MINGGU 8
Capacitor, Camera & Context

DESKRIPSI TEMA

Menggunakan Capacitor, Camera dan Context

CAPAIAN PEMBELAJARAN MINGGUAN (SUB-CAPAIAN PEMBELAJARAN)

Mahasiswa mampu menggunakan React Context untuk mengelola data dan state aplikasi
Mahasiswa mampu menerapkan fitur camera pada aplikasi Ionic

PERALATAN YANG DIGUNAKAN

WebStorm / VS Code
Android Studio

LANGKAH-LANGKAH PRAKTIKUM

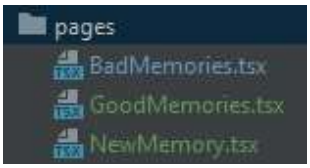
Buat Project Ionic-React Baru

1. Buatlah sebuah Project Ionic-React yang baru, dengan template Blank.
2. Hapus folder components dan pages default.
3. Ubah code pada App.tsx menjadi seperti berikut

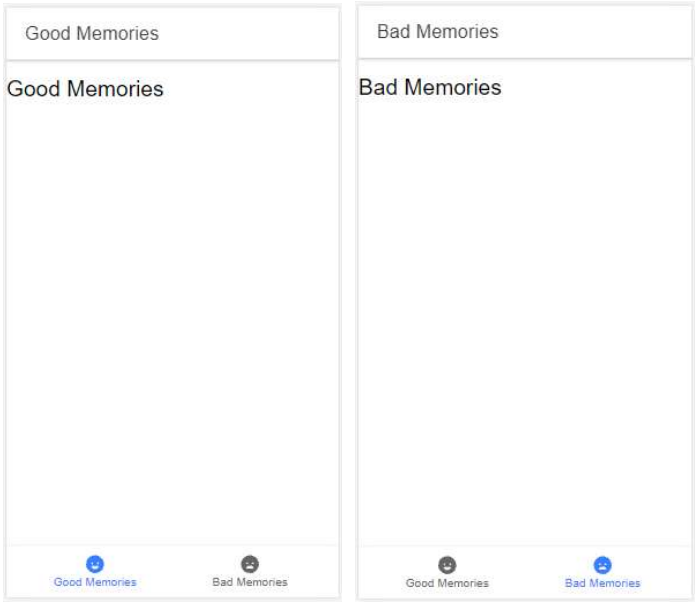
```
const App: React.FC = () => (
  <IonApp>
    <h2>This works!</h2>
  </IonApp>
);
```

Setup MemoryApp

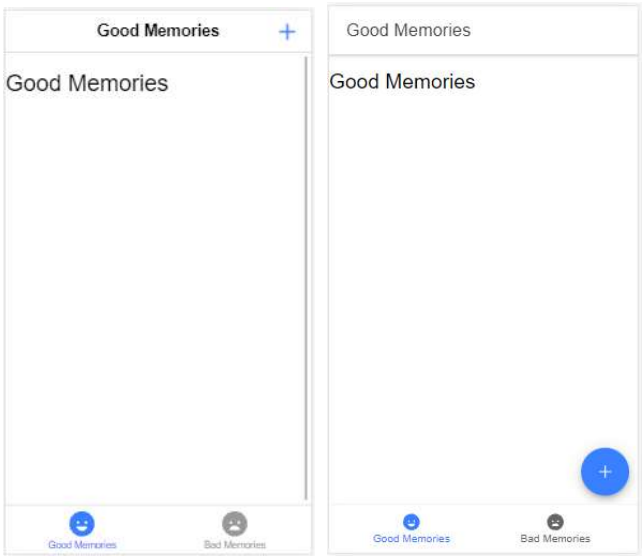
1. Buat tiga buah file dalam folder pages: GoodMemories.tsx, BadMemories.tsx dan NewMemory.tsx



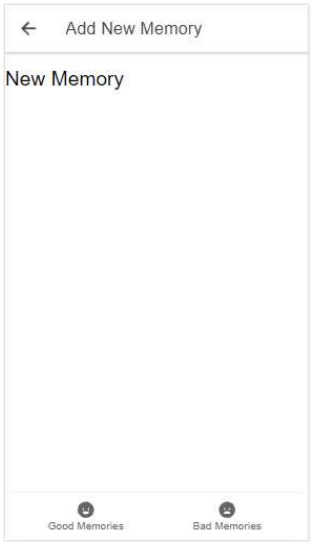
2. Buat navigasi Tabs, serta tampilan page GoodMemories dan BadMemories seperti gambar berikut:



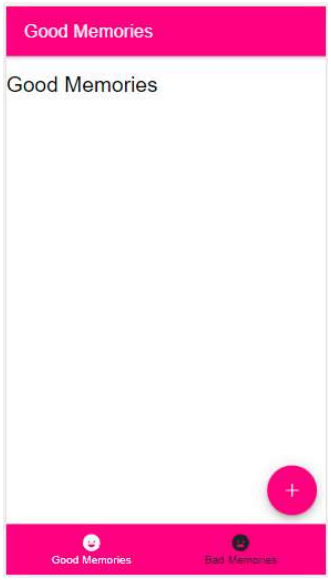
3. Tambahkan Tombol untuk mengakses page NewMemory berupa Toolbar Button dan Fab Button. Toolbar Button akan tampil pada platform iOS (gambar kiri), dan FabButton akan tampil pada platform Android (gambar kanan)::



4. Berikut ini adalah tampilan page NewMemory:

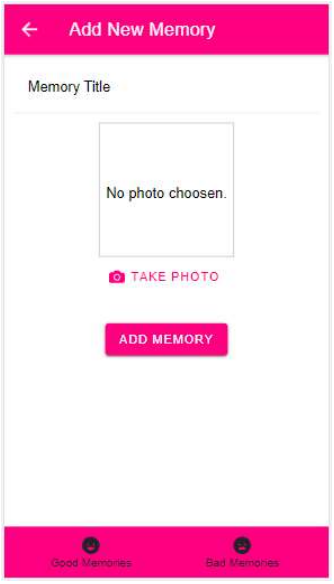


5. Lakukan styling sesuai selera Anda.



New Memory - Open Camera

1. Buat tampilan page New Memory seperti gambar berikut ini (source-code merupakan sebagian dari code NewMemory.tsx). Buat file NewMemory.css untuk menambahkan styling untuk class "image-preview".




```
<IonRow className="ion-text-center">
    <IonCol>
        <div className="image-preview">
            <h3>No photo choosen.</h3>
        </div>
        <IonButton fill="clear">
            <IonIcon slot="start" icon={camera} />
            <IonLabel>Take Photo</IonLabel>
        </IonButton>
    </IonCol>
</IonRow>
<IonRow className="ion-margin-top">
    <IonCol className="ion-text-center">
        <IonButton>Add Memory</IonButton>
    </IonCol>
</IonRow>
```

2. Jalankan command:

```
npm install @capacitor/camera
```

3. Tambahkan method takePhotoHandler untuk menangani ketika tombol Take Photo di-click.

```
import {Camera, CameraResultType, CameraSource} from '@capacitor/camera';

const NewMemory: React.FC = () => {
    const takePhotoHandler = async () => {
        const image = Camera.getPhoto( options: {
            resultType: CameraResultType.Uri,
            source: CameraSource.Camera,
            quality: 80,
            width: 500
        });
        console.log(image);
    };
```

```
<IonButton fill="clear" onClick={takePhotoHandler}>
    <IonIcon slot="start" icon={camera} />
    <IonLabel>Take Photo</IonLabel>
</IonButton>
```

4. Sesuaikan code pada file capacitor.config.json

```
{
  "appId": "com.alexwawo.ionic.memories",
  "appName": "Memories",
  "webDir": "build",
  "bundledWebRuntime": false
}
```

5. Jalankan command:

```
ionic capacitor add android
```

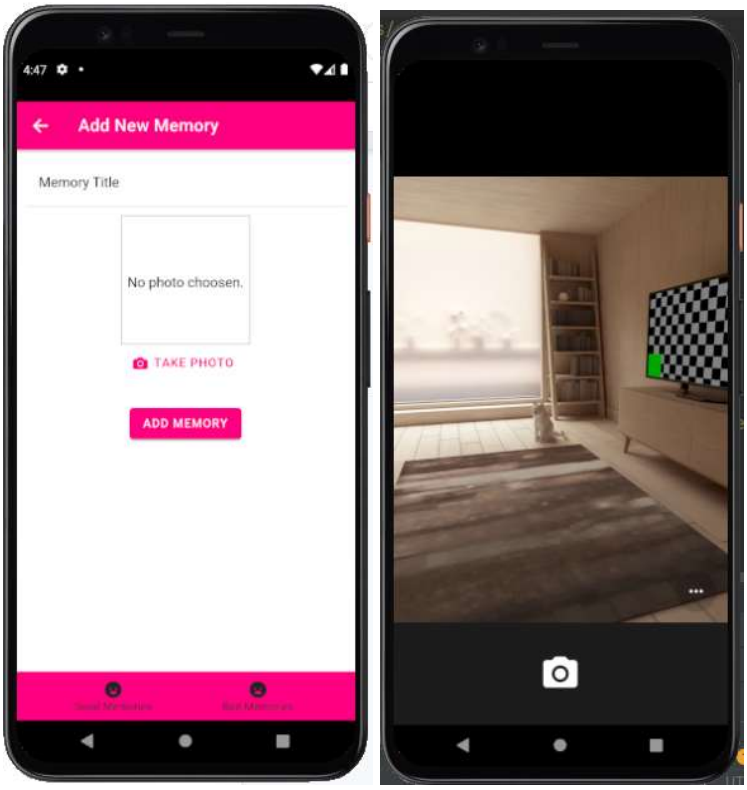
```
ionic build
```

```
capacitor copy android
```

6. Buka folder android sebagai project pada Android Studio, pastikan AndroidManifest memiliki permission berikut ini:

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

7. Coba jalankan pada Emulator, apakah berhasil membuka camera dan mengambil foto?



New Memory - Image Preview

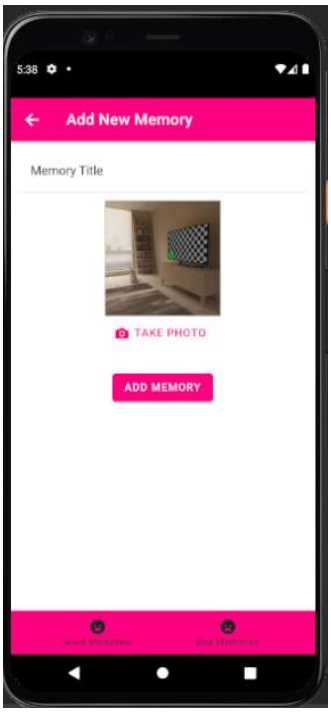
1. Modifikasi fungsi takePhotoHandler menjadi seperti berikut:

```
const [takenPhoto, setTakenPhoto] = useState<{
    path: string; // will store original URL
    preview: string // will store preview URL for web
}>();
const takePhotoHandler = async () => {
    const photo = await Camera.getPhoto( options: {
        resultType: CameraResultType.Uri,
        source: CameraSource.Camera,
        quality: 80,
        width: 500
    });
    console.log(photo);

    if(!photo || !photo.path || !photo.webPath) {
        return;
    }

    setTakenPhoto( value: {
        path: photo.path,
        preview: photo.webPath
    });
};
```

2. Modifikasi div image-preview menjadi seperti berikut:

```
<div className="image-preview">
    {!takenPhoto && <h3>No photo choosen.</h3>}
    {takenPhoto && <img src={takenPhoto.preview} alt="Preview" />}
</div>
```

3. Jalankan command "ionic capacitor copy android", kemudian jalankan kembali aplikasi melalui Android Studio, coba ambil gambar menggunakan camera, apakah hasil preview muncul?



Menyimpan Foto di FileSystem dan Mendapatkan User Input

1. Tambahkan code berikut pada NewMemory.tsx

```
import {Directory, Filesystem} from "@capacitor/filesystem";
import {base64FromPath} from "@ionic/react-hooks/filesystem";
```

```
const addMemoryHandler = async () => {
    const fileName = new Date().getTime() + '.jpeg';
    const base64 = await base64FromPath(takenPhoto!.preview);
    await Filesystem.writeFile( options: {
        path: fileName,
        data: base64,
        directory: Directory.Data
    });
};
```

```
<IonButton onClick={addMemoryHandler}>Add Memory</IonButton>
```

2. Jalankan command "ionic capacitor copy android", kemudian buka kembali project android melalui Android Studio.
3. Tambahkan code berikut pada NewMemory.tsx

```
const [chosenMemoryType, setChosenMemoryType] = useState<'good' | 'bad'>( initialState: 'good');
const titleRef = useRef<HTMLIonInputElement>( initialValue: null);

const selectMemoryTypeHandler = (event: CustomEvent) => {
    const selectedMemoryType = event.detail.value;
    setChosenMemoryType(selectedMemoryType);
};
```

```
const addMemoryHandler = async () => {
    const enteredTitle = titleRef.current?.value;
    if(!enteredTitle || enteredTitle.toString().trim().length === 0 || !takenPhoto || !chosenMemoryType) {
        return;
    }

    const fileName = new Date().getTime() + '.jpeg';
    const base64 = await base64FromPath(takenPhoto!.preview);
    await Filesystem.writeFile( options: {
        path: fileName,
        data: base64,
        directory: Directory.Data
    });
};
```

```
<IonInput type="text" ref={titleRef}></IonInput>
```

```
<IonRow>
    <IonCol>
        <IonSelect onIonChange={selectMemoryTypeHandler} value="good">
            <IonSelectOption value="good">Good Memory</IonSelectOption>
            <IonSelectOption value="bad">Bad Memory</IonSelectOption>
        </IonSelect>
    </IonCol>
</IonRow>
```

Menyimpan Data di Context

1. Buat folder data, kemudian buat file "memories-context.ts" dan MemoriesContextProvider.tsx"

2. memories-context.ts

```tsx
import React from 'react';

export interface Memory {
    id: string;
    imagePath: string;
    title: string;
    type: 'good' | 'bad';
}

const MemoriesContext = React.createContext<{
    memories: Memory[];
    addMemory: (path: string, title: string, type: 'good'|'bad') => void;
}>( defaultValue: {
    memories: [],
    addMemory: () => {}
});

export default MemoriesContext;
```

3. MemoriesContextProvider.tsx

```tsx
import React, {useState} from 'react';

import MemoriesContext, {Memory} from "./memories-context";

const MemoriesContextProvider: React.FC = props => {
    const [memories, setMemories] = useState<Memory[]>( initialState: []);
    const addMemory = (path: string, title: string, type: 'good'|'bad') => {
        const newMemory: Memory = {
            id: Math.random().toString(),
            title,
            type,
            imagePath: path
        }
        setMemories( value: curMemories => {
            return [...curMemories, newMemory];
        });
    };
    return (
        <MemoriesContext.Provider value={{memories, addMemory}}>
            {props.children}
        </MemoriesContext.Provider>
    );
}

export default MemoriesContextProvider;
```

4. Tambahkan code berikut pada NewMemory.tsx

```
const memoriesCtx = useContext(MemoriesContext);
const history = useHistory();
```

```
const addMemoryHandler = async () => {
    const enteredTitle = titleRef.current?.value;
    if(!enteredTitle || enteredTitle.toString().trim().length === 0 || !takenPhoto || !chosenMemoryType) {
        return;
    }

    const fileName = new Date().getTime() + '.jpeg';
    const base64 = await base64FromPath(takenPhoto!.preview);
    await Filesystem.writeFile( options: {
        path: fileName,
        data: base64,
        directory: Directory.Data
    });

    memoriesCtx.addMemory(fileName, enteredTitle.toString(), chosenMemoryType);
    history.length > 0 ? history.goBack() : history.replace( path: '/good-memories');
};
```

5. Tambahkan code berikut pada GoodMemories.tsx

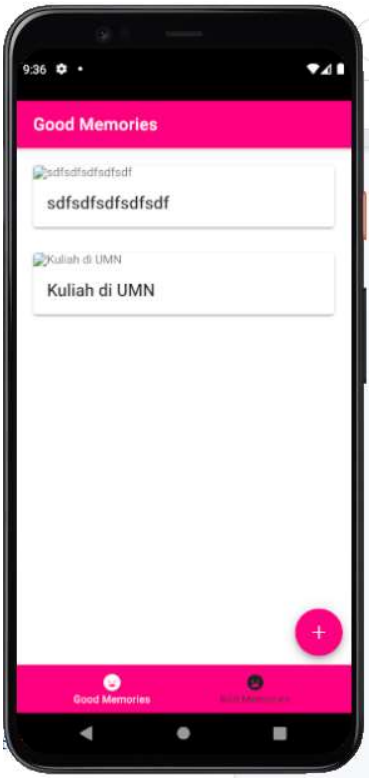
```
const memoriesCtx = useContext(MemoriesContext);
const goodMemories = memoriesCtx.memories.filter(memory => memory.type === 'good');
```

```
<IonContent>
    <IonGrid>
        {goodMemories.length === 0 && (
            <IonRow>
                <IonCol className="ion-text-center">
                    <h2>No good memories found.</h2>
                </IonCol>
            </IonRow>
        )}
        {goodMemories.map(memory => (
            <IonRow key={memory.id}>
                <IonCol>
                    <IonCard>
                        <img src={memory.imagePath} alt={memory.title} />
                        <IonCardHeader>
                            <IonCardTitle>{memory.title}</IonCardTitle>
                        </IonCardHeader>
                    </IonCard>
                </IonCol>
            </IonRow>
        ))}
    </IonGrid>
```

6. Tambahkan MemoriesContextProvider pada App.tsx

```
<IonReactRouter>
    <MemoriesContextProvider>
      <IonTabs...>
    </MemoriesContextProvider>
</IonReactRouter>
```

7. Jalankan command "ionic capacitor copy android", kemudian buka kembali project android melalui AndroidStudio, tambahkan Good Memory, apakah tampil? (catatan: gambar belum tampil)



Menampilkan Image Preview via Context

1. Tambahkan code berikut pada memories-context.ts

```
export interface Memory {
    id: string;
    imagePath: string;
    title: string;
    type: 'good' | 'bad';
    base64Url: string
}

const MemoriesContext = React.createContext<{
    memories: Memory[];
    addMemory: (path: string, base64Data: string, title: string, type: 'good'|'bad') => void;
}>( defaultValue: {
    memories: [],
    addMemory: () => {}
});
```

2. Tambahkan code berikut pada MemoriesContextProvider.tsx

```
const addMemory = (path: string, base64Data: string, title: string, type: 'good'|'bad') => {
    const newMemory: Memory = {
        id: Math.random().toString(),
        title,
        type,
        imagePath: path,
        base64Url: base64Data
    }
    setMemories( value: curMemories => {
        return [...curMemories, newMemory];
    });
};
```

3. Tambahkan code berikut pada NewMemory.tsx

```
memoriesCtx.addMemory(fileName, base64, enteredTitle.toString(), chosenMemoryType);
history.length > 0 ? history.goBack() : history.replace( path: '/good-memories');
```

4. Tambahkan code berikut pada GoodMemories.tsx

```
<img src={memory.base64Url} alt={memory.title} />
```

5. Jalankan command "ionic capacitor copy android", kemudian jalankan aplikasi, apakah berhasil menampilkan foto?

6. **Kerjakan TUGAS 1**

Menyimpan dan Mengambil Data dari Storage

1. Tambahkan code berikut pada MemoriesContextProvider.tsx

```
useEffect( effect: () => {
    const storableMemories = memories.map(memory => {
        return {
            id: memory.id,
            title: memory.title,
            imagePath: memory.imagePath,
            type: memory.type
        }
    });
    Storage.set({key: 'memories', value: JSON.stringify(storableMemories)});
}, deps: [memories]);
```

```
const initContext = useCallback( callback: async () => {
    const memoriesData = await Storage.get({key: 'memories'});
    const storedMemories = memoriesData.value ? JSON.parse(memoriesData.value) : [];
    const loadedMemories: Memory[] = [];
    for (const storedMemory of storedMemories) {
        const file = await Filesystem.readFile( options: {
            path: storedMemory.imagePath,
            directory: Directory.Data
        })
        loadedMemories.push({
            id: storedMemory.id,
            title: storedMemory.title,
            type: storedMemory.type,
            imagePath: storedMemory.imagePath,
            base64Url: '' + file.data
        });
    }
    setMemories(loadedMemories);
}, deps: []);
```

2. Tambahkan code berikut pada memories-context.ts

```
const MemoriesContext = React.createContext<{
    memories: Memory[];
    addMemory: (path: string, base64Data: string, title: string, type: 'good'|'bad') => void;
    initContext: () => void;
}>( defaultValue: {
    memories: [],
    addMemory: () => {},
    initContext: () => {}
});
```

3. Tambahkan command berikut pada App.tsx

```
const App: React.FC = () => {
    const memoriesCtx = useContext(MemoriesContext);
    const {initContext} = memoriesCtx;
    useEffect( effect: () => {
        initContext();
    }, deps: [initContext]);
    return(
  <IonApp>
      <IonReactRouter>
          {/*<MemoriesContextProvider>*/}
            <IonTabs...>
          {/*</MemoriesContextProvider>*/}
      </IonReactRouter>
  </IonApp>
)};
```

4. Tambahkan code berikut pada index.tsx

```
ReactDOM.render(
  <React.StrictMode>
      <MemoriesContextProvider>
        <App />
      </MemoriesContextProvider>
  </React.StrictMode>,
  document.getElementById( elementId: 'root')
);
```

5. Jalankan command "ionic capacitor copy android", kemudian buka kembali project android dan jalankan. Apakah memory yang tersimpan dapat di lihat kembali setelah aplikasi ditutup?
6. **Kerjakan TUGAS 2 & 3**

Native APIs in Browser

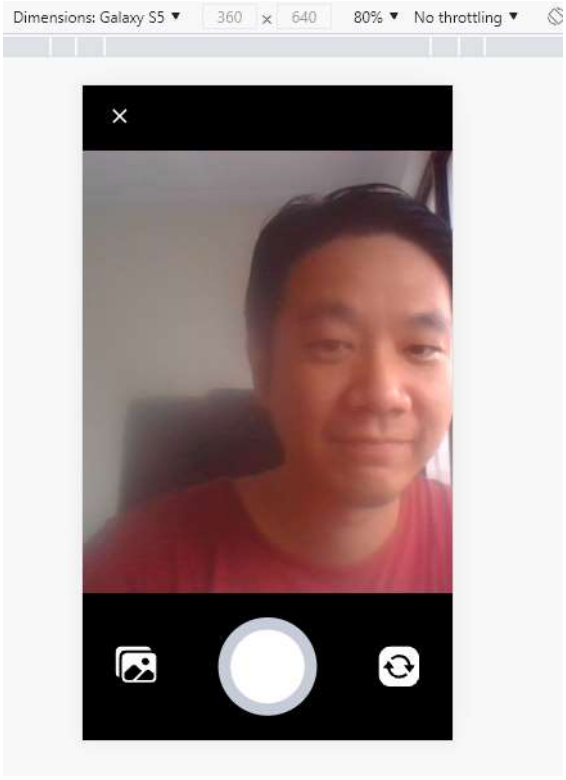
1. Jalankan command:

```
npm install --save @ionic/pwa-elements
```

2. Tambahkan code berikut pada index.tsx

```
import {defineCustomElements} from "@ionic/pwa-elements/loader";

defineCustomElements(window);
```

3. Coba jalankan aplikasi di browser, apakah berhasil membuka camera?



4. Perhatikan console log setelah mengambil gambar menggunakan camera. Preview Image tidak tampil.

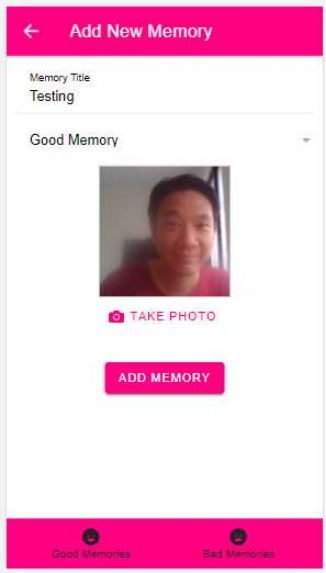
```
NewMemory.tsx:40
{webPath: 'blob:http://localhost:8100/71991c21-08a4-49f8-bb78-3973ba47f393', format: 'png', saved: false}
  format: "png"
  saved: false
  webPath: "blob:http://localhost:8100/71991c21-08a4-49f8-bb78-3973ba47…
  [[Prototype]]: Object
```

5. Lakukan modifikasi code pada NewMemory.tsx

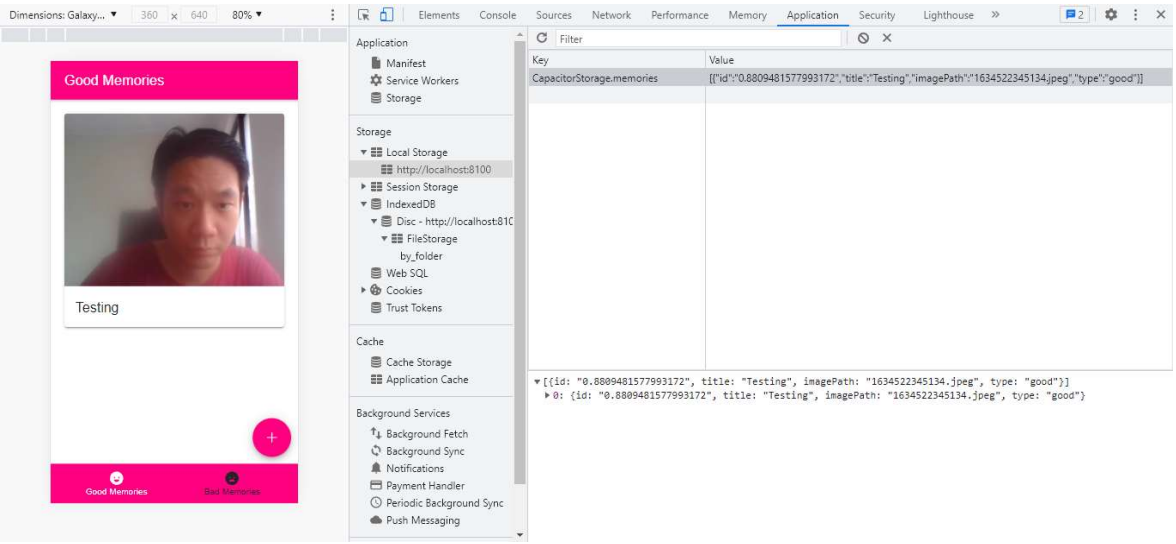
```
const [takenPhoto, setTakenPhoto] = useState<{
    path: string | undefined; // will store original URL
    preview: string // will store preview URL for web
}>();
```

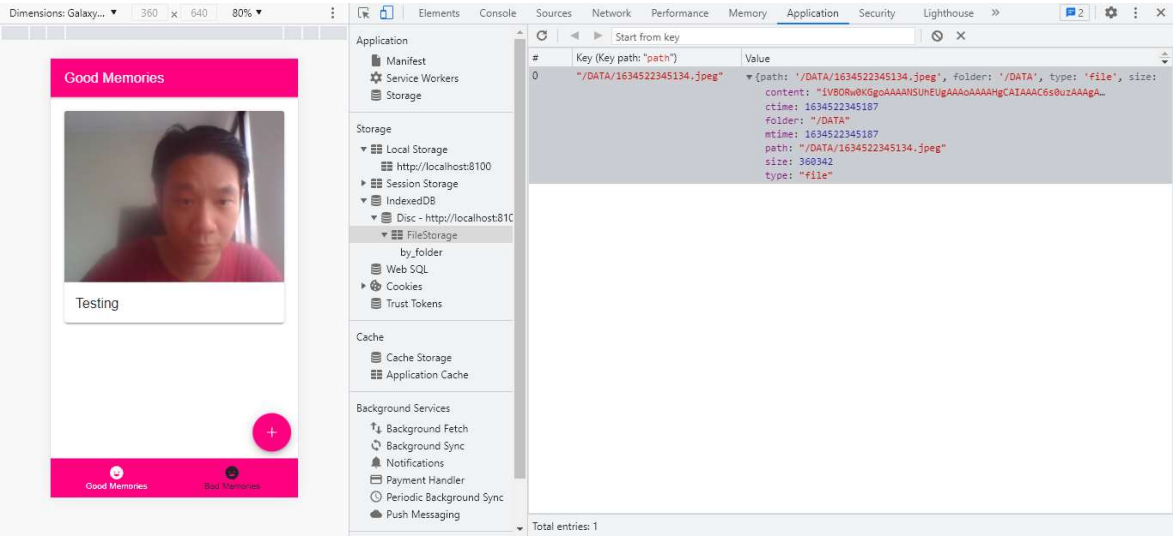
```
if(!photo || /*!photo.path ||*/ !photo.webPath) {
    return;
}
```

6. Coba kembali, apakah setelah mengambil gambar menggunakan camera, image preview muncul?



7. Coba simpan memory, kemudian refresh browser, apakah data masih ada? Perhatikan tab Application pada jendela Web Developer Tools, perhatikan Local Storage dan Indexed DB





TUGAS

1. Buat tampilan untuk page BadMemories sehingga menampilkan data Bad Memories dari context.
2. Buat komponen untuk Memory Item, sehingga bisa di re-use pada page Good dan Bad Memory
3. Buat tampilan untuk page Bad Memories sehingga menampilkan data Bad Memories dari Storage.