```
from question_model import Question
from data import question_data
from quiz_brain import QuizBrain
from ui import QuizInterface

question_bank = []
for question in question_data:
    question_text = question["question"]
    question_answer = question["correct_answer"]
    new_question = Question(question_text, question_answer)
    question_bank.append(new_question)


quiz = QuizBrain(question_bank)
quiz_ui = QuizInterface(quiz)

#question_model.py
class Question:

    def __init__(self, q_text, q_answer):
        self.text = q_text
        self.answer = q_answer

#ui.py
from tkinter import *
from quiz_brain import QuizBrain

THEME_COLOR = "#375362"


class QuizInterface:

    def __init__(self, quiz_brain: QuizBrain):
        self.quiz = quiz_brain

        self.window = Tk()
        self.window.title("Quizzler")
        self.window.config(padx=20, pady=20, bg=THEME_COLOR)

```

```
40          self.score_label = Label(text="Score: 0", fg=
    "white", bg=THEME_COLOR)
41          self.score_label.grid(row=0, column=1)
42
43          self.canvas = Canvas(width=300, height=250,
    bg="white")
44          self.question_text = self.canvas.create_text(
45              150,
46              125,
47              width=280,
48              text="Some Question Text",
49              fill=THEME_COLOR,
50              font=("Arial", 20, "italic")
51          )
52          self.canvas.grid(row=1, column=0, columnspan=
    2, pady=50)
53
54          true_image = PhotoImage(file="images/true.png
    ")
55          self.true_button = Button(image=true_image,
    highlightthickness=0, command=self.true_pressed)
56          self.true_button.grid(row=2, column=0)
57
58          false_image = PhotoImage(file="images/false.
    png")
59          self.false_button = Button(image=false_image
    , highlightthickness=0, command=self.false_pressed)
60          self.false_button.grid(row=2, column=1)
61
62          self.get_next_question()
63
64          self.window.mainloop()
65
66      def get_next_question(self):
67          self.canvas.config(bg="white")
68          if self.quiz.still_has_questions():
69              self.score_label.config(text=f"Score: {
    self.quiz.score}")
70              q_text = self.quiz.next_question()
71              self.canvas.itemconfig(self.question_text
    , text=q_text)
```

```
        else:
            self.canvas.itemconfig(self.question_text, text="You've reached the end of the quiz.")
            self.true_button.config(state="disabled")
            self.false_button.config(state="disabled")

    def true_pressed(self):
        self.give_feedback(self.quiz.check_answer("True"))

    def false_pressed(self):
        is_right = self.quiz.check_answer("False")
        self.give_feedback(is_right)

    def give_feedback(self, is_right):
        if is_right:
            self.canvas.config(bg="green")
        else:
            self.canvas.config(bg="red")
        self.window.after(1000, self.get_next_question)

#data.py
import requests

parameters = {
    "amount": 10,
    "type": "boolean",
}

response = requests.get("https://opentdb.com/api.php", params=parameters)
response.raise_for_status()
data = response.json()
question_data = data["results"]

#quiz_brain
import html
```

```
106
107
108 class QuizBrain:
109
110     def __init__(self, q_list):
111         self.question_number = 0
112         self.score = 0
113         self.question_list = q_list
114         self.current_question = None
115
116     def still_has_questions(self):
117         return self.question_number < len(self.
    question_list)
118
119     def next_question(self):
120         self.current_question = self.question_list[
    self.question_number]
121         self.question_number += 1
122         q_text = html.unescape(self.current_question
    .text)
123         return f"Q.{self.question_number}: {q_text}"
124
125     def check_answer(self, user_answer):
126         correct_answer = self.current_question.
    answer
127         if user_answer.lower() == correct_answer.
    lower():
128             self.score += 1
129             return True
130         else:
131             return False
132
133
134
135
136
```