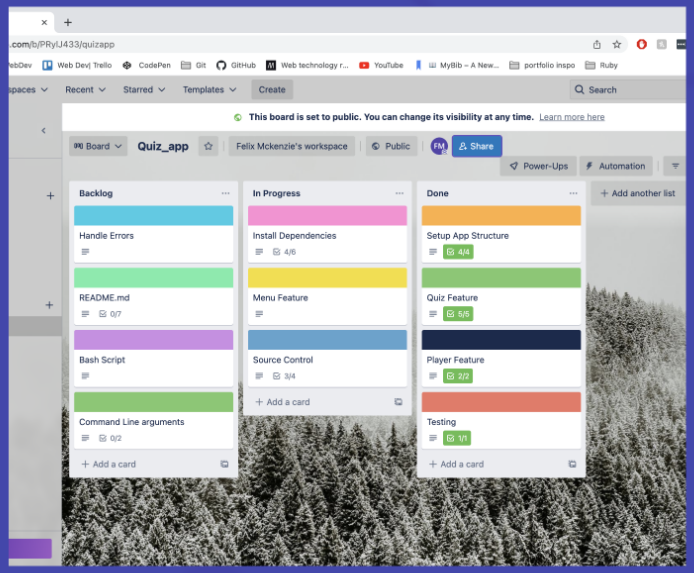
Who Wants To Be a Ruby Millionaire

An app that allows a user to log in and play a Ruby programming language quiz based on the gameshow "Who Wants To Be A Millionaire"




Terminal App
Felix McKenzie
MY CODE
TO DOs
PROGRESS
OVERVIEW
IDEA

SO FAR....

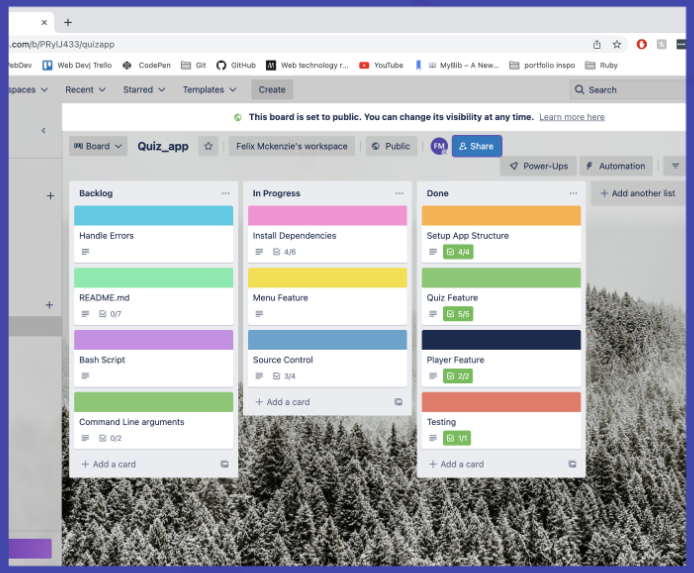


CHALLENGES

Passing methods and variables between classes. "Gluing it all together"

Debugging code where it uses multiple classes. I need to remind myself to slow down and read the error messages properly

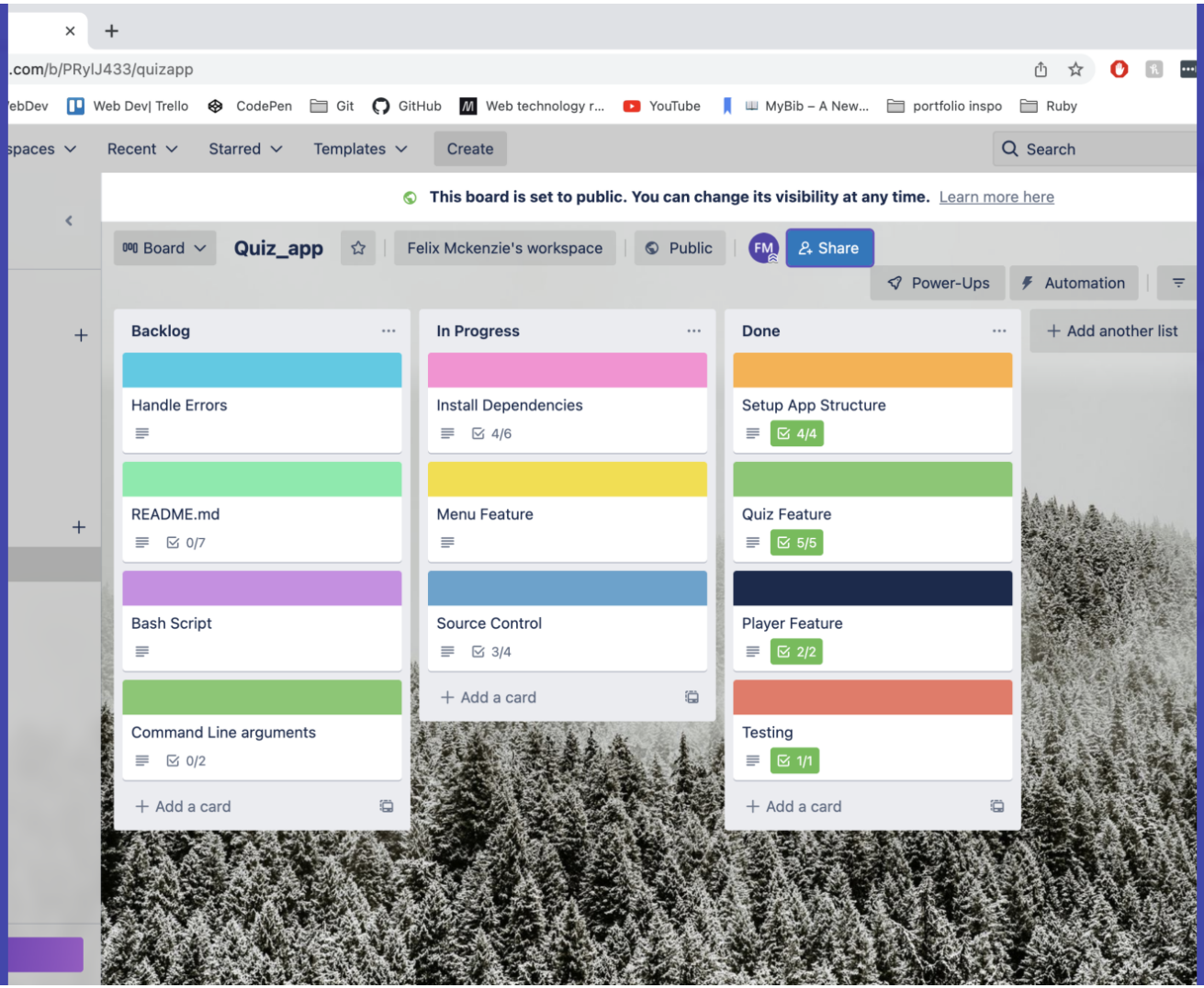
SO FAR....

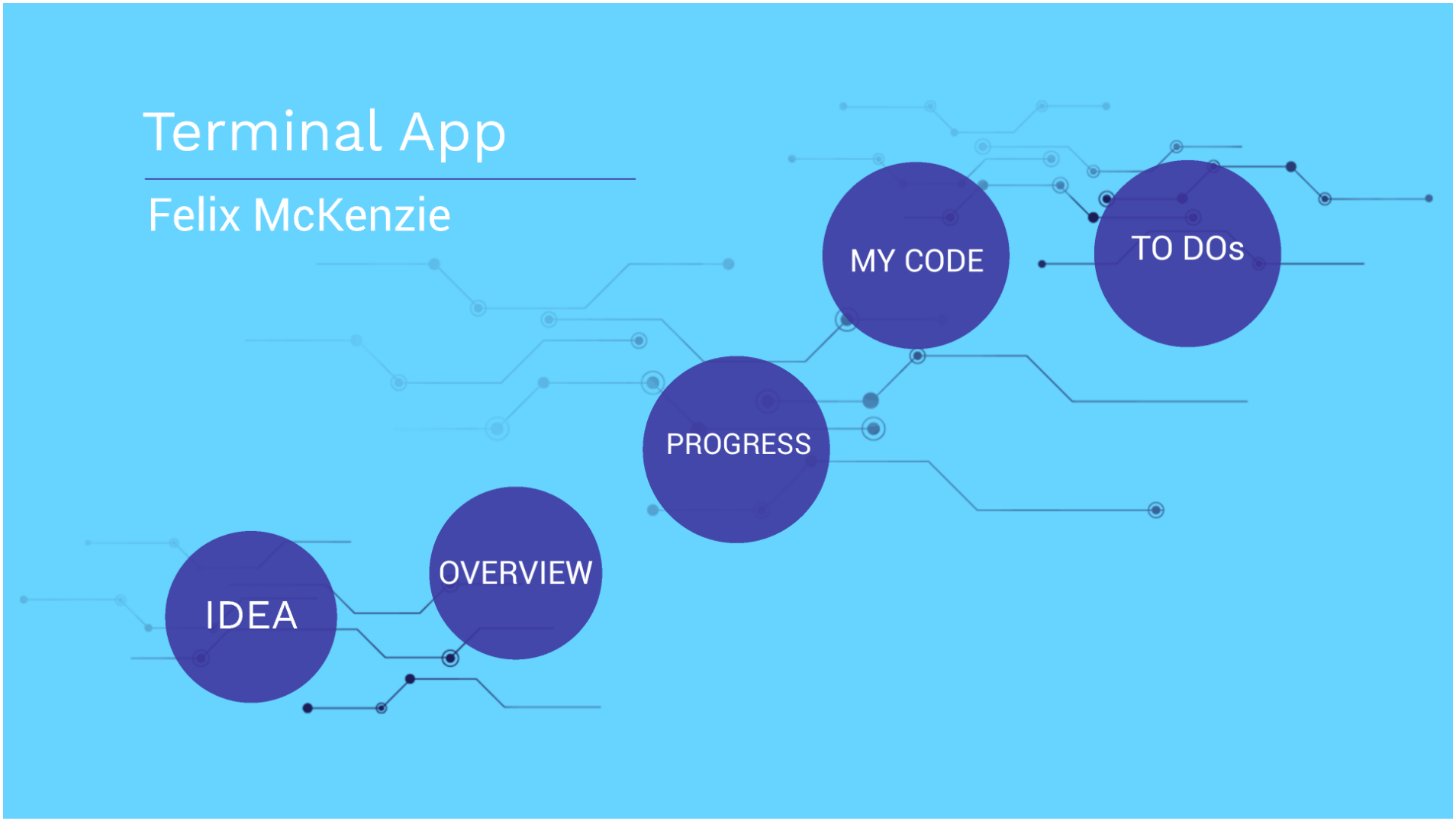


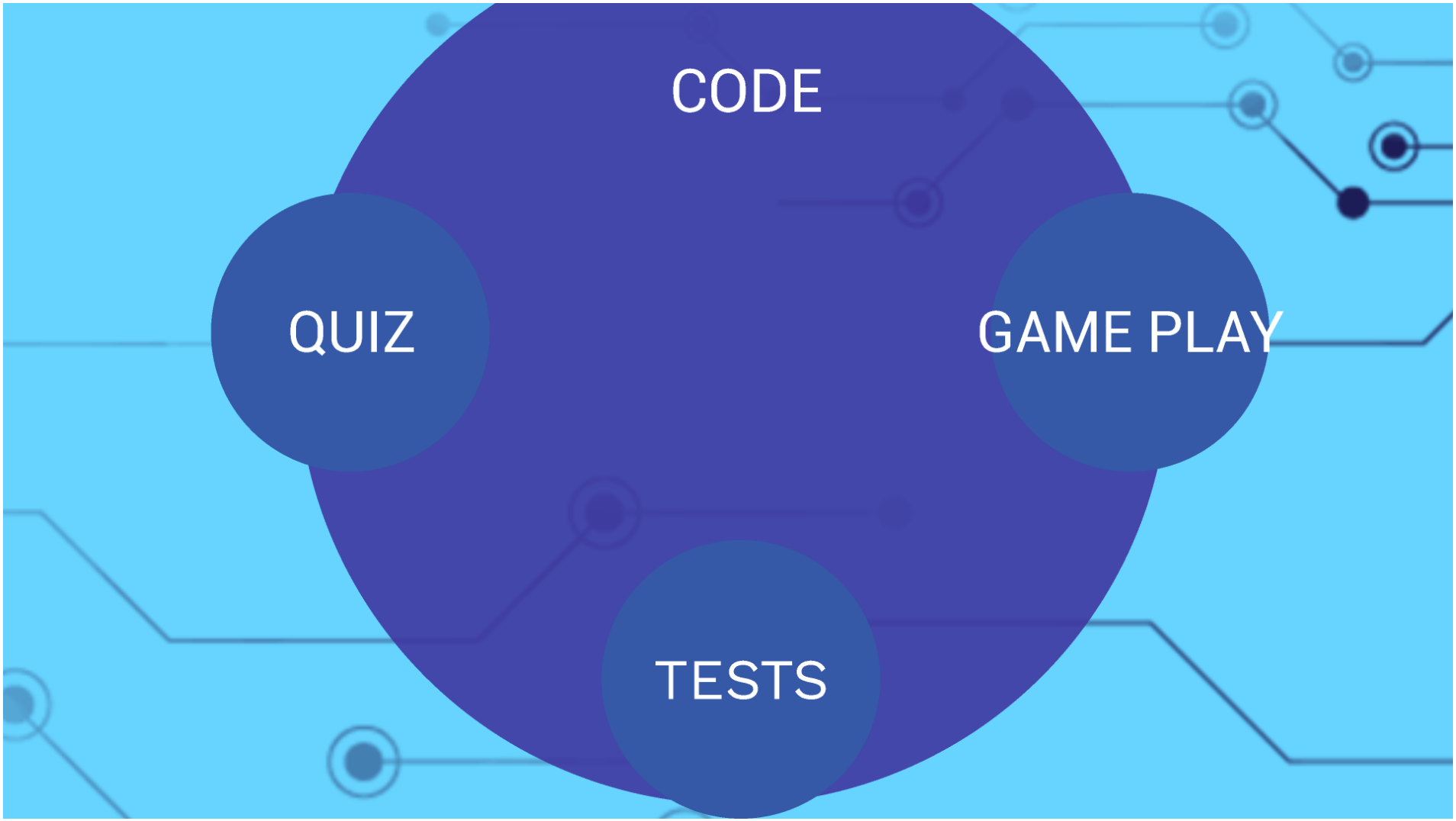
CHALLENGES

Passing methods and variables between classes. "Gluing it all together"

Debugging code where it uses multiple classes. I need to remind myself to slow down and read the error messages properly

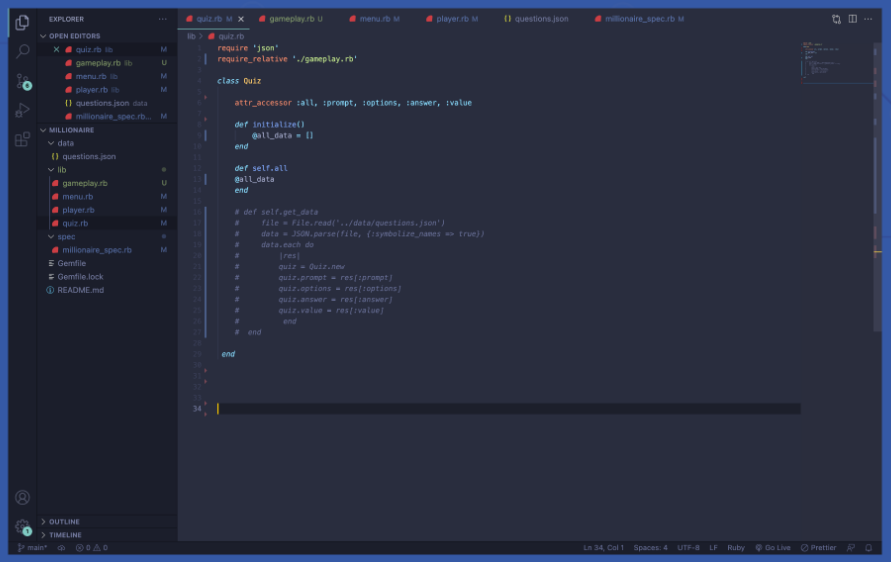

.com/b/PRylJ433/quizapp
Web Dev| Trello
CodePen
Git
GitHub
Web technology r...
YouTube
MyBib – A New...
portfolio inspo
Ruby
Recent
Starred
Templates
Create
Search
This board is set to public. You can change its visibility at any time. Learn more here
Board
Quiz_app
Felix Mckenzie's workspace
Public
FM
Share
Power-Ups
Automation
Backlog
In Progress
Done
Add another list
Handle Errors
README.md
0/7
Bash Script
Command Line arguments
0/2
Add a card
Install Dependencies
4/6
Menu Feature
Source Control
3/4
Add a card
Setup App Structure
4/4
Quiz Feature
5/5
Player Feature
2/2
Testing
1/1
Add a card


Terminal App
Felix McKenzie
MY CODE
TO DOs
PROGRESS
OVERVIEW
IDEA


CODE
QUIZ
GAME PLAY
TESTS

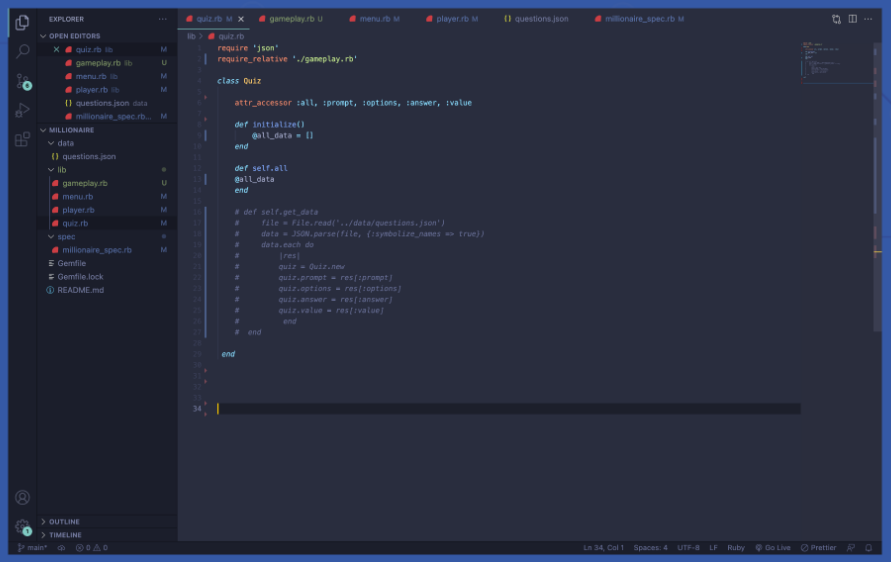
QUIZ CLASS

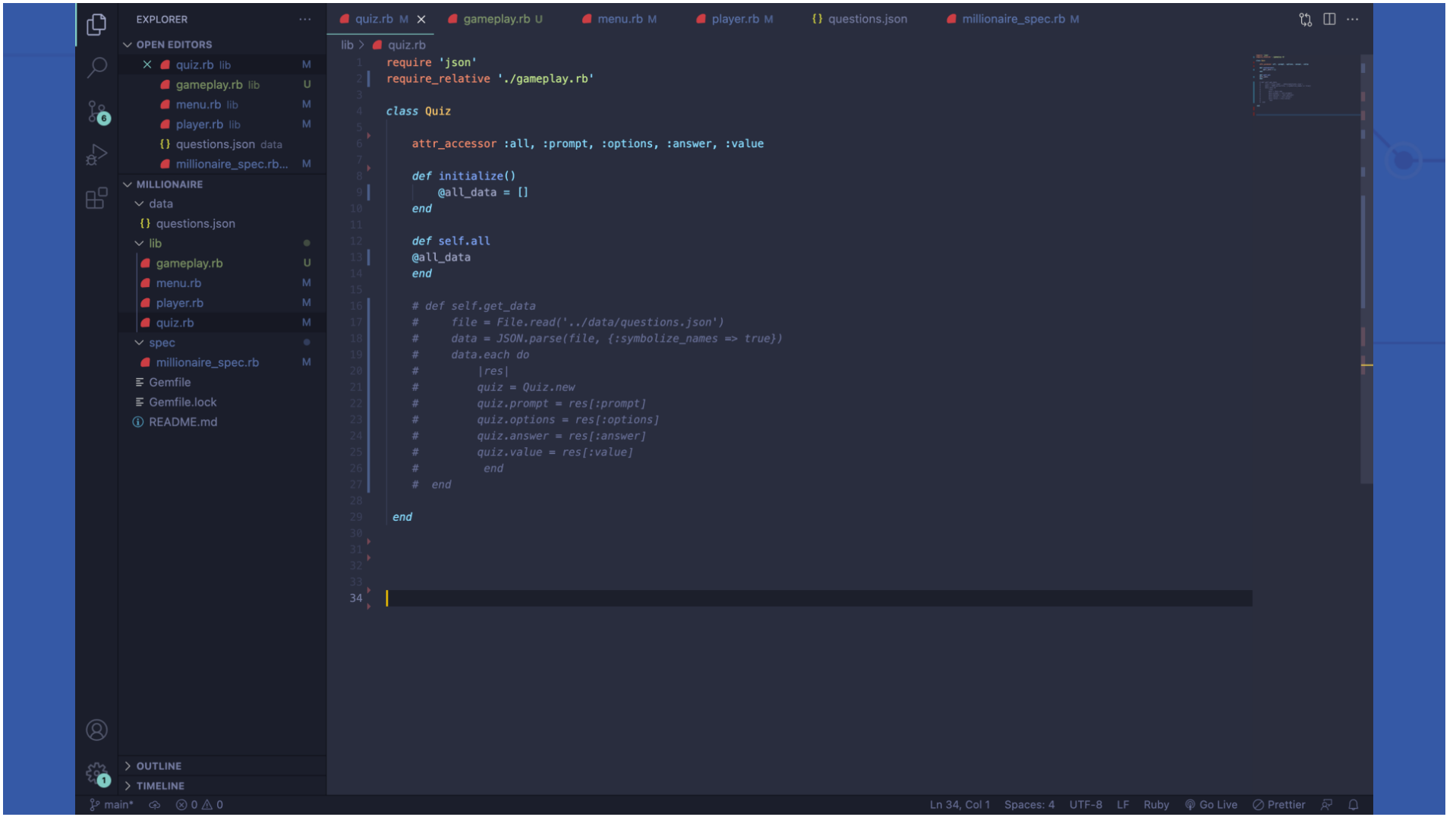
New instance created when data is parsed from JSON file, has an array instance variable that holds the questions

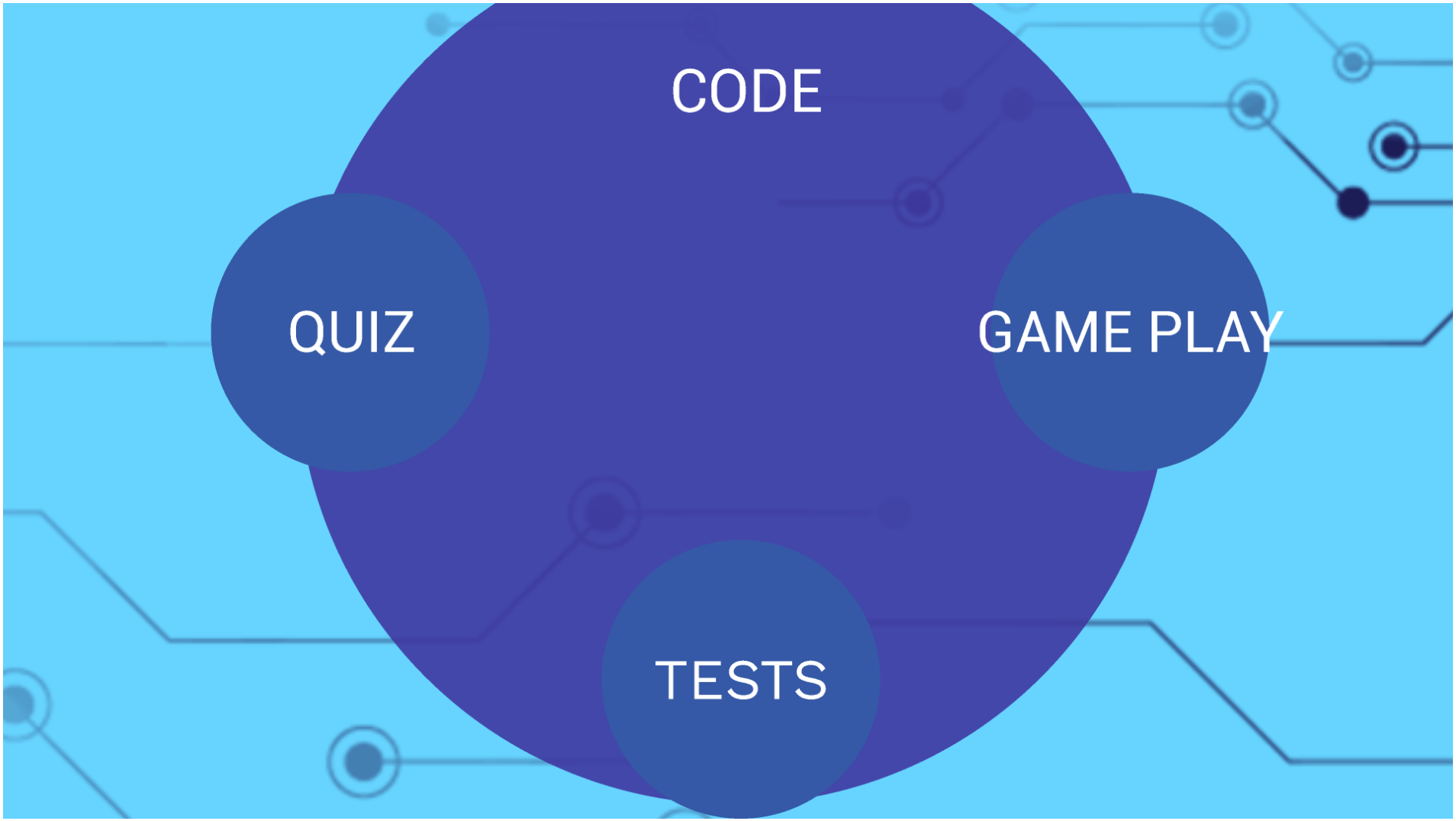


QUIZ CLASS

New instance created when data is parsed from JSON file, has an array instance variable that holds the questions

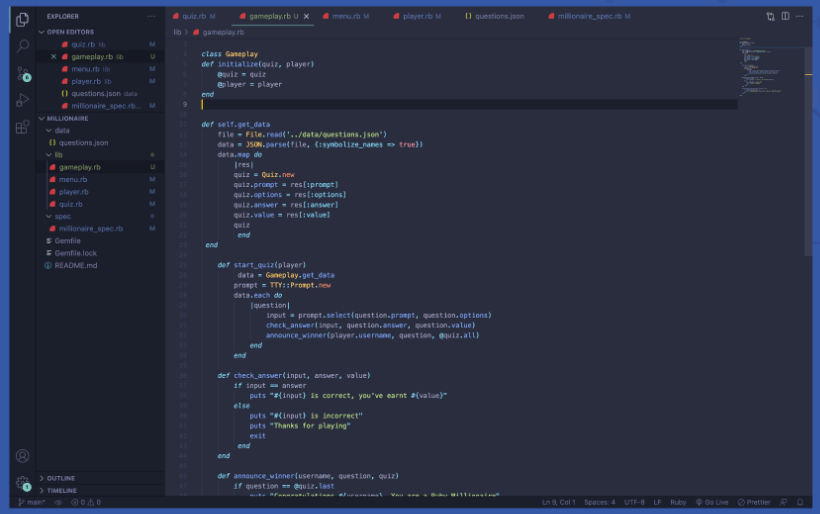



EXPLORER
OPEN EDITORS
quiz.rb lib M
gameplay.rb lib U
menu.rb lib M
player.rb lib M
questions.json data
millionaire_spec.rb... M
MILLIONAIRE
data
questions.json
lib
gameplay.rb U
menu.rb M
player.rb M
quiz.rb M
spec
millionaire_spec.rb M
Gemfile
Gemfile.lock
README.md
OUTLINE
TIMELINE
quiz.rb M
gameplay.rb U
menu.rb M
player.rb M
questions.json
millionaire_spec.rb M
lib > quiz.rb
require 'json'
require_relative './gameplay.rb'
class Quiz
attr_accessor :all, :prompt, :options, :answer, :value
def initialize()
@all_data = []
end
def self.all
@all_data
end
def self.get_data
file = File.read('../data/questions.json')
data = JSON.parse(file, {:symbolize_names => true})
data.each do
|res|
quiz = Quiz.new
quiz.prompt = res[:prompt]
quiz.options = res[:options]
quiz.answer = res[:answer]
quiz.value = res[:value]
end
end
end
main*
0 0
Ln 34, Col 1
Spaces: 4
UTF-8
LF
Ruby
Go Live
Prettier

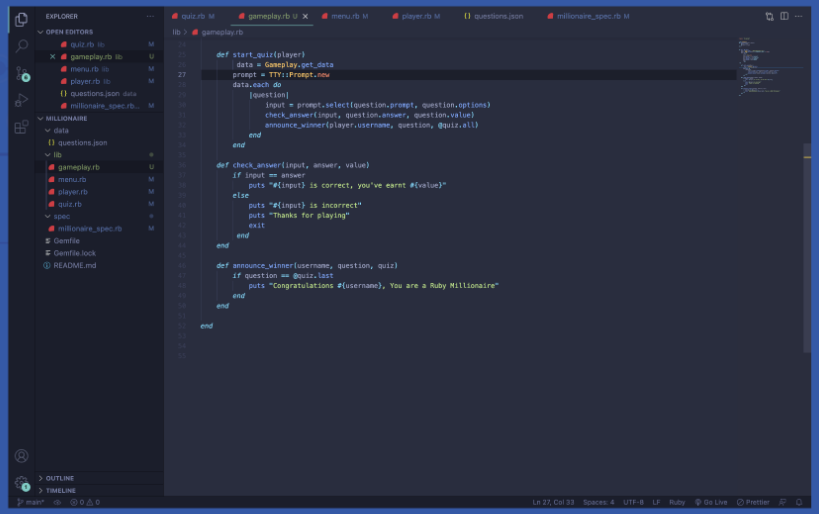

CODE
QUIZ
GAME PLAY
TESTS

GAMEPLAY CLASS

Contains core methods to parse data, execute a quiz, allow the user to interact and determine outcomes.



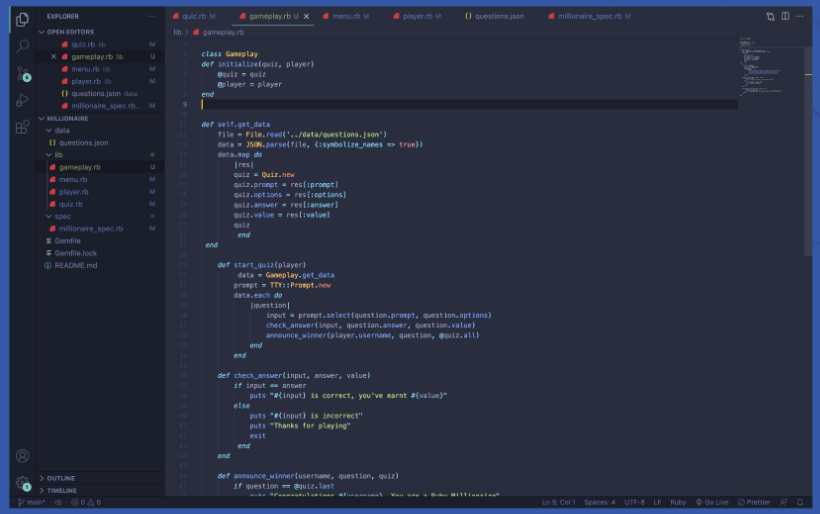
```ruby
class Gameplay
def initialize(quiz, player)
    @quiz = quiz
    @player = player
end

def self.get_data
    file = File.read('../data/questions.json')
    data = JSON.parse(file, {:symbolize_names => true})
    data.map do
        |res|
        quiz = Quiz.new
        quiz.prompt = res[:prompt]
        quiz.options = res[:options]
        quiz.answer = res[:answer]
        quiz.value = res[:value]
        quiz
    end
end

def start_quiz(player)
    data = Gameplay.get_data
    prompt = TTY::Prompt.new
    data.each do
        |question|
        input = prompt.select(question.prompt, question.options)
        check_answer(input, question.answer, question.value)
        announce_winner(player.username, question, @quiz.all)
    end
end

def check_answer(input, answer, value)
    if input == answer
        puts "#{input} is correct, you've earnt #{value}"
    else
        puts "#{input} is incorrect"
        puts "Thanks for playing"
        exit
    end
end

def announce_winner(username, question, quiz)
    if question == @quiz.last
```



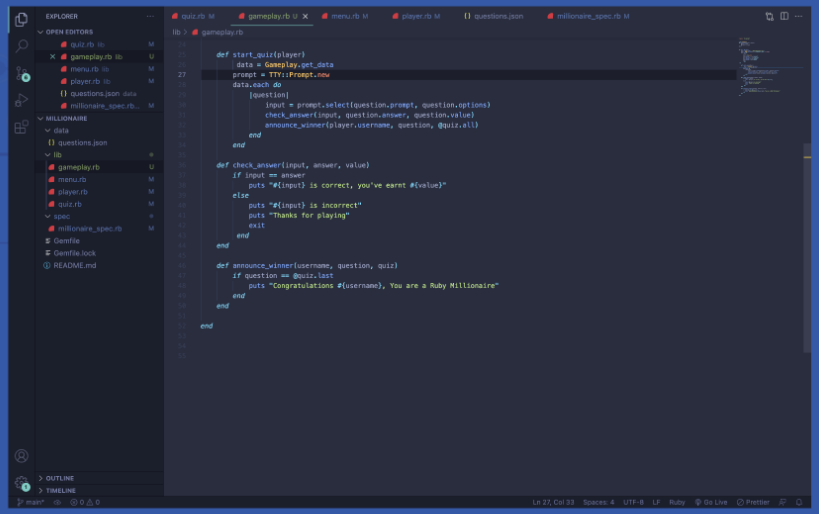
```ruby
def start_quiz(player)
    data = Gameplay.get_data
    prompt = TTY::Prompt.new
    data.each do
        |question|
        input = prompt.select(question.prompt, question.options)
        check_answer(input, question.answer, question.value)
        announce_winner(player.username, question, @quiz.all)
    end
end

def check_answer(input, answer, value)
    if input == answer
        puts "#{input} is correct, you've earnt #{value}"
    else
        puts "#{input} is incorrect"
        puts "Thanks for playing"
        exit
    end
end

def announce_winner(username, question, quiz)
    if question == @quiz.last
        puts "Congratulations #{username}, You are a Ruby Millionaire"
    end
end

end
```

GAMEPLAY CLASS

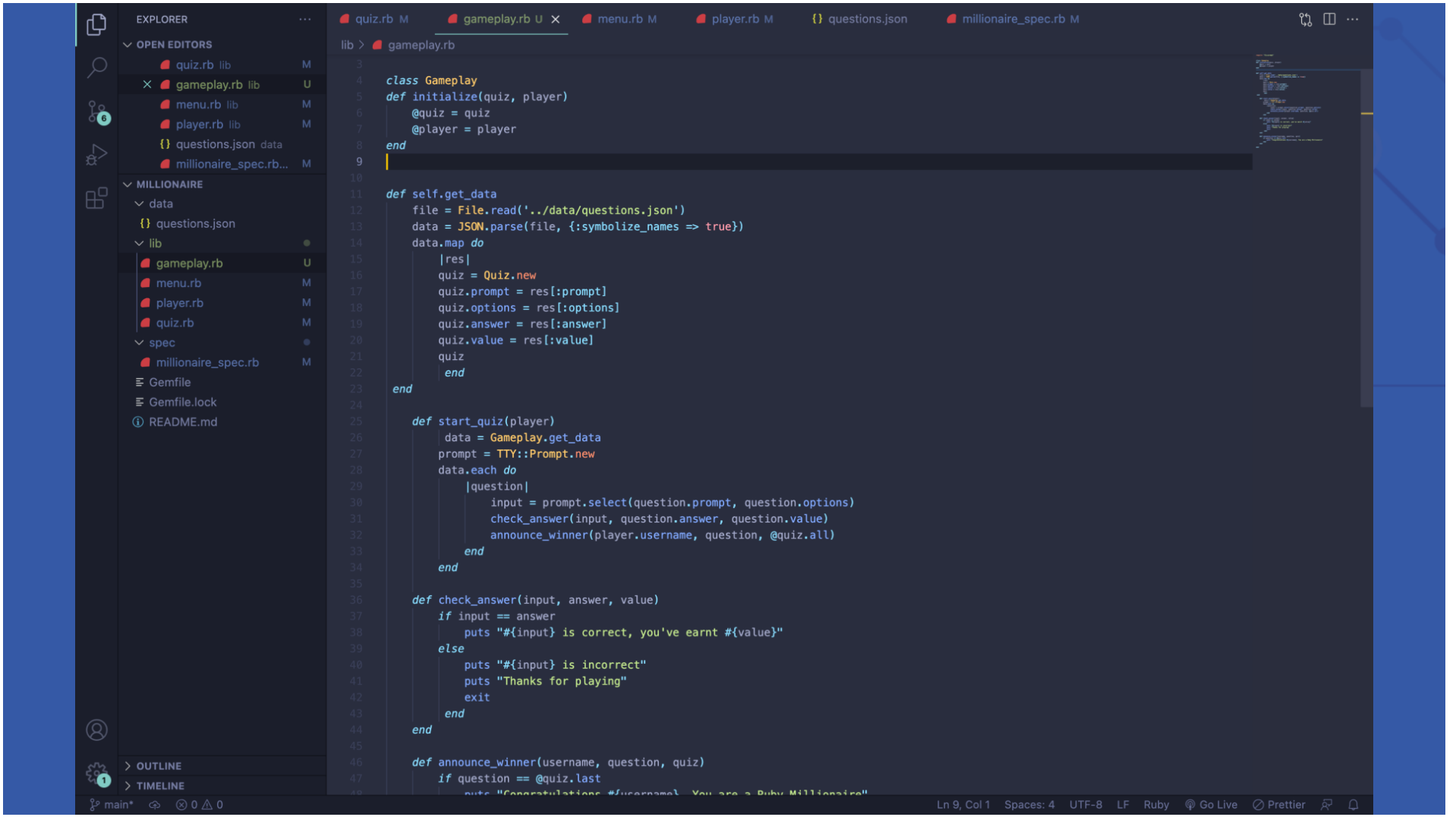
Contains core methods to parse data, execute a quiz, allow the user to interact and determine outcomes.

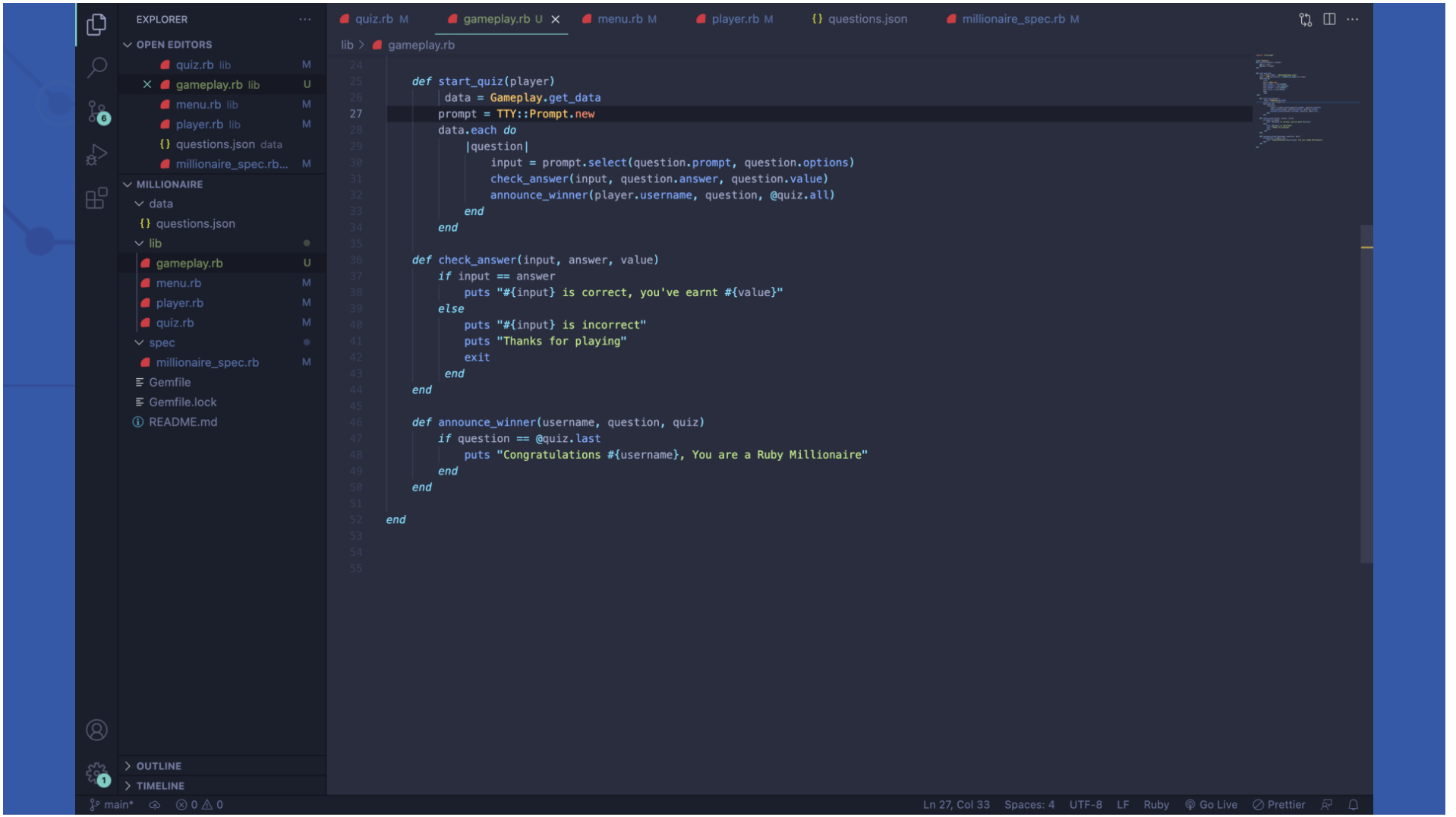


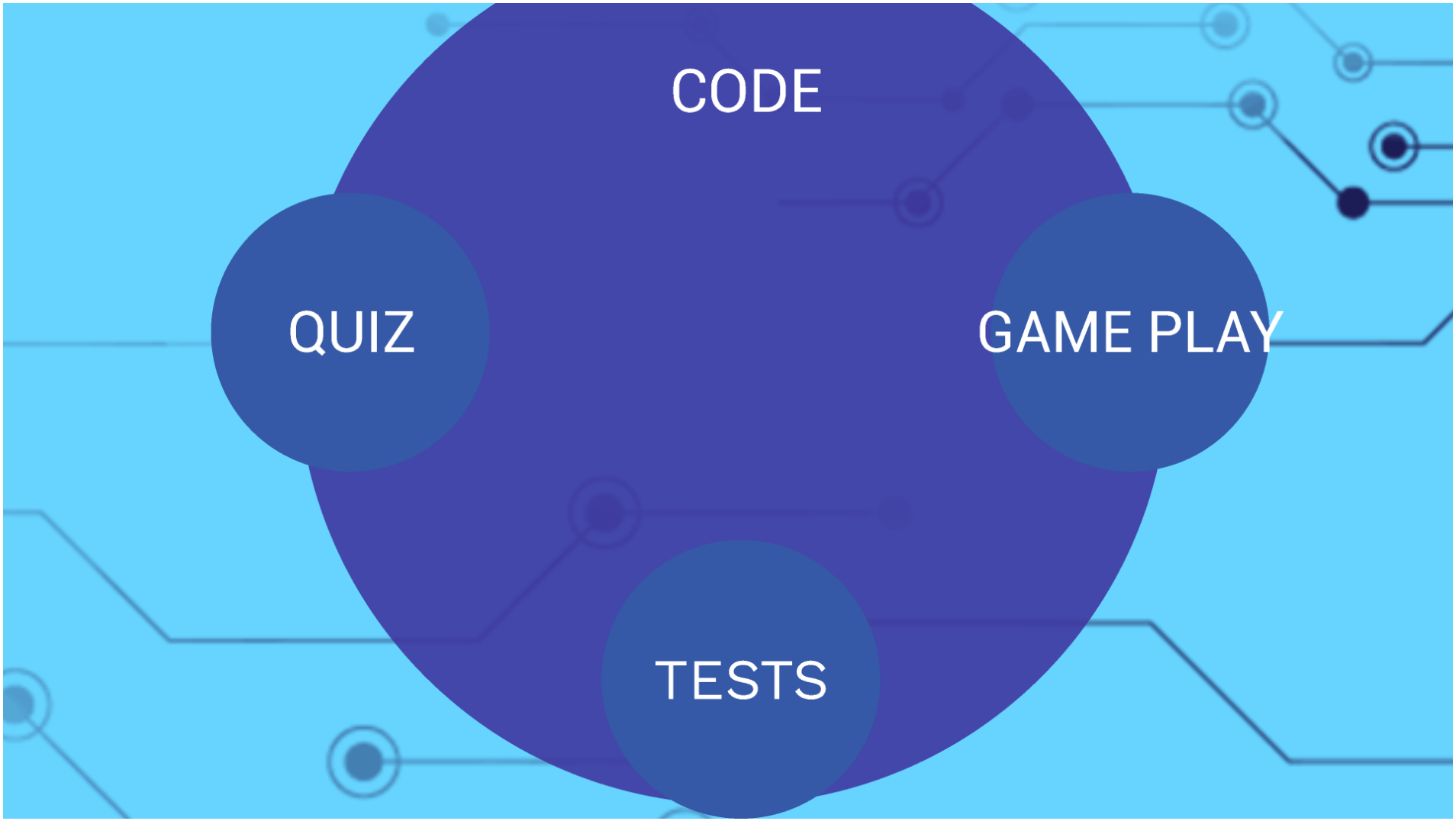
```ruby
class Gameplay
def initialize(quiz, player)
    @quiz = quiz
    @player = player
end

def self.get_data
    file = File.read('../data/questions.json')
    data = JSON.parse(file, {:symbolize_names => true})
    data.map do
        |res|
        quiz = Quiz.new
        quiz.prompt = res[:prompt]
        quiz.options = res[:options]
        quiz.answer = res[:answer]
        quiz.value = res[:value]
        quiz
        end
end

    def start_quiz(player)
        data = Gameplay.get_data
        prompt = TTY::Prompt.new
        data.each do
            |question|
                input = prompt.select(question.prompt, question.options)
                check_answer(input, question.answer, question.value)
                announce_winner(player.username, question, @quiz.all)
            end
        end

    def check_answer(input, answer, value)
        if input == answer
            puts "#{input} is correct, you've earnt #{value}"
        else
            puts "#{input} is incorrect"
            puts "Thanks for playing"
            exit
        end
    end

    def announce_winner(username, question, quiz)
        if question == @quiz.last
```



```ruby
    def start_quiz(player)
        data = Gameplay.get_data
        prompt = TTY::Prompt.new
        data.each do
            |question|
                input = prompt.select(question.prompt, question.options)
                check_answer(input, question.answer, question.value)
                announce_winner(player.username, question, @quiz.all)
            end
        end

    def check_answer(input, answer, value)
        if input == answer
            puts "#{input} is correct, you've earnt #{value}"
        else
            puts "#{input} is incorrect"
            puts "Thanks for playing"
            exit
        end
    end

    def announce_winner(username, question, quiz)
        if question == @quiz.last
            puts "Congratulations #{username}, You are a Ruby Millionaire"
        end
    end

end
```

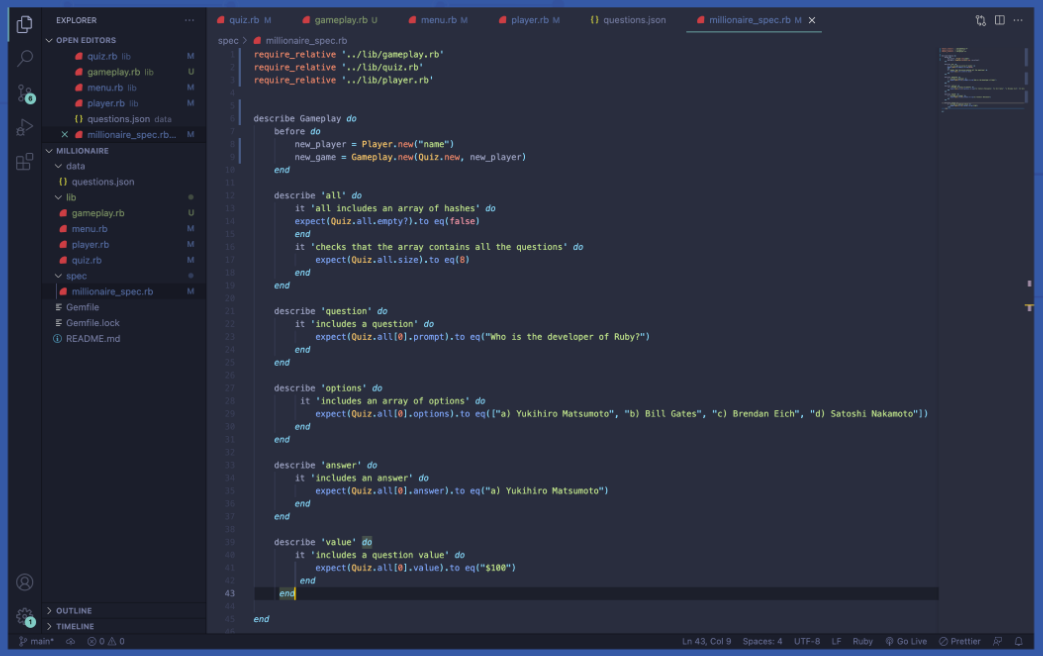

EXPLORER
OPEN EDITORS
quiz.rb lib M
gameplay.rb lib U
menu.rb lib M
player.rb lib M
questions.json data
millionaire_spec.rb... M
MILLIONAIRE
data
questions.json
lib
gameplay.rb U
menu.rb M
player.rb M
quiz.rb M
spec
millionaire_spec.rb M
Gemfile
Gemfile.lock
README.md
OUTLINE
TIMELINE
quiz.rb M
gameplay.rb U
menu.rb M
player.rb M
questions.json
millionaire_spec.rb M
lib > gameplay.rb
class Gameplay
def initialize(quiz, player)
@quiz = quiz
@player = player
end
def self.get_data
file = File.read('../data/questions.json')
data = JSON.parse(file, {:symbolize_names => true})
data.map do
|res|
quiz = Quiz.new
quiz.prompt = res[:prompt]
quiz.options = res[:options]
quiz.answer = res[:answer]
quiz.value = res[:value]
quiz
end
end
def start_quiz(player)
data = Gameplay.get_data
prompt = TTY::Prompt.new
data.each do
|question|
input = prompt.select(question.prompt, question.options)
check_answer(input, question.answer, question.value)
announce_winner(player.username, question, @quiz.all)
end
end
def check_answer(input, answer, value)
if input == answer
puts "#{input} is correct, you've earnt #{value}"
else
puts "#{input} is incorrect"
puts "Thanks for playing"
exit
end
end
def announce_winner(username, question, quiz)
if question == @quiz.last
main*
0 0
Ln 9, Col 1
Spaces: 4
UTF-8
LF
Ruby
Go Live
Prettier


EXPLORER
OPEN EDITORS
quiz.rb lib M
gameplay.rb lib U
menu.rb lib M
player.rb lib M
questions.json data
millionaire_spec.rb... M
MILLIONAIRE
data
questions.json
lib
gameplay.rb U
menu.rb M
player.rb M
quiz.rb M
spec
millionaire_spec.rb M
Gemfile
Gemfile.lock
README.md
OUTLINE
TIMELINE
quiz.rb M
gameplay.rb U
menu.rb M
player.rb M
questions.json
millionaire_spec.rb M
lib > gameplay.rb
def start_quiz(player)
data = Gameplay.get_data
prompt = TTY::Prompt.new
data.each do
|question|
input = prompt.select(question.prompt, question.options)
check_answer(input, question.answer, question.value)
announce_winner(player.username, question, @quiz.all)
end
end
def check_answer(input, answer, value)
if input == answer
puts "#{input} is correct, you've earnt #{value}"
else
puts "#{input} is incorrect"
puts "Thanks for playing"
exit
end
end
def announce_winner(username, question, quiz)
if question == @quiz.last
puts "Congratulations #{username}, You are a Ruby Millionaire"
end
end
end
main*
0 0
Ln 27, Col 33
Spaces: 4
UTF-8
LF
Ruby
Go Live
Prettier


CODE
QUIZ
GAME PLAY
TESTS

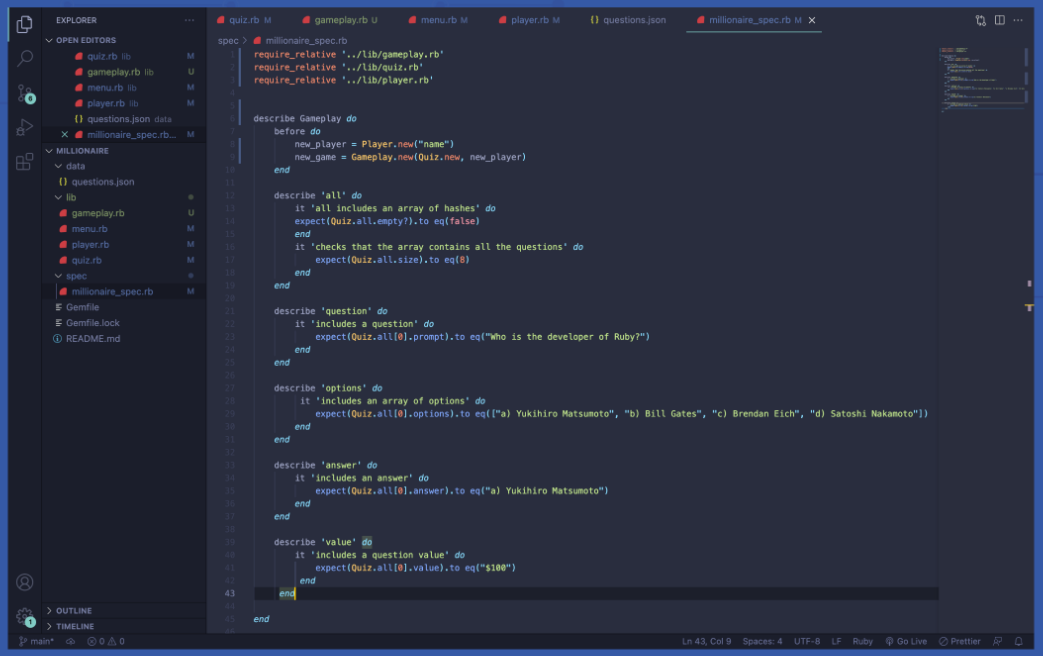
RSPEC

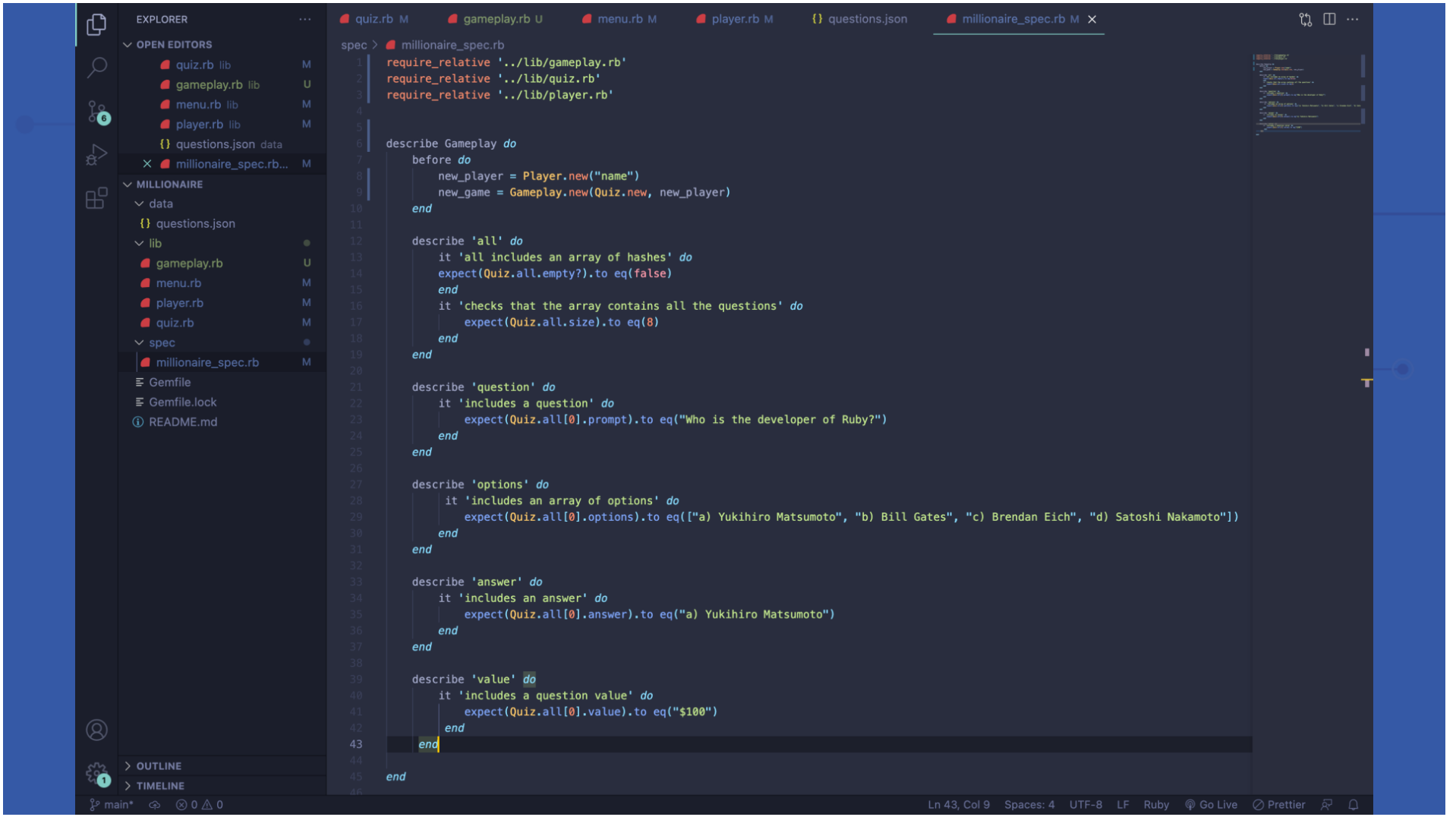
Tests were passing now they're not, to be continued.....



RSPEC

Tests were passing now they're not, to be continued.....




EXPLORER
OPEN EDITORS
quiz.rb lib M
gameplay.rb lib U
menu.rb lib M
player.rb lib M
questions.json data
millionaire_spec.rb... M
MILLIONAIRE
data
questions.json
lib
gameplay.rb U
menu.rb M
player.rb M
quiz.rb M
spec
millionaire_spec.rb M
Gemfile
Gemfile.lock
README.md
OUTLINE
TIMELINE
quiz.rb M
gameplay.rb U
menu.rb M
player.rb M
questions.json
millionaire_spec.rb M
spec > millionaire_spec.rb
require_relative '../lib/gameplay.rb'
require_relative '../lib/quiz.rb'
require_relative '../lib/player.rb'
describe Gameplay do
before do
new_player = Player.new("name")
new_game = Gameplay.new(Quiz.new, new_player)
end
describe 'all' do
it 'all includes an array of hashes' do
expect(Quiz.all.empty?).to eq(false)
end
it 'checks that the array contains all the questions' do
expect(Quiz.all.size).to eq(8)
end
end
describe 'question' do
it 'includes a question' do
expect(Quiz.all[0].prompt).to eq("Who is the developer of Ruby?")
end
end
describe 'options' do
it 'includes an array of options' do
expect(Quiz.all[0].options).to eq(["a) Yukihiro Matsumoto", "b) Bill Gates", "c) Brendan Eich", "d) Satoshi Nakamoto"])
end
end
describe 'answer' do
it 'includes an answer' do
expect(Quiz.all[0].answer).to eq("a) Yukihiro Matsumoto")
end
end
describe 'value' do
it 'includes a question value' do
expect(Quiz.all[0].value).to eq("$100")
end
end
end
main*
0 0
Ln 43, Col 9
Spaces: 4
UTF-8
LF
Ruby
Go Live
Prettier