

OPEN EDITORS1 unsaved

JS App.js src

JS studentSlice.js src\redux

JS StudentPage.js src\pages\studentPage

PRO1

node_modules

public

src

components

pages

home

studentPage

JS StudentPage.js

redux

store.js

JS studentSlice.js

App.css

App.js

index.js

package-lock.json

package.json

README.md

src > redux > JS studentSlice.js > addNewStudent > createAsyncThunk('student/addNewStudent') callback

11 export const getAll= createAsyncThunk('student/getAll', async ({ currentPage, limit },thunkAPI) => {
20 }
21 });
22 export const addNewStudent= createAsyncThunk('student/addNewStudent', async (student,thunkAPI) => {
23 const url= '/student';
24 try {
25 const response = await axios.post(url,student);
26 return response.data; // Trả về dữ liệu từ phản hồi
27 } catch (error) {
28 return thunkAPI.rejectWithValue(error.response.data); // Trả về lỗi nếu có
29 }
30 });
31
32
33
34 const studentSlice = createSlice({
35 name: 'student',
36 initialState: {
37 status: 'idle',
38 error: null,
39 students:null,
40 totalPages:10
41 },
42 reducers: {
43
44 },
45 extraReducers: (builder) => {
46 builder
47 .addCase(getAll.fulfilled, (state, action) => {
48 state.students = action.payload.data.studentResonseList
49 state.totalPages = action.payload.data.totalPages
50 })
51 .addCase(addNewStudent.fulfilled, (state, action) => {
52
53 })
54 };
55 },
56 });
57

PROBLEMS OUTPUT DEBUGCONSOLE TERMINAL

```

return (
  <div>
    <Header />
    <Button onClick={toggle} className='btn btn-success'>Add new student</Button>
    <h1>Student page</h1>
    <Student />
    <Modal isOpen={modal} toggle={toggle}>
      <ModalHeader toggle={toggle}>Add new student</ModalHeader>
      <ModalBody>
        <FormGroup>
          <Label for="exampleEmail">
            Họ tên
          </Label>
          <Input
            id="exampleEmail"
            name="ten"
            placeholder="Họ tên"
            type="text"
            value={student.ten}
            onChange={handleChange}
          />
        </FormGroup>
        { ' ' }
        <FormGroup>
          <Label for="exampleEmail">
            Thành phố
          </Label>
          <Input
            id="exampleEmail"
            name="thanhPho"
            placeholder="Họ tên"
            type="text"
            value={student.thanhPho}
            onChange={handleChange}
          />
        </FormGroup>
        <FormGroup>

```

```
<FormGroup>
  <Label for="exampleSelect">
    | Xếp loại
  </Label>
  <Input
    id="exampleSelect"
    name="xepLoai"
    type="select"
    value={student.xepLoai}
    onChange={handleChange}
  >
    <option>
      | Giỏi
    </option>
    <option>
      | Khá
    </option>
    <option>
      | Trung bình
    </option>
    <option>
      | Yếu
    </option>
  </Input>
</FormGroup>
<FormGroup>
  <Label for="exampleDate">
    | Date
  </Label>
  <Input
    id="exampleDate"
    name="ngaySinh"
    placeholder="date placeholder"
    type="date"
    value={student.ngaySinh}
    onChange={handleChange}
  />
</FormGroup>
```

OUTPUT DEBUG CONSOLE TERMINAL

```

export default function StudentPage() {
  <option>
    Trung bình
  </option>
  <option>
    Yếu
  </option>
</Input>
</FormGroup>
<FormGroup>
  <Label for="exampleDate">
    Date
  </Label>
  <Input
    id="exampleDate"
    name="ngaySinh"
    placeholder="date placeholder"
    type="date"
    value={student.ngaySinh}
    onChange={handleChange}
  />
</FormGroup>
</ModalBody>
<ModalFooter>
  <Button color="primary" onClick={handle_add}>
    Save
  </Button>{' '}
  <Button color="secondary" onClick={toggle}>
    Cancel
  </Button>
</ModalFooter>
</Modal>
</div>
)
}

```

Sửa code:

EXPLORER

OPEN EDITORS

App.js src

studentSlice.js src/redux

StudentPage.js src/pages/studentPage

Student.js src/components/student

PROJ

node_modules

public

src

components

header

student

Student.js

pages

home

studentPage

StudentPage.js

redux

store.js

studentSlice.js

App.css

App.js

index.js

package-lock.json

package.json

README.md

JS App.js

JS studentSlice.js X

JS StudentPage.js

JS Student.js

src > redux > JS studentSlice.js > [0] studentSlice

11 export const getAll= createAsyncThunk('student/getAll', async ({ currentPage, limit },thunkAPI) => {

17

18 } catch (error) {

19 return thunkAPI.rejectWithValue(error.response.data); // Trả về lỗi nếu có

20 }

21 });

22 export const addNewStudent= createAsyncThunk('student/addNewStudent', async (student,thunkAPI) => {

23 const url= BASE_URL+'/' +student';

24 try {

25 console.log(student)

26 const response = await axios.post(url, student);

27

28 return response.data; // Trả về dữ liệu từ phản hồi

29 } catch (error) {

30 return thunkAPI.rejectWithValue(error.response.data); // Trả về lỗi nếu có

31 }

32 });

33

34

35

36 const studentSlice = createSlice({

37 name: 'student',

38 initialState: {

39 status: 'idle',

40 error: null,

41 students:null,

42 totalPages:10

43 },

44 reducers: {

45

46 },

47 extraReducers: (builder) => {

48 builder

49 .addCase(getAll.fulfilled, (state, action) => {

50 state.students = action.payload.data.studentResonseList

51 state.totalPages = action.payload.data.totalPages

52 })

53 .addCase(addNewStudent.fulfilled, (state, action) => {

54

55 })

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL



p.js JS studentSlice.js JS StudentPage.js X JS Student.js

pages > studentPage > JS StudentPage.js > StudentPage > handleChange > setStudent() callback > [name]

```

import { Button, FormGroup, Input, Label, Modal, ModalBody, ModalFooter, ModalHeader } from 'reactstrap';
import { useDispatch } from 'react-redux';
import { addNewStudent } from "../../redux/studentSlice";

export default function StudentPage() {
  const [modal, setModal] = useState(false);
  const [student, setStudent] = useState({
    ten: "Lê Mèo",
    thanhPho: "meo",
    xepLoai: "Giỏi",
    ngaySinh: "01-08-2000" // Initial format: DD-MM-YYYY
  });

  const dispatch = useDispatch();

  const toggle = () => setModal(!modal);

  const handle_add = () => {
    toggle();
    console.log(student)
    dispatch(addNewStudent(student));
  };

  const handleChange = (e) => {
    const { name, value } = e.target;

    if (name === 'ngaySinh') {

      setStudent(prevStudent => ({
        ...prevStudent,
        [name]: convertDateToDDMMYYYY([value])
      }));
    } else {
      setStudent(prevStudent => ({
        ...prevStudent,
        [name]: value
      }));
    }
  };
}

```

```
const handleChange = (e) => {
  const { name, value } = e.target;

  if (name === 'ngaySinh') {
    setStudent(prevStudent => ({
      ...prevStudent,
      [name]: convertDateToDDMMYYYY(value)
    }));
  } else {
    setStudent(prevStudent => ({
      ...prevStudent,
      [name]: value
    }));
  }
};

const convertDateToYYYYMMDD = (date) => {
  const [day, month, year] = date.split('-');
  return `${year}-${month}-${day}`;
};

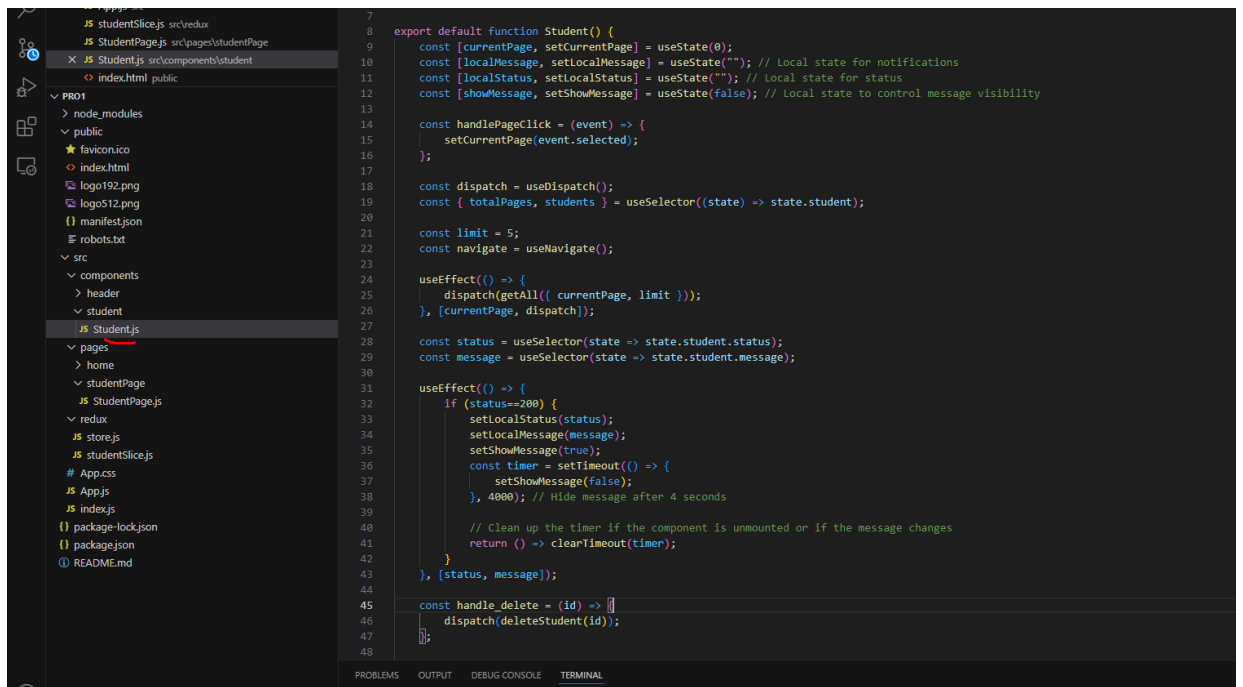
const convertDateToDDMMYYYY = (date) => {
  const [year, month, day] = date.split('-');
  return `${day}-${month}-${year}`;
};
```

```

return (
  <div>
    <Header />
    <Button onClick={toggle} className='btn btn-success'>Add new student</Button>
    <h1>Student page</h1>
    <Student />
    <Modal isOpen={modal} toggle={toggle}>
      <ModalHeader toggle={toggle}>Add new student</ModalHeader>
      <ModalBody>
        <FormGroup>...
      </FormGroup>
      <FormGroup>...
      </FormGroup>
      <FormGroup>...
      </FormGroup>
      <FormGroup>...
      <Label for="ngaySinh">Ngày sinh</Label>
      <Input
        id="ngaySinh"
        name="ngaySinh"
        type="date"
        value={convertDateToYYYYMMDD(student.ngaySinh)} // Convert date format for input
        onChange={handleChange}
      />
      </FormGroup>
    </ModalBody>
    <ModalFooter>
      <Button color="primary" onClick={handle_add}>
        Save
      </Button>{' '}
      <Button color="secondary" onClick={toggle}>
        Cancel
      </Button>
    </ModalFooter>
  </Modal>
</div>

```

Chức năng delete

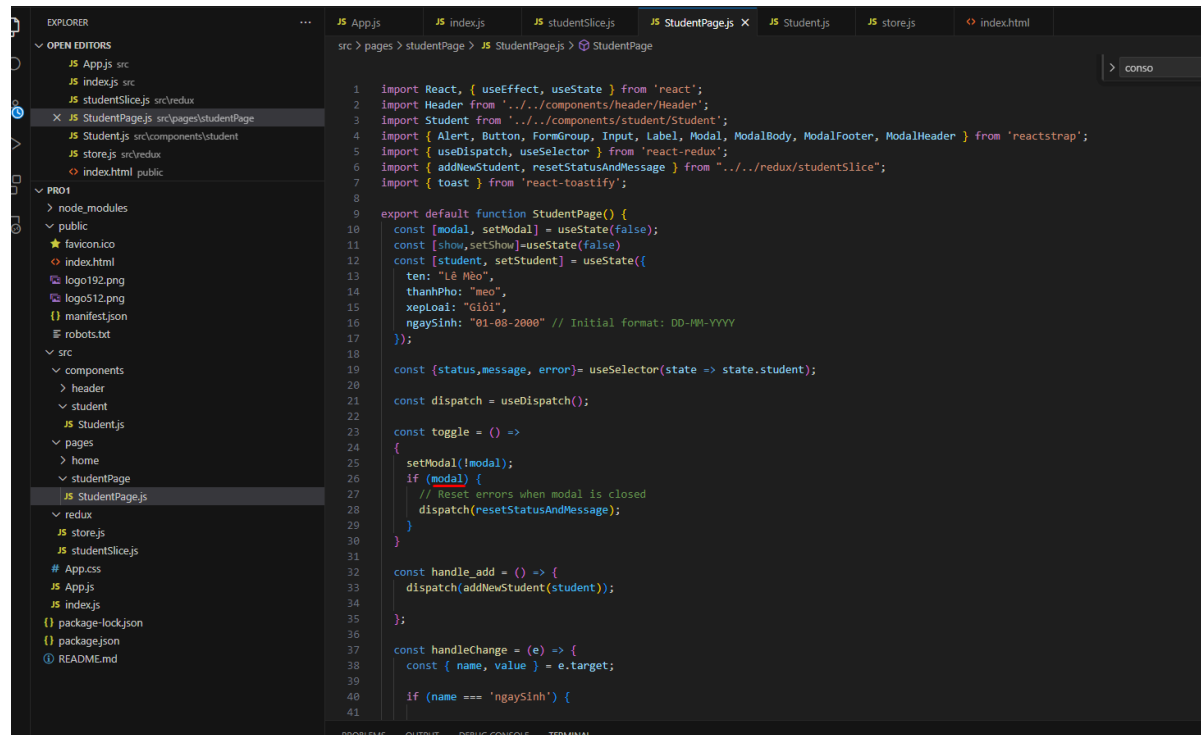
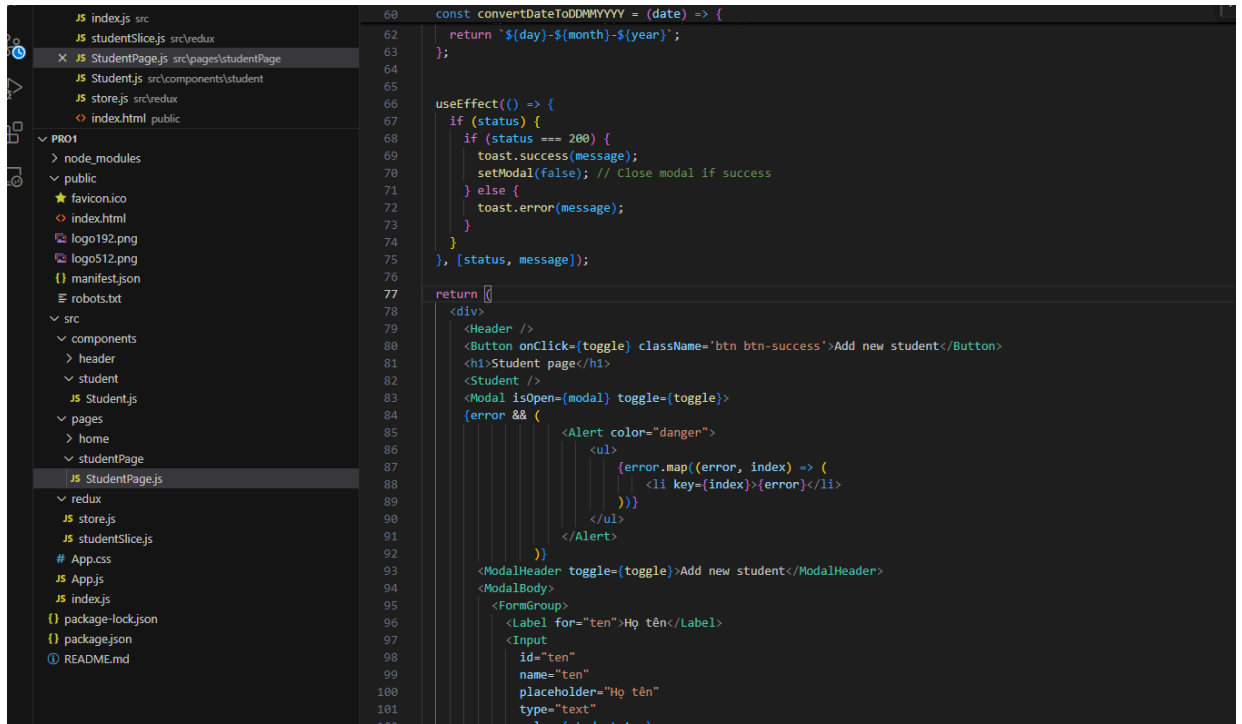


```
7
8
9 export default function Student() {
10   const [currentPage, setCurrentPage] = useState(0);
11   const [localMessage, setLocalMessage] = useState(""); // Local state for notifications
12   const [localStatus, setLocalStatus] = useState(""); // Local state for status
13   const [showMessage, setShowMessage] = useState(false); // Local state to control message visibility
14
15   const handlePageClick = (event) => {
16     setCurrentPage(event.selected);
17   };
18
19   const dispatch = useDispatch();
20   const { totalPages, students } = useSelector((state) => state.student);
21
22   const limit = 5;
23   const navigate = useNavigate();
24
25   useEffect(() => {
26     dispatch(getAll({ currentPage, limit }));
27   }, [currentPage, dispatch]);
28
29   const status = useSelector(state => state.student.status);
30   const message = useSelector(state => state.student.message);
31
32   useEffect(() => {
33     if (status !== 200) {
34       setLocalStatus(status);
35       setLocalMessage(message);
36       setShowMessage(true);
37       const timer = setTimeout(() => {
38         setShowMessage(false);
39       }, 4000); // Hide message after 4 seconds
40
41       // Clean up the timer if the component is unmounted or if the message changes
42       return () => clearTimeout(timer);
43     }
44   }, [status, message]);
45
46   const handle_delete = (id) => {
47     dispatch(deleteStudent(id));
48   };
49 }
```

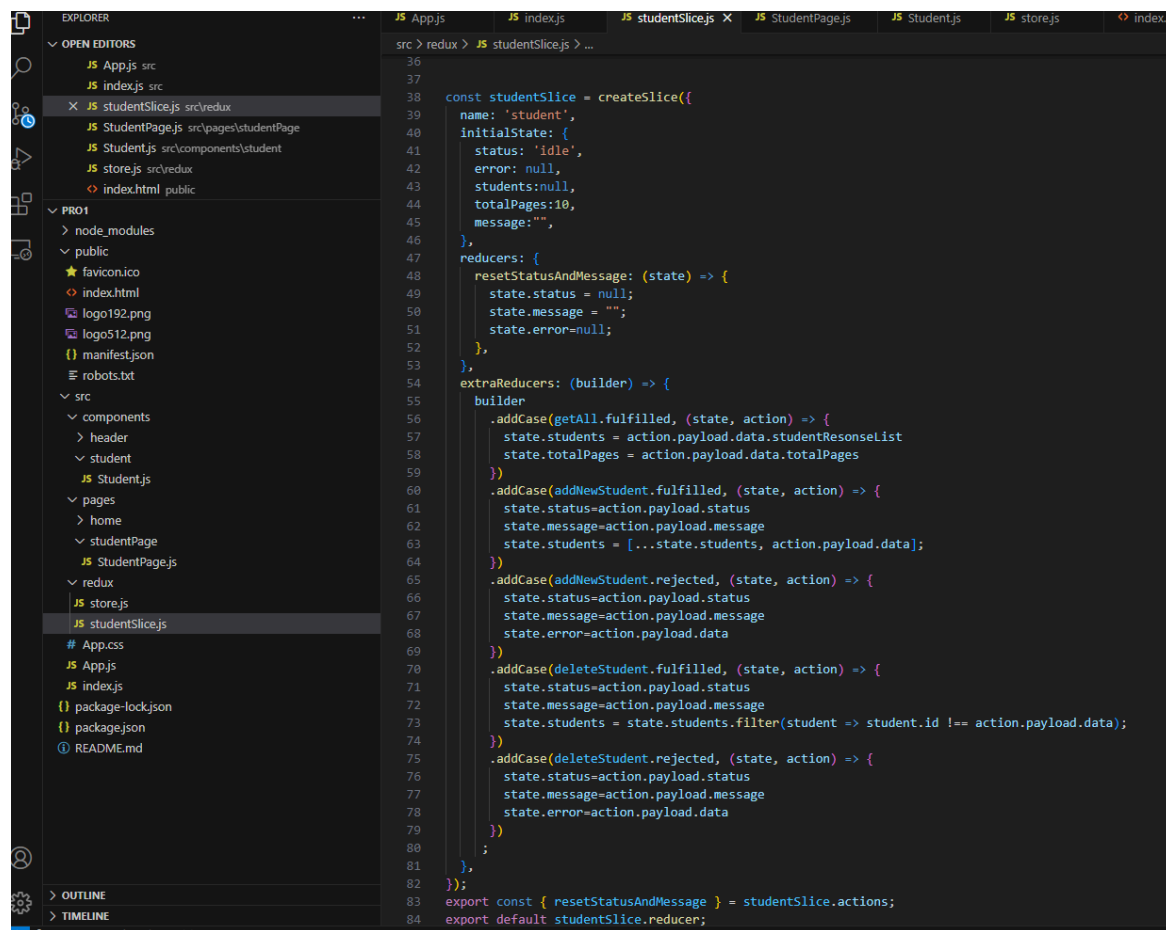
Cách hiển thị mess(lỗi, thành công)

```
src > components > student > JS Student.js > Student > handle_delete
8  export default function Student() {
16
17    const limit = 5;
18    const navigate = useNavigate();
19
20    useEffect(() => {
21      dispatch(getAll({ currentPage, limit }));
22    }, [currentPage, dispatch]);
23
24    useEffect(() => {
25      if (status && message) {
26        setShowMessage(true);
27
28        const timer = setTimeout(() => {
29          setShowMessage(false);
30          dispatch(resetStatusAndMessage()); // Reset status and message
31        }, 2000);
32
33        return () => clearTimeout(timer);
34      }
35    }, [status, message, dispatch]);
36
37    const handle_delete = (id) => {
38      dispatch(deleteStudent(id));
39    };
40
41    return (
42      <div className="products">
43        <Container>
44          {showMessage && (
45            <Alert color={status === 200 ? "success" : "danger"}>
46              {message}
47            </Alert>
48          )}
49
50          <Table hover>
51            <thead>
52              <tr>
53                <th>#</th>
54                <th>Student ID</th>
55                <th>Student Name</th>
56                <th>Delete</th>
57              </tr>
58            </thead>
59            <tbody>
60              <tr>
61                <td>1</td>
62                <td>1</td>
63                <td>1</td>
64                <td>1</td>
65              </tr>
66            </tbody>
67          </Table>
68        </Container>
69      </div>
70    );
71  }
72}
```

cách hiển thị danh sách error



student reducer



```
36
37
38 const studentSlice = createSlice({
39   name: 'student',
40   initialState: {
41     status: 'idle',
42     error: null,
43     students: null,
44     totalPages: 10,
45     message: '',
46   },
47   reducers: {
48     resetStatusAndMessage: (state) => {
49       state.status = null;
50       state.message = '';
51       state.error = null;
52     },
53   },
54   extraReducers: (builder) => {
55     builder
56       .addCase(getAll.fulfilled, (state, action) => {
57         state.students = action.payload.data.studentResonseList;
58         state.totalPages = action.payload.data.totalPages;
59       })
60       .addCase(addNewStudent.fulfilled, (state, action) => {
61         state.status = action.payload.status;
62         state.message = action.payload.message;
63         state.students = [...state.students, action.payload.data];
64       })
65       .addCase(addNewStudent.rejected, (state, action) => {
66         state.status = action.payload.status;
67         state.message = action.payload.message;
68         state.error = action.payload.data;
69       })
70       .addCase(deleteStudent.fulfilled, (state, action) => {
71         state.status = action.payload.status;
72         state.message = action.payload.message;
73         state.students = state.students.filter(student => student.id !== action.payload.data);
74       })
75       .addCase(deleteStudent.rejected, (state, action) => {
76         state.status = action.payload.status;
77         state.message = action.payload.message;
78         state.error = action.payload.data;
79       })
80   },
81 });
82
83 export const { resetStatusAndMessage } = studentSlice.actions;
84 export default studentSlice.reducer;
```

```
import { toast } from 'react-toastify';
```

```
=> import { toast } from  
'react-toastify';
```

Link source code frontend:

https://drive.google.com/drive/folders/18ognxK0HcSvtz_BE0dNzQ5eH0ahKS8MO?usp=sharing