API Documentation

Student API

POST /register

- **request.body**: `{ email, password, roll_number, student_name }`
- **Description**: Registers a new user with the provided details (email, password, roll number, and student name). The password is encrypted before being saved to the database.

POST /login

- **request.body**: `{ email, password }`
- **Description**: Logs the user in with the provided email and password. The password is encrypted and checked against the database. If valid, a JWT token is generated and returned.

GET /show-student-mark/:id

- **request.body**: `{}`
- **request.params**: `{ id }`
- **Description**: Fetches and returns the marks for a specific student, identified by the student ID in the request parameters.

Staff API

POST /register

- **request.body**: `{ email, password, employee_number, staff_name }`

- **Description**: Registers a new staff member with the provided details (email, password, employee number, and staff name). The password is encrypted before being saved to the database.

POST /login

- **request.body**: `{ email, password }`
- **Description**: Logs the staff member in with the provided email and password. The password is encrypted and checked against the database. If valid, a JWT token is generated and returned.

POST /mark-entry

- **request.body**: `{ student_name, subject_name, group_name, staff_name, component, marks, entry_date }`

- **Description**: Allows the staff to enter marks for a student. The student's, subject's, group's, and staff's existence is validated before the marks are inserted into the database.

GET /show-staff-completed-tasks/:staff_id

- **request.params**: { staff_id }
- **Description**: show the list of tasks completed by the staff

---

PUT /update-marks/:task_id

- **request.params**: { task_id }
- **Description**: staff can update the marks (theory, lab, ia1, ia2) of a task based on the task_id

---

Coordinator API

POST /subject

- **request.body**: `{ subject_name, course_id }`
- **Description**: Adds a new subject to a course using the provided `subject_name` and `course_id`.

POST /add-schema

- **request.body**: `{ theory_weightage, lab_weightage, ia1_weightage, ia2_weightage, subject_id }`
- **Description**: Adds an evaluation scheme (weightages for theory, lab, IA1, IA2) for a subject using the provided `subject_id`.

POST /add-marking-schema

- **request.body**: `{ theory_weightage, lab_weightage, ia1_weightage, ia2_weightage, subject_name, course_name }`
- **Description**: Adds a marking schema by first retrieving the `course_id` and `subject_id` from their respective tables, then inserting the evaluation scheme for the subject.

GET /show-schema

- **request.body**: `{}`
- **Description**: Fetches the evaluation schemes for a specific course. The course name is passed as a query parameter (`course_name`).

POST /add-group

- **request.body**: `{ course_id, group_name }`
- **Description**: Adds a new group to a course with the provided `course_id` and `group_name`.

GET /show-groups

- **request.body**: `{}`
- **Description**: Fetches all groups along with their course names.

POST /add-student-group

- **request.body**: `{ email, group_name }`
- **Description**: Adds a student to a group. The student is identified by their `email`, and the group is identified by the `group_name`.

GET /students-with-groups

- **request.body**: `{}`
- **Description**: Fetches a list of all students along with their assigned groups.

POST /mark-entry

- **request.body**: `{ student_name, subject_name, group_name, staff_name, component, marks, entry_date }`
- **Description**: Marks the entry of a student for a subject and group. The student, subject, group, and staff are validated before the marks entry is created.

GET /show-marks

- **request.body**: `{}`
- **Description**: Fetches all student marks, including roll number, student name, group name, subject name, course name, and marks.

POST /course-groups

- **request.body**: `{ course_name }`
- **Description**: Fetches all course groups, present in a course

---

POST /course-subjects

- **request.body**: `{ course_name }`
- **Description**: Fetches all subjects, present in a course

POST /add-student-to-course

- **request.body**: `{ email, course_name }`
- **Description**: takes student email and course name and adds them to the course

---

GET /show-students

- **request.body**: `{ }`
- **Description**: show all students that are NOT present in any course or group

---

GET /student-with-course

- **request.body**: `{ }`
- **Description**: show all students that are present in a course or but NOT in any group

---

GET /student-with-course-groups

- **request.body**: `{ }`
- **Description**: show all students that are present in a course and a group

---

POST /show-staff

- **request.body**: `{ course_name }`
- **Description**: show all staffs that are present in a course

---

POST /assign-task

- **request.body**: `{ course_name, subject_name, group_name, staff_name, from_date, till_date, }`

- **Description**: assign a task to staff for a given course for a particular subject and group

---

GET /show-unapproved-task/:course_name

- **request.body**: `{ }`
- **request.params**: `{ course_name }`
- **description**: show all assigned tasks (unapproved tasks) that have been assigned to all the staff in a course

---

PUT /update-approval/:task_id

- **request.body**: `{ approved }`
- **request.params**: `{ task_id }`
- **description**: update the status of the given task based on task_id

---

GET /show-approved-tasks/:course_name

- **request.body**: `{ }`
- **request.params**: `{ course_name }`

- **description**: show all assigned tasks with status approved that have been assigned to all the staff in a course

---

Admin APIs

GET /show-all-staffs

- **request.body**: `{ }`
- **description**: shows all the staffs that are present in a database (might be present in a course)

PUT /update-staff-details

- **request.body**: `{ course_name, role, email }`
- **description**: update the staff details (like updating the role (coordinator or staff) of staff or assigning them to a course)

POST /add-course

- **request.body**: `{ course_name, description }`
- **description**: adds a new course to the database

GET /show-courses

- **description**: show all courses present in the database

POST /add-subject

- **request.body**: `{ subject_name, course_name }`
- **description**: adds a new subject to a course

GET /show-subjects/:course_name

- **request.params**: `{ course_name }`
- **description**: show all subjects present in a course

POST /add-group

- **request.body**: `{ group_name, course_name }`
- **description**: adds a new group to a course

GET /show-groups/:course_name

- **request.params**: `{ course_name }`

- **description**: show all groups present in a course

GET /show-all-students

- **description**: show all students that are NOT present in any course

---

GET /add-student-to-course

- **request.body**: `{ email, course_name }`
- **description**: add a student to the specified course based on the email of the student

---

GET /students-with-course

- **description**: show all students that are present in the specified course

---