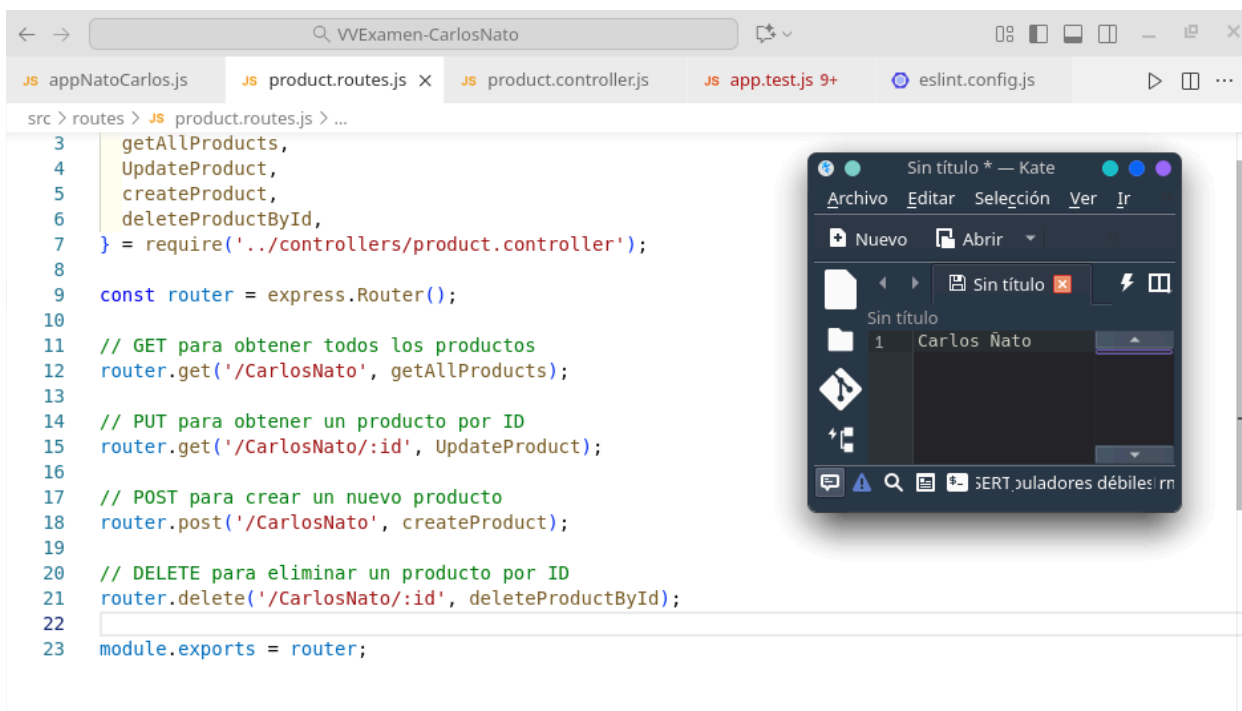


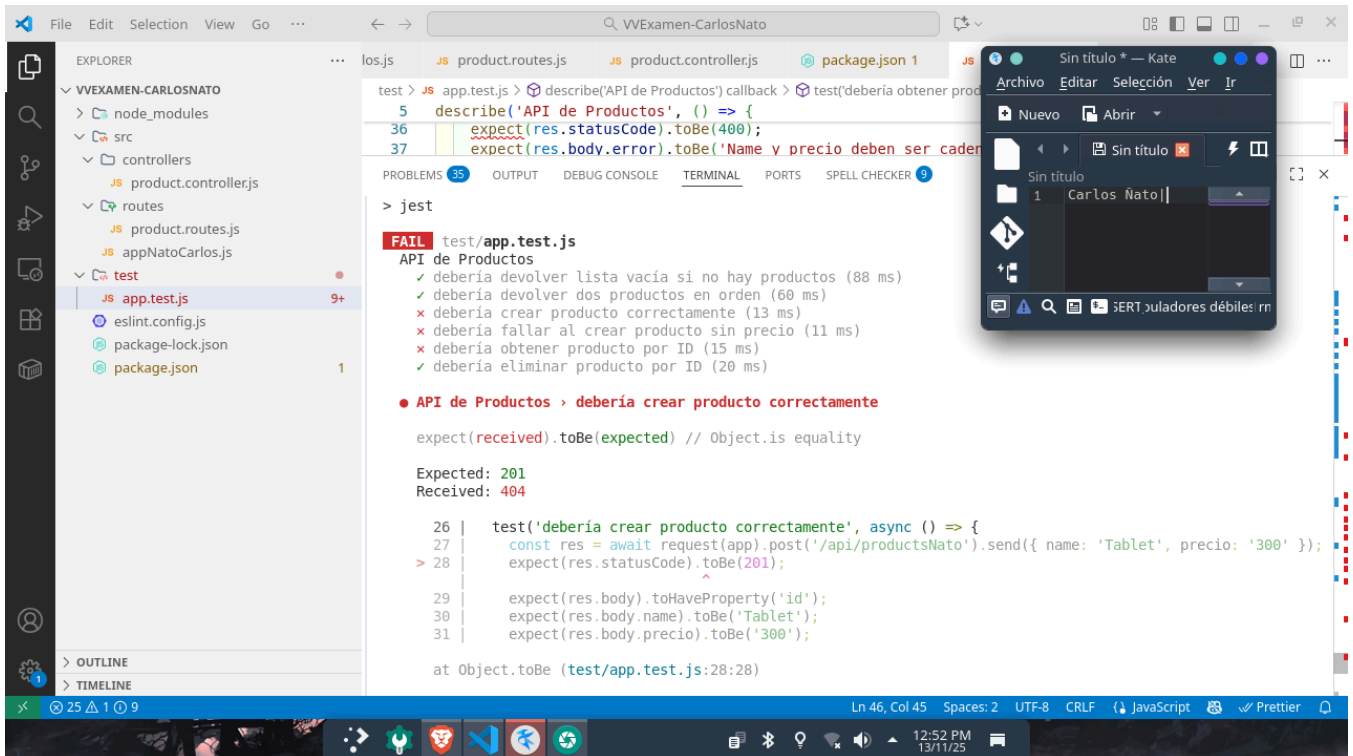
DEPARTAMENTO:	Ciencias de la computación	CARRERA:	Software		
ASIGNATURA:	Pruebas de software	NIVEL:	6to	FECHA:	25/10/2025
DOCENTE:	Luis Alberto Castillo Salinas	PRÁCTICA N°:	1	CALIFICACIÓN:	

Tema: Verificación y Validación en API REST – Node.js

1. Endpoints de productos (2.0 pts)



```
src > routes > JS product.routes.js > ...
3  getAllProducts,
4  UpdateProduct,
5  createProduct,
6  deleteProductById,
7  } = require('../controllers/product.controller');
8
9  const router = express.Router();
10
11  // GET para obtener todos los productos
12  router.get('/CarlosNato', getAllProducts);
13
14  // PUT para obtener un producto por ID
15  router.put('/CarlosNato/:id', UpdateProduct);
16
17  // POST para crear un nuevo producto
18  router.post('/CarlosNato', createProduct);
19
20  // DELETE para eliminar un producto por ID
21  router.delete('/CarlosNato/:id', deleteProductById);
22
23  module.exports = router;
```



The screenshot shows the VS Code interface with the file explorer on the left displaying the project structure. The main editor shows the file `app.test.js` with Jest test results. The tests are as follows:

- API de Productos**
 - ✓ debería devolver lista vacía si no hay productos (88 ms)
 - ✓ debería devolver dos productos en orden (60 ms)
 - ✗ debería crear producto correctamente (13 ms)
 - ✗ debería fallar al crear producto sin precio (11 ms)
 - ✗ debería obtener producto por ID (15 ms)
 - ✓ debería eliminar producto por ID (20 ms)

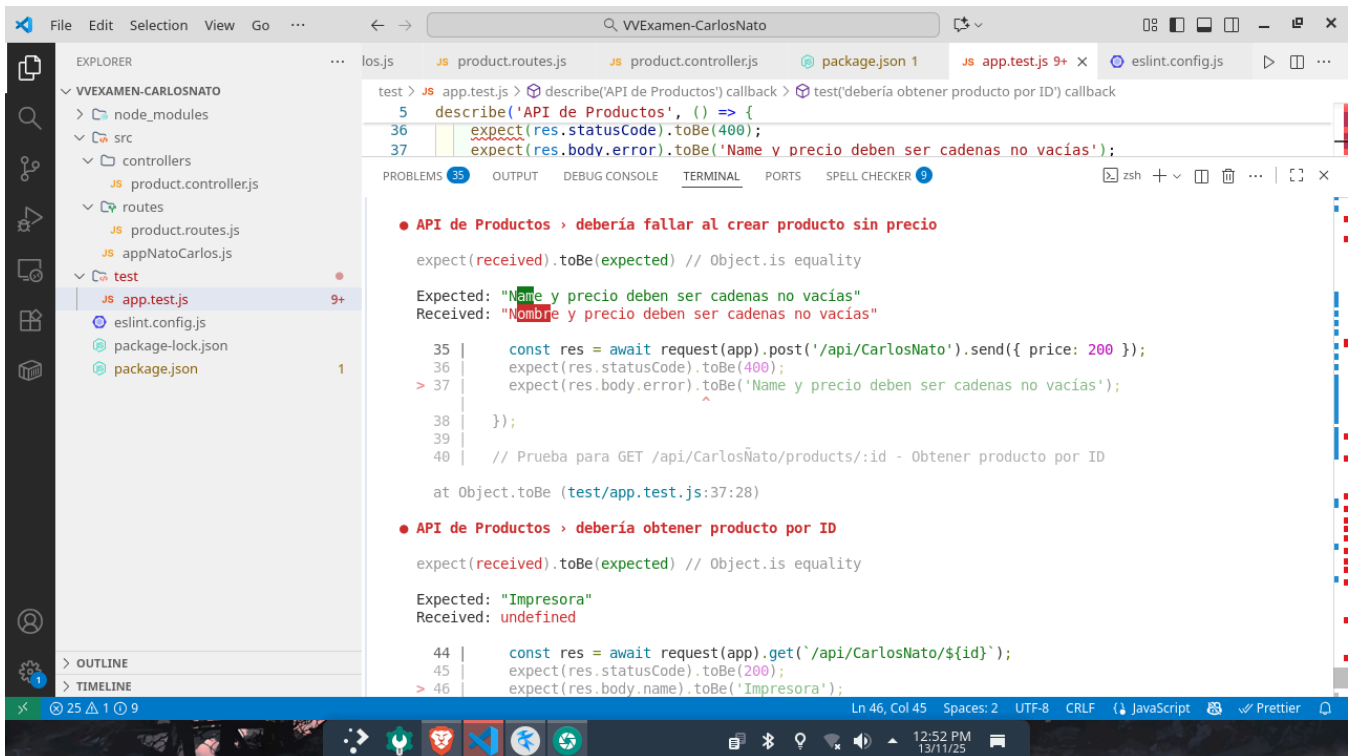
The failed test "debería crear producto correctamente" is expanded, showing the error message: "API de Productos > debería crear producto correctamente". The error details are:

```
expect(received).toBe(expected) // Object.is equality

Expected: 201
Received: 404
```

The test code snippet is:

```
26 | test('debería crear producto correctamente', async () => {
27 |   const res = await request(app).post('/api/productsNato').send({ name: 'Tablet', precio: '300' });
28 |   expect(res.statusCode).toBe(201);
29 |   expect(res.body).toHaveProperty('id');
30 |   expect(res.body.name).toBe('Tablet');
31 |   expect(res.body.precio).toBe('300');
  at Object.toBe (test/app.test.js:28:28)
```



The screenshot shows the VS Code interface with the file explorer on the left displaying the project structure. The main editor shows the file `app.test.js` with Jest test results. The tests are as follows:

- API de Productos**
 - ✗ debería fallar al crear producto sin precio
 - ✗ debería obtener producto por ID

The failed test "debería fallar al crear producto sin precio" is expanded, showing the error message: "API de Productos > debería fallar al crear producto sin precio". The error details are:

```
expect(received).toBe(expected) // Object.is equality

Expected: "Nombre y precio deben ser cadenas no vacías"
Received: "Nombre y precio deben ser cadenas no vacías"
```

The test code snippet is:

```
35 |   const res = await request(app).post('/api/CarlosNato').send({ price: 200 });
36 |   expect(res.statusCode).toBe(400);
37 |   expect(res.body.error).toBe('Nombre y precio deben ser cadenas no vacías');
38 | });
39 |
40 | // Prueba para GET /api/CarlosNato/products/:id - Obtener producto por ID

at Object.toBe (test/app.test.js:37:28)
```

The failed test "debería obtener producto por ID" is expanded, showing the error message: "API de Productos > debería obtener producto por ID". The error details are:

```
expect(received).toBe(expected) // Object.is equality

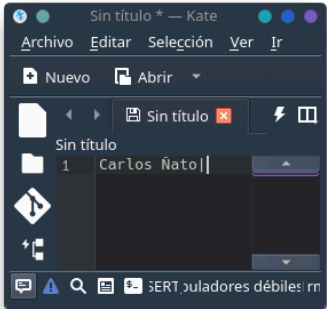
Expected: "Impresora"
Received: undefined
```

The test code snippet is:

```
44 |   const res = await request(app).get(`/api/CarlosNato/${id}`);
45 |   expect(res.statusCode).toBe(200);
46 |   expect(res.body.name).toBe('Impresora');
```

2. Reglas de verificación con ESLint (1.5 pts)

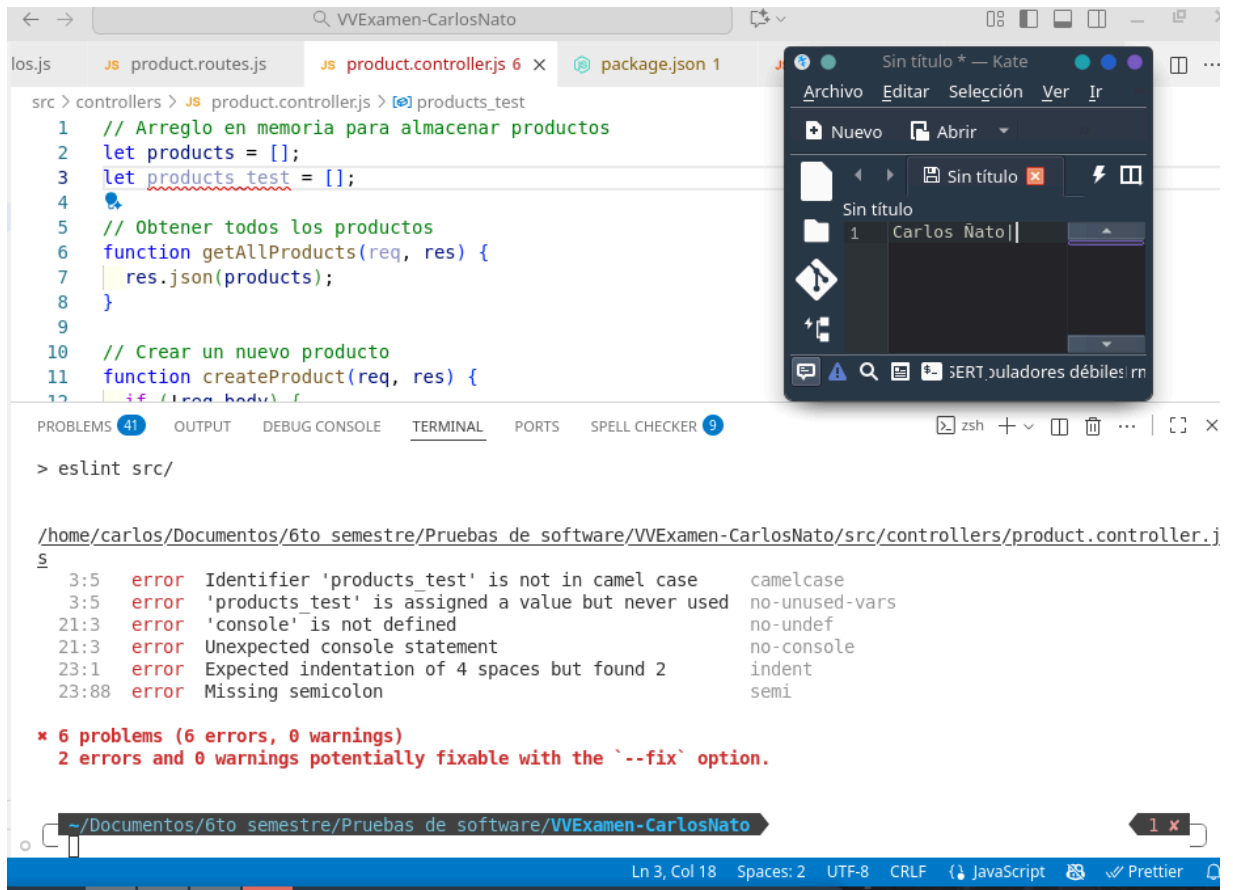
```
eslint.config.js > <unknown> > rules
1  const js = require('@eslint/js');
2
3  module.exports = [
4    {
5      files: ['src/**/*.js', 'test/**/*.js'],
6      languageOptions: {
7        ecmaVersion: 2025,
8        sourceType: 'commonjs',
9      },
10     rules: {
11       ...js.configs.recommended.rules,
12       'no-console': ['error'],
13       'eqeqeq': ['error', 'always'],
14       'camelcase': ['error', { properties: 'always' }],
15       'no-unused-vars': ['error'],
16       "indent": ['error', 2],
17       'semi': ['error', 'always'],
18       'quotes': ['error', 'single'],
19     },
20   ],
21 ];
```



PROBLEMS 31 OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 6

Time: 0.785 s, estimated 1 s

Evidencia PDF:



The screenshot shows the VS Code editor interface. The active file is `product.controller.js` in the `src/controllers` directory. The code contains several ESLint errors. A file explorer overlay is visible on the right, showing the file structure.

```
src > controllers > JS product.controller.js > products_test
1 // Arreglo en memoria para almacenar productos
2 let products = [];
3 let products_test = [];
4
5 // Obtener todos los productos
6 function getAllProducts(req, res) {
7   res.json(products);
8 }
9
10 // Crear un nuevo producto
11 function createProduct(req, res) {
12   if (!req.body) {
```

PROBLEMS (41) OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER (9)

> eslint src/

/home/carlos/Documentos/6to semestre/Pruebas de software/VVExamen-CarlosNato/src/controllers/product.controller.js

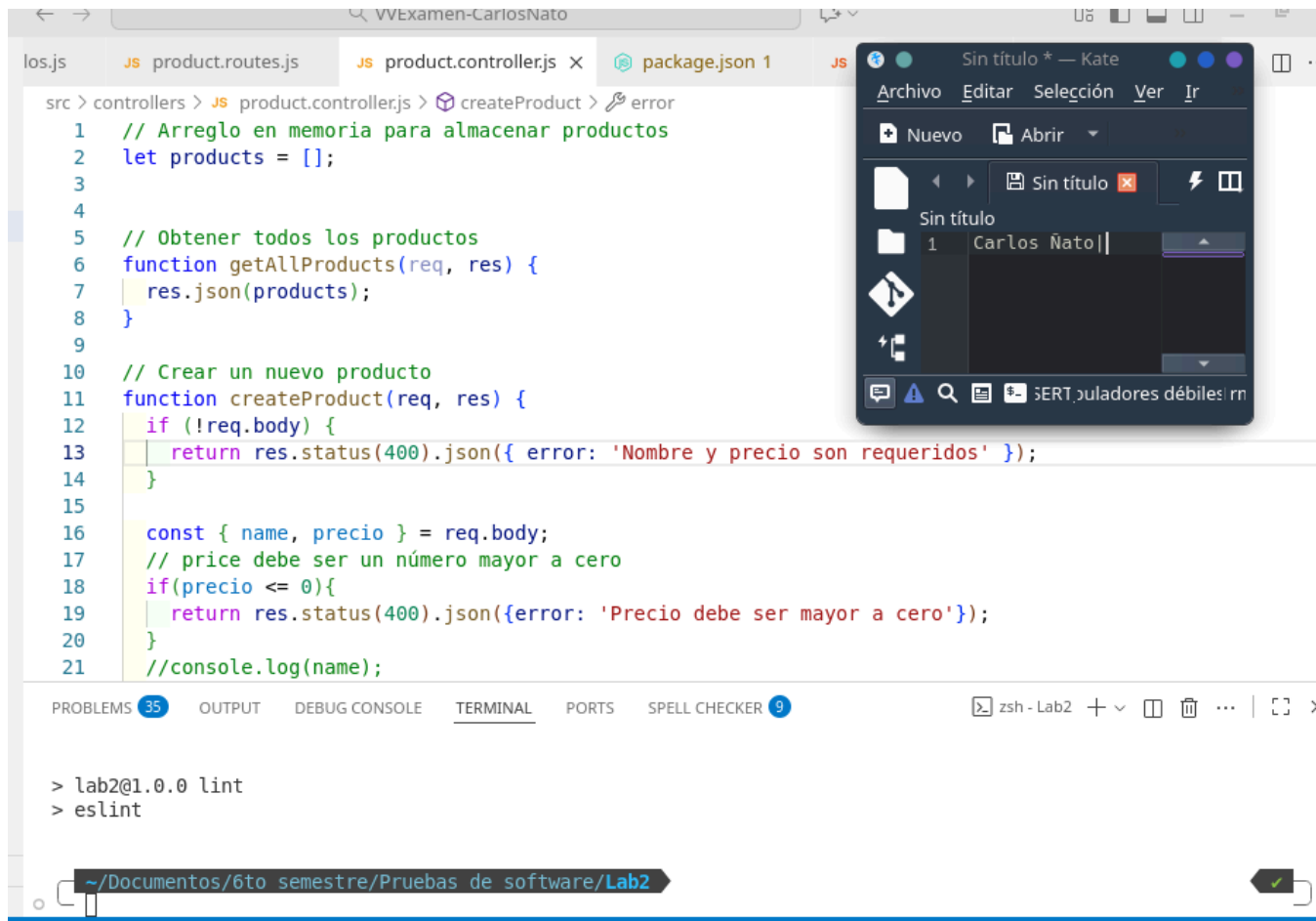
Line	Column	Severity	Message	Source
3:5		error	Identifier 'products_test' is not in camel case	camelcase
3:5		error	'products_test' is assigned a value but never used	no-unused-vars
21:3		error	'console' is not defined	no-undef
21:3		error	Unexpected console statement	no-console
23:1		error	Expected indentation of 4 spaces but found 2	indent
23:88		error	Missing semicolon	semi

✖ 6 problems (6 errors, 0 warnings)
2 errors and 0 warnings potentially fixable with the `--fix` option.

~/Documentos/6to semestre/Pruebas de software/VVExamen-CarlosNato

Ln 3, Col 18 Spaces: 2 UTF-8 CRLF JavaScript Prettier

3. Validaciones de producto (1.5 pts)



```
src > controllers > JS product.controller.js > createProduct > error
1 // Arreglo en memoria para almacenar productos
2 let products = [];
3
4
5 // Obtener todos los productos
6 function getAllProducts(req, res) {
7   res.json(products);
8 }
9
10 // Crear un nuevo producto
11 function createProduct(req, res) {
12   if (!req.body) {
13     return res.status(400).json({ error: 'Nombre y precio son requeridos' });
14   }
15
16   const { name, precio } = req.body;
17   // price debe ser un número mayor a cero
18   if (precio <= 0){
19     return res.status(400).json({error: 'Precio debe ser mayor a cero'});
20   }
21   //console.log(name);
```

PROBLEMS 35 OUTPUT DEBUG CONSOLE TERMINAL PORTS SPELL CHECKER 9

zsh - Lab2

lab2@1.0.0 lint
eslint

~/Documentos/6to semestre/Pruebas de software/Lab2