

## Estándares y convenciones

- Usar siempre jQuery (ver anexo jQuery)
- Todo el código en inglés (Objetos, métodos, variables, constantes) (ver anexo nomenclaturas)
- Seleccionar SIEMPRE por id. Nunca class, ni valor, ni por hijos o padres en el DOM, etc. (ver anexo jQuery)
- No dejar comentarios.
- No dejar console.logs
- No copiar y pegar del chat. Usar como un recurso.
- No usar <script></script> en los .ejs .html
- No usar style="display: none;" para ocultar elementos. Usar .hide(); de jquery
- No usar disable para deshabilitar inputs. Usar readonly
- Usar variables globales (credenciales de DB, keys, etc) en las .env

### EJECUTAR EN PC LOCAL CON NODEMON

npm run start:dev

npm run start:prod

### EJECUTAR EN SERVER CON PM2

pm2 start pm2.config.js --env prod

pm2 start pm2.config.js --env dev

## Prácticas de programación

Mala práctica: Muchos \$(document).ready por archivo .js.



```
$(function () {  
  console.log('Logica 1')  
});  
  
$(function () {  
  console.log('Logica 2')  
});
```



```
$(function () {  
  logica1;  
  logica2;  
});  
  
function logica1() {  
  console.log('Logica 1')  
}  
  
function logica2() {  
  console.log('Logica 2')  
}
```

# Código generico

Ejemplo de un método en un controller:



```
delegarNota: async (req, res) => {
  const { idNota } = req.body;
  try {
    const updatedRow = await models.cliente_nota.update(
      { state: 4 },
      { where: { id: idNota } }
    );

    if (updatedRow[0] === 0) {
      return res.status(404).json({ error: 'Nota no encontrada o el estado no se actualizó.' });
    }

    res.status(200).json({ message: 'Estado de la nota actualizado exitosamente.' });
  } catch (error) {
    res.status(500).json({ error: 'Error al actualizar el estado de la nota.' });
    console.error(error);
  }
}
```



```
updateNotaState: async (req, res) => {
  const { idNota, newState } = req.body;

  try {
    const updatedRows = await models.cliente_nota.update(
      { state: nuevoEstado },
      { where: { id: idNota } }
    );

    if (updatedRows[0] === 0) {
      return res.status(404).json({ error: 'Nota no encontrada o el estado no se actualizó.' });
    }

    res.status(200).json({ message: 'Estado de la nota actualizado exitosamente.' });
  } catch (error) {
    res.status(500).json({ error: 'Error al actualizar el estado de la nota.' });
    console.error(error);
  }
}
```

## Como hacer y enviar formularios

En caso de post y get, no usar `methodOverride`. Simplemente poner `get/post` en `method`.

Ejemplo de front y back en:

<https://github.com/arielsoriano/Node-Ejemplo-Base/tree/main/8%20-%20formularios>