## 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?**

Jawaban:
Pada dasarnya sebuah browser hanya mengizinkan web application melakukan pemanggilan API dari *resource origin* yang sama atas alasan keamanan, misalnya web application yg memanggil API dari http://domain-a.com tidak bisa memanggil API dari http://api.domain-b.com/data.json

Agar browser mengizinkan pemanggilan API dari resource origin berbeda, diperlukan cors. Jika cors dihilangkan akan muncul error access to fetch pada url API di block oleh CORS policy

**2. Pada fungsi handleFormSubmit(e) di UpdatePasien.jsx terdapat klausa:**
**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 sebuah input yang memiliki attribute name="statusPasien.id". Jika tidak ada **"dataJson[key]=val"** data dikirim akan menjadi:

```
{
        statusPasien.id = "id_pasien"
}
```

Dengan adanya **"dataJson[key]=val"** menjadi:

```
{
        "statusPasien": {
                "Id" : "id_pasien"
        }
}
```

**LATIHAN**
Lihat daftar staff farmasi

1. Tambahkan menu daftar staff farmasi pada navbar di App.js

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

   Link url tersebut ke komponen baru yang telah dibuat yaitu DaftarStaff

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

2. Membuat fungsi getAllStaffFarmasi() pada appointment.js

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

3. Buat komponen DaftarStaffRow

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

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

```
export class DaftarStaff extends React.Component {
    /**
     * TODO: Akses method getAllPasien() pada Appointment dan lakukan
     * TODO: Lakukan pemanggilan pada constructor() atau pada lifecycl
     */

    constructor(props) {
        super(props)
        this.state = {
            loading: true,
            listStaff: []
        }

        Appointment.getAllStaffFarmasi().then(response => {
            this.setState({
                loading: false,
                listStaff: response.result
            })
        })
    }
```

Tambah hasil lab

1. Tambahkan kolom button pada DaftarPasienRow

```
<td>
    <Link to={`/add-lab-pasien/${pasien.id}`} className="btn btn-success">Add hasil lab</Link>
</td>
```

2. Pada App.js route link tersebut ke komponen AddLab yang sudah dibuat

```
<Route path="/add-lab-pasien/:id" exact component={AddLab} />
```

3. Implementasikan komponen AddLab

```
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')
        }
    })
}
```

```
Appointment.addHasilLab(dataJson).then(response => {
    console.warn(dataJson)
    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 ${this.state.pasien.nama}`)
    }
    console.log(response)
})
```

```
handleFormSubmit(e) {
    e.preventDefault()
    /**
     * TODO: Akses method updateStatusPasien(requestBody) pada Appointment dan lak
     */
    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
            }
        }
    })
```