Mocking Request

Undici have its own mocking utility. It allow us to intercept undici HTTP request and return mocked value instead. It can be useful for testing purposes.

Example:

```
// index.mjs
import { request } from 'undici'

export async function bankTransfer(recepient, ammount) {
  const { body } = await request('http://localhost:3000/bank-transfer',
    {
      method: 'POST',
      headers: {
        'X-TOKEN-SECRET': 'SuperSecretToken',
      },
      body: JSON.stringify({ recepient })
    }
  )
  return await body.json()
}
```

And this is what the test file looks like:

```
// index.test.mjs
import { strict as assert } from 'assert'
import { MockAgent, setGlobalDispatcher, } from 'undici'
import { bankTransfer } from './undici.mjs'

const mockAgent = new MockAgent();

setGlobalDispatcher(mockAgent);

// Provide the base url to the request
const mockPool = mockAgent.get('http://localhost:3000');

// intercept the request
mockPool.intercept({
  path: '/bank-transfer',
  method: 'POST',
  headers: {
    'X-TOKEN-SECRET': 'SuperSecretToken',
  },
  body: JSON.stringify({
    recepient: '1234567890',
    ammount: '100'
  })
```

```
}).reply(200, {
  message: 'transaction processed'
})

const success = await bankTransfer('1234567890', '100')

assert.deepEqual(success, { message: 'transaction processed' })

// if you dont want to check whether the body or the headers contain the same value
// just remove it from interceptor
mockPool.intercept({
  path: '/bank-transfer',
  method: 'POST',
}).reply(400, {
  message: 'bank account not found'
})

const badRequest = await bankTransfer('1234567890', '100')

assert.deepEqual(badRequest, { message: 'bank account not found' })
```

Explore other MockAgent functionality here

Debug Mock Value

When the interceptor we wrote are not the same undici will automatically call real HTTP request. To debug our mock value use `mockAgent.disableNetConnect()`

```
const mockAgent = new MockAgent();

setGlobalDispatcher(mockAgent);
mockAgent.disableNetConnect()

// Provide the base url to the request
const mockPool = mockAgent.get('http://localhost:3000');

mockPool.intercept({
  path: '/bank-tanfer',
  method: 'POST',
}).reply(200, {
  message: 'transaction processed'
})

const badRequest = await bankTransfer('1234567890', '100')
// Will throw an error
// MockNotMatchedError: Mock dispatch not matched for path '/bank-transfer':
// subsequent request to origin http://localhost:3000 was not allowed (net.connect disabled)
```