Wason Selection Task Experiment: Technical Documentation

Cognitive Systems Project

January 8, 2026

Contents

1 Experiment Overview

The experiment implements a single-trial, browser-based version of the Wason Selection Task to study the effect of contextual framing on conditional reasoning.

Participants evaluate a conditional rule of the form *If P, then Q* by selecting which of four cards must be turned over to test the rule.

The logical structure is held constant across conditions; only the semantic context varies.

Three conditions are used:

- abstract (numbers and colors)
- familiar real-world (age and alcohol)
- unfamiliar structured (rule-based access scenario)

Participants are randomly assigned to exactly one condition.

2 Experimental Design

2.1 Design

Between-subjects, single-factor design with one trial per participant.

2.2 Independent Variable

Context type with three levels:

- abstract
- familiar
- unfamiliar structured

Assignment is uniform random.

2.3 Dependent Variables

Recorded per participant:

- **Correctness**: exact match between final selection and condition-specific correct set
- **Time to submission**: task submit time minus task start time (ms)
- **Selection changes**: number of card toggle actions
- **First card selected**: identifier of first toggled card
- **Confidence**: self-report, 0–100

2.4 Participant Flow

Introduction → task → confidence rating → debrief. Backward navigation is not permitted.

3 Front-End Architecture

3.1 Stack

Client-side application using:

- Vite
- React
- TypeScript
- Tailwind CSS

The application is deployed as a static website.

3.2 State Model

A single global session state controls the application. A discrete screen variable determines the rendered view:

- `start`
- `task`
- `grade`
- `end`

Transitions are strictly forward-only.

3.3 Component Structure

- `App.tsx`: global state and transitions.
- `screens/`: UI per experimental stage.
- `tasks.ts`: condition definitions and correct sets.
- `types.ts`: shared type definitions.
- `api.ts`, `supabase.ts`: authentication and persistence.

Screen components are stateless with respect to experimental logic.

4 Task Logic and Measurement

4.1 Card Model

Each task consists of four cards identified as `c1--c4`. Identifiers are constant; labels vary by condition.

4.2 Selection Logic

Card interaction is implemented as a toggle:

- selecting adds the card to the selection set
- selecting again removes it

All toggles are centrally handled.

4.3 First Card Selected

The first toggle event sets `first_card_selected`. This value is immutable after initialization.

4.4 Selection Changes

Each toggle increments `selection_changes`. This counts exploratory behavior independently of correctness.

4.5 Final Selection

The final selection is the unordered set of cards selected at submission.

4.6 Correctness

A response is correct iff:

- all required cards are selected
- no additional cards are selected

Partial or superset selections are scored as incorrect.

4.7 Timing

Task duration is measured using `performance.now()`:

$$\text{time} = \text{task\_submit\_ms} - \text{task\_start\_ms}$$

4.8 Confidence

Confidence is collected post-task on a 0–100 scale.

5 Back-End and Data Storage

5.1 Infrastructure

Data are stored using Supabase (PostgreSQL). No custom server is used.

5.2 Authentication

Participants are authenticated via anonymous Supabase auth. The resulting UUID serves as the participant identifier.

No personally identifying data are collected.

5.3 Security Model

Row Level Security enforces:

- insert-only access
- authenticated users only
- user may insert only their own rows
- no client-side reads, updates, or deletes

The public API key does not grant privileged access.

5.4 Write Procedure

Data are written exactly once, after confidence submission. A saving state prevents duplicate inserts. Errors are surfaced to the participant and allow retry.

6 Data Schema

Each row represents one completed session.

6.1 Core Fields

- `session_id`: client-generated primary key
- `experiment_id`: experiment/version label
- `user_id`: anonymous Supabase UUID
- `condition`: assigned context
- `task_start_ms`, `task_submit_ms`
- `selection_changes`

- `first_card_selected`
- `final_selection`
- `correct`
- `confidence`
- `created_at`

6.2 Derived Measures

- time to submission
- exploratory intensity
- initial reasoning strategy

6.3 Data Handling

The database is immutable from the client. All exclusions or corrections are performed offline.

7 Conclusion

This document specifies the complete experimental logic, measurement definitions, and data guarantees of the implemented Wason Selection Task system.