Task 03

DAY 01:

Action: Transferring money

Considered Intent:
-Allow money transfer
-Refuse money transfer
-Time sensitive money transfer

Frozen Structure:
1. Actor
2. Goal
3. Required Setup
4. Decision condition
5. Result if condition pass
6. Result if condition fails

Intent 01: Allow money transfer

Actor: User linked to source account
Goal: To transfer a specified amount of money from source to destination account
Required Setup: Source and destination should exist and transfer mechanism
Decision condition:
-The user has authority to initiate transfer from source to destination at time of evaluation.
-The source account balance >=the specified transfer amount.
Result if condition pass:
The specified money is deducted from source and adds to destination account
Result if condition fails: Balance remains unchanged.

Intent 02: Refuse money transfer

Actor: User linked to source account

Goal: To transfer a specified amount of money from source to destination account
Required Setup: Source and destination should exist and transfer mechanism
Decision condition:
-The user has authority to initiate transfer from source to destination at time of evaluation.
-The source account balance >=the specified transfer amount.
Result if condition pass:
The specified money is deducted from source and adds to destination account
Result if condition fails: Balance remains unchanged.

Intent 03: Time sensitive money transfer

Actor: User linked to source account
Goal: To transfer a specified amount of money from source to destination account
Required Setup: Source and destination should exist and transfer mechanism
Decision condition:
-The user has authority to initiate transfer from source to destination at time of evaluation.
-The source account balance >=the specified transfer amount.
-Transfer request is evaluated outside a defined restricted time window.
Result if condition pass:
The specified money is deducted from source and adds to destination account
Result if condition fails: Balance remains unchanged.

DAY 02:

a) Python: Clear procedural flow, explicit condition checks and one decision leads to one outcome.

Mapping:

Actor: Input to process
Goal: Purpose of process
Required setup: Precondition that must exist before execution
Decision conditions: Explicit conditional checks evaluated in order
Result if condition pass: Return successful outcome
Result if condition fail: Return refusal outcome

– Where authority is checked
  Authority is checked during the evaluation of decision conditions before the process proceeds to execution.
– Where time is evaluated
  Time is evaluated at the moment the decision conditions are checked, not at the time the request is created.
– Where refusal occurs
  Refusal occurs when the conditional checks fail, causing the process to return the refusal outcome.
– Why the outcome matches the canonical intent
  The process follows the same ordered decision conditions as the canonical intent, ensuring that only the defined pass or fail outcome is produced.

b)Javascript: Event driven

Mapping:

Actor: Source of triggering event
Goal: Intent carrired by event
Required setup: Context required before handling event
Decision conditions: checks perform within event handler
Result if condition pass: side effects triggered by handler
Result if condition fail: handler exist without producing effects

– Where authority is checked
  Authority is checked inside the event handler when the event is processed.
– Where time is evaluated

Time is evaluated when the handler runs, not when the event is triggered.

– Where refusal occurs

  Refusal occurs when the handler exits without producing side effects.

– Why the outcome matches the canonical intent

  Side effects occur only if decision conditions pass, preserving the same allow or refuse behaviour defined in the canonical intent.

c) Declarative form: describe intent, do not execute logic

Mapping:

Actor: User interaction (form, button, request)
Goal: Declared action being requested
Required setup: What must be present/enabled to request
Decision conditions: Not evaluated here
Result if condition pass: Request is sent upward
Result if condition fail: Request is not sent or ignored

– Where authority is checked

  Authority is not checked at this layer.

– Where time is evaluated

  Time is not evaluated at this layer.

– Where refusal occurs

  Refusal does not occur here; requests may simply not be forwarded.

– Why the outcome matches the canonical intent

  This layer only expresses intent. Enforcement of authority, time, and refusal occurs in the grammar layer above, preserving canonical meaning.

DAY 03:

a) Python

– Identify what was easy to express

  Sequential decision flow aligned naturally with the canonical intent. Decision conditions could be expressed explicitly and in a readable order.

Refusal fit cleanly as an alternate outcome of the same process.

– Identify what was awkward or verbose.
Maintaining clarity about when evaluation happens requires explicit wording.
– Identify any place where meaning was at risk of being lost
If decision conditions are evaluated at request time instead of execution time.

b) JS
– Identify what was easy to express
Event-triggered intent mapped well to the actor and goal.
Side effects clearly represented successful execution.
– Identify what was awkward or verbose
Separating event creation from event handling required extra care.
– Identify any place where meaning was at risk of being lost
If authority or time is checked when the event is triggered instead of when it is handled.

c)Declarative
– Identify what was easy to express
User intent and requested action were clearly representable.
Actor and goal mapped cleanly to user interaction elements.
– Identify what was awkward or verbose
Decision conditions could not be expressed directly.
– Identify any place where meaning was at risk of being lost
If visibility or enablement is mistaken for permission.

– Which parts of your intent grammar resist translation?

The parts of the grammar that resist translation are those that depend on precise evaluation boundaries, especially authority and time. While all representations can reference these concepts, not all of them naturally enforce when and where they must be evaluated. Declarative representations in particular struggle to express refusal and decision conditions directly, as they are limited to describing intent rather than enforcing outcomes.

– Which parts translate cleanly across all languages?

The actor, goal, and explicit outcomes translate cleanly across all representations. These elements represent intent and result, which are conceptually universal and do not depend on execution style. The separation between successful and refused outcomes also translates consistently, as every representation can express the presence or absence of effects.

– What assumptions do programming languages force that your grammar avoids?

Evaluation happens immediately after intent is expressed.
Authority can be inferred from context or identity.
Refusal is an exceptional case rather than a valid outcome.