INTRODUCTION:

Ajira has developed a rover for extra-terrestrial exploration.
It can be configured to act differently in different environments.
Before sending it on a voyage, we need to simulate an extra-terrestrial environment and test the rover The rover consists of the following internal modules:

Rover Modules

Inventory

This is a storage unit which lets Ajirayaan store important samples and items it has retrieved during exploration.

For example,
Water Samples - 2 units - Priority 2
Rock Samples - 3 units - Priority 3
Storm Shield - 2 units - Priority 1

The inventory has a finite space.
We can add and remove items from the inventory. If the inventory is full and we need to add an item, we must remove items of a lower priority as required.

Battery

The rover has limited battery charge and initially has a battery level of 11.
For every step the rover takes, the battery level decreases by 1.
Every 10 steps, the rover's battery recharges to a level of 10 (it uses kinetic energy or movement to recharge itself).
If the battery level reaches 0, the rover dies.

Extra-terrestrial Environment

The rover has been designed to be configurable to adapt to any kind of environment.
For the purposes of the simulation, we can consider the following environmental factors:

- Temperature
- Humidity
- Solar Flare. A solar flare recharges the rover's battery fully
- Storm. If a storm occurs and the rover isn't shielded, it will be destroyed
- Terrain. There are different kinds of terrain such as: "dirt", "water", "rock", "sand"

Simulation

Initial configuration of the environment is as follows:

```
POST /api/environment/configure
Content-Type: application/json

{
    "temperature": 60,
    "humidity": 65,
    "solar-flare": false,
    "storm": false,
    "area-map": [
        [  "dirt",  "dirt",  "dirt", "water",  "dirt" ],
        [  "dirt",  "dirt", "water", "water", "water" ],
        [  "dirt",  "dirt",  "dirt", "water",  "dirt" ],
        [  "dirt",  "dirt",  "dirt",  "dirt",  "dirt" ],
        [  "dirt",  "dirt",  "dirt",  "dirt",  "dirt" ]
    ]
}
```

Subsequent modifications to the environment should only be allowed through:

```
PATCH /api/environment
Content-Type: application/json
Accept: application/json

{
    "temperature": 20
}
```

Initial configuration of the rover is done as follows:

```
 POST /api/rover/configure
Content-Type: application/json

{
    "scenarios": [
        {
            "name": "battery-low",
            "conditions": [
                {
                    "type": "rover",
                    "property": "battery",
                    "operator": "lte",
                    "value": 2
                }
            ],
            "rover": [
                { "is": "immobile" }
            ]
        },
        {
            "name": "encountering-water",
            "conditions": [
                {
                    "type": "environment",
                    "property": "terrain",
                    "operator": "eq",
                    "value": "water"
                }
            ],
            "rover": [
                {
                    "performs": {
                        "collect-sample": {
                            "type": "water-sample",
                            "qty": 2
                        }
                    }
                }
            ]
        },
        {
            "name": "encountering-storm",
            "conditions": [
                {
                    "type": "environment",
                    "property": "storm",
                    "operator": "eq",
                    "value": true
                }
            ],
            "rover": [
                {
                    "performs": {
                        "item-usage": {
                            "type": "storm-shield",
                            "qty": 1
                        }
                    }
                }
            ]
        }
    ],
    "states": [
        {
```

```
        {
            "name": "normal",
            "allowedActions": [ "move", "collect-sample" ]
        },
        {
            "name": "immobile",
            "allowedActions": [ "collect-sample" ]
        }
    ],

    "deploy-point": {
        "row": 3,
        "column": 1
    },

    "initial-battery": 11,

    "inventory": [
        {
            "type": "storm-shield",
            "quantity": 1,
            "priority": 1
        }
    ]
}
```

Rover movement is simulated using

```
POST /api/rover/move
Content-Type: application/json
Accept: application/json

{
    "direction": "up"
}
```

Rover status is obtained using

```
GET /api/rover/status
Accept: application/json
```

Spec

`POST /api/environment/configure`
`PATCH /api/environment`

Request Interface

```
type TerrainType = "dirt" | "water" | "rock" | "sand"
type Map2D = Array<Array<TerrainType>>

{
    "temperature": number,
    "humidity": number,
    "solar-flare": boolean,
    "storm": boolean,
    "area-map": Map2D
}
```

Response Interface

If successful,

```
200 OK
```

Use standard error codes as necessary

`POST /api/rover/move`

Request Interface

```
type Direction = "up" | "down" | "left" | "right"

{
    "direction": Direction
}
```

Response Interface

If successful,

```
200 OK
```

If at edge of map and request is to move outside mapped area,

```
428 Precondition Required
Content-Type: application/json

{
    "message": "Can move only within mapped area"
}
```

If during a storm,

```
428 Precondition Required
Content-Type: application/json

{
    "message": "Cannot move during a storm"
}
```

If the rover's battery dies or the rover is destroyed, do not return a response

Use standard error codes as necessary

`GET /api/rover/status`

Response Interface

If successful,

```
200 OK
Content-Type: application/json

{
    "rover": {
        "location": {
            "row": 0,
            "column": 0
        },
        "battery": 10,
        "inventory": [ ]
    },
    "environment": {
        "temperature": 60,
        "humidity": 65,
        "solar-flare": false,
        "storm": false,
        "terrain": "dirt"
    }
}
```

```
type SampleType = "water-sample" | "rock-sample"
type InventoryItemType = "storm-shield" | SampleType
type InventoryItem = {
    "type": InventoryItemType,
    "qty": number,
    "priority": number
}
type TerrainType = "dirt" | "water" | "rock" | "sand"

{
    "rover": {
        "location": {
            "row": number,
            "column": number
        },
        "battery": number,
        "inventory": InventoryItem[]
    },
    "environment": {
        "temperature": number,
        "humidity": number,
        "solar-flare": boolean,
        "storm": boolean,
        "terrain": TerrainType
    }
}
```

If the rover's battery dies or the rover is destroyed, do not return a response

Use standard error codes as necessary

`POST /api/rover/configure`

Request Interface

```
type RoverProperty = "battery"
type EnvironmentProperty = "terrain" | "temperature" | "humidity" | "solar-flare" | "storm"
type Operator = "eq" | "ne" | "lte" | "gte" | "lt" | "gt"

type ScenarioCondition = {
    "type": "rover" | "environment"
    "property": RoverProperty | EnvironmentProperty
    "operator": Operator
    "value": number | string | boolean
}
type Scenario = {
    "name": string,
    "conditions": ScenarioCondition[]
    "rover": {
        "is":  string | undefined
        "performs": {
            "collect-sample": undefined | {
                "type": SampleType
                "qty": number
            }
            "item-usage": undefined | {
                "type": InventoryItemType
                "qty": number
            }
        }
    }
}

type Action = "move" | "collect-sample"
type State = {
    "name": string
    "allowed-actions": Action[]
}

{
    "scenarios": Scenario[],
    "states": State[]

    "deploy-point": {
        "row": number,
        "column": number
    },

    "initial-battery": number,

    "inventory": InventoryItem[]
}
```

Test Case

Configure environment and rover using the same values given in the explanation. Then do the following

```
POST /api/rover/move
Content-Type: application/json
Accept: application/json

{
    "direction": "right"
}
```

```
POST /api/rover/move
Content-Type: application/json
Accept: application/json

{
    "direction": "right"
}
```

```
GET /api/rover/status
Accept: application/json
```

should give

```
200 OK
Content-Type: application/json

{
    "rover": {
        "location": {
            "row": 3,
            "column": 3
        },
        "battery": 9,
        "inventory": [
            {
                "type": "storm-shield",
                "quantity": 1,
                "priority": 1
            }
        ]
    },
    "environment": {
        "temperature": 60,
        "humidity": 65,
        "solar-flare": false,
        "storm": false,
        "terrain": "dirt"
    }
}
```

Next

```
PATCH /api/environment
Content-Type: application/json
Accept: application/json

{
    "storm": true
}
```

```
GET /api/rover/status
Accept: application/json
```

should give

```
200 OK
Content-Type: application/json

{
    "rover": {
        "location": {
            "row": 3,
            "column": 3
        },
        "battery": 9,
        "inventory": [ ]
    },
    "environment": {
        "temperature": 60,
        "humidity": 65,
        "solar-flare": false,
        "storm": true,
        "terrain": "dirt"
    }
}
```

Next

```
POST /api/rover/move
Content-Type: application/json
Accept: application/json

{
    "direction": "up"
}
```

should give

```
428 Precondition Required
Content-Type: application/json

{
    "message": "Cannot move during a storm"
}
```

Next

```
PATCH /api/environment
Content-Type: application/json
Accept: application/json

{
    "storm": false
}
```

```
POST /api/rover/move
Content-Type: application/json
Accept: application/json

{
    "direction": "up"
}
```

```
GET /api/rover/status
Accept: application/json
```

should give

```
200 OK
Content-Type: application/json

{
    "rover": {
        "location": {
            "row": 2,
            "column": 3
        },
        "battery": 8,
        "inventory": [
            {
                "type": "water-sample",
                "qty": 2,
                "priority": 2
            }
        ]
    },
    "environment": {
        "temperature": 60,
        "humidity": 65,
        "solar-flare": false,
        "storm": false,
        "terrain": "water"
    }
}
```

Next

```
PATCH /api/environment
Content-Type: application/json
Accept: application/json

{
    "storm": true
}
```

Any subsequent requests to `/api/rover` and subroutes should not return a response