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:**

```
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?

   Jawaban:

   Pada form updatePasien terdapat sebuah input yang memiliki attribute name="statusPasien.id". Jika tidak ada **"dataJson[key]=val"** data dikirim menjadi:

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

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

```
{
    "statusPasien":{
        "id" :  "id_pasien"
    }
}
```

LATIHAN

1. **Daftar Seluruh Staff Farmasi**
   - Untuk melakukan request getAllStaffFarmasi, dilakukan implementasi pada fungsi getAllStaffFarmasi() pada Appointment.js seperti berikut:

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

   - Membuat file bernama DaftarStaffRow.jsx yang berisikan:

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

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

   - Melakukan pemanggilan function getAllStaffFarmasi() pada DaftarStaffFarmasi.jsx, sehingga dapat menampilkan daftar staff farmasi sesuai dengan data SI-Appointment
     - Pada Constructor

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

```
                loading: false,
                listStaffFarmasi: response.result
            })
        })
    }
```

- Pada method render():

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

- Menambahkan pilihan di navbar yang akan mengarahkan kepada halaman daftar staff farmasi pada App.js
  - Pada div dengan nama “App” tambahkan

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

  - Pada div dengan nama “container” tambahkan

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

2. **Menambahkan Hasil Lab Pasien**
   - Pada Appointment.js, tambahkan *request* POST melalui method AddLabResult() seperti berikut:

```
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 bernama FormAddLabResult.jsx untuk menerima *input* hasil lab dari pasien, yang berisikan:

```
import React from 'react';


export const FormAddLabResult = (props) => {
    return (
        <form onSubmit={props.onSubmit}>
            <h2>Tambah Hasil Lab</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" name="pasien.nama" 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>
    )
}
```

- Membuat file bernama AddLabResult.jsx
    - Melakukan pemanggilan function getDetailPasien() pada Constructor

```
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')
            }
        })
```

    - Melakukan implementasi pada fungsi handleFormSubmit(e) di AddLabResult.jsx, seperti berikut:

```
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.AddLabResult(dataJson).then(response => {
            console.log(dataJson)
            if (response.status === 200){
                this.setState({
                    loading : false,
                })
                alert(`Sukses update pasien ${this.state.pasien.nama}`)
            } else {
                this.setState({
                    loading: false
                })
                alert(`Gagal update pasien ${this.state.pasien.nama}`)
            }
            console.log(response)
        })
    }
```

- Agar formulir penambahan hasil lab dapat dimunculkan, pada method render() implementasikan bagian berikut:

```
if (this.state.loading) {
                return (
                    <Loading msg="Fetching Data..." />
                )
            } else {
                return (
                    <FormAddLabResult pasien={this.state.pasien}
onSubmit={this.handleFormSubmit} />
                )
            }
```

Dengan demikian, jika button “Add Hasil Lab” diklik, maka akan dilakukan request POST dan melakukan penambahan hasil lab pada pasien tertentu

- Menambahkan pilihan di tabel daftar pasien yang akan mengarahkan kepada halaman formulir tambah hasil lab pada DaftarPasienRow.js

```
<td>
    <Link to={`/update-pasien/${pasien.id}`} className="btn btn-info">Update</Link>
    <Link to={`/add-lab/${pasien.id}`} className="btn btn-info">Add Hasil Lab</Link>
</td>
```

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

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