Fiber Splice Sheet Generator API

A REST API for generating fiber optic splice sheets from Excel input files. This API automatically creates comprehensive splice documentation that tracks fiber connections from main feeder cables through distribution points to customer terminals.

Features

- 🔌 **Multiple Cable Support**: Handle various cable configurations (144F, 48F, etc.)
- 🎨 **Standard Color Coding**: Implements TIA-598-C fiber color standards
- 📊 **Excel Integration**: Input and output Excel file support
- 🏠 **Address Management**: Track customer premises and terminal locations
- 📋 **Unused Fiber Tracking**: Automatically mark unused fibers for future expansion
- 🔧 **Flexible Configuration**: Support for different port counts and cable layouts

Installation

```bash
# Clone the repository
git clone <repository-url>
cd fiber-splice-sheet-generator

# Install dependencies
npm install

# Start the server
npm start

# For development with auto-reload
npm run dev
```

API Endpoints

Health Check

```
GET /health
```

Returns the API status.

Response:

```json
{
  "status": "OK",
  "message": "Splice Sheet Generator API is running"
}
```

Generate Splice Sheet from Excel File

```
POST /generate-splice-sheet
Content-Type: multipart/form-data
```

Upload an Excel file and generate a splice sheet.

Parameters:

- `inputFile` (file): Excel file with cable configuration

Response:

```json
{
  "success": true,
  "message": "Splice sheet generated successfully",
  "filename": "splice_sheet_1234567890.xlsx",
  "rowCount": 360,
  "data": [
    ["Port #", "Cable Name", "Port #", "Cable", "B#", "(B)", "F#", "(F)", "MST", "Address"],
    [1, "FDH108_144F_1-96", 1, "144F(1)", 1, "BL", 1, "bl", "MST_F1000ECOATSAVE.210820", "2101
  ]
}
```

Generate Custom Splice Sheet

```
POST /generate-custom-splice-sheet
Content-Type: application/json
```

Generate a splice sheet with custom parameters.

Request Body:

```json
{
  "ports": 96,
  "mainCableName": "FDH108_144F_1-96",
  "cables": [
    {
      "name": "144F(1)",
      "fiberCount": 144
    },
    {
      "name": "144F(2)",
      "fiberCount": 144
    },
    {
      "name": "48F(3)",
      "fiberCount": 48
    }
  ],
  "addresses": [
    {
      "mst": "MST_F1245ECOATSAVE.210819",
      "address": "2101 MARENGO LK RD",
      "sheet": 14,
      "terminal": "T1"
    }
  ]
}
```

Get Fiber Standards

```
GET /fiber-standards
```

Returns fiber optic color coding standards.

Response:

json

```json
{
  "bufferColors": ["BL", "OR", "GR", "BR", "SL", "WH", "RD", "BK", "YL", "VT", "RS", "AQ"],
  "fiberColors": ["bl", "or", "gr", "br", "sl", "wh", "rd", "bk", "yl", "vt", "rs", "aq"],
  "standardFibersPerTube": 12,
  "colorCodingStandard": "TIA-598-C"
}
```

Input File Format

The input Excel file should have the following structure:

Ports	Cable 1	Cable 2	Cable 3	MST	Address

Column Descriptions:

- **Ports**: Total number of ports to generate
- **Cable 1-N**: Cable configurations (name and fiber count)
- **MST**: Main Service Terminal prefix
- **Address**: Customer premise addresses

Output Format

The generated splice sheet includes:

1. **Port #**: Sequential port numbers
2. **Cable Name**: Main feeder cable identifier
3. **Cable Sections**: Multiple columns for each cable:
   - Port # (for this cable)
   - Cable name
   - B# (Buffer tube number)
   - (B) (Buffer tube color)
   - F# (Fiber number within tube)
   - (F) (Fiber color)
4. **MST**: Main Service Terminal identifier
5. **Address**: Customer premise address
6. **Additional Info**: Sheet numbers, terminal designations

Fiber Color Coding

The API uses standard TIA-598-C color coding:

Buffer Tube Colors (12-tube sequence):

1. Blue (BL)
2. Orange (OR)
3. Green (GR)
4. Brown (BR)
5. Slate (SL)
6. White (WH)
7. Red (RD)
8. Black (BK)
9. Yellow (YL)
10. Violet (VT)
11. Rose (RS)
12. Aqua (AQ)

Fiber Colors (12-fiber sequence):

1. Blue (bl)
2. Orange (or)
3. Green (gr)
4. Brown (br)
5. Slate (sl)
6. White (wh)
7. Red (rd)
8. Black (bk)
9. Yellow (yl)
10. Violet (vt)
11. Rose (rs)
12. Aqua (aq)

Example Usage

Using cURL

bash

```bash
# Generate splice sheet with file upload
curl -X POST \
  -F "inputFile=@input.xlsx" \
  http://localhost:3000/generate-splice-sheet

# Generate custom splice sheet
curl -X POST \
  -H "Content-Type: application/json" \
  -d '{
    "ports": 144,
    "mainCableName": "FDH201_288F_1-144",
    "cables": [
      {"name": "144F(1)", "fiberCount": 144},
      {"name": "144F(2)", "fiberCount": 144}
    ]
  }' \
  http://localhost:3000/generate-custom-splice-sheet
```

Using Node.js

javascript

```javascript
const axios = require('axios');
const FormData = require('form-data');
const fs = require('fs');

// Upload file and generate splice sheet
const form = new FormData();
form.append('inputFile', fs.createReadStream('input.xlsx'));

axios.post('http://localhost:3000/generate-splice-sheet', form, {
  headers: form.getHeaders()
})
.then(response => {
  console.log('Splice sheet generated:', response.data);
})
.catch(error => {
  console.error('Error:', error.response.data);
});
```

Using Python

```python
import requests

# Generate custom splice sheet
data = {
    "ports": 96,
    "mainCableName": "FDH108_144F_1-96",
    "cables": [
        {"name": "144F(1)", "fiberCount": 144},
        {"name": "48F(2)", "fiberCount": 48}
    ]
}

response = requests.post(
    'http://localhost:3000/generate-custom-splice-sheet',
    json=data
)

print(response.json())
```

Error Handling

The API returns appropriate HTTP status codes and error messages:

- **200**: Success
- **400**: Bad request (invalid input)
- **500**: Internal server error

Error response format:

```json
{
  "success": false,
  "message": "Error description",
  "error": "Detailed error message"
}
```

Configuration

Environment variables:

- `PORT`: Server port (default: 3000)
- `NODE_ENV`: Environment (development/production)

Testing

```
bash

# Run tests
npm test

# Run tests in watch mode
npm run test:watch
```

Contributing

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Add tests for new functionality
5. Submit a pull request

License

MIT License - see LICENSE file for details.

Support

For support and questions:

- Create an issue on GitHub
- Email: support@example.com

Changelog

v1.0.0

- Initial release
- Basic splice sheet generation
- Excel file input/output support

- Standard fiber color coding
- Multiple cable configuration support