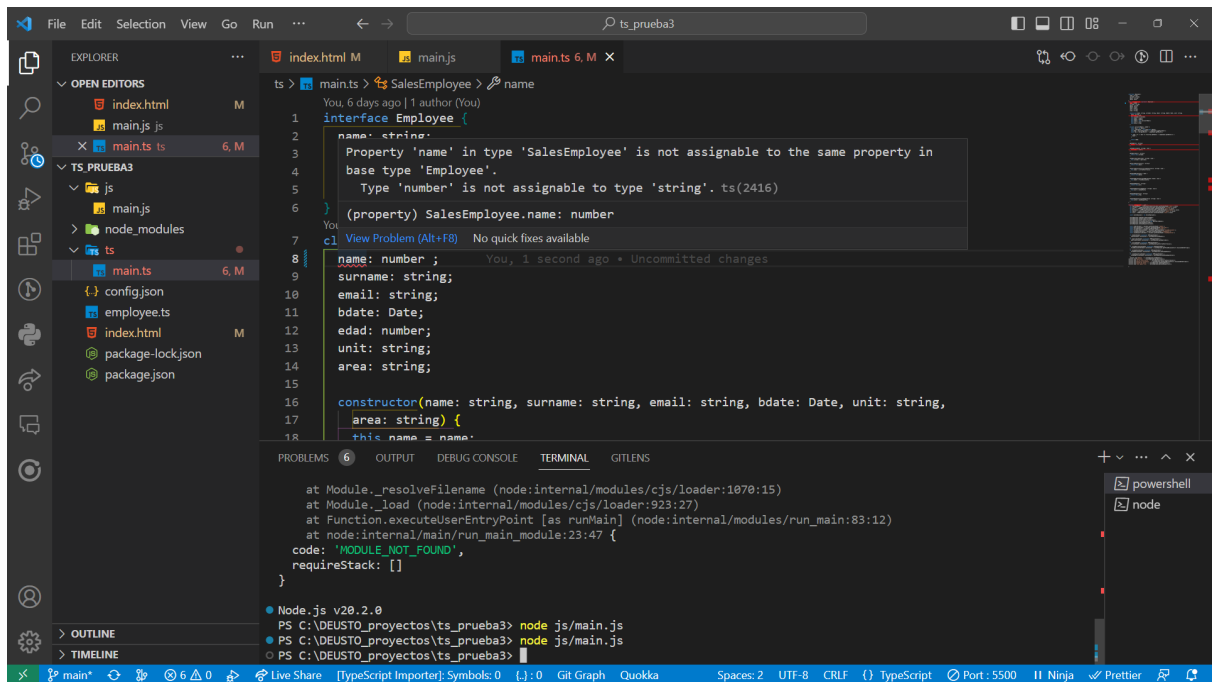


1. Creamos un archivo html y dentro creamos el formulario con los campos de empleado: nombre, apellidos, Correo electrónico, Fecha de nacimiento, Unidad de venta, Zona geográfica.
2. El campo nombre tiene que ser de tipo string.



En el ejemplo anterior, se declara un campo nombre de tipo string y devuelve un valor de tipo string. Si cambiamos el tipo de dato del campo nombre a number en lugar de string, TypeScript muestra un error de compilación, indicando que el tipo es incorrecto.

El tipado estático en TypeScript permite detectar este tipo de errores en tiempo de compilación, lo que ayuda a evitar errores comunes y proporciona un mejor control sobre el código.

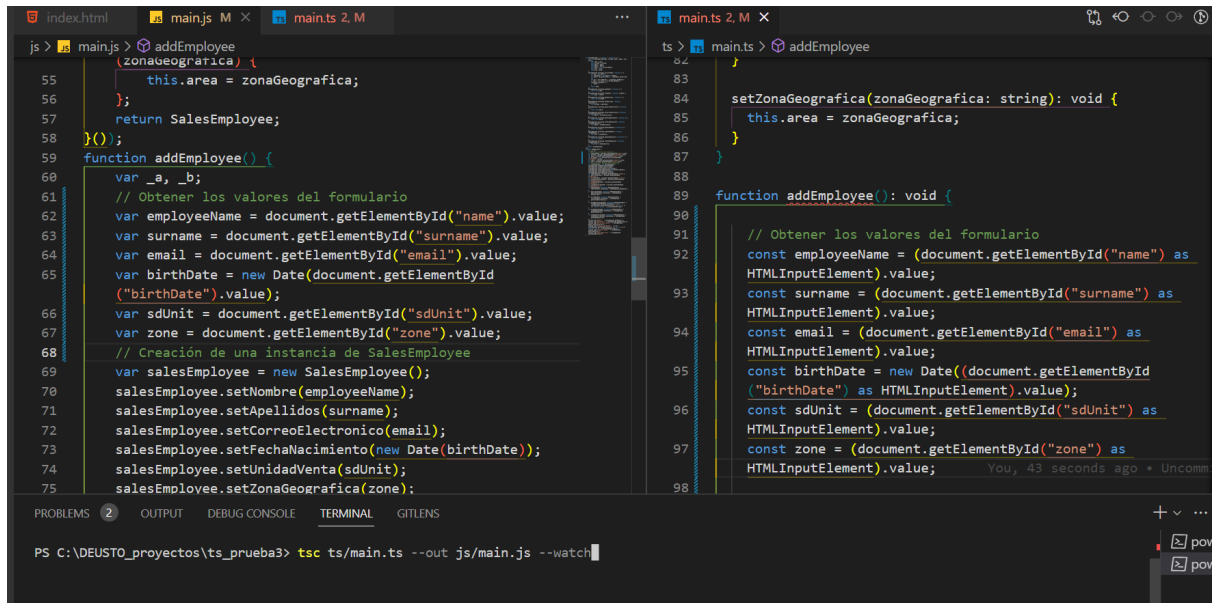
3. La clase SalesEmployee implementa la interfaz Employee con tipado estatico, lo que significa que debe tener todas las propiedades requeridas por la interfaz (name, surname, email y bdate). Además, se agregan propiedades adicionales (edad, unit y area) específicas de la clase SalesEmployee.

```
interface Employee {
  name: string;
  surname: string;
  email: string;
  bdate: Date;
}

class SalesEmployee implements Employee {
  name: string;
  surname: string;
  email: string;
  bdate: Date;
  edad: number;
```

```
unit: string;
area: string;
```

4. Luego creamos la clase SalesEmployee El objeto en cuestión es salesEmployee, que es una instancia de la clase SalesEmployee. Este objeto representa a un empleado de ventas y contiene las propiedades y métodos definidos en la clase SalesEmployee, así como las propiedades requeridas por la interfaz Employee.
5. Transpilamos el código.



The screenshot shows a VS Code editor with two files open: `main.js` and `main.ts`. The `main.js` file contains the JavaScript transpiled version of the TypeScript code, and the `main.ts` file contains the original TypeScript code. The `main.ts` file defines a `SalesEmployee` class and an `addEmployee` function. The `addEmployee` function uses `document.getElementById` to retrieve form values and creates a new `SalesEmployee` instance, setting its properties. The `main.js` file shows the transpiled version of this code, where the `SalesEmployee` class is converted to a constructor function and the `addEmployee` function is converted to a regular function. The terminal at the bottom shows the command used to transpile the code: `PS C:\DEUSTO_proyectos\ts_prueba3> tsc ts/main.ts --out js/main.js --watch`.

```
js > main.js > addEmployee
(zonaGeografica) {
  this.area = zonaGeografica;
};
return SalesEmployee;
}();
function addEmployee() {
  var _a, _b;
  // Obtener los valores del formulario
  var employeeName = document.getElementById("name").value;
  var surname = document.getElementById("surname").value;
  var email = document.getElementById("email").value;
  var birthDate = new Date(document.getElementById("birthDate").value);
  var sdUnit = document.getElementById("sdUnit").value;
  var zone = document.getElementById("zone").value;
  // Creación de una instancia de SalesEmployee
  var salesEmployee = new SalesEmployee();
  salesEmployee.setNombre(employeeName);
  salesEmployee.setApellidos(surname);
  salesEmployee.setCorreoElectronico(email);
  salesEmployee.setFechaNacimiento(new Date(birthDate));
  salesEmployee.setUnidadVenta(sdUnit);
  salesEmployee.setZonaGeografica(zone);
}

ts > main.ts > addEmployee
setZonaGeografica(zonaGeografica: string): void {
  this.area = zonaGeografica;
}
function addEmployee(): void {
  // Obtener los valores del formulario
  const employeeName = (document.getElementById("name") as
HTMLInputElement).value;
  const surname = (document.getElementById("surname") as
HTMLInputElement).value;
  const email = (document.getElementById("email") as
HTMLInputElement).value;
  const birthDate = new Date((document.getElementById("birthDate") as HTMLInputElement).value);
  const sdUnit = (document.getElementById("sdUnit") as
HTMLInputElement).value;
  const zone = (document.getElementById("zone") as
HTMLInputElement).value;
```

```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL GIT LENS
PS C:\DEUSTO_proyectos\ts_prueba3> tsc ts/main.ts --out js/main.js --watch
```