**User Story 14: Pickup Scheduling**

**As a scheduler, I want to configure recurring or ad-hoc waste pickup schedules for each zone, so that I can ensure timely waste collection.**

**Acceptance Criteria:**

* **GIVEN** a scheduler accesses the pickup scheduling API endpoint **WHEN** they provide valid scheduling details **THEN** a new pickup schedule is created, and a success message is returned.
* **GIVEN** a scheduler accesses the pickup scheduling API endpoint **WHEN** they provide invalid scheduling details **THEN** an error message is returned indicating the issue.

**Backend Requirements:**

* **API Endpoint:**
* **URL:** /wastewise/scheduler
* **Method:** POST
* **Request Body:** {

  "zoneId": "zone123",

  "frequency": "Weekly",

  "timeSlot": "08:00-10:00",

  "vehicleId": "vehicle456",

“workerId” : “001”,

“pickup\_point”: “CTS Office SIPCOT”

}

* **Response:** A JSON object with a success message and the schedule ID.
* **Success Response Status Code:** 201 Created
* **Success Response Body:** {

  "message": "new pickup schedule created with ID and pickup location",

  "scheduleId": "schedule789",

“Pickup\_point”: “CTS office SIPCOT”

}

* **Failure Response Status Code:** 409 Conflict
* **Failure Response Body**: {

  "message": "pickup schedule already exists for the given zone and time slot",

  "zoneId": "zone123",

  "timeSlot": "08:00-10:00"

}

* **Controller Layer:**
* **Create an Endpoint:** Implement PickupScheduleController to handle pickup scheduling requests.
* **Validate Input Data:** Validate the incoming scheduling details.
* **Generate Response:** Provide a success message or an error message if the details are invalid.
* **Service Layer:**
* **Business Logic:** Implement PickupScheduleService to validate and process pickup scheduling requests.
* **Save Schedule Data:** Save the new schedule data to the database.
* **Validate Input Data:** Ensure proper validation in the Service Layer, including:
* Zone ID Validation: Correct format and existence.
* Frequency Validation: Correct format and not empty.
* Time Slot Validation: Correct format and not empty.
* Vehicle ID Validation: Correct format and existence.
* **Add Logger:** Implement logging to capture significant pickup scheduling events.
* **Repository Layer:**
* **Database Interaction:** PickupScheduleRepository to save schedule information.
* **Entities and DTOs:**
* **Define PickupSchedule Entity:** Create a PickupSchedule entity with fields for id, zone\_id, frequency, time\_slot, and vehicle\_id.
* **PickupSchedule DTO:** Create a PickupScheduleDTO to map the request body.
* **Exception Handling:**

Should handle invalid requests.

* **Testing:**
* **Unit Tests:**
* Test Service Layer Methods: Validate and save pickup scheduling details.
* Exception Handling: Ensure proper handling of invalid scheduling details.

A screenshot of a computer

AI-generated content may be incorrect., Picture

**User Story 15: Pickup Schedule Delete**

**As a scheduler, I want to delete recurring or ad-hoc waste pickup schedules for each zone, so that I can ensure timely waste collection.**

**Acceptance Criteria:**

* **GIVEN** a scheduler accesses the pickup scheduling API endpoint **WHEN** they provide valid scheduling details **THEN** a pickup schedule is DELETED, and a success message is returned.
* **GIVEN** a scheduler accesses the pickup scheduling API endpoint **WHEN** they provide invalid scheduling details **THEN** an error message is returned indicating the issue.

**Backend Requirements:**

* **API Endpoint:**
* **URL:** /wastewise/scheduler
* **Method:** DELETE
* **Request Body:** {

  "pickup\_id”: “001”

}

* **Response:** A JSON object with a success message and the schedule ID.
* **Success Response Status Code:** 200 OK
* **Success Response Body:** {

  "message": " pickup schedule deleted with ID and pickup location",

  "scheduleId": "schedule789",

“Pickup\_point”: “CTS office SIPCOT”

}

* **Failure Response Status Code:** 409 Conflict
* **Failure Response Body**: {

  "message": "pickup schedule does not exists for the given id",

  "pickup\_id": "123"

}

* **Controller Layer:**
* **Create an Endpoint:** Implement PickupScheduleController to handle pickup scheduling requests.
* **Validate Input Data:** Validate the incoming scheduling details.
* **Generate Response:** Provide a success message or an error message if the details are invalid.
* **Service Layer:**
* **Business Logic:** Implement PickupScheduleService to validate and process pickup scheduling requests.
* **Save Schedule Data:** Save the new schedule data to the database.
* **Validate Input Data:** Ensure proper validation in the Service Layer, including:
* **Add Logger:** Implement logging to capture significant pickup scheduling events.
* **Repository Layer:**
* **Database Interaction:** PickupScheduleRepository to save schedule information.
* **Entities and DTOs:**
* **Define PickupSchedule Entity:** Create a PickupSchedule entity with fields for id, zone\_id, frequency, time\_slot, and vehicle\_id.
* **PickupSchedule DTO:** Create a PickupScheduleDTO to map the request body.
* **Exception Handling:**

Should handle invalid requests.

* **Testing:**
* **Unit Tests:**
* Test Service Layer Methods: Validate and save pickup scheduling details.
* Exception Handling: Ensure proper handling of invalid scheduling details.

A screenshot of a computer

AI-generated content may be incorrect., Picture