Unit Testing (Backend)

Plans

Objective: Test individual components and functions of the timetable system to ensure they work as expected.

Scope:

- Test individual modules such as the scheduling algorithm, constraint validation functions, and API integrations.
- Verify the correctness of each function by providing known inputs and checking outputs against expected results.

Test Cases:

- **Constraint 1 (Student Constraint):** Verify that the function preventing overlapping classes for a student works correctly.
- **Constraint 2 (Classroom Constraint):** Ensure that the algorithm prevents double-booking of classrooms.
- **Constraint 3 (Room Capacity Constraint):** Verify that the room assignment function correctly matches class sizes with room capacities.
- **Constraint 4 (Lab Constraint):** A unit with a lab component will be prioritized to be assigned to a lab room.

Tools:

- Use testing frameworks like JUnit (for Java).
- Use timefold's ConstraintVerifier.
- Use quarkus's QuarkusTest.

Acceptance Criteria: Refer to Jira.

Progress + Results

Link: jetedge/backend/src/test/java/org/acme/solver/TimetableConstraintProviderTest.java at 34bd8d6f0a06def3aee6ca83ee4cdd62eaeb5c56 · hotungkhanh/jetedge

There are 4 unit tests, one for each constraint.

- **studentConflictTest**:
  - Testing for the student constraint.
  - Penalize 1 if a student is attending two units at the same time.
- **roomConflict**:
  - Testing for the room constraint.
  - Penalize 1 if a room is occupied by two units at the same time.
- **roomCapacityConclict**:
  - Testing for the room capacity constraint.
  - Penalize 1 if a room is occupied by more students than its capacity.
- **labConflict**:
  - Testing for the unit with lab components is prioritized with a lab room.
  - Penalize 1 if a non-lab room is occupied by a lab unit.