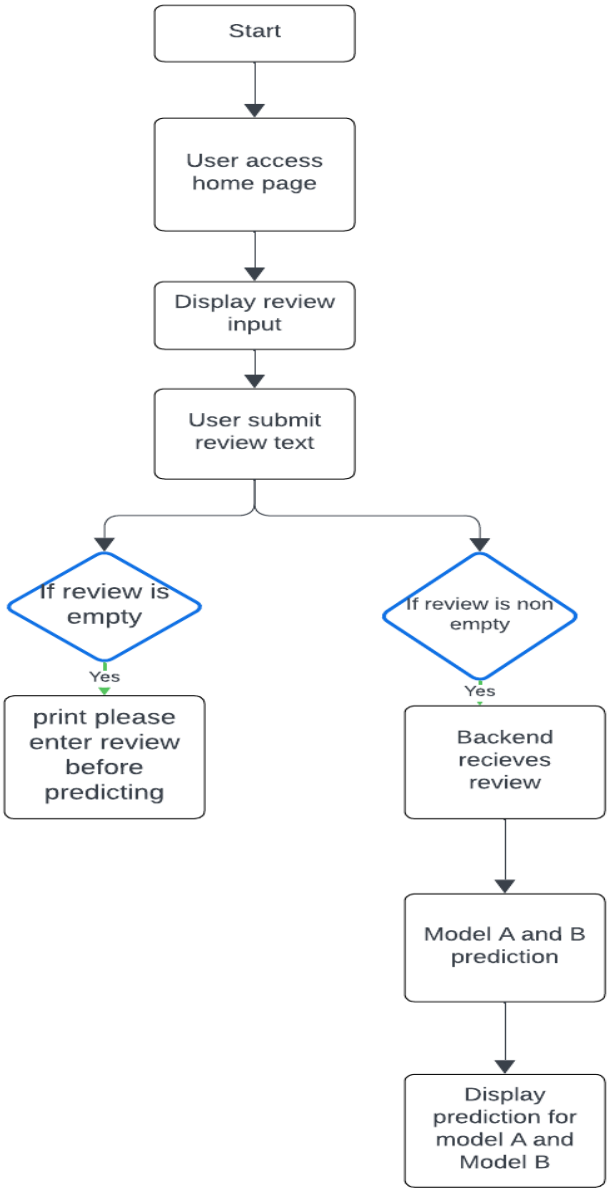
AUTOMATED REVIEW SYSTEM

Flow chart:



HTML:

```
<!DOCTYPE html>
<html>
<head>
    <title>Automated Review System</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            max-width: 700px;
            margin: auto;
            padding: 20px;
        }
        textarea {
            width: 100%;
            height: 120px;
            font-size: 16px;
            padding: 10px;
        }
        input[type="submit"] {
            padding: 10px 20px;
            font-size: 16px;
            background-color: #007BFF;
            color: white;
            border: none;
            border-radius: 5px;
        }
        .result {
```

```
            margin-top: 20px;
            padding: 15px;
            background: #f1f1f1;
            border-radius: 5px;
        }
        .success { color: green; }
        .info { color: orange; }
    </style>
</head>
<body>
    <h1>📝 AUTOMATED REVIEW SYSTEM</h1>
    <p>This app compares predictions from two models: Balanced and Imbalanced.</p>

    <form method="post">
        <label for="review">✍️ Enter a product review:</label><br>
        <textarea name="review" required>{{ review or '' }}</textarea><br><br>
        <input type="submit" value="🔍 Predict">
    </form>

    {% if prediction_A is not none and prediction_B is not none %}
        <div class="result">
            <h3>☑️ Model Predictions</h3>
            <p><strong>Balanced Model:</strong> {{ prediction_A }}</p>
            <p><strong>Imbalanced Model:</strong> {{ prediction_B }}</p>

            {% if agreement %}
                <p class="success">☑️ Both models agree on the prediction.</p>
            {% else %}
                <p class="info">ℹ️ The models gave <strong>different</strong> predictions.</p>
```

```
            {% endif %}
        </div>
    {% endif %}

    <hr>
    <small>Built with Flask</small>
</body>
</html>
```

Flask:

```
from flask import Flask, render_template, request
from transformers import AutoTokenizer, AutoModelForSequenceClassification
import torch
import torch.nn.functional as F

app = Flask(__name__)

# Load both models
tokenizer_A = AutoTokenizer.from_pretrained("./Model A")
model_A = AutoModelForSequenceClassification.from_pretrained("./Model A")
model_A.eval()

tokenizer_B = AutoTokenizer.from_pretrained("./Model B")
model_B = AutoModelForSequenceClassification.from_pretrained("./Model B")
model_B.eval()

def predict_rating(model, tokenizer, text):
    inputs = tokenizer(text, return_tensors="pt", truncation=True, padding=True, max_length=128)
    with torch.no_grad():
        outputs = model(**inputs)
```

```
        probs = F.softmax(outputs.logits, dim=1)
        pred_class = torch.argmax(probs, dim=1).item()
    return pred_class + 1  # Classes from 1 to 5

@app.route("/", methods=["GET", "POST"])
def index():
    prediction_A = prediction_B = review = None
    agreement = None

    if request.method == "POST":
        review = request.form["review"]

        prediction_A = predict_rating(model_A, tokenizer_A, review)
        prediction_B = predict_rating(model_B, tokenizer_B, review)

        agreement = (prediction_A == prediction_B)

    return render_template(
        "index.html",
        review=review,
        prediction_A=prediction_A,
        prediction_B=prediction_B,
        agreement=agreement
    )
if __name__ == "__main__":
    app.run(debug=True)
```

UI Design Decisions

- **When the page first loads (GET request)**:
  - Display the form with:
    - A header: "📝 AUTOMATED REVIEW SYSTEM"
    - A short description about comparing two models.
    - A textarea for review input (empty by default).
    - A "🔍 Predict" button.
  - No results or predictions are shown yet.

- **When the user submits the form (POST request)**:
  - The app checks if the review field is filled:
    - If **empty**: HTML form validation prevents submission (`required` in textarea).
    - If **not empty**: process continues.

- **After receiving input**:
  - The backend runs predictions using both models (Model A and Model B).
  - Both predictions (rating values from 1 to 5) are returned to the frontend.

- **Frontend then displays the results section**:
  - A titled block appears: "☑ Model Predictions"
  - It shows:
    - Balanced Model prediction (e.g., "4")
    - Imbalanced Model prediction (e.g., "5")

- **Based on comparison of predictions**:
  - If the two predictions are the same:
    - Show a green message: ☑ Both models agree on the prediction.
  - If the two predictions are different:
    - Show an orange message: **ℹ** The models gave different predictions.

- **User may edit the review and submit again**:
  - The previous review remains in the textarea (sticky input).
  - Updated predictions are shown after new submission.

- **At the bottom**:
    - A horizontal line (`<hr>`)
    - A footer note: “Built with Flask”

User Flow

1. Entry Point

- User visits `/` (home page of the app).

2. Interaction

- The user sees a header titled “ 📝 AUTOMATED REVIEW SYSTEM”.
- Below that, a short explanation: “This app compares predictions from two models: Balanced and Imbalanced.”
- A text box is presented for the user to **enter a product review**.
- A submit button labeled 🔍 **Predict** is displayed.

3. Action

- The user writes a review (e.g., “The product quality was amazing”) and clicks **Predict**.

4. Processing

- The app:
    - Passes the review to **Model A** (balanced data model) and **Model B** (imbalanced data model).
    - Tokenizes, classifies, and computes softmax scores.
    - Retrieves the predicted class (1 to 5 rating) from both models.

5. Outcome

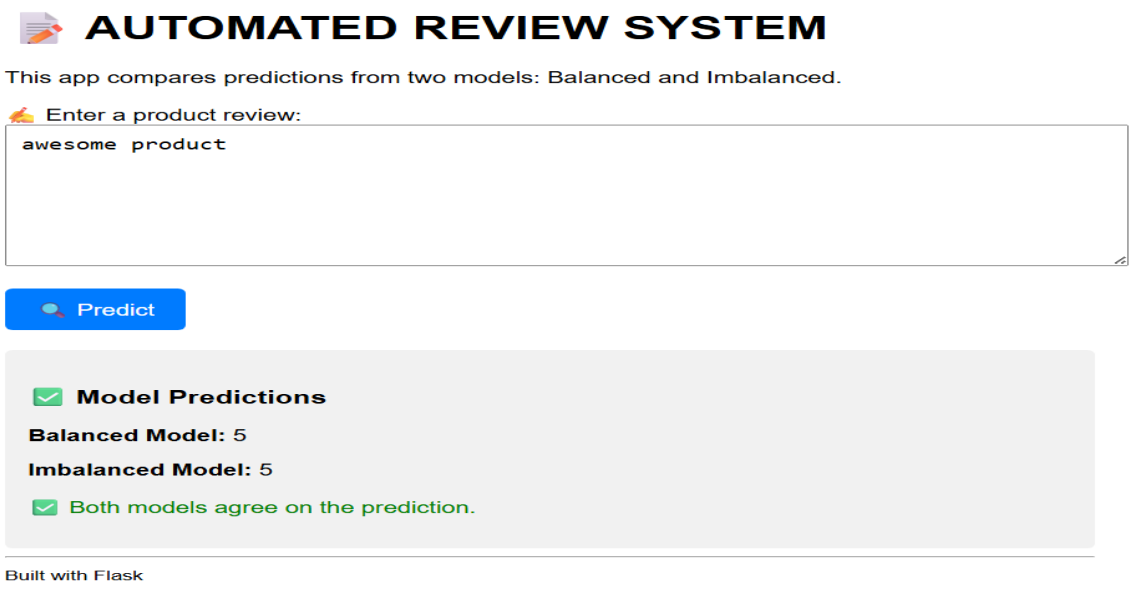
- The system displays:
    - **Balanced Model Prediction**

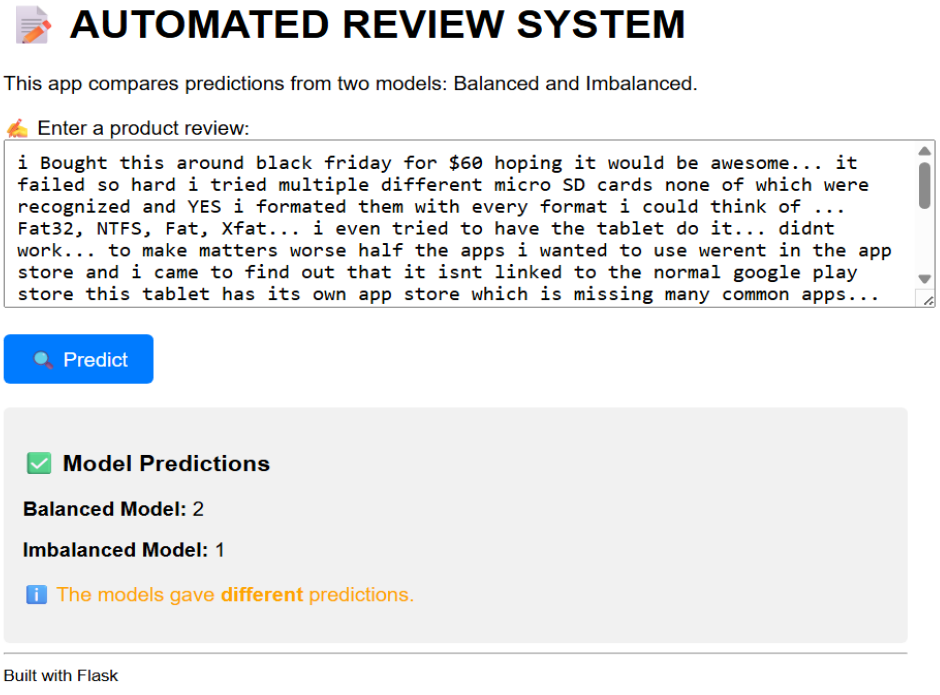
- **Imbalanced Model Prediction**
- Agreement status:
  - If predictions match → ☑ Both models agree on the prediction.
  - If predictions differ → **ℹ** The models gave different predictions.

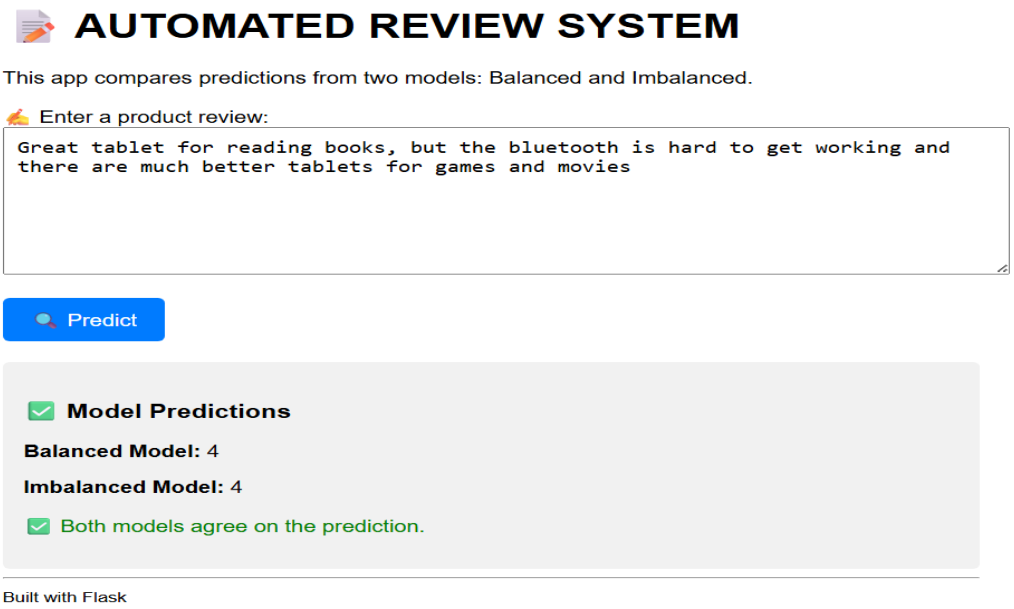
6. End

- The user can modify the input and repeat the prediction cycle.

Samples of prediction:




AUTOMATED REVIEW SYSTEM
This app compares predictions from two models: Balanced and Imbalanced.
Enter a product review:
i Bought this around black friday for $60 hoping it would be awesome... it failed so hard i tried multiple different micro SD cards none of which were recognized and YES i formated them with every format i could think of ... Fat32, NTFS, Fat, Xfat... i even tried to have the tablet do it... didnt work... to make matters worse half the apps i wanted to use werent in the app store and i came to find out that it isnt linked to the normal google play store this tablet has its own app store which is missing many common apps...
Predict
Model Predictions
Balanced Model: 2
Imbalanced Model: 1
The models gave different predictions.
Built with Flask


AUTOMATED REVIEW SYSTEM
This app compares predictions from two models: Balanced and Imbalanced.
Enter a product review:
Great tablet for reading books, but the bluetooth is hard to get working and there are much better tablets for games and movies
Predict
Model Predictions
Balanced Model: 4
Imbalanced Model: 4
Both models agree on the prediction.
Built with Flask
