File upload (klijent)

1. Koristiti `template.zip`
2. Dodati `change` handler na kontrolu forme za fajl, promijeniti `type="file"` te ukloniti `formControlName`

```
<input name="file" type="file" class="form-control-
file" (change)="onFileChange($event)">
```

3. Tijelo `AppComponent` komponente:

```
onFileChange(event: Event) {
  const file = (event.target as any).files[0];
  this.form.patchValue({ file });
}


percentage: number = 0;
progressHandler(percentage: number, done: boolean) {
  if (!isNaN(percentage))
    this.percentage = percentage;

  if (done) {
    console.log("Successfully uploaded");
    this.form.reset();
    setTimeout(() => {
      this.percentage = 0;
    }, 2000);
  }
}
```

4. U predlošku `AppComponent`:

```
<p class="w-50">
  <ngb-progressbar type="primary" [value]="percentage"></ngb-progressbar>
</p>
```

5. Također, u `AppComponent` promijeniti `submit` (pokazati prvo bez `bind`). Isto tako, umjesto poziva `bind`, `progressHandler` može biti i lambda funkcija pa neće biti greške.

```
  submit() {
    this.filesService
      .upload(this.form.value, this.progressHandler.bind(this))
      .subscribe();
  }
```

6. Sukladno, promijeniti `upload` u `FilesService`-u:

```
upload(file: any, handler: (percentage: number, done: boolean) => void): Observab
le<any> {
  return this.http.post(this.url, file)
    .pipe(
      tap((res: any) => {
        console.log(res);
        handler(100, true);
      })
    );
}
```

7. Na serveru, dodati `setTimeout`:

```
    app.post("/api/files/", (req, res) => {
        console.log(req.body);
        setTimeout(() => {
            res.send({ message: "Received" });
        }, 2000)
    });
```

8. Opcionalno u `AppComponent`:
   - `#fileinput`
   - `@ViewChild("fileinput") fi: ElementRef = new ElementRef<any>(null);`
   - `this.fi.nativeElement.value = ""` prilikom submit-a

9. `import { HttpClient, HttpEventType, HttpRequest } from '@angular/common/http';`

10. Dodati u `FilesService`:

```
  uploadStatus(event: any) {
    if (event.body) {
      // upload je gotov, pridodaj novi fajl
    }
    const percentage = Math.round(100 * event.loaded / event.total);
    const done = event.type === HttpEventType.Response;
    return {percentage, done};
  }
```

11. Promijeniti `upload` u `FilesService`-u:

```
upload(data: { description: string, file: any }, handler: (percentage: number, do
ne: boolean) => void): Observable<any> {
    let formData: FormData = new FormData();
    formData.append("file", data.file, data.file.name);
    formData.append("description", data.description);
    const req = new HttpRequest("POST", this.url, formData,
      { reportProgress: true }
    );


    return this.http.request(req)
      .pipe(
        map(event => this.uploadStatus(event)),
        tap((status: { percentage: number, done: boolean }) => {
          handler(status.percentage, status.done)
        }),
        last()
      );
  }
```

12. Na serveru pokazati da se upload poziva. Skoro pa smo s klijentskim dijelom uploada.

File upload (server, multer)

1. `npm install multer`

2. Unijeti i inicijalizirati multer:

```
const multer = require("multer");
const upload = multer({ dest: "./uploads/" });
```

3. Dodati middleware koji će parsirati ono što je klijent poslao:

```
app.post("/api/files/", upload.single("file"), async (req, res) => {
    console.log(req.file);
    const db_res = await db.collection("files").insertOne(req.file);
    req.file["_id"] = db_res.insertedId;
    res.send({ message: "Successfully uploaded", file: req.file });
});
```

4. Pokazati neku datoteku u `server/uploads/`

5. U scope-u `file.service.ts`:

```
files = signal<any[]>([]);
```

6. Modificirati `uploadStatus`:

```
if (event.body) {
  // upload je gotov, pridodaj novi fajl
  const newFile = event.body.file;
  this.files.update(files => [...files, newFile])
}
```

Dohvaćanje svega & download

1. Get za sve datoteke u `index.js`:

```
app.get("/api/files", async (req, res) => {
    const files = await db.collection("files")
        .find().toArray();
    res.send(files);
});
```

2. Get na klijentu, `FilesService`:

```
getFiles(): Observable<any[]> {
  return this.http.get<any[]>(this.url)
    .pipe(
      tap(
        (files: any[]) => {
          this.files.set(files);
        }
      )
    )
}

constructor(public http: HttpClient) {
  this.getFiles().subscribe();
}
```

3. Get na klijentu, predložak `AppComponent` (prvo bez href):

```
<ul>
  @for(file of filesService.files(); track file._id) {
    <li>
      <a href="{{ '/api/files/' + file._id }}">
        {{ file.originalname }}
      </a>
    </li>
  }
</ul>
```

4. Download, server:

```
app.get("/api/files/:_id", async (req, res) => {
    const _id = mongodb.ObjectId.createFromHexString(req.params._id);
    const file = await db.collection("files").findOne({ _id });
    res.download(file.path, file.originalname);
});
```