Error & Constraint Handling Specification

1. Purpose

This document defines how the timetable scheduler must detect, classify, record, and report constraint-related issues during schedule generation.

It strictly governs error handling behavior and diagnostic output, even when scheduling is successful.

A scheduling run is considered incomplete if constraint violations are not properly documented.

2. Error Handling Philosophy

The scheduler MUST follow these principles:

1. **No silent degradation**
   Any deviation from expected constraints must be explicitly reported.

2. **Separation of validity and quality**

   - Hard constraints define *validity*
   - Soft constraints define *quality*

3. **Explain, don't just detect**
   Every issue must include:

   - What failed
   - Why it failed
   - What can be done about it

3. Error Classification Model

All errors are classified using two orthogonal dimensions:

3.1 Constraint Type

Type	Meaning
HARD	Must Never Be Violated
SOFT	May be violated if unavoidable

3.2 Severity

Type	Meaning
ERROR	Invalid or blocked scheduling
WARNING	Valid schedule with compromises

4. Hard Constraint Error Handling

4.1 Definition

A hard constraint error occurs when the scheduler cannot find a valid allocation that satisfies a mandatory rule.

Hard constraint errors directly affect schedule validity.

4.2 Required Behavior

When a hard constraint conflict is encountered, the scheduler MUST:

1. Attempt all viable alternative allocations
2. Backtrack where supported
3. If unresolved:
    - Record a **hard constraint diagnostic**
      Mark the issue with `severity = ERROR`
    - Clearly identify the blocking entities
4. Decide whether scheduling can continue in unaffected scopes

The scheduler MUST NOT:

- Ignore the conflict
- Downgrade it to a warning
- Return a “successful” status without recording it

4.3 Diagnostic Requirements (Hard Constraints)

Each hard constraint error MUST include:

- `constraint_type = HARD`
- `severity = ERROR`
- A stable, unique `code`
- Clear explanation of the conflict
- Precise affected entities
- One or more root causes
- A human-actionable resolution hint

5. Soft Constraint Error Handling

5.1 Definition

A soft constraint error occurs when the scheduler **cannot meet an optimization target** while still respecting all hard constraints.

Soft constraint errors affect **schedule quality**, not validity.

5.2 Required Behavior

When a soft constraint cannot be satisfied, the scheduler MUST:

1. Accept the best valid allocation
2. Continue scheduling remaining entities
3. Record a **soft constraint diagnostic**
4. Ensure the final timetable is still returned

The scheduler MUST NOT:

- Retry indefinitely
- Fail the entire schedule
- Suppress the warning

5.3 Diagnostic Requirements (Soft Constraints)

Each soft constraint violation MUST include:

- `constraint_type = SOFT`
- `severity = WARNING`
- A stable `code`
- Expected vs actual outcome
- Affected entities
- One or more root causes
- A mitigation or configuration hint

6. Diagnostic Recording Rules

- One diagnostic entry MUST be created per distinct violation
- Multiple violations of the same rule on different entities MUST be recorded separately
- Diagnostics MUST be appended in the order they are encountered
- Diagnostics MUST be included even if scheduling completes successfully

7. Scheduler Status Determination

situation	status
No violations	SUCCESS
Only soft violations	PARTIAL_SUCCESS
One or more hard violations	`FAILED` or `PARTIAL_SUCCESS` (if partial output exists)

8. Worked Example (Expected Behavior)

Scenario

Soft Constraint

- Minimum courses per day = 4

Observed Scheduling Result

- Monday → 4 courses

- Tuesday → 4 courses
- Wednesday → 2 courses
- Thursday → 4 courses
- Friday → 4 courses

Scheduler Behavior (Correct)

1. Scheduler attempts Wednesday allocations
2. All hard constraints are satisfied
3. Only 2 feasible courses exist
4. Scheduler accepts the best valid outcome
5. Scheduler records a soft constraint diagnostic
6. Scheduler continues scheduling Thursday and Friday
7. Scheduler returns timetable + diagnostics

Expected Diagnostic Output (Excerpt)


```
{
  "constraint_type": "SOFT",
  "severity": "WARNING",
  "code": "MIN_COURSES_PER_DAY_NOT_MET",
  "description": "Only 2 courses were scheduled on Wednesday instead of the required minimum of 4.",
  "affected_entities": [
    {
      "entity_type": "DAY",
      "day": "Wednesday",
      "required": 4,
      "actual": 2
    }
  ],
```

```
  "root_causes": [
    {
      "cause": "Teacher unavailability",
      "details": "Most qualified teachers were unavailable on Wednesday."
    }
  ]
}
```

9. What This Guarantees

This error-handling model guarantees that:

- A working scheduler is not penalized for unavoidable compromises
- Administrators understand *why* compromises happened
- Developers can debug and improve constraint configuration

10. Design Principle

> A schedule without explanations is an incomplete schedule.

2. Hard Constraint Definitions & Handling

HC-01: Teacher Preferred Teaching Period Enforcement

Definition

A teacher MUST ONLY be scheduled within their declared preferred teaching periods.

Derived From
teacher_prefered_teaching_period

Rule

- The scheduler MUST NOT assign a teacher to any time slot outside their preferred periods.
- Preferred periods are **restrictive**, not optional.

Violation Condition

- A class is scheduled for a teacher outside the specified day or time range.

Expected Error Handling Behavior

1. Scheduler attempts allocation
2. Detects teacher assignment outside preferred period
3. Attempts alternative valid slots
4. If no valid slot exists:
    - Scheduler MUST block the allocation
    - Record a HARD constraint diagnostic
    - Mark severity as `ERROR`


```
{
  "constraint_type": "HARD",
  "severity": "ERROR",
  "code": "TEACHER_OUTSIDE_PREFERRED_PERIOD",
  "title": "Teacher scheduled outside preferred teaching period",
  "description": "Stanley Dach was scheduled on Thursday outside his preferred teaching period of 12:00–13:00.",
  "affected_entities": [
    {
      "entity_type": "TEACHER",
      "teacher_id": "904deb4a-2fdd-494f-93bc-e7df4744c2e8",
      "teacher_name": "Stanley Dach"
    },
    {
      "entity_type": "TIME_SLOT",
```

```
    "day": "thursday",
    "start_time": "10:00",
    "end_time": "11:00"
   }
  ],
  "root_causes": [
   {
    "cause": "Restricted availability",
    "details": "No other available time slots align with the teacher’s preferred period."
   }
  ],
  "resolution_hint": "Extend the teacher’s preferred teaching period or adjust course scheduling requirements."
}
```


HC-02: Hall Busy Period Enforcement

Definition

Classes MUST NEVER be scheduled in a hall during its busy period.

Derived From
hall_busy_periods

Rule

- A hall is considered unavailable during its busy period.
- The scheduler MUST treat busy halls as fully blocked resources.

Expected Error Handling Behavior

1. Scheduler selects hall
2. Detects overlap with hall busy period
3. Attempts alternate halls or times
4. If none exist:
    - HARD constraint diagnostic is generated
    - Scheduling is blocked for that allocation


```
{
  "constraint_type": "HARD",
  "severity": "ERROR",
  "code": "HALL_BUSY_PERIOD_CONFLICT",
  "title": "Hall unavailable during scheduled period",
  "description": "Main Lecture Theatre A was scheduled on Tuesday between 14:00–16:00, which overlaps with its busy period.",
  "affected_entities": [
    {
      "entity_type": "HALL",
      "hall_id": "c1d2e3f4-a5b6-7c8d-9e0f-1a2b3c4d5e6f",
      "hall_name": "Main Lecture Theatre A"
    },
    {
      "entity_type": "TIME_SLOT",
      "day": "tuesday",
      "start_time": "15:00",
      "end_time": "16:00"
    }
  ],
  "root_causes": [
    {
      "cause": "Hall reservation conflict",
      "details": "The hall is reserved during this time window."
    }
  ],
  "resolution_hint": "Select an alternative hall or reschedule the class outside the busy period."
}
```


HC-03: Teacher–Course Ownership Enforcement

Definition

Courses MUST ONLY be taught by the teachers explicitly assigned to them.

Derived From

teacher_courses

Rule

- Course → Teacher mapping is immutable
- The scheduler MUST NOT substitute teachers

Expected Error Handling Behavior

1. Scheduler attempts to assign a course
2. Assigned teacher does not match course ownership
3. Scheduler MUST reject the allocation immediately
4. HARD diagnostic is generated


```
{
  "constraint_type": "HARD",
  "severity": "ERROR",
  "code": "COURSE_TEACHER_MISMATCH",
  "title": "Course assigned to unauthorized teacher",
  "description": "The course 'Adipisci Qui Quas Minima' was assigned to a teacher not authorized to teach it.",
  "affected_entities": [
    {
      "entity_type": "COURSE",
      "course_id": "aa4f8ba8-514f-4811-9b99-5199ec785a59",
      "course_title": "Adipisci Qui Quas Minima"
    }
  ],
  "root_causes": [
    {
      "cause": "Strict course ownership",
      "details": "The course is assigned exclusively to Stanley Dach."
```

```
    }

  ],

  "resolution_hint": "Assign the course to its designated teacher or update course-teacher mappings."

}
```


HC-04: Teacher Availability (No Conflict Rule)

Definition

A teacher MUST NEVER be scheduled during their busy periods.

Derived From

teacher_busy_period

Rule

- Busy periods represent absolute unavailability
- Overlaps are not allowed under any circumstance

Expected Error Handling Behavior

1. Scheduler detects overlap with busy period
2. Attempts alternative slots
3. If unresolved:
    - HARD constraint diagnostic
    - Allocation blocked

Expected Diagnostic Example


```
{
  "constraint_type": "HARD",
  "severity": "ERROR",
  "code": "TEACHER_BUSY_PERIOD_CONFLICT",
  "title": "Teacher scheduled during busy period",
  "description": "Dr. Chen was scheduled on Tuesday between 16:00–17:00, which overlaps with a declared busy period.",
  "affected_entities": [
    {
      "entity_type": "TEACHER",
      "teacher_id": "99z8y7x6-w5v4-u3t2-s1r0-q9p8o7n6m5l4",
```

```
      "teacher_name": "Dr Chen"
    }
  ],
  "root_causes": [
    {
      "cause": "Teacher unavailability",
      "details": "The teacher is explicitly unavailable during this time window."
    }
  ],
  "resolution_hint": "Reschedule the class or update the teacher's busy periods."
}
```


3. Hard Constraint Guarantee

The scheduler guarantees that:

- No timetable is returned with unresolved hard constraint violations
- All blocked allocations are **explicitly explained**
- Every hard constraint failure is **traceable to input data**
- Error handling remains consistent regardless of scheduling success

4. Design Principle (Hard Constraints)

> **Hard constraints define correctness, not preference.**
> Violating them makes a schedule invalid—no exceptions.