Documentació del Projecte: Aplicació de Gestió d'Incidències

1. Objectius del Projecte

En aquest projecte s'ha desenvolupat una aplicació web de gestió d'incidències utilitzant **Node.js, Express, Sequelize** i **EJS** com a motor de plantilles. Les funcionalitats inclouen la creació i visualització d'incidències, gestió de tècnics i actuacions, i una interfície responsive per a diferents rols d'usuari.

2. Estructura del Projecte

```
project/
│
├── models/          → Models Sequelize (Incidencia, Tecnic, Departament, Actuacio)
├── routes/          → Rutes Express (incidencies.js, actuacions.js, tecnics.js)
├── views/           → Plantilles EJS (llistats, formularis, parcial header/footer)
├── public/          → Recursos públics (css, js)
├── config/          → Configuració de la base de dades Sequelize
├── app.js           → Fitxer principal d'Express
└── package.json     → Dependències i scripts del projecte
```

3. Modelatge de Dades (Sequelize)

Exemple d'ús de `ENUM`:

```
prioritat: {
  type: DataTypes.ENUM('Baixa', 'Mitjana', 'Alta'),
  allowNull: true,
}
```

Relacions entre taules:

```
Tecnic.hasMany(Incidencia, { foreignKey: 'tecnicId', onDelete: 'CASCADE' });
Incidencia.belongsTo(Tecnic, { foreignKey: 'tecnicId' });

Departament.hasMany(Incidencia, { foreignKey: 'departamentId', onDelete: 'CASCADE' });
Incidencia.belongsTo(Departament, { foreignKey: 'departamentId' });
```

4. Funcionalitats Implementades

Creació d'incidències

Es pot crear una incidència des d'un formulari. La incidència creada es mostra a una vista especial:

```
router.post('/create', async (req, res) => {
  try {
    const { descripcio, departament } = req.body;
    inc = await Incidencia.create({ descripcio, departament });
    res.render('incidencies/creada', { inc });
  } catch (error) {
    res.status(500).send(`Error: ${error.message}`);
  }
});
```

Vista de llistat d'incidències amb dades relacionades

```
const incidencies = await Incidencia.findAll({
  include: [
    { model: Departament, attributes: ['id', 'nom'] },
    { model: Tecnic, attributes: ['id', 'nom'] }
  ]
});
```

Condicional per mostrar dades només si existeixen:

```
<td><%= incidencia.Tecnic ? incidencia.Tecnic.nom : "No assignat" %></td>
<td><%= incidencia.Departament ? incidencia.Departament.nom : "No assignat" %></td>
```

Llistat d'actuacions d'una incidència

Es mostra la informació de la incidència i totes les seves actuacions:

```
router.get('/:id', async (req, res) => {
  const actuacions = await Actuacio.findAll({ where: { incidenciaId: req.params.id } });
  const infoIncidencia = await Incidencia.findByPk(req.params.id, {
    include: [
      { model: Departament, attributes: ['id', 'nom'] },
      { model: Tecnic, attributes: ['id', 'nom'] }
    ]
  });
  res.render('actuacions/list', { infoIncidencia, actuacions });
});
```

5. Gestió d’Errors amb Sequelize

Totes les crides importants es controlen amb `try...catch`. Els errors es mostren de forma clara al terminal o al navegador.

```
catch (error) {
  console.error('Error:', error);
  res.status(500).send(`Error: ${error.message}`);
}
```

6. Interfície d’Usuari (EJS + Bootstrap + JS)

S’ha aplicat **Bootstrap** per a l’estil i s’han afegit **scripts JavaScript** per millorar l’experiència de l’usuari:

Validació de formularis:

```
document.getElementById("formulariIncidencia").addEventListener("submit", function(e) {
  // validació bàsica de camps obligatoris
});
```

Filtrat d’incidències:

```
document.getElementById("filtre").addEventListener("change", function() {
  // mostra/oculta files per prioritat o estat
});
```

7. Desplegament

L’aplicació està preparada per funcionar tant en local com en plataformes com Heroku o Render.

Passos per executar en local:

1. `npm install`
2. Configura `.env` amb credencials de base de dades
3. `npm run dev` o `node app.js`

Preparació per desplegament:

- Port dinàmic amb `process.env.PORT`

- `start` script a `package.json`:

```
"scripts": {
  "start": "node app.js"
}
```