**File Upload Notes**

**----------------------------**

**FIleUpload Field**

---------------------------

*<input  
 accept="multipart/form-data"  
 id="raised-button-file"  
 type="file"  
 />  
 <label htmlFor="raised-button-file">  
 <MButton variant="contained" size="medium" color="primary" style={{margin: '5px'}} onClick={this.handleUpload}>  
 upload  
 </MButton>  
 </label>*

**Fetch the File from Form , sending file to Server**

======================================

const formData = new FormData();

const fileField = document.querySelector('input[type="file"]');

formData.append('username', 'abc123');

formData.append('avatar', fileField.files[0]);

try {

const response = await fetch('https://example.com/profile/avatar', {

method: 'PUT',

body: formData

});

const result = await response.json();

console.log('Success:', JSON.stringify(result));

} catch (error) {

console.error('Error:', error);

}

handleUpload(event){  
  
 **const** fileField = ***document***.querySelector(**'input[type="file"]'**);  
 **let** osPackageObj = **this**.**state**.newOSPackage;  
 osPackageObj.**ISO\_http\_path** = fileField.**files**[0];  
 ***console***.log(**"HTTPPath:"**,fileField.**files**[0]);  
 **this**.setState({  
 **osPackageDetails**: osPackageObj  
 })  
}

<https://stackoverflow.com/questions/36067767/how-do-i-upload-a-file-with-the-js-fetch-api>

**An important note for sending Files with Fetch API**

One needs to omit content-type header for the Fetch request. Then the browser will automatically add the Content type header including the Form Boundary which looks like

Content-Type: multipart/form-data; boundary=—-WebKitFormBoundaryfgtsKTYLsT7PNUVD

Form boundary is the delimiter for the form data

