PERTANYAAN

1. **Pada setiap fungsi di Appointment.js, parameter pertama pasti memiliki prefix "${cors}" yang merefer pada variable cors. Mengapa hal tersebut perlu dilakukan? Apa yang terjadi jika prefix tersebut dihilangkan?**

   Cors berfungsi untuk mengizinkan pemanggilan API dari *resource origin* yang berbeda. Hal ini karena sebuah *browser* hanya mengizinkan *web application* untuk memanggil API dari *resource origin* yang sama untuk keamanan. Jika cors dihilangkan maka akan muncul *error access to fetch* pada url API, di-*block* oleh CORS policy.

2. **Pada fungsi handleFormSubmit(e) di UpdatePasien.jsx terdapat klausa:**

```
let last = name.pop()
name.reduce((prev,next) => {
    return prev[next] = prev[next] || {};
}, dataJson)[last] = val
```

   Apa yang dilakukan pada code tersebut? Jika klausa tersebut diganti dengan "dataJson[key] = val", apa yang terjadi dengan isi variable dataJson?

   Pada form updatePasien, terdapat *input* dengan atribut name="statusPasien.id". Jika tidak ada "dataJson[key]=val" maka data yang dikirim menjadi:

   {
   statusPasien.id = "id_pasien"
   }

   Dengan "dataJson[key]=val" maka akan menjadi:
   {
   "statusPasien": {
   "Id" : "id_pasien"
   }
   }

LATIHAN

1. **Tambahkan menu pada navbar untuk menampilkan daftar seluruh staf Farmasi, dan implementasikan pagenya hingga menampilkan data seperti pada page daftar pasien**

- Pada Appointment.js tambahkan method getAllStaffFarmasi() untuk melakukan *request* getAllStaffFarmasi sebagai berikut:

```
getAllStaffFarmasi() {
    return fetch(`${cors}${baseUrl}/1/getAllStaffFarmasi`, {
        method: 'GET',
    })
    .then(response => {
        return response.json()
    })
    .then(jsonResponse => {
        return jsonResponse
    })
}
```

- Pada folder *components*, tambahkan file DaftarStaffRow.jsx dengan isi sebagai berikut:

```
import React from 'react';
import { Link } from "react-router-dom";

export const DaftarStaffRow = (props) => {
    return (
        <tbody>
            {props.listStaff.map(staff => {
                return (
                    <tr key={staff.id}>
                        <td>{staff.nama}</td>
                        <td>{staff.jenis}</td>
                    </tr>
                )
            })}
        </tbody>
    )
}
```

- Pada folder *screens*, tambahkan *file* DaftarStaffFarmasi.jsx dengan isi yang termasuk pemanggilan *function* getAllStaffFarmasi() sebagai berikut:

```
export class DaftarStaffFarmasi extends React.Component {

    constructor(props) {
        super(props)
        this.state = {
            loading: true,
            listStaff: []
        }
        Appointment.getAllStaffFarmasi().then(response => {
            this.setState({
                loading: false,
                listStaff: response.result
            })
        })
    }
```

```
render() {
    if (this.state.loading) {
        return (
            <Loading msg="Fetching Data..."/>
        )
    } else {
        return (
            <TableContainer title="Daftar Staff Farmasi" header={['Nama Staff', 'Jenis Staff']}>
                <DaftarStaffRow listStaff={this.state.listStaff}/>
            </TableContainer>
        )
    }
}
```

- Pada *file* App.js, tambahkan navigasi pada navbar dengan menambahkan potongan kode berikut:

```
<li className='nav-item'>
    <NavLink to="/all-staff-farmasi" exact className="nav-link"
    activeClassName="active">Daftar Staff Farmasi</NavLink>
</li>
```

dan tambahkan juga *route path*-nya:

```
<Route path="/all-staff-farmasi" exact component={DaftarStaffFarmasi} />
```

2. **Pada daftar pasien, tambahkan button untuk menambahkan hasil lab pasien, yang jika diklik maka akan menampilkan form untuk menginput hasil lab pasien. Implementasikan form hingga berhasil melakukan request POST ke SI-Appointment.**
    - Pada Appointment.js, tambahkan *request* POST melalui *method* addHasilLabPasien dengan isi sebagai berikut:

```
addHasilLabPasien(requestBody) {
    return fetch(`${cors}${baseUrl}/1/addLabResult`, {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(requestBody)
    })
    .then(response => {
        return response.json()
    })
    .then(jsonResponse => {
        return jsonResponse
    })
},
```

    - Membuat *file* FormHasilLab.jsx untuk menerima *input* hasil lab dari pasien, dengan isi sebagai berikut:

```
export const FormHasilLab = (props) => {
    return (
        <form onSubmit={props.onSubmit}>
            <h2>Tambah Hasil Lab Pasien</h2>
            <input type="hidden" name="pasien.id" value={props.pasien.id} />
            <div className="form-group">
                <label>Nama Pasien<span style={{ color: 'red' }}>*</span></label>
                <input type="text" className="form-control"  defaultValue={props.pasien.nama} readOnly/>
            </div>
            <div className="form-group">
                <label>Jenis</label>
                <input type="text" className="form-control" name="jenis" />
            </div>
            <div className="form-group">
                <label>Hasil</label>
                <input type="text" className="form-control" name="hasil" />
            </div>
            <div className="form-group">
                <label>Tanggal Pengajuan</label>
                <input type="date" className="form-control" name="tanggalPengajuan" />
            </div>
            <button className="btn btn-success" value="submit">Tambah</button>
        </form>
    )
}
```

- Pada *screens,* tambahkan *file* HasilLab.jsx yang di dalamnya memanggil method getDetailPasien:

```
constructor(props) {
    super(props)
    this.state = {
        loading: true,
        pasien: {},
    }
    this.handleFormSubmit = this.handleFormSubmit.bind(this)
    Appointment.getDetailPasien(this.props.match.params.id).then(response => {
        if (response.status === 200) {
            this.setState({
                loading: false,
                pasien: response.result
            })
        } else {
            alert('Data tidak ditemukan')
            this.props.history.push('/all-pasien')
        }
    })
}
```

dan method handleFormSubmit untuk membuat *button* “Tambah Hasil Lab” menjadi fungsional:

```
handleFormSubmit(e) {
    e.preventDefault()
    this.setState({
        loading: true
    })

    const data = new FormData(e.target)
    const dataJson = {}

    data.forEach((val, key) => {
        if (val !== "") {
            let name = key.split('.');
            if (name.length > 1) {
                let last = name.pop()
                name.reduce((prev, next) => {
                    return prev[next] = prev[next] || {};
                }, dataJson)[last] = val
            } else {
                dataJson[key] = val
            }
        }
    })
```

```
Appointment.addHasilLabPasien(dataJson).then(response => {
    if (response.status === 200) {
        this.setState({
            loading: false,
        })
        alert(`Sukses tambah hasil lab pasien ${this.state.pasien.nama}`)
    } else {
        this.setState({
            loading: false
        })
        alert(`Gagal tambah hasil lab pasien ${this.state.pasien.nama}`)
    }
})
```

- Pada *file* App.js tambahkan *route path* untuk menambah hasil lab, sebagai berikut:

```
<Route path="/hasil-lab/:id" exact component={HasilLab} />
```