In [ ]:
```
🧩 Integration Design Description
🔁 Interaction Flow:
1.User Interaction (Angular):
* A user records a voice input or submits data through a form.
* Angular sends this input to the Kotlin backend via a REST API.
```

In [ ]:
```
2.Voice Module (Python Speech Recognition):
* Kotlin receives the audio file and forwards it to a Python-based voice module.
* The Python module (using libraries like SpeechRecognition, pydub, or Google Cloud
```

In [ ]:
```
3.Machine Learning Module (Python + ML Model):
* Kotlin sends the processed text or structured data to another Python module that
* The model returns prediction results (e.g., lead_score: 0.87 or churn_risk: high)
```

In [ ]:
```
4.Perl Integration (Optional):
* For tasks like cleaning legacy CSVs, converting logs, or formatting prediction da
```

In [ ]:
```
5.Response to Frontend:
* Kotlin compiles the results and sends them back to the Angular frontend.
* Angular displays prediction outcomes and lifecycle insights.
```

In [ ]:
```
@PostMapping("/analyzeClient")
fun analyzeClient(@RequestParam file: MultipartFile): ResponseEntity<Any> {
    // 1. Send audio to voice module (Python)
    val speechText = http.post("http://localhost:5000/speech-to-text", file)

    // 2. Send transcribed text or features to ML model
    val mlPrediction = http.post("http://localhost:5001/predict", speechText)

    // 3. Optional: Clean the result with Perl script
    val cleanedOutput = runPerlScript(mlPrediction)

    // 4. Send final result to frontend
    return ResponseEntity.ok(cleanedOutput)
}
```