AMCBRIDGE
Engineering Software Development Made Easy

3D files browser

By: Mykola Golovach

3D files browser

- 3 months
- Web-based 3D files browser, with authentication and admin tools where admins can manage 3D files and users.

Implemented Functions

Lab Work 1

- Implementation of Files Storage Server;
- Ready Files Storage has ability to:
  - upload;
  - update;
  - get;
  - delete files.

Lab Work 2

- Creation of users management, which consists of server API and client part;
- The app has a web interface with the ability to:
  - view users list;
  - create;
  - update;
  - remove users.

Implemented Functions

Lab Work 3

- Creation of models management, that consists of server API and client part;
- The app should has a web interface with the ability to manipulate models like creating, updating and deleting;

Lab Work 4

- Implement authentication and authorization;
- The app should has a web interface with the ability to login user ;
- Implement a permission to create/update/delete users only for Admin role;

Technologies Used

Laboratory work 1

node

yarn

Express JS

mongoDB

Laboratory work 2

ASP.NET Core

KARMA

ANGULAR

Entity Framework Core

Technologies Used

AMCBRIDGE
Engineering Software Development Made Easy

Laboratory work 3

ASP.NET Core

KARMA

ANGULAR

Entity Framework Core

Laboratory work 4

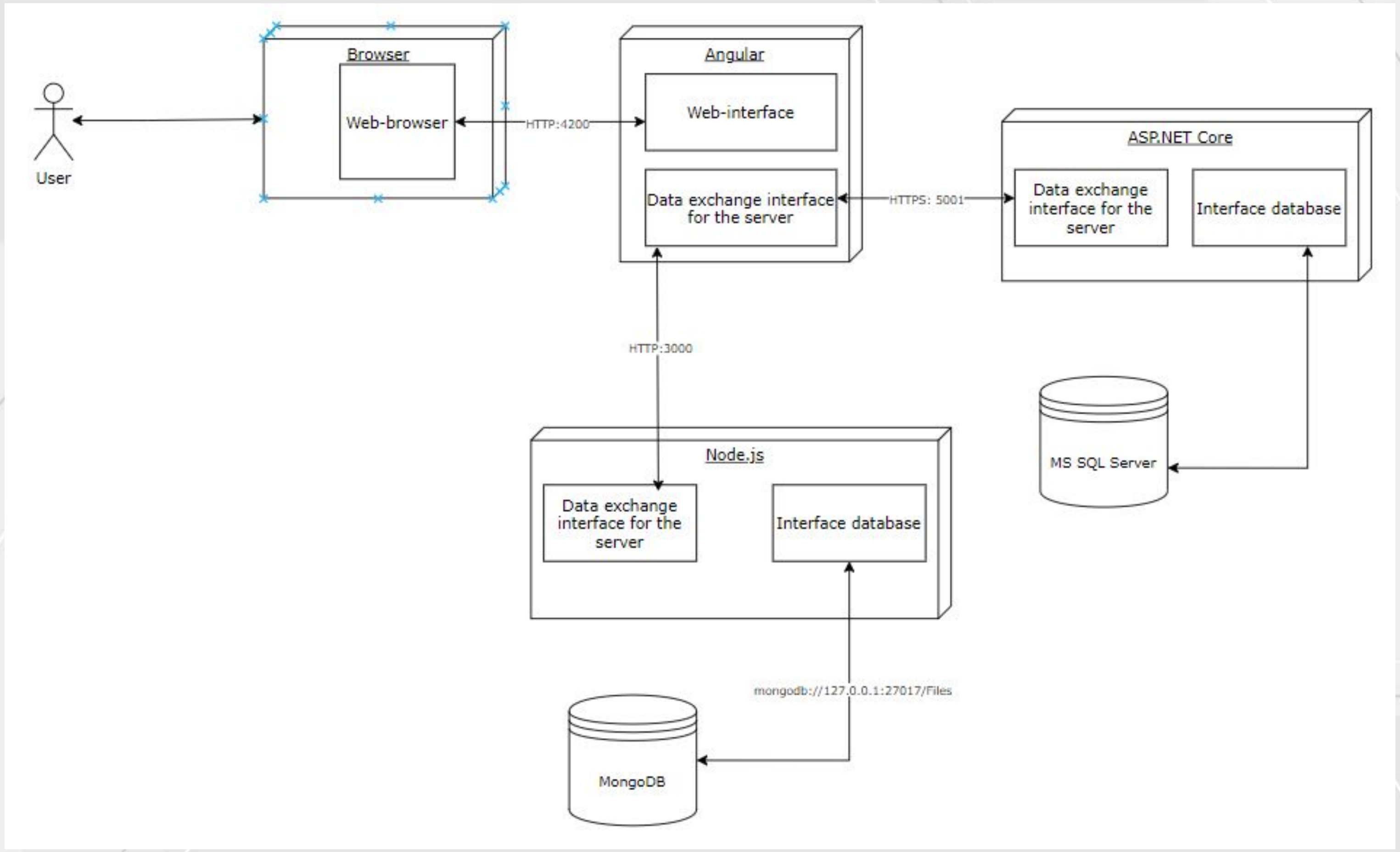
ASP.NET Core

Express JS

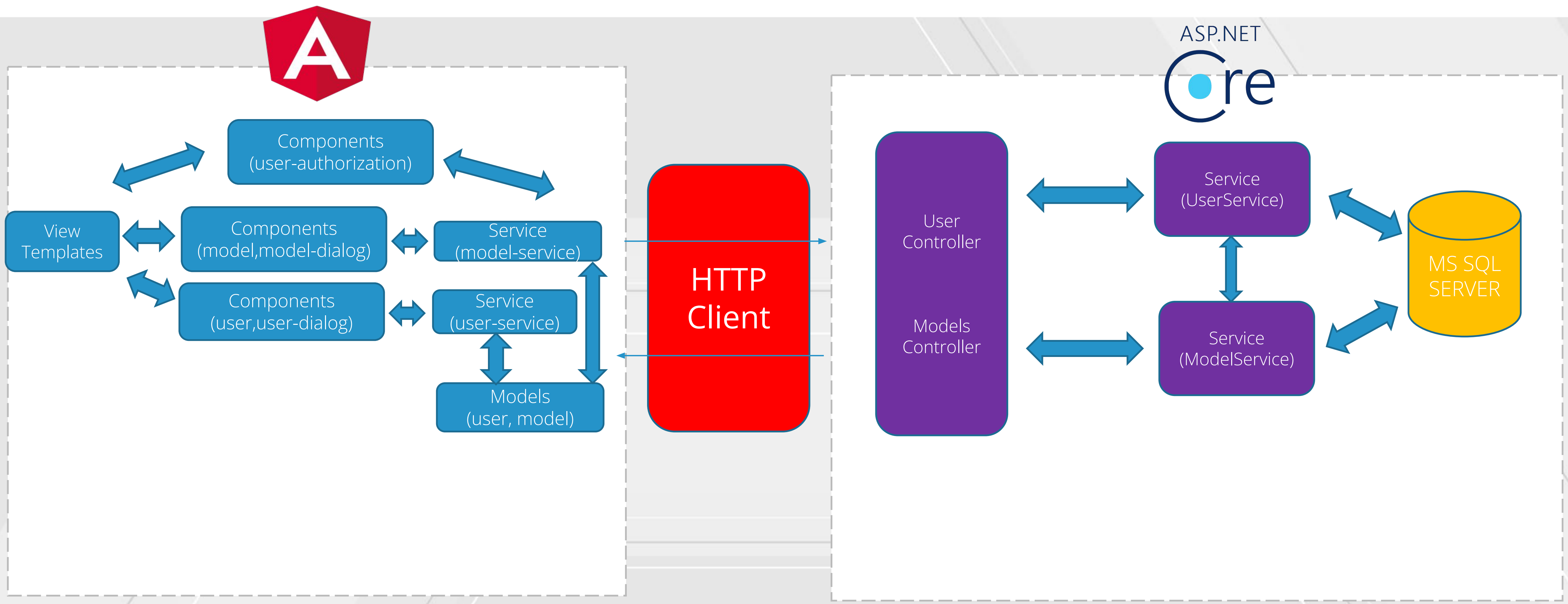
ANGULAR

node JS

Deployment diagram



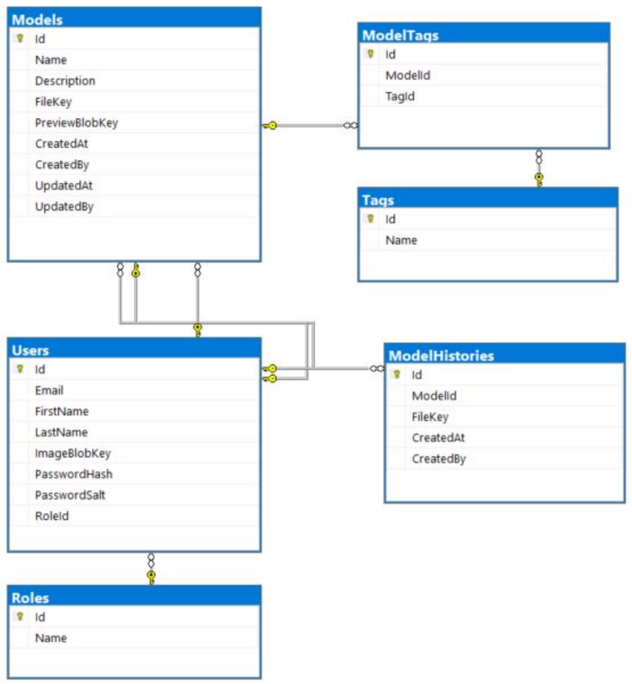
App Architecture



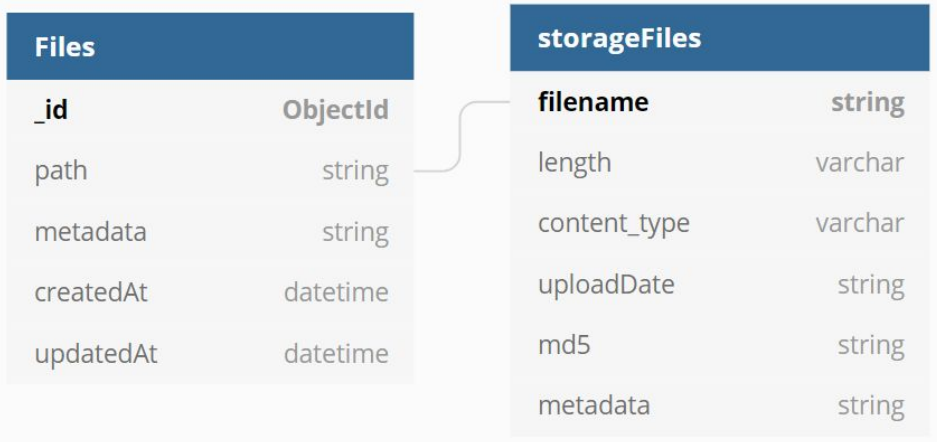
Database relationships

Engineering Software Development Made Easy

Microsoft SQL Server



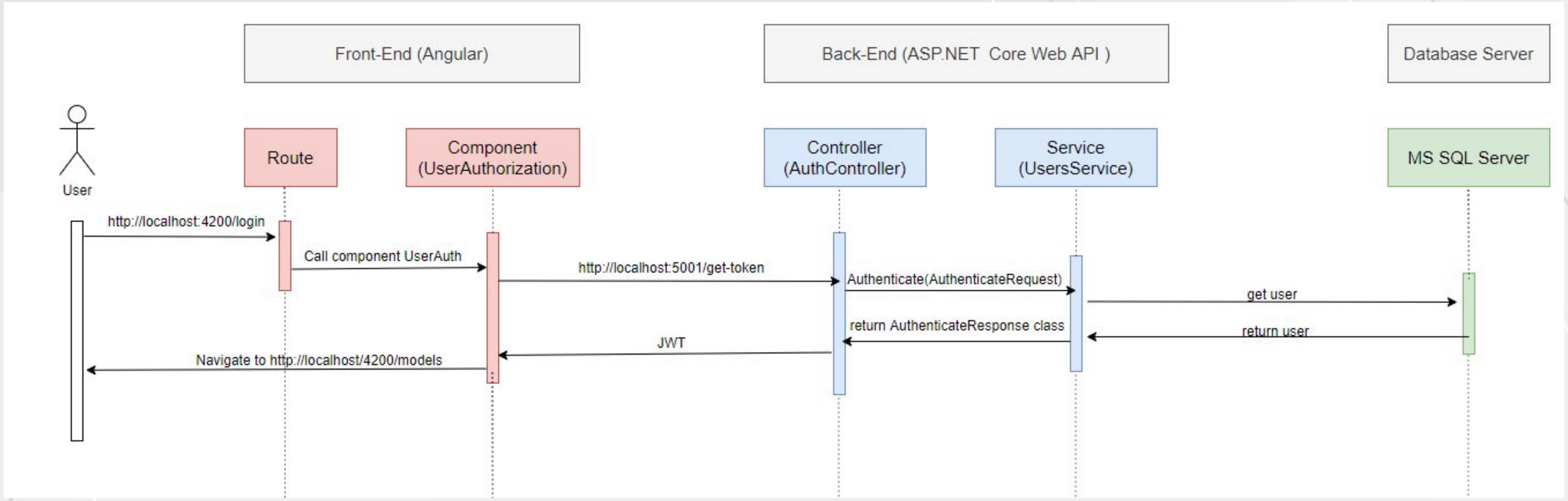
mongoDB



Files	
_id	ObjectId
path	string
metadata	string
createdAt	datetime
updatedAt	datetime

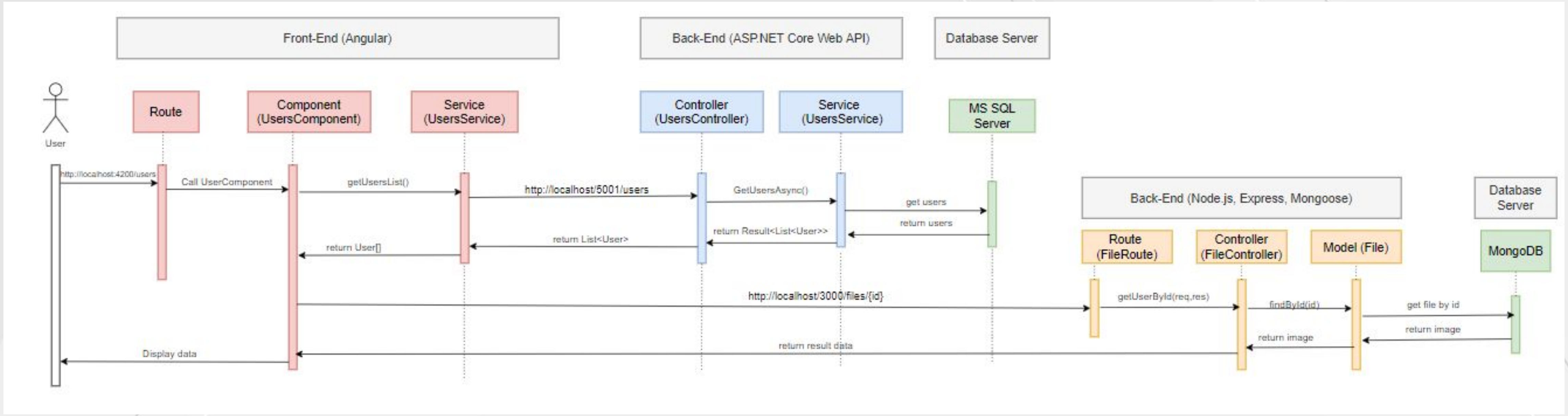
storageFiles	
filename	string
length	varchar
content_type	varchar
uploadDate	string
md5	string
metadata	string

Sequence diagram (login page)

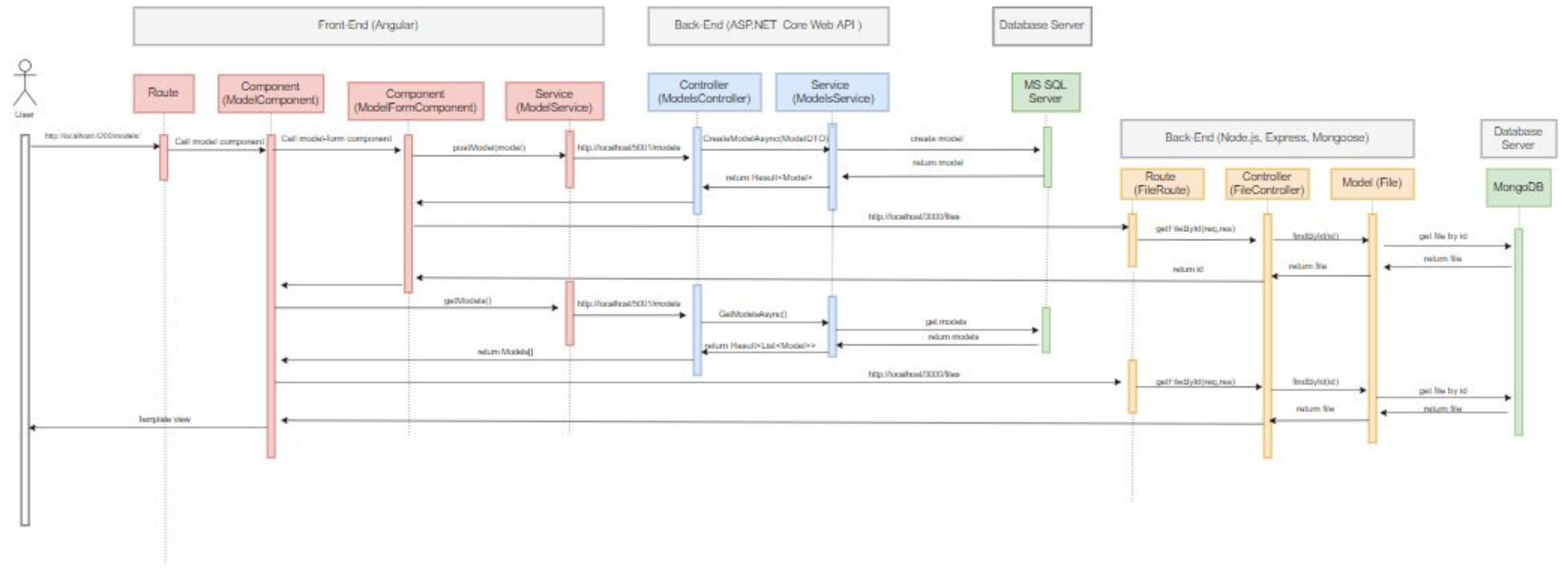


Front-End (Angular)
Back-End (ASP.NET Core Web API)
Database Server
User
Route
Component (UserAuthorization)
Controller (AuthController)
Service (UsersService)
MS SQL Server
http://localhost:4200/login
Call component UserAuth
http://localhost:5001/get-token
Authenticate(AuthenticateRequest)
get user
return user
return AuthenticateResponse class
JWT
Navigate to http://localhost/4200/models

Sequence diagram (user page)

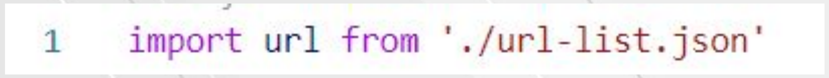


Sequence diagram (create model)



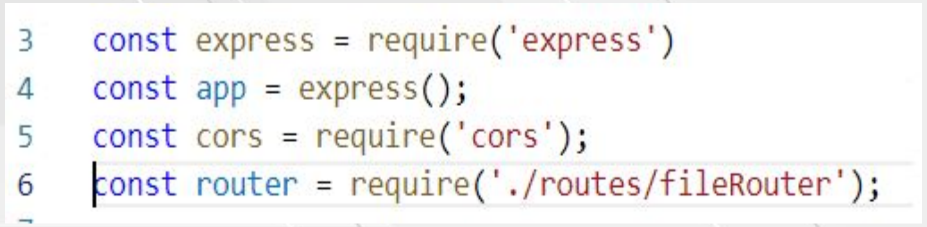
Nodejs & express

1. Import URL of database server



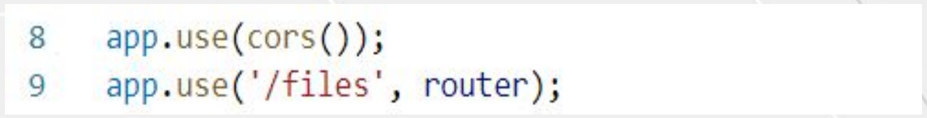
```
import url from './url-list.json'
```

2. Connect express, routers and create application object



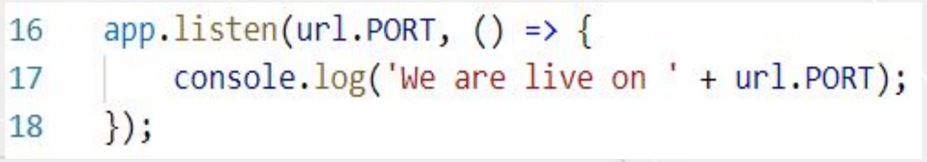
```
const express = require('express')
const app = express();
const cors = require('cors');
const router = require('./routes/fileRouter');
```

3. Install routers to handle all requests along the route (/file)



```
app.use(cors());
app.use('/files', router);
```

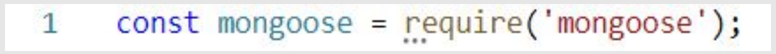
4. Setup connection through port(3000)



```
app.listen(url.PORT, () => {
    console.log('We are live on ' + url.PORT);
});
```

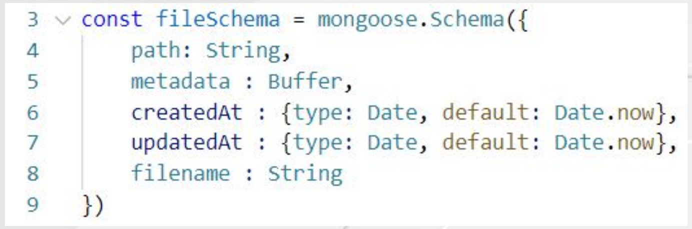
mongoDB & mongoose

1. Connecting mongoose



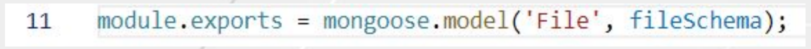
```
const mongoose = require('mongoose');
```

2. Creating Schema



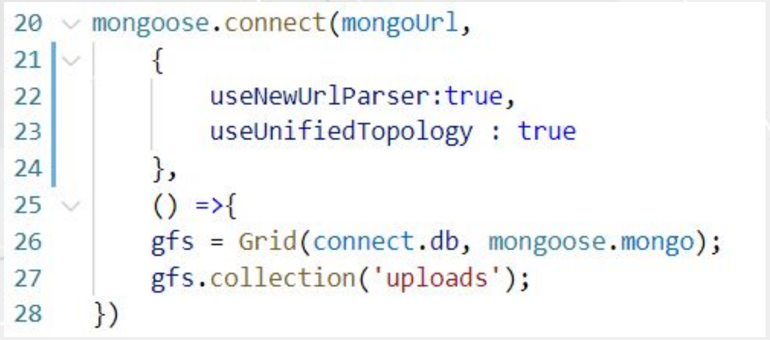
```
const fileSchema = mongoose.Schema({
    path: String,
    metadata : Buffer,
    createdAt : {type: Date, default: Date.now},
    updatedAt : {type: Date, default: Date.now},
    filename : String
})
```

3. Creating model and exporting it



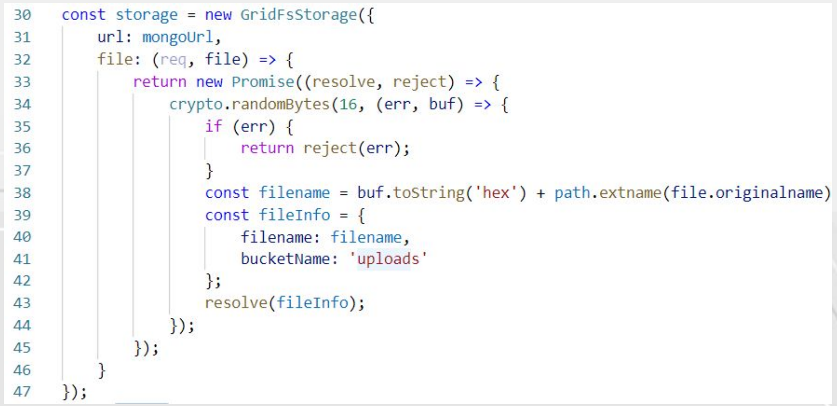
```
module.exports = mongoose.model('File', fileSchema);
```

4. Create connection with mongoDB



```
mongoose.connect(mongoUrl,
    {
        useNewUrlParser:true,
        useUnifiedTopology : true
    },
    () =>{
    gfs = Grid(connect.db, mongoose.mongo);
    gfs.collection('uploads');
})
```

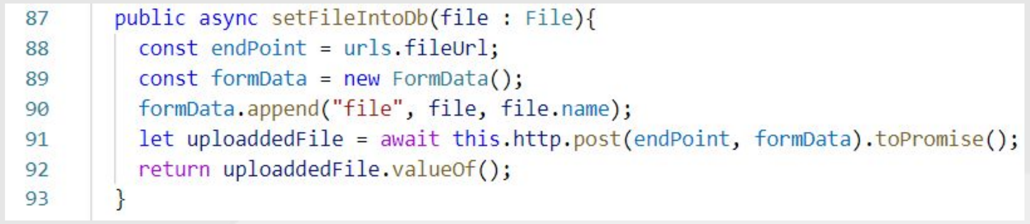
5. Create file storage



```
const storage = new GridFsStorage({
    url: mongoUrl,
    file: (req, file) => {
        return new Promise((resolve, reject) => {
            crypto.randomBytes(16, (err, buf) => {
                if (err) {
                    return reject(err);
                }
                const filename = buf.toString('hex') + path.extname(file.originalname)
                const fileInfo = {
                    filename: filename,
                    bucketName: 'uploads'
                };
                resolve(fileInfo);
            });
        });
    }
});
```

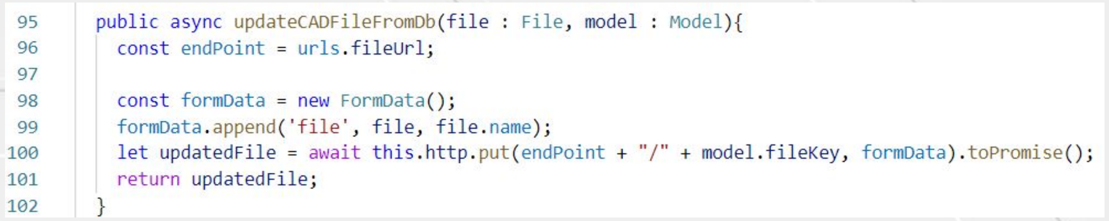
Angular (Implement operations with files from service)

1. Implement adding file in database



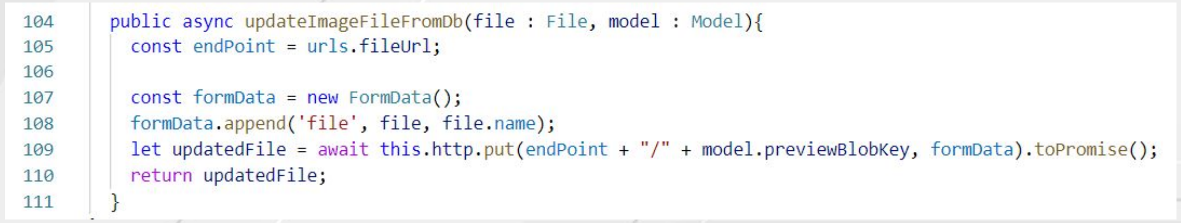
```typescript
public async setFileIntoDb(file : File){
  const endPoint = urls.fileUrl;
  const formData = new FormData();
  formData.append("file", file, file.name);
  let uploaddedFile = await this.http.post(endPoint, formData).toPromise();
  return uploaddedFile.valueOf();
}
```

2. Implement updating CAD file in database



```typescript
public async updateCADFileFromDb(file : File, model : Model){
  const endPoint = urls.fileUrl;

  const formData = new FormData();
  formData.append('file', file, file.name);
  let updatedFile = await this.http.put(endPoint + "/" + model.fileKey, formData).toPromise();
  return updatedFile;
}
```

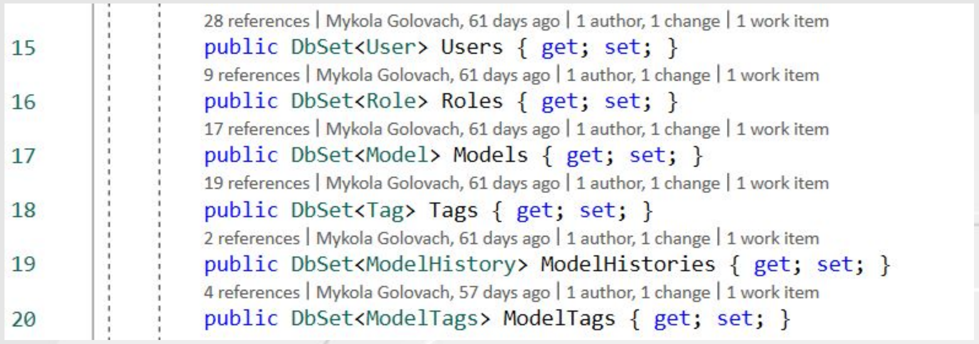
3. Implement updating images in database



```typescript
public async updateImageFileFromDb(file : File, model : Model){
  const endPoint = urls.fileUrl;

  const formData = new FormData();
  formData.append('file', file, file.name);
  let updatedFile = await this.http.put(endPoint + "/" + model.previewBlobKey, formData).toPromise();
  return updatedFile;
}
```

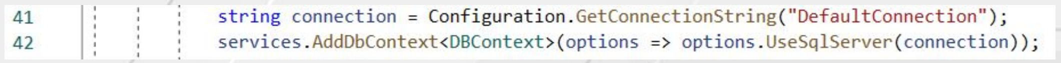
Entity Framework Core (Create a database context for MS SQL Server)

1. Creating a collection for entities that are mapped to database tables.



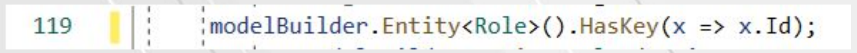
```
28 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
public DbSet<User> Users { get; set; }
9 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
public DbSet<Role> Roles { get; set; }
17 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
public DbSet<Model> Models { get; set; }
19 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
public DbSet<Tag> Tags { get; set; }
2 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
public DbSet<ModelHistory> ModelHistories { get; set; }
4 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
public DbSet<ModelTags> ModelTags { get; set; }
```

2. Creating connection with database in Startup class.



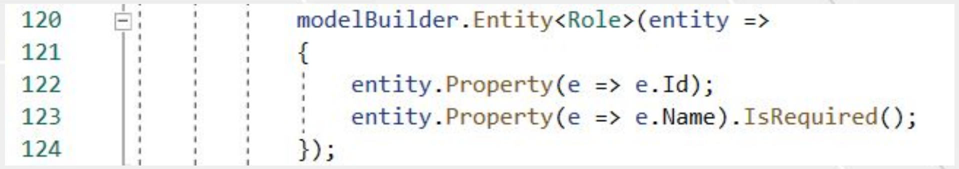
```
string connection = Configuration.GetConnectionString("DefaultConnection");
services.AddDbContext<DBContext>(options => options.UseSqlServer(connection));
```

3. Creating primary key of table.



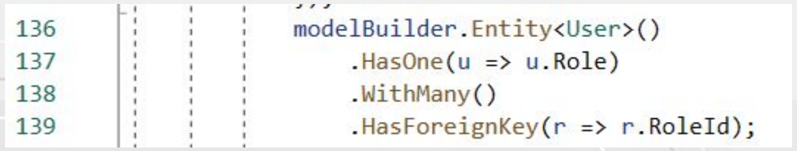
```
modelBuilder.Entity<Role>().HasKey(x => x.Id);
```

4. Creating columns and adding Required configuration.



```
modelBuilder.Entity<Role>(entity =>
{
    entity.Property(e => e.Id);
    entity.Property(e => e.Name).IsRequired();
});
```

5. Creating relationships.

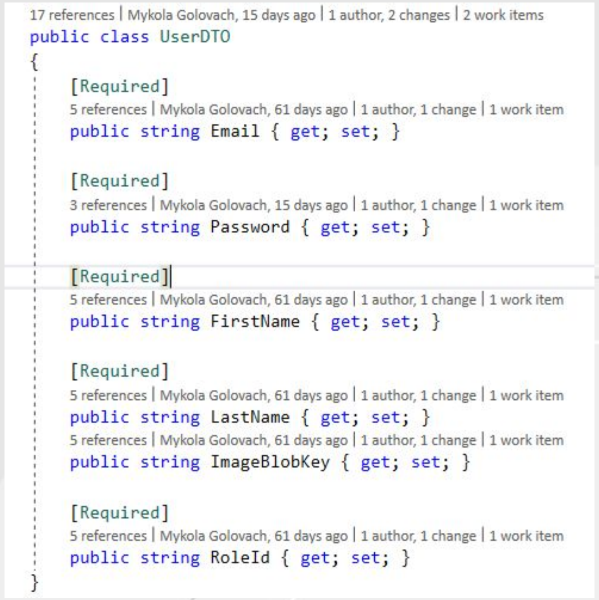


```
modelBuilder.Entity<User>()
    .HasOne(u => u.Role)
    .WithMany()
    .HasForeignKey(r => r.RoleId);
```

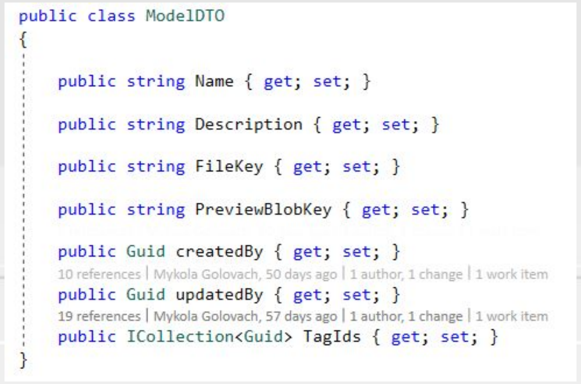
ASP.NET Core (Implement Data Transfer Object and AutoMapper)

AMCBRIDGE
Engineering Software Development Made Easy

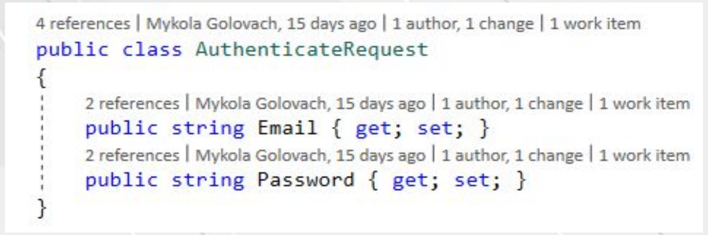
1. Implement Data Transfer Objects to transfer data between application subsystems



```
17 references | Mykola Golovach, 15 days ago | 1 author, 2 changes | 2 work items
public class UserDTO
{
    [Required]
    5 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    public string Email { get; set; }

    [Required]
    3 references | Mykola Golovach, 15 days ago | 1 author, 1 change | 1 work item
    public string Password { get; set; }

    [Required]
    5 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    public string FirstName { get; set; }

    [Required]
    5 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    public string LastName { get; set; }
    5 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    public string ImageBlobKey { get; set; }

    [Required]
    5 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    public string RoleId { get; set; }
}
```

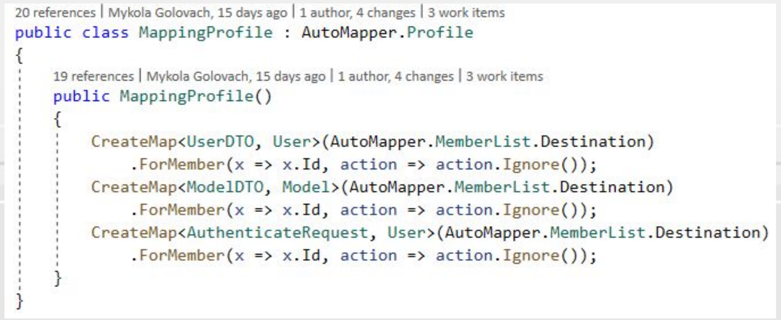


```
public class ModelDTO
{

    public string Name { get; set; }

    public string Description { get; set; }

    public string FileKey { get; set; }

    public string PreviewBlobKey { get; set; }

    public Guid createdBy { get; set; }
    10 references | Mykola Golovach, 50 days ago | 1 author, 1 change | 1 work item
    public Guid updatedBy { get; set; }
    19 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
    public ICollection<Guid> TagIds { get; set; }
}
```

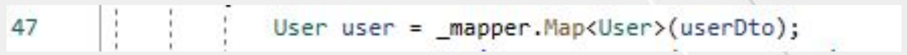


```
4 references | Mykola Golovach, 15 days ago | 1 author, 1 change | 1 work item
public class AuthenticateRequest
{
    2 references | Mykola Golovach, 15 days ago | 1 author, 1 change | 1 work item
    public string Email { get; set; }
    2 references | Mykola Golovach, 15 days ago | 1 author, 1 change | 1 work item
    public string Password { get; set; }
}
```

2. Realize mapping from DTO to models



```
20 references | Mykola Golovach, 15 days ago | 1 author, 4 changes | 3 work items
public class MappingProfile : AutoMapper.Profile
{
    19 references | Mykola Golovach, 15 days ago | 1 author, 4 changes | 3 work items
    public MappingProfile()
    {
        CreateMap<UserDTO, User>(AutoMapper.MemberList.Destination)
            .ForMember(x => x.Id, action => action.Ignore());
        CreateMap<ModelDTO, Model>(AutoMapper.MemberList.Destination)
            .ForMember(x => x.Id, action => action.Ignore());
        CreateMap<AuthenticateRequest, User>(AutoMapper.MemberList.Destination)
            .ForMember(x => x.Id, action => action.Ignore());
    }
}
```

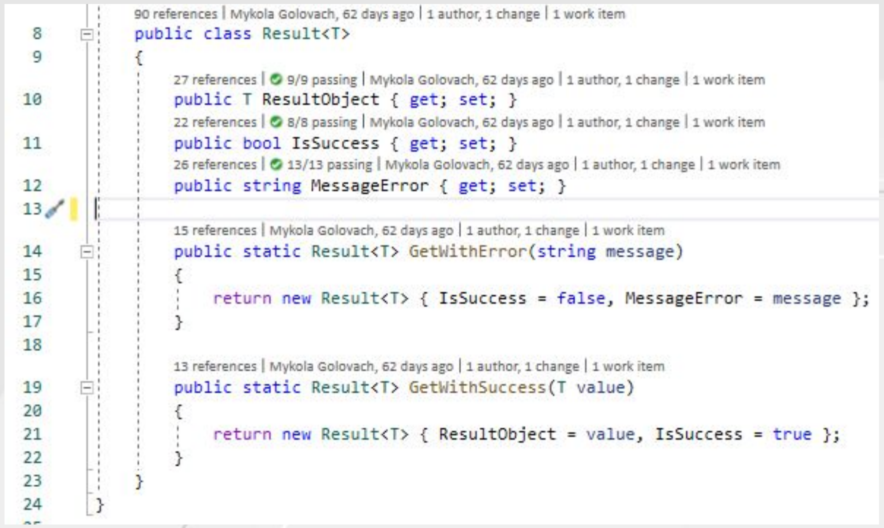


```
User user = _mapper.Map<User>(userDto);
```

ASP.NET Core (Creating Generic class Result<T>)

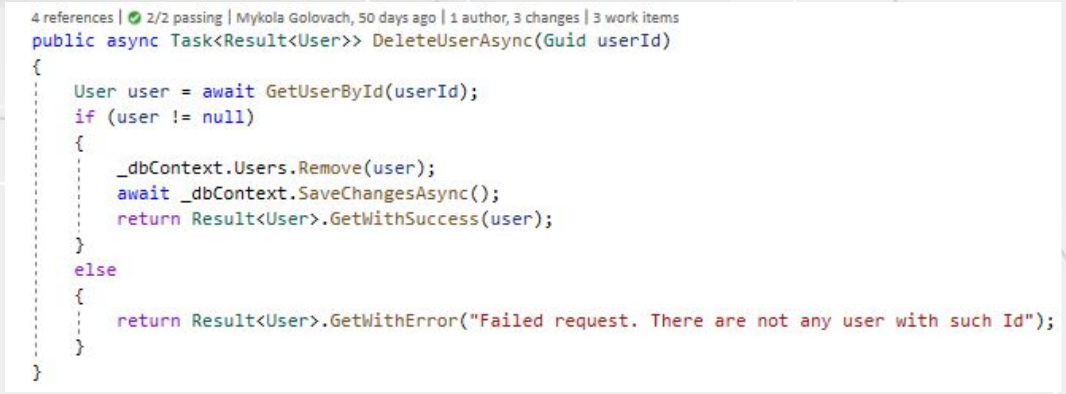
1. Creating generic class to return from service to controller

more advanced concept of the result



```
90 references | Mykola Golovach, 62 days ago | 1 author, 1 change | 1 work item
public class Result<T>
{
    27 references | 9/9 passing | Mykola Golovach, 62 days ago | 1 author, 1 change | 1 work item
    public T ResultObject { get; set; }
    22 references | 8/8 passing | Mykola Golovach, 62 days ago | 1 author, 1 change | 1 work item
    public bool IsSuccess { get; set; }
    26 references | 13/13 passing | Mykola Golovach, 62 days ago | 1 author, 1 change | 1 work item
    public string MessageError { get; set; }

    15 references | Mykola Golovach, 62 days ago | 1 author, 1 change | 1 work item
    public static Result<T> GetWithError(string message)
    {
        return new Result<T> { IsSuccess = false, MessageError = message };
    }

    13 references | Mykola Golovach, 62 days ago | 1 author, 1 change | 1 work item
    public static Result<T> GetWithSuccess(T value)
    {
        return new Result<T> { ResultObject = value, IsSuccess = true };
    }
}
```

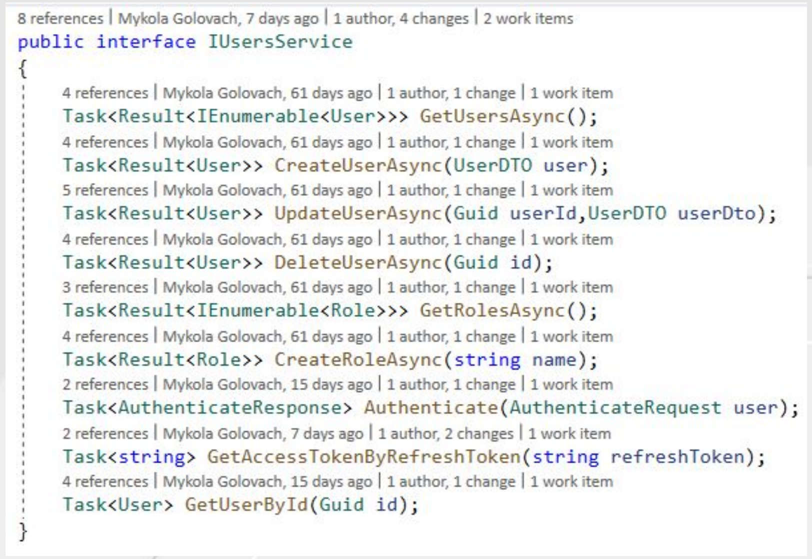
2. Example of using:



```
4 references | 2/2 passing | Mykola Golovach, 50 days ago | 1 author, 3 changes | 3 work items
public async Task<Result<User>> DeleteUserAsync(Guid userId)
{
    User user = await GetUserById(userId);
    if (user != null)
    {
        _dbContext.Users.Remove(user);
        await _dbContext.SaveChangesAsync();
        return Result<User>.GetWithSuccess(user);
    }
    else
    {
        return Result<User>.GetWithError("Failed request. There are not any user with such Id");
    }
}
```

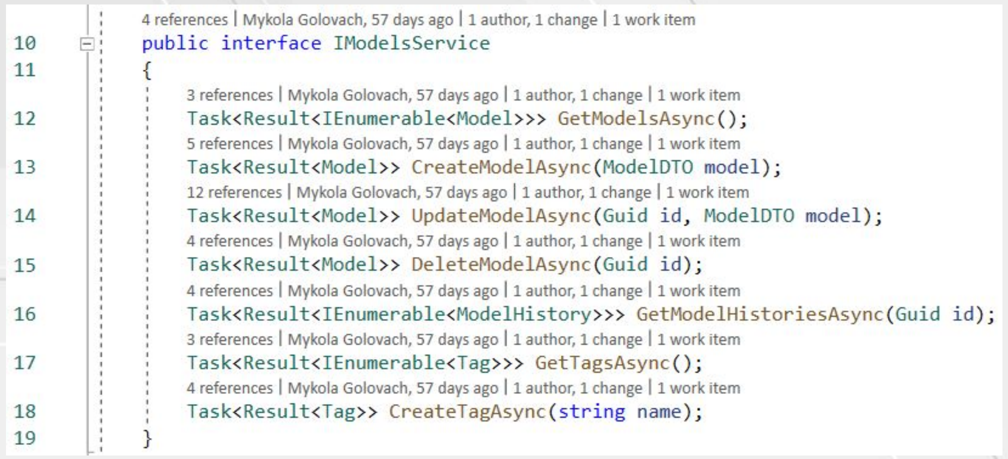
ASP.NET Core (Creating interfaces for services)

1. Creating IUsersService service



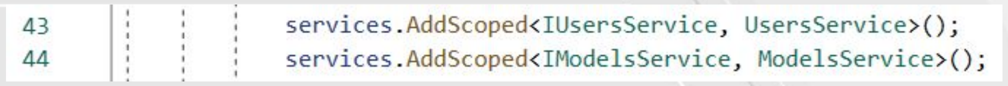
```
8 references | Mykola Golovach, 7 days ago | 1 author, 4 changes | 2 work items
public interface IUsersService
{
    4 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    Task<Result<IEnumerable<User>>> GetUsersAsync();
    4 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    Task<Result<User>> CreateUserAsync(UserDTO user);
    5 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    Task<Result<User>> UpdateUserAsync(Guid userId,UserDTO userDto);
    4 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    Task<Result<User>> DeleteUserAsync(Guid id);
    3 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    Task<Result<IEnumerable<Role>>> GetRolesAsync();
    4 references | Mykola Golovach, 61 days ago | 1 author, 1 change | 1 work item
    Task<Result<Role>> CreateRoleAsync(string name);
    2 references | Mykola Golovach, 15 days ago | 1 author, 1 change | 1 work item
    Task<AuthenticateResponse> Authenticate(AuthenticateRequest user);
    2 references | Mykola Golovach, 7 days ago | 1 author, 2 changes | 1 work item
    Task<string> GetAccessTokenByRefreshToken(string refreshToken);
    4 references | Mykola Golovach, 15 days ago | 1 author, 1 change | 1 work item
    Task<User> GetUserById(Guid id);
}
```

2. Creating IModelsService service



```
4 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
public interface IModelsService
{
    3 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
    Task<Result<IEnumerable<Model>>> GetModelsAsync();
    5 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
    Task<Result<Model>> CreateModelAsync(ModelDTO model);
    12 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
    Task<Result<Model>> UpdateModelAsync(Guid id, ModelDTO model);
    4 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
    Task<Result<Model>> DeleteModelAsync(Guid id);
    4 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
    Task<Result<IEnumerable<ModelHistory>>> GetModelHistoriesAsync(Guid id);
    3 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
    Task<Result<IEnumerable<Tag>>> GetTagsAsync();
    4 references | Mykola Golovach, 57 days ago | 1 author, 1 change | 1 work item
    Task<Result<Tag>> CreateTagAsync(string name);
}
```

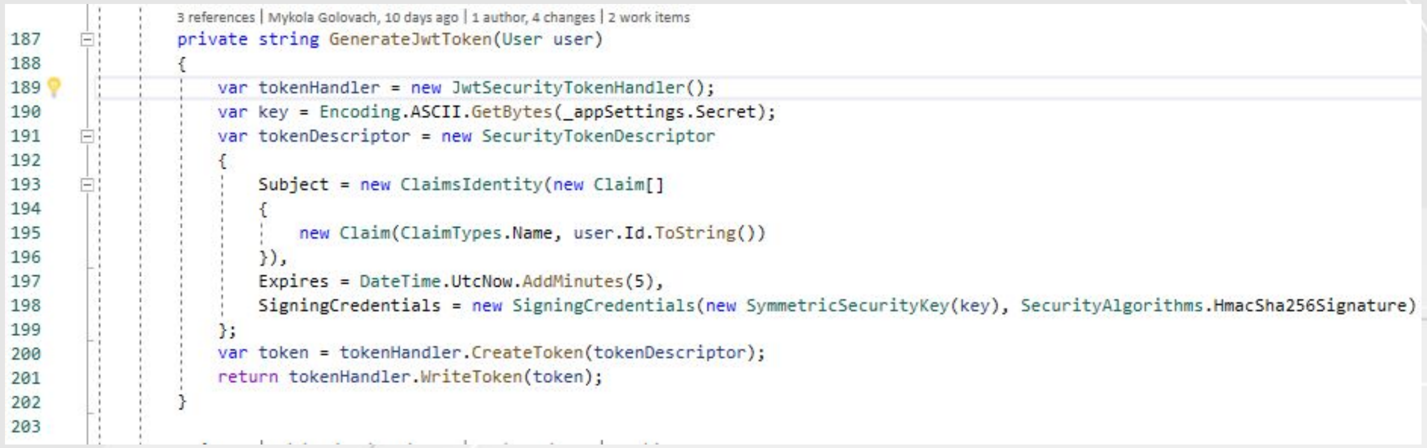
3. Calling method AddScoped to implement independence



```
services.AddScoped<IUsersService, UsersService>();
services.AddScoped<IModelsService, ModelsService>();
```

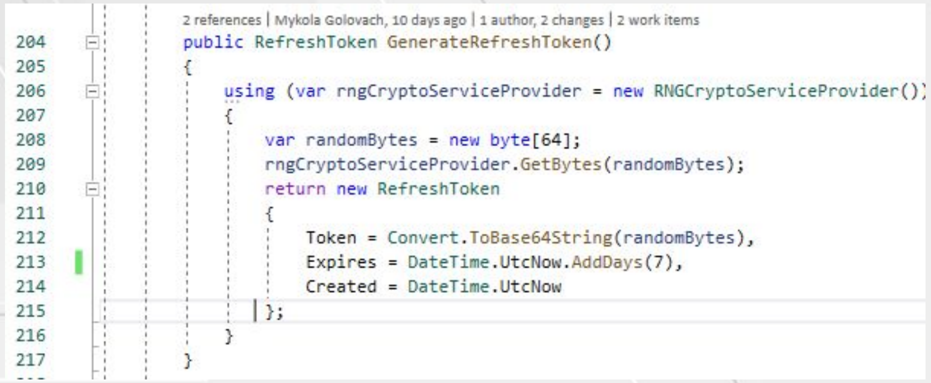
ASP.NET Core (Generating JSON Web Token)

1. Creating generating JSON Web Token



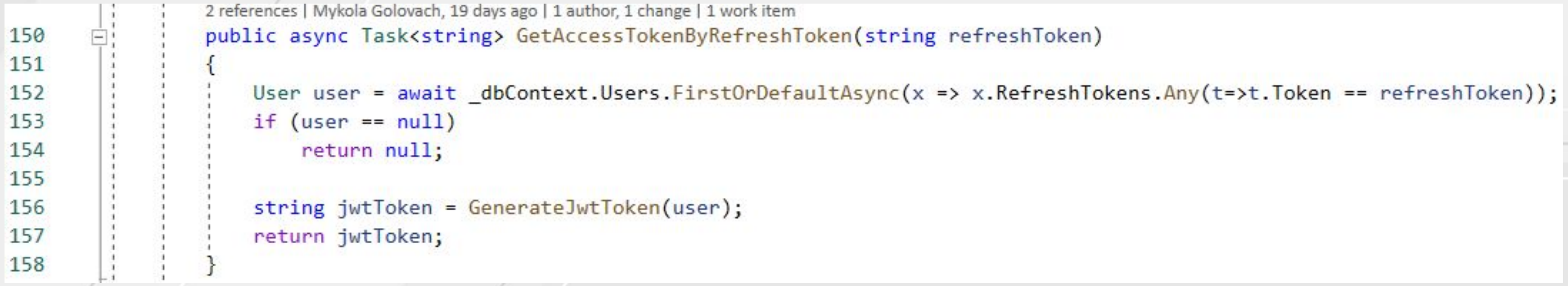
```
3 references | Mykola Golovach, 10 days ago | 1 author, 4 changes | 2 work items
private string GenerateJwtToken(User user)
{
    var tokenHandler = new JwtSecurityTokenHandler();
    var key = Encoding.ASCII.GetBytes(_appSettings.Secret);
    var tokenDescriptor = new SecurityTokenDescriptor
    {
        Subject = new ClaimsIdentity(new Claim[]
        {
            new Claim(ClaimTypes.Name, user.Id.ToString())
        }),
        Expires = DateTime.UtcNow.AddMinutes(5),
        SigningCredentials = new SigningCredentials(new SymmetricSecurityKey(key), SecurityAlgorithms.HmacSha256Signature)
    };
    var token = tokenHandler.CreateToken(tokenDescriptor);
    return tokenHandler.WriteToken(token);
}
```

2. Creating generating refresh token



```
2 references | Mykola Golovach, 10 days ago | 1 author, 2 changes | 2 work items
public RefreshToken GenerateRefreshToken()
{
    using (var rngCryptoServiceProvider = new RNGCryptoServiceProvider()
    {
        var randomBytes = new byte[64];
        rngCryptoServiceProvider.GetBytes(randomBytes);
        return new RefreshToken
        {
            Token = Convert.ToBase64String(randomBytes),
            Expires = DateTime.UtcNow.AddDays(7),
            Created = DateTime.UtcNow
        };
    }
}
```

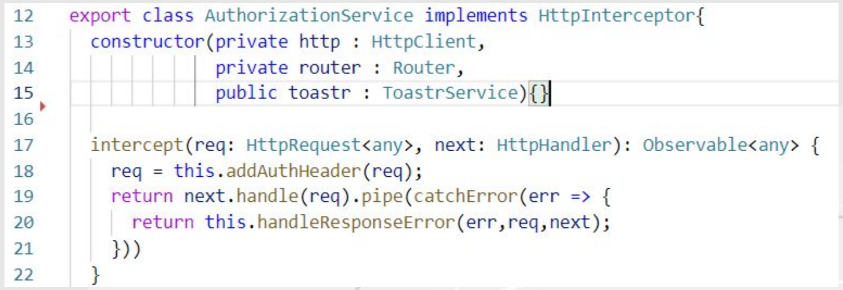
3. Creating generating access token by refresh token



```
2 references | Mykola Golovach, 19 days ago | 1 author, 1 change | 1 work item
public async Task<string> GetAccessTokenByRefreshToken(string refreshToken)
{
    User user = await _dbContext.Users.FirstOrDefaultAsync(x => x.RefreshTokens.Any(t=>t.Token == refreshToken));
    if (user == null)
        return null;

    string jwtToken = GenerateJwtToken(user);
    return jwtToken;
}
```

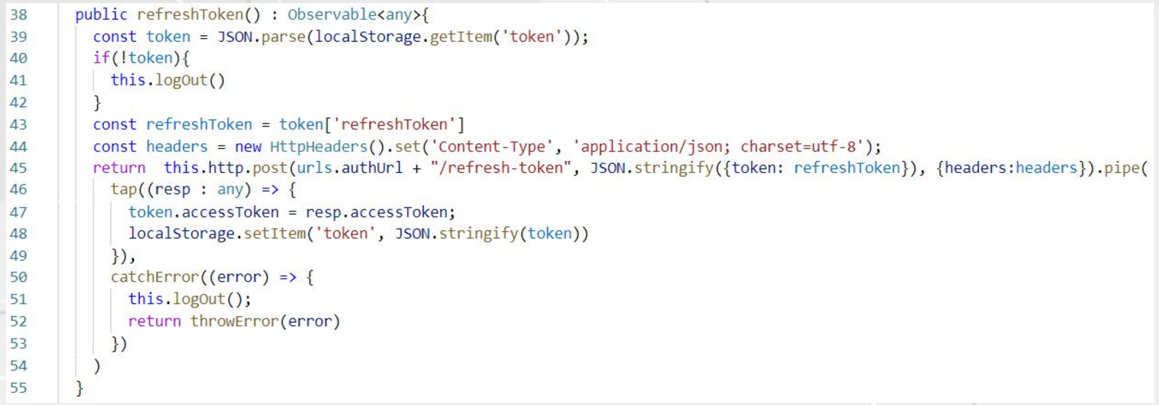
Angular (Implement HttpInterceptor)

1. Creating AuthorizationService class to implement HttpInterceptor



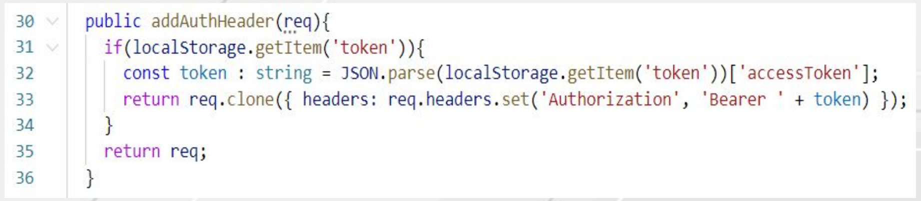
```
export class AuthorizationService implements HttpInterceptor{
  constructor(private http : HttpClient,
              private router : Router,
              public toastr : ToastrService){}

  intercept(req: HttpRequest<any>, next: HttpHandler): Observable<any> {
    req = this.addAuthHeader(req);
    return next.handle(req).pipe(catchError(err => {
      return this.handleResponseError(err,req,next);
    }))
  }
```

2. Creating refreshToken function



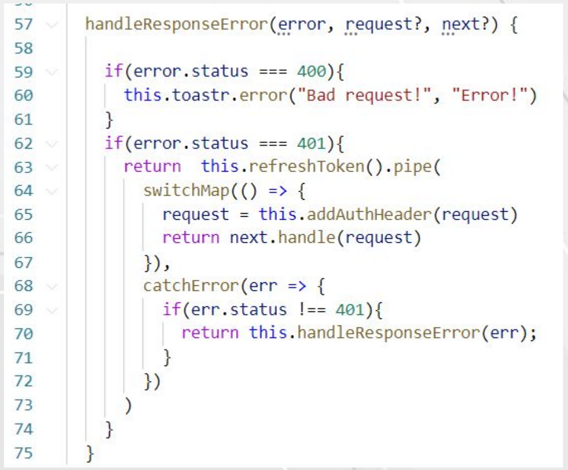
```
public refreshToken() : Observable<any>{
  const token = JSON.parse(localStorage.getItem('token'));
  if(!token){
    this.logOut()
  }
  const refreshToken = token['refreshToken']
  const headers = new HttpHeaders().set('Content-Type', 'application/json; charset=utf-8');
  return  this.http.post(urls.authUrl + "/refresh-token", JSON.stringify({token: refreshToken}), {headers:headers}).pipe(
    tap((resp : any) => {
      token.accessToken = resp.accessToken;
      localStorage.setItem('token', JSON.stringify(token))
    }),
    catchError((error) => {
      this.logOut();
      return throwError(error)
    })
  )
}
```

3. Creating addAuthHeader function

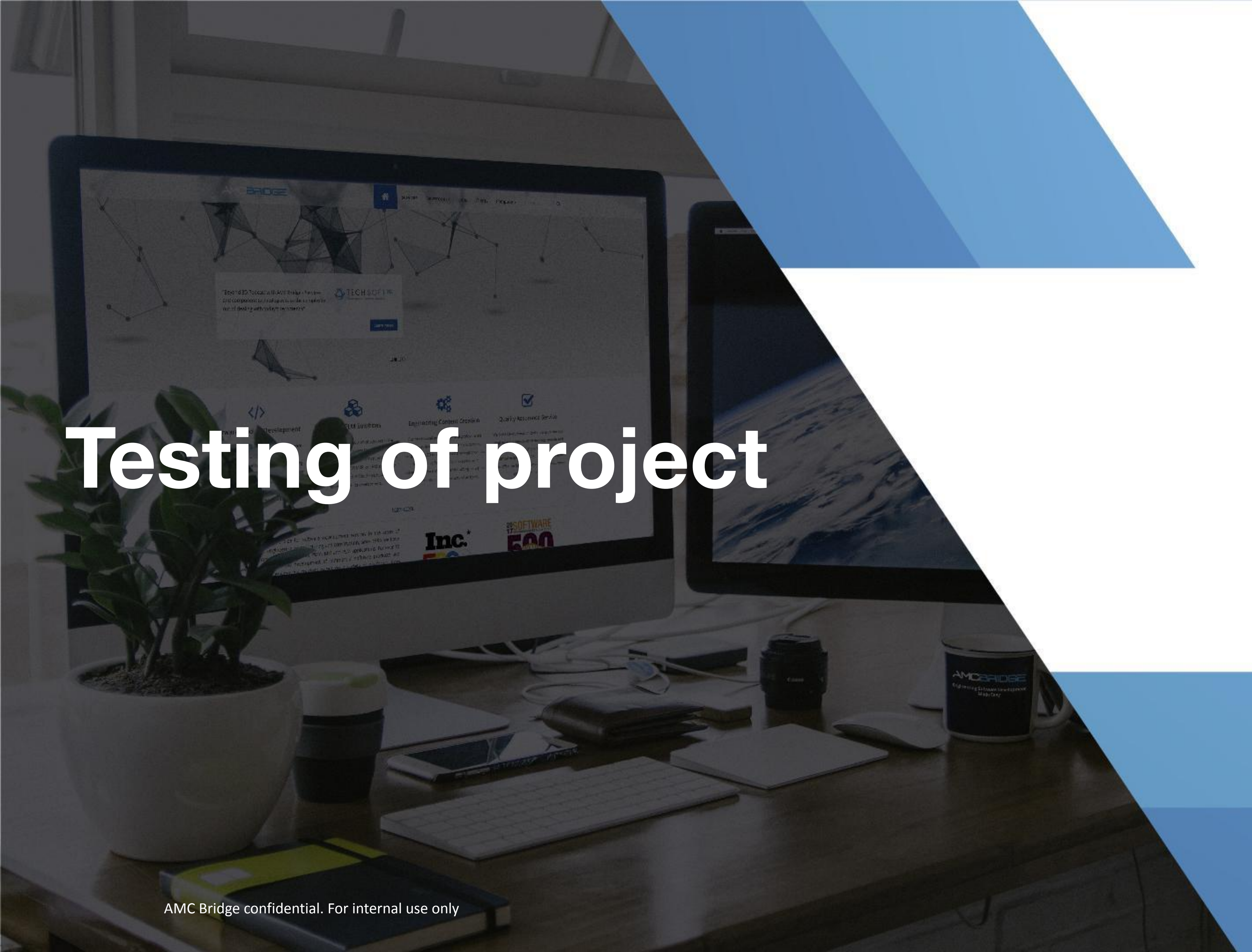


```
public addAuthHeader(req){
  if(localStorage.getItem('token')){
    const token : string = JSON.parse(localStorage.getItem('token'))['accessToken'];
    return req.clone({ headers: req.headers.set('Authorization', 'Bearer ' + token) });
  }
  return req;
}
```

4. Implement handleResponseError function to handle the errors

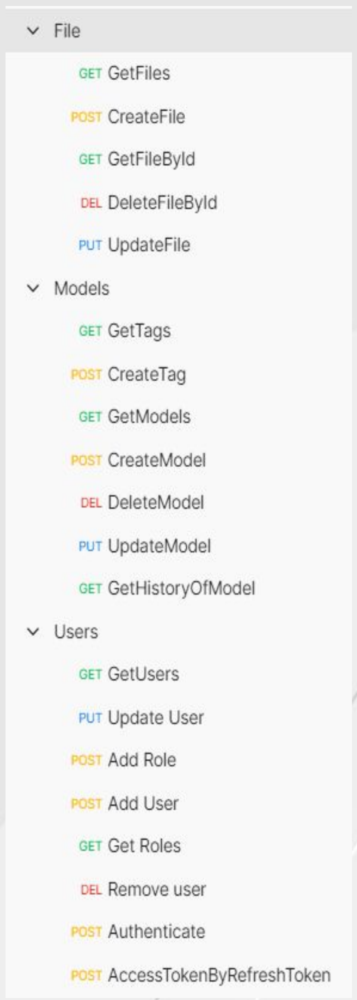


```
handleResponseError(error, request?, next?) {

  if(error.status === 400){
    this.toastr.error("Bad request!", "Error!")
  }
  if(error.status === 401){
    return  this.refreshToken().pipe(
      switchMap(() => {
        request = this.addAuthHeader(request)
        return next.handle(request)
      }),
      catchError(err => {
        if(err.status !== 401){
          return this.handleResponseError(err);
        }
      })
    )
  }
}
```



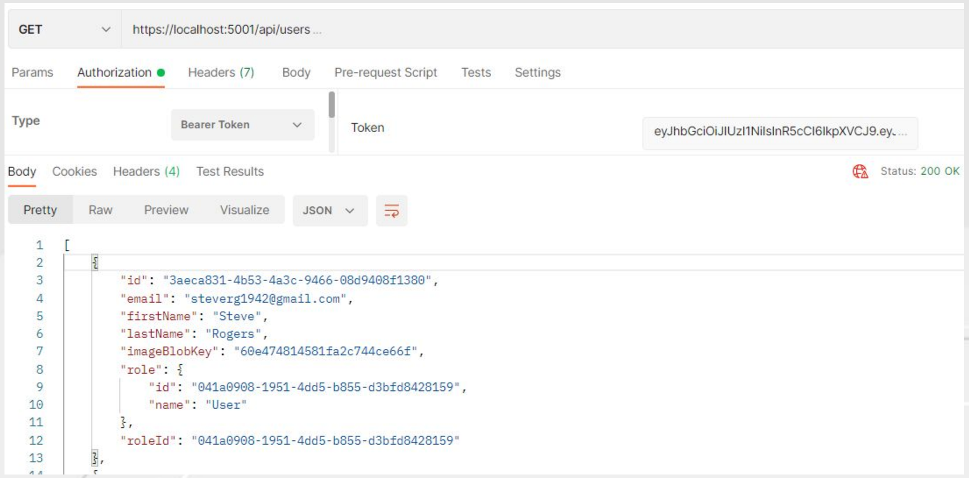
Testing of project

Postman



- File
  - GET GetFiles
  - POST CreateFile
  - GET GetFileById
  - DEL DeleteFileById
  - PUT UpdateFile
- Models
  - GET GetTags
  - POST CreateTag
  - GET GetModels
  - POST CreateModel
  - DEL DeleteModel
  - PUT UpdateModel
  - GET GetHistoryOfModel
- Users
  - GET GetUsers
  - PUT Update User
  - POST Add Role
  - POST Add User
  - GET Get Roles
  - DEL Remove user
  - POST Authenticate
  - POST AccessTokenByRefreshToken

1. Get the list of users



GET https://localhost:5001/api/users ...

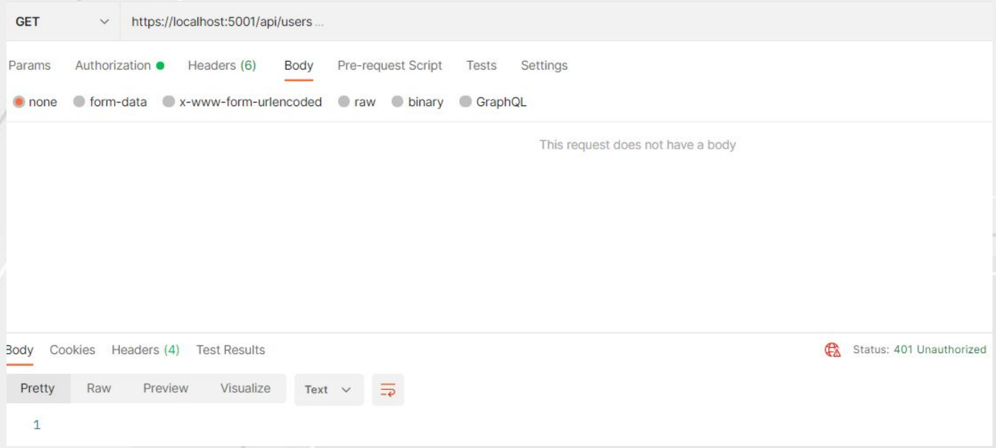
Params | Authorization ● | Headers (7) | Body | Pre-request Script | Tests | Settings

Type: Bearer Token
Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey...

Body | Cookies | Headers (4) | Test Results — Status: 200 OK

Pretty | Raw | Preview | Visualize | JSON

```
[
    {
        "id": "3aeca831-4b53-4a3c-9466-08d9408f1380",
        "email": "steverg1942@gmail.com",
        "firstName": "Steve",
        "lastName": "Rogers",
        "imageBlobKey": "60e474814581fa2c744ce66f",
        "role": {
            "id": "041a0908-1951-4dd5-b855-d3bfd8428159",
            "name": "User"
        },
        "roleId": "041a0908-1951-4dd5-b855-d3bfd8428159"
    },
```



GET https://localhost:5001/api/users ...

Params | Authorization ● | Headers (6) | Body | Pre-request Script | Tests | Settings

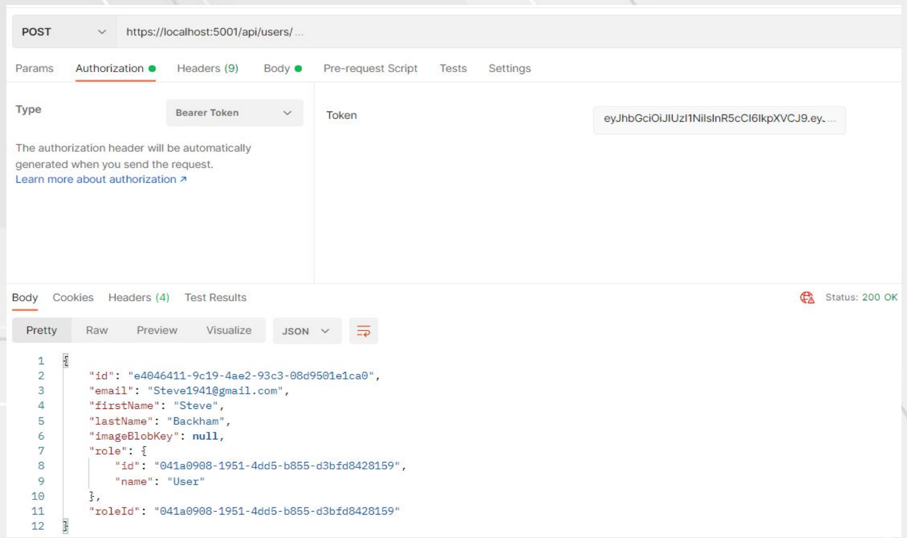
● none | form-data | x-www-form-urlencoded | raw | binary | GraphQL

This request does not have a body

Body | Cookies | Headers (4) | Test Results — Status: 401 Unauthorized

Pretty | Raw | Preview | Visualize | Text

2. Create user by Admin



POST https://localhost:5001/api/users/ ...

Params | Authorization ● | Headers (9) | Body ● | Pre-request Script | Tests | Settings

Type: Bearer Token
Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey...

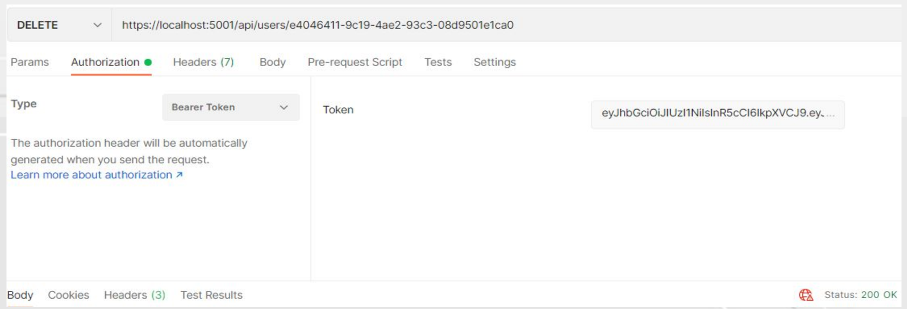
The authorization header will be automatically generated when you send the request.
Learn more about authorization ↗

Body | Cookies | Headers (4) | Test Results — Status: 200 OK

Pretty | Raw | Preview | Visualize | JSON

```
{
    "id": "e4046411-9c19-4ae2-93c3-08d9501e1ca0",
    "email": "Steve1941@gmail.com",
    "firstName": "Steve",
    "lastName": "Backham",
    "imageBlobKey": null,
    "role": {
        "id": "041a0908-1951-4dd5-b855-d3bfd8428159",
        "name": "User"
    },
    "roleId": "041a0908-1951-4dd5-b855-d3bfd8428159"
}
```

3. Delete user by Admin



DELETE https://localhost:5001/api/users/e4046411-9c19-4ae2-93c3-08d9501e1ca0

Params | Authorization ● | Headers (7) | Body | Pre-request Script | Tests | Settings

Type: Bearer Token
Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey...

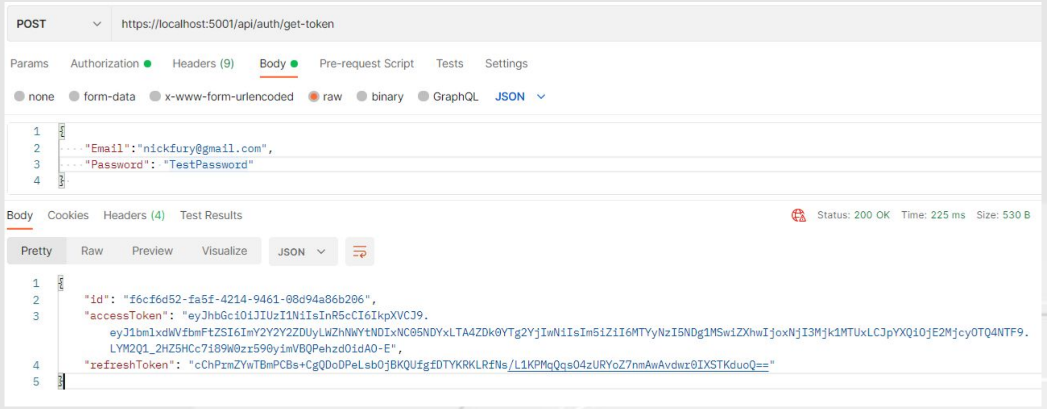
The authorization header will be automatically generated when you send the request.
Learn more about authorization ↗

Body | Cookies | Headers (3) | Test Results — Status: 200 OK

Postman

AMCBRIDGE
Engineering Software Development Made Easy

Authenticate user



POST https://localhost:5001/api/auth/get-token

Params | Authorization | Headers (9) | Body | Pre-request Script | Tests | Settings

none | form-data | x-www-form-urlencoded | [✓ raw] | binary | GraphQL | JSON

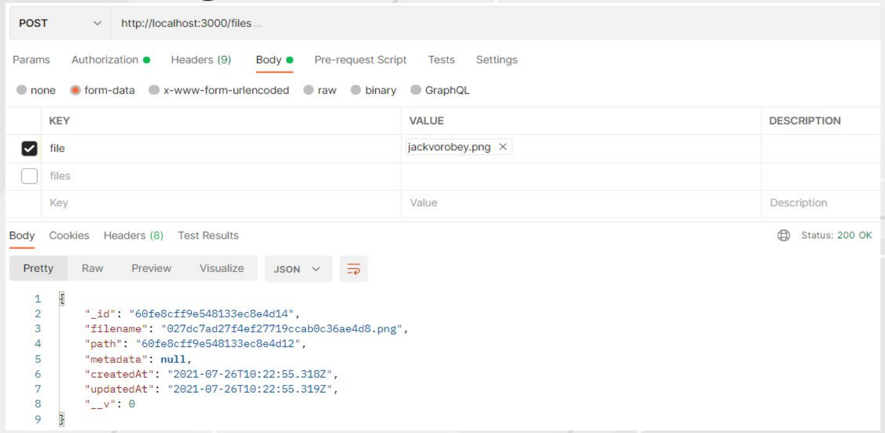
```
{
    "Email":"nickfury@gmail.com",
    "Password": "TestPassword"
}
```

Body | Cookies | Headers (4) | Test Results — Status: 200 OK Time: 225 ms Size: 530 B

Pretty | Raw | Preview | Visualize | JSON

```
{
    "id": "f6cf6d52-fa5f-4214-9461-08d94a86b206",
    "accessToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJ1bmlxdWVfbmFtZSI6ImY2Y2Y2ZDUyLWZhNWYtNDIxNC05NDYxLTA4ZDk0YTg2YjIwNiIsIm5iZiI6MTYyNzI5NDg1MSwiZXhwIjoxNjI3Mjk1MTUxLCJpYXQiOjE2MjcyOTQ4NTF9.
        LYM2Q1_2HZ5HCc7i89W0zr590yimVBQPehzd0idA0-E",
    "refreshToken": "cChPrmZYwTBmPCBs+CgQDoDPeLsb0jBKQUfgfDTYKRKLRfNs/L1KPMqQqs04zURYoZ7nmAwAvdwr0IXSTKduoQ=="
}
```

Save image in database



POST http://localhost:3000/files

Params | Authorization | Headers (9) | Body | Pre-request Script | Tests | Settings

none | [✓ form-data] | x-www-form-urlencoded | raw | binary | GraphQL

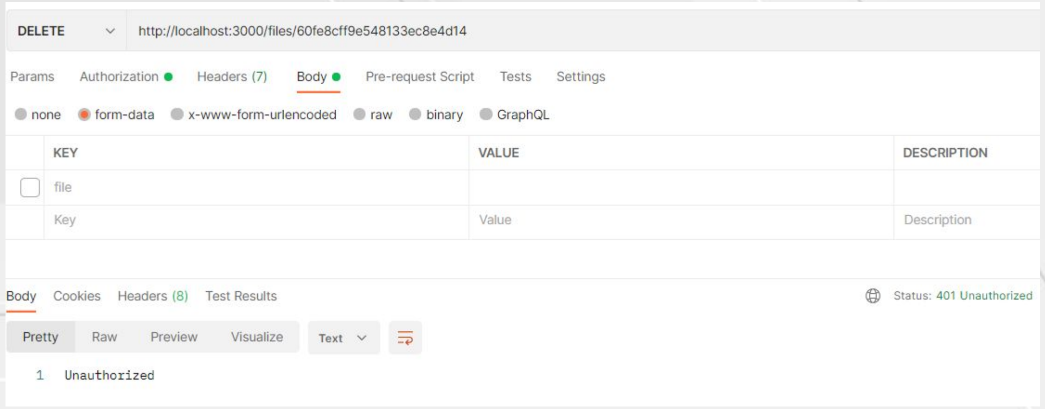
	KEY	VALUE	DESCRIPTION
✓	file	jackvorobey.png	
	files		
	Key	Value	Description

Body | Cookies | Headers (8) | Test Results — Status: 200 OK

Pretty | Raw | Preview | Visualize | JSON

```
{
    "_id": "60fe8cff9e548133ec8e4d14",
    "filename": "027dc7ad27f4ef27719ccab0c36ae4d8.png",
    "path": "60fe8cff9e548133ec8e4d12",
    "metadata": null,
    "createdAt": "2021-07-26T10:22:55.318Z",
    "updatedAt": "2021-07-26T10:22:55.319Z",
    "__v": 0
}
```

Delete image from database



DELETE http://localhost:3000/files/60fe8cff9e548133ec8e4d14

Params | Authorization | Headers (7) | Body | Pre-request Script | Tests | Settings

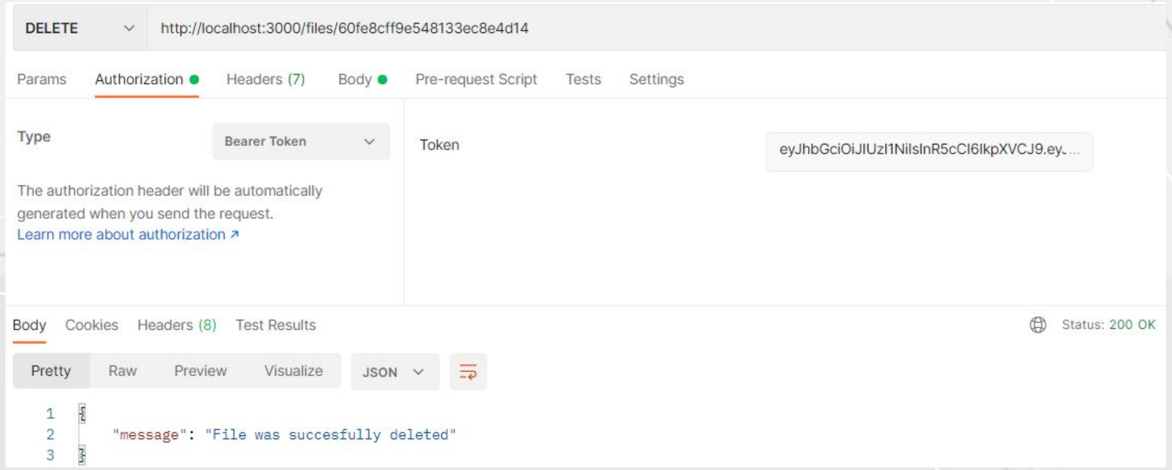
none | [✓ form-data] | x-www-form-urlencoded | raw | binary | GraphQL

	KEY	VALUE	DESCRIPTION
	file		
	Key	Value	Description

Body | Cookies | Headers (8) | Test Results — Status: 401 Unauthorized

Pretty | Raw | Preview | Visualize | Text

```
Unauthorized
```



DELETE http://localhost:3000/files/60fe8cff9e548133ec8e4d14

Params | Authorization | Headers (7) | Body | Pre-request Script | Tests | Settings

Type: Bearer Token

Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey...

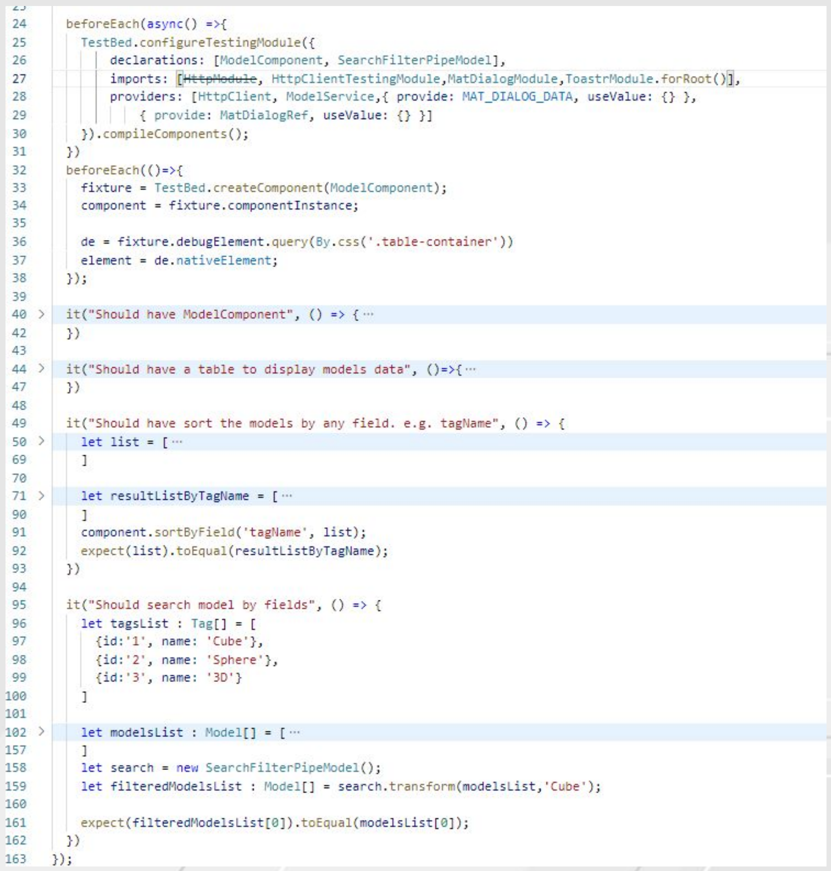
The authorization header will be automatically generated when you send the request.
Learn more about authorization ↗

Body | Cookies | Headers (8) | Test Results — Status: 200 OK

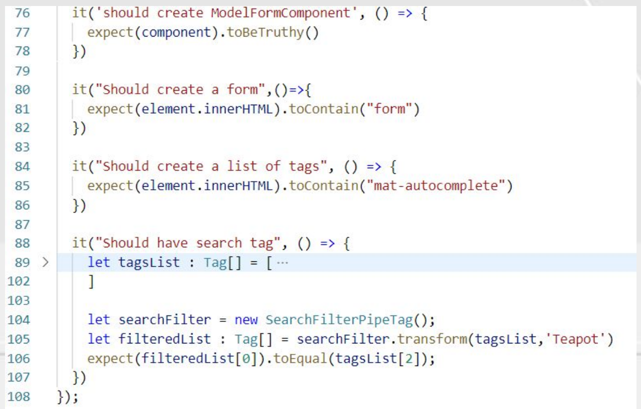
Pretty | Raw | Preview | Visualize | JSON

```
{
    "message": "File was succesfully deleted"
}
```

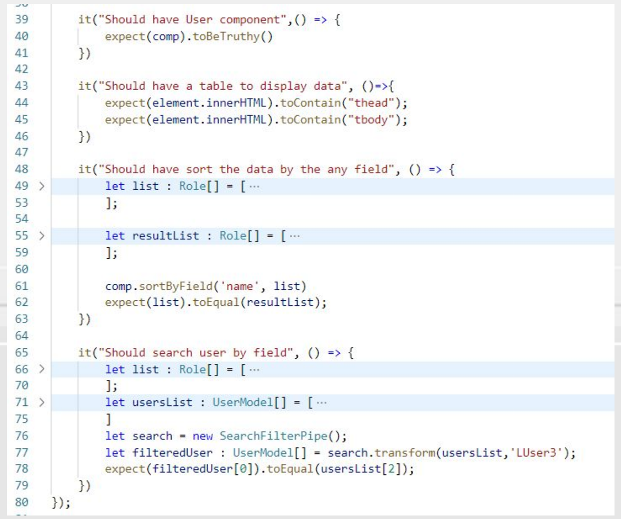
Karma



```
  beforeEach(async() =>{
    TestBed.configureTestingModule({
      declarations: [ModelComponent, SearchFilterPipeModel],
      imports: [~~HttpModule~~, HttpClientTestingModule,MatDialogModule,ToastrModule.forRoot()],
      providers: [HttpClient, ModelService,{ provide: MAT_DIALOG_DATA, useValue: {} },
          { provide: MatDialogRef, useValue: {} }]
    }).compileComponents();
  })
  beforeEach(()=>{
    fixture = TestBed.createComponent(ModelComponent);
    component = fixture.componentInstance;

    de = fixture.debugElement.query(By.css('.table-container'))
    element = de.nativeElement;
  });

  it("Should have ModelComponent", () => { ···
  })

  it("Should have a table to display models data", ()=>{ ···
  })

  it("Should have sort the models by any field. e.g. tagName", () => {
    let list = [ ···
    ]

    let resultListByTagName = [ ···
    ]
    component.sortByField('tagName', list);
    expect(list).toEqual(resultListByTagName);
  })

  it("Should search model by fields", () => {
    let tagsList : Tag[] = [
      {id:'1', name: 'Cube'},
      {id:'2', name: 'Sphere'},
      {id:'3', name: '3D'}
    ]

    let modelsList : Model[] = [ ···
    ]
    let search = new SearchFilterPipeModel();
    let filteredModelsList : Model[] = search.transform(modelsList,'Cube');

    expect(filteredModelsList[0]).toEqual(modelsList[0]);
  })
});
```

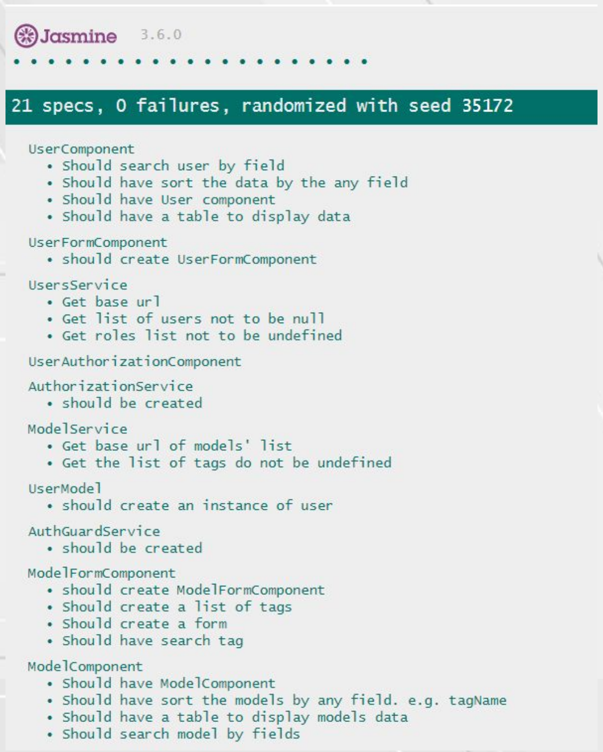


```
  it('should create ModelFormComponent', () => {
    expect(component).toBeTruthy()
  })

  it("Should create a form",()=>{
    expect(element.innerHTML).toContain("form")
  })

  it("Should create a list of tags", () => {
    expect(element.innerHTML).toContain("mat-autocomplete")
  })

  it("Should have search tag", () => {
    let tagsList : Tag[] = [ ···
    ]

    let searchFilter = new SearchFilterPipeTag();
    let filteredList : Tag[] = searchFilter.transform(tagsList,'Teapot')
    expect(filteredList[0]).toEqual(tagsList[2]);
  })
});
```



```
    it("Should have User component",() => {
        expect(comp).toBeTruthy()
    })

    it("Should have a table to display data", ()=>{
        expect(element.innerHTML).toContain("thead");
        expect(element.innerHTML).toContain("tbody");
    })

    it("Should have sort the data by the any field", () => {
        let list : Role[] = [ ···
        ];

        let resultList : Role[] = [ ···
        ];

        comp.sortByField('name', list)
        expect(list).toEqual(resultList);
    })

    it("Should search user by field", () => {
        let list : Role[] = [ ···
        ];
        let usersList : UserModel[] = [ ···
        ]
        let search = new SearchFilterPipe();
        let filteredUser : UserModel[] = search.transform(usersList,'LUser3');
        expect(filteredUser[0]).toEqual(usersList[2]);
    })
});
```

Result



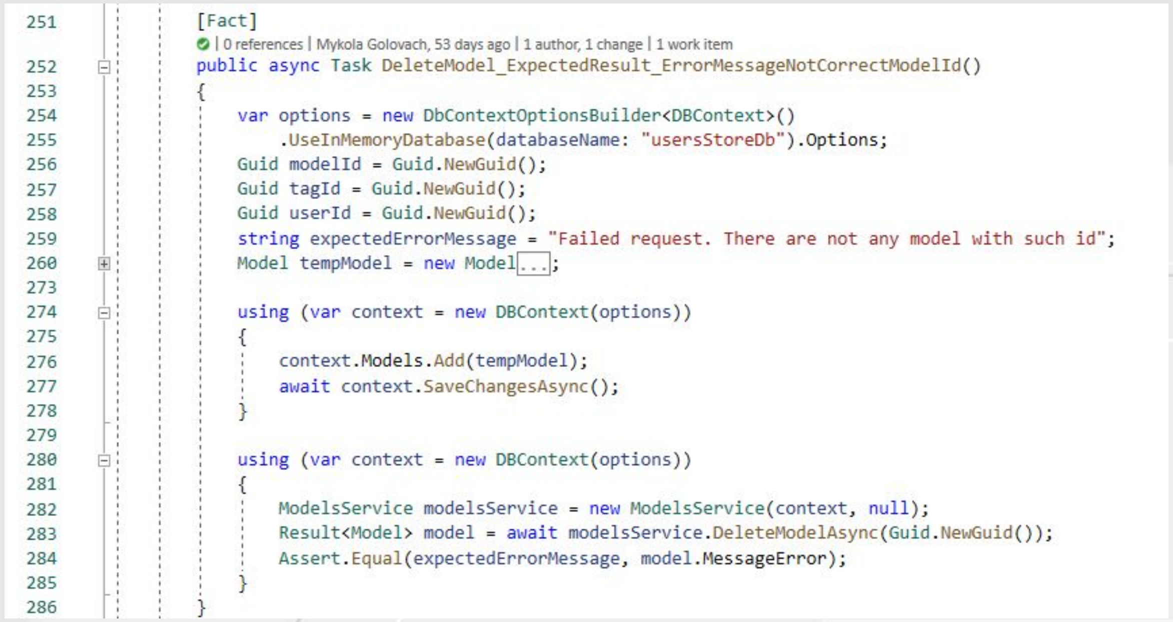
Jasmine 3.6.0

```
• • • • • • • • • • • • • • • • • • • • •
21 specs, 0 failures, randomized with seed 35172

UserComponent
  • Should search user by field
  • Should have sort the data by the any field
  • Should have User component
  • Should have a table to display data
UserFormComponent
  • should create UserFormComponent
UsersService
  • Get base url
  • Get list of users not to be null
  • Get roles list not to be undefined
UserAuthorizationComponent
AuthorizationService
  • should be created
ModelService
  • Get base url of models' list
  • Get the list of tags do not be undefined
UserModel
  • should create an instance of user
AuthGuardService
  • should be created
ModelFormComponent
  • should create ModelFormComponent
  • Should create a list of tags
  • Should create a form
  • Should have search tag
ModelComponent
  • Should have ModelComponent
  • Should have sort the models by any field. e.g. tagName
  • Should have a table to display models data
  • Should search model by fields
```

XUnit

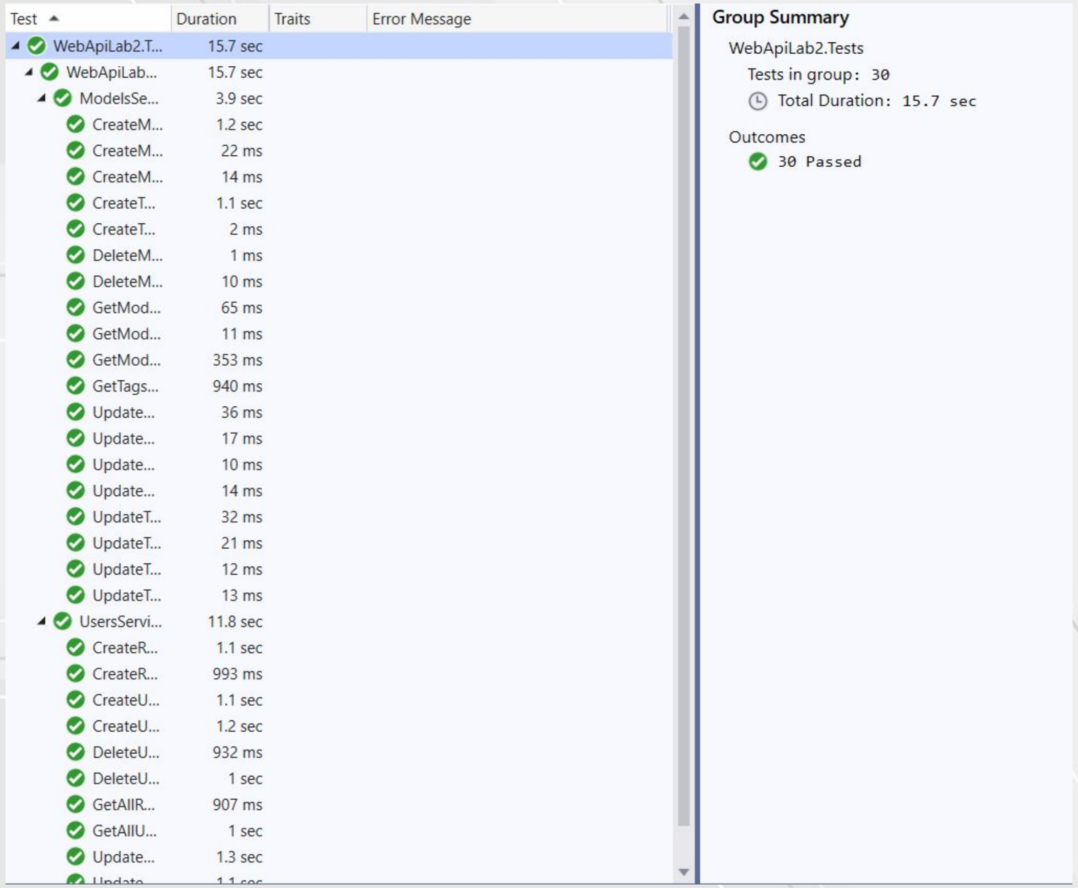
AMCBRIDGE
Engineering Software Development Made Easy

Example



```
[Fact]
| 0 references | Mykola Golovach, 53 days ago | 1 author, 1 change | 1 work item
public async Task DeleteModel_ExpectedResult_ErrorMessageNotCorrectModelId()
{
    var options = new DbContextOptionsBuilder<DBContext>()
        .UseInMemoryDatabase(databaseName: "usersStoreDb").Options;
    Guid modelId = Guid.NewGuid();
    Guid tagId = Guid.NewGuid();
    Guid userId = Guid.NewGuid();
    string expectedErrorMessage = "Failed request. There are not any model with such id";
    Model tempModel = new Model...;

    using (var context = new DBContext(options))
    {
        context.Models.Add(tempModel);
        await context.SaveChangesAsync();
    }

    using (var context = new DBContext(options))
    {
        ModelsService modelsService = new ModelsService(context, null);
        Result<Model> model = await modelsService.DeleteModelAsync(Guid.NewGuid());
        Assert.Equal(expectedErrorMessage, model.MessageError);
    }
}
```

Result

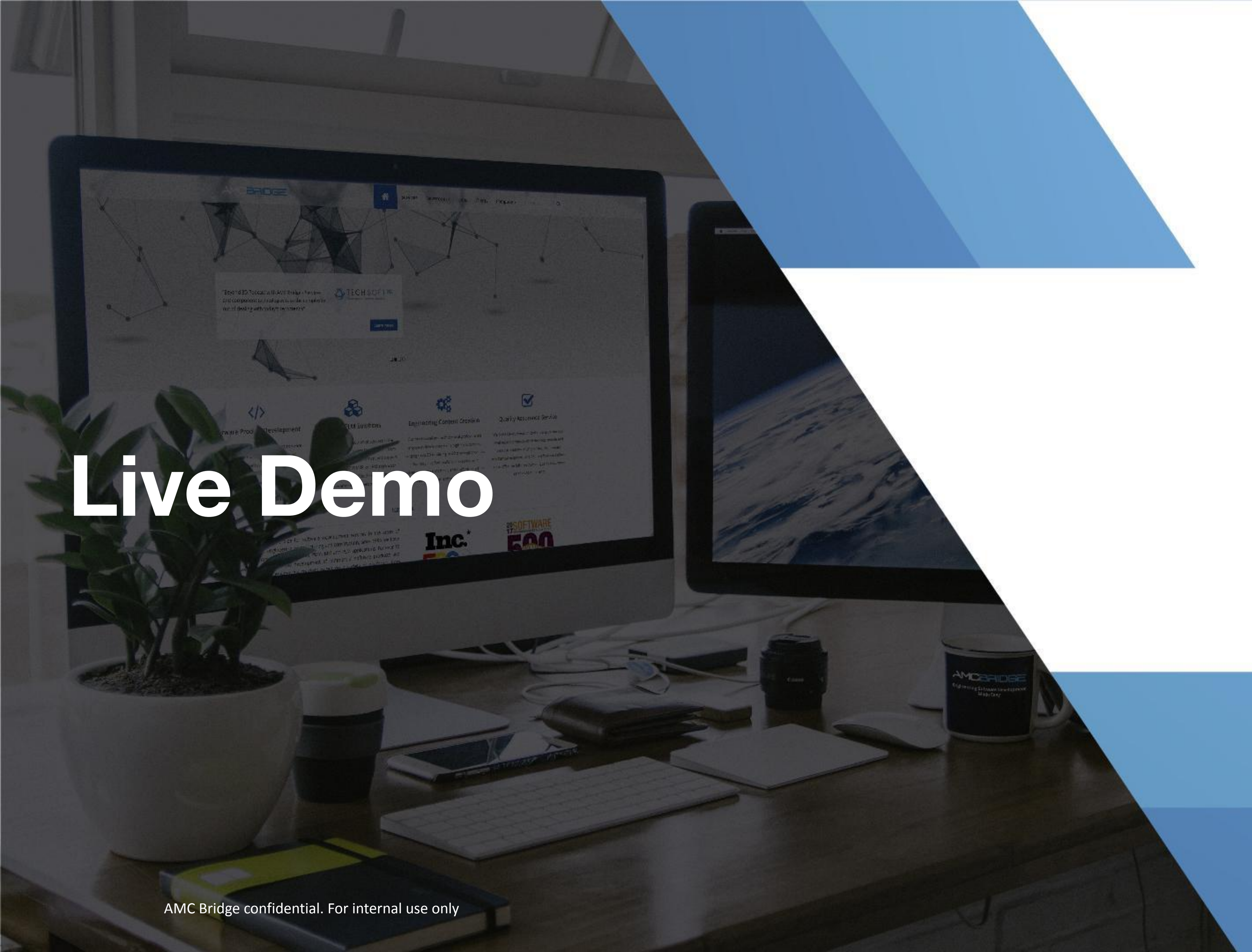


Test	Duration	Traits	Error Message
WebApiLab2.T...	15.7 sec		
WebApiLab...	15.7 sec		
ModelsSe...	3.9 sec		
CreateM...	1.2 sec		
CreateM...	22 ms		
CreateM...	14 ms		
CreateT...	1.1 sec		
CreateT...	2 ms		
DeleteM...	1 ms		
DeleteM...	10 ms		
GetMod...	65 ms		
GetMod...	11 ms		
GetMod...	353 ms		
GetTags...	940 ms		
Update...	36 ms		
Update...	17 ms		
Update...	10 ms		
Update...	14 ms		
UpdateT...	32 ms		
UpdateT...	21 ms		
UpdateT...	12 ms		
UpdateT...	13 ms		
UsersServi...	11.8 sec		
CreateR...	1.1 sec		
CreateR...	993 ms		
CreateU...	1.1 sec		
CreateU...	1.2 sec		
DeleteU...	932 ms		
DeleteU...	1 sec		
GetAllR...	907 ms		
GetAllU...	1 sec		
Update...	1.3 sec		
Update	1.1 sec		

Group Summary

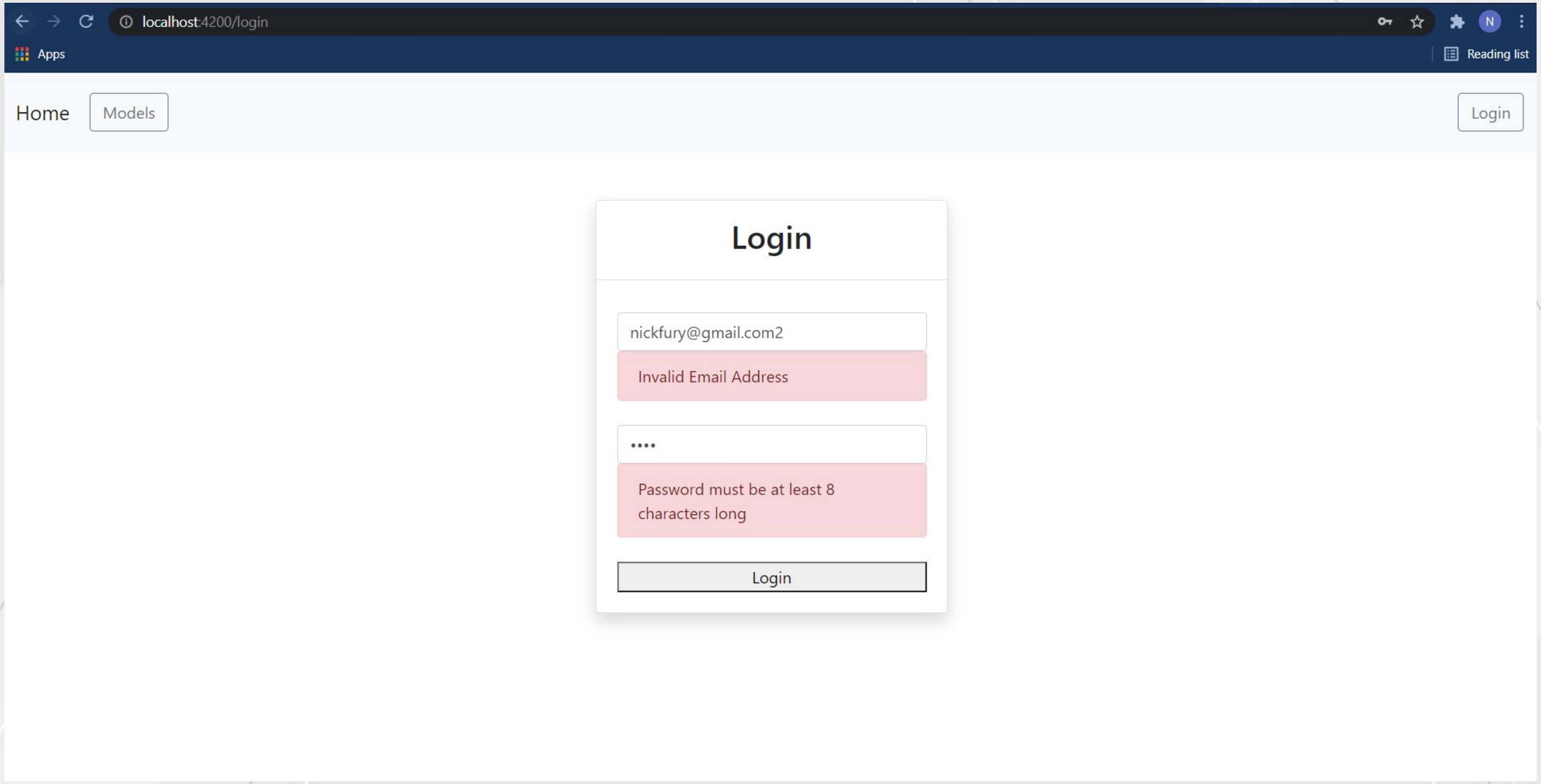
WebApiLab2.Tests
Tests in group: 30
Total Duration: 15.7 sec

Outcomes
✓ 30 Passed



Live Demo

Login page


localhost:4200/login

Home | Models | Login

Login

nickfury@gmail.com2

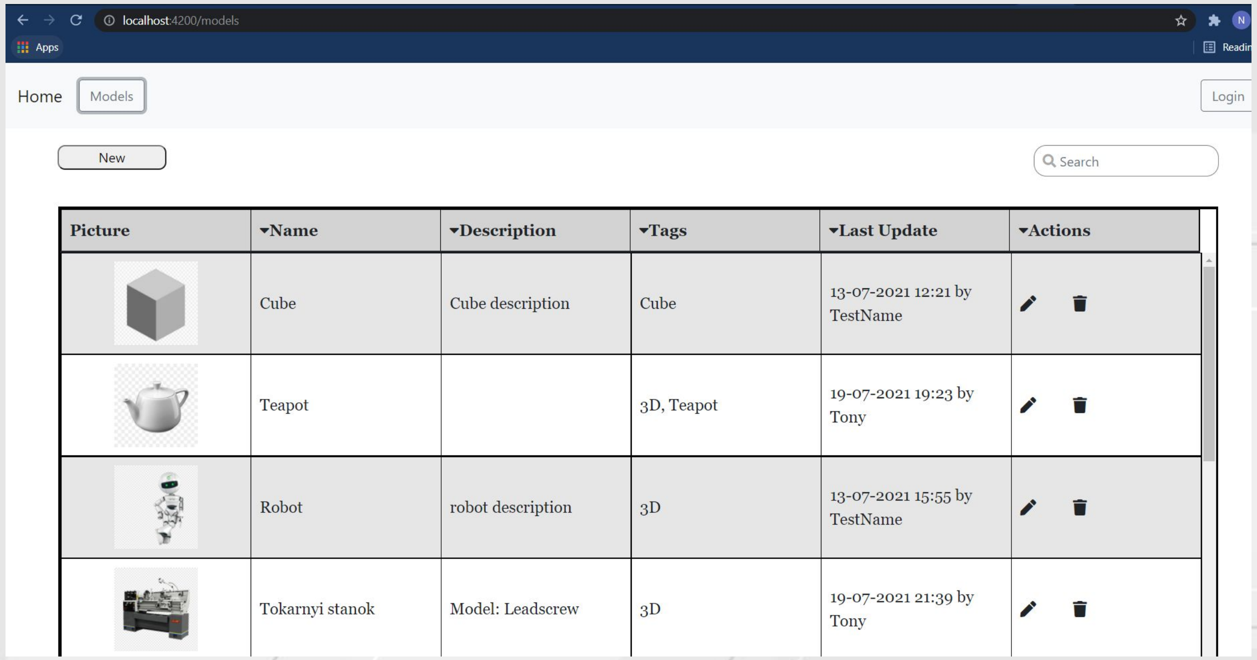
Invalid Email Address

••••

Password must be at least 8 characters long

Login

Models list page (Unauthorized user)

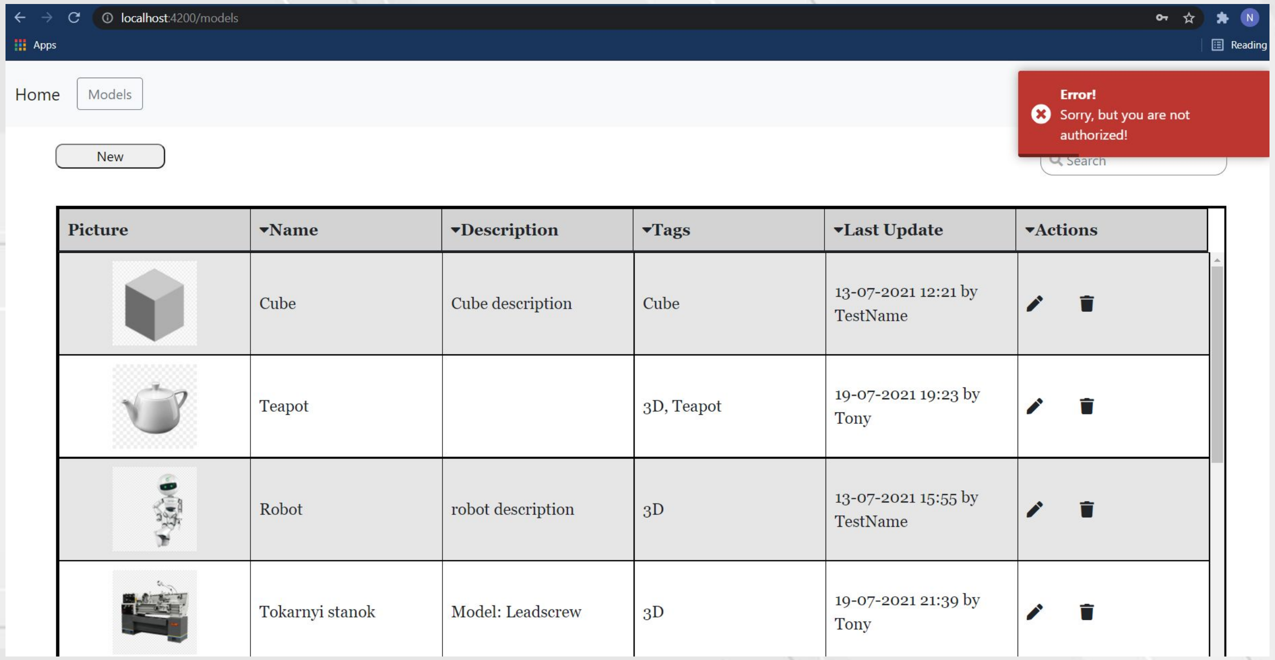


localhost:4200/models

Home | Models | Login

New | Search

Picture	▾Name	▾Description	▾Tags	▾Last Update	▾Actions
	Cube	Cube description	Cube	13-07-2021 12:21 by TestName	
	Teapot		3D, Teapot	19-07-2021 19:23 by Tony	
	Robot	robot description	3D	13-07-2021 15:55 by TestName	
	Tokarnyi stanok	Model: Leadscrew	3D	19-07-2021 21:39 by Tony	



localhost:4200/models

Home | Models

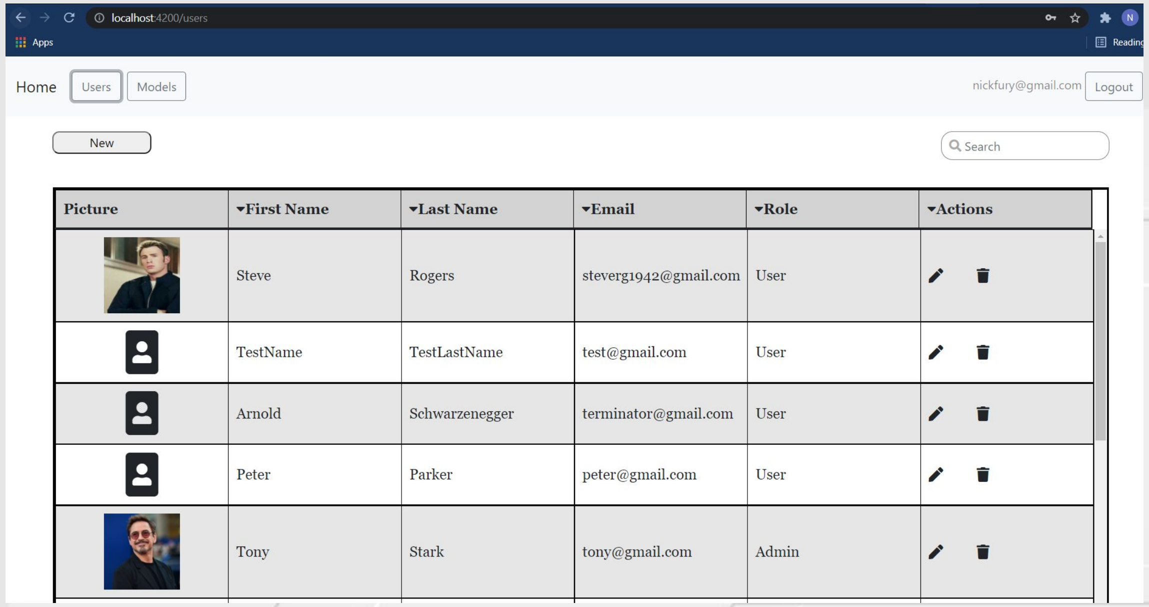
Error!
Sorry, but you are not authorized!

New | Search

Picture	▾Name	▾Description	▾Tags	▾Last Update	▾Actions
	Cube	Cube description	Cube	13-07-2021 12:21 by TestName	
	Teapot		3D, Teapot	19-07-2021 19:23 by Tony	
	Robot	robot description	3D	13-07-2021 15:55 by TestName	
	Tokarnyi stanok	Model: Leadscrew	3D	19-07-2021 21:39 by Tony	

Users list page

Admin role



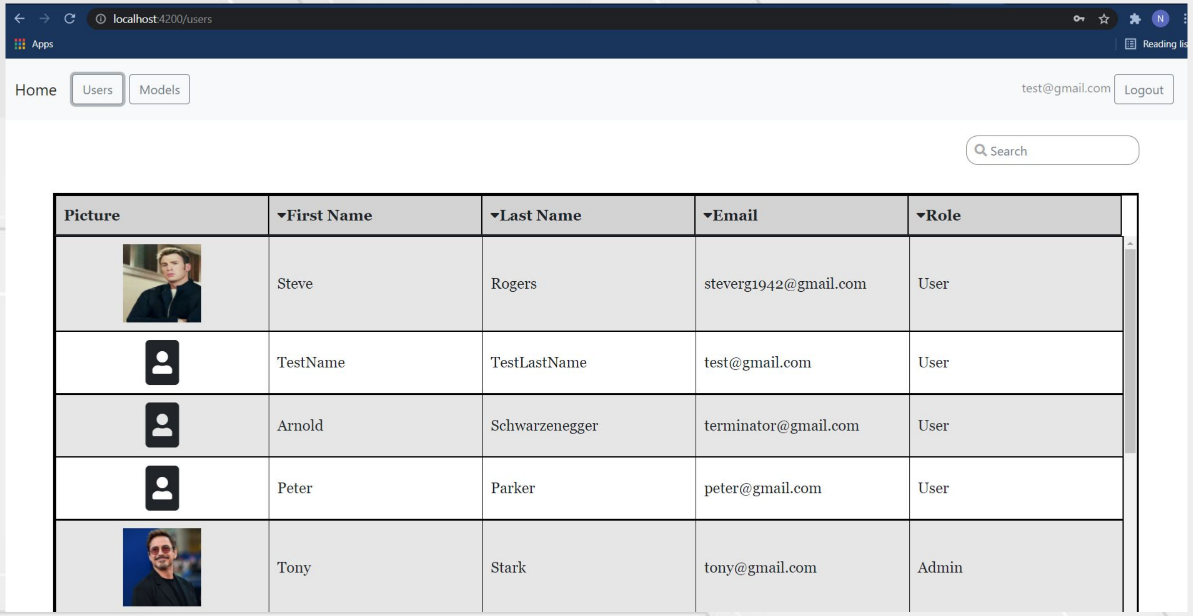
localhost:4200/users

Home | Users | Models | nickfury@gmail.com | Logout

New | Search

Picture	▾First Name	▾Last Name	▾Email	▾Role	▾Actions
	Steve	Rogers	steverg1942@gmail.com	User	✎ 🗑
	TestName	TestLastName	test@gmail.com	User	✎ 🗑
	Arnold	Schwarzenegger	terminator@gmail.com	User	✎ 🗑
	Peter	Parker	peter@gmail.com	User	✎ 🗑
	Tony	Stark	tony@gmail.com	Admin	✎ 🗑

Another role



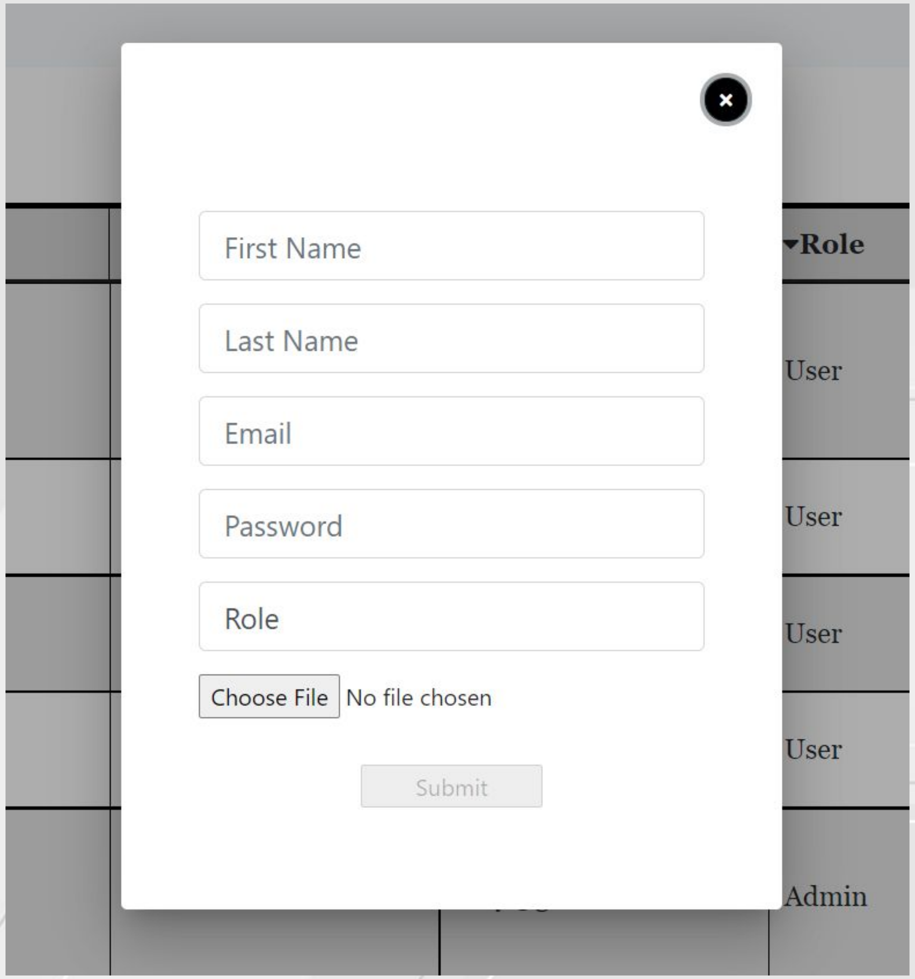
localhost:4200/users

Home | Users | Models | test@gmail.com | Logout

Search

Picture	▾First Name	▾Last Name	▾Email	▾Role
	Steve	Rogers	steverg1942@gmail.com	User
	TestName	TestLastName	test@gmail.com	User
	Arnold	Schwarzenegger	terminator@gmail.com	User
	Peter	Parker	peter@gmail.com	User
	Tony	Stark	tony@gmail.com	Admin

Create and update user dialog


First Name

Last Name

Email

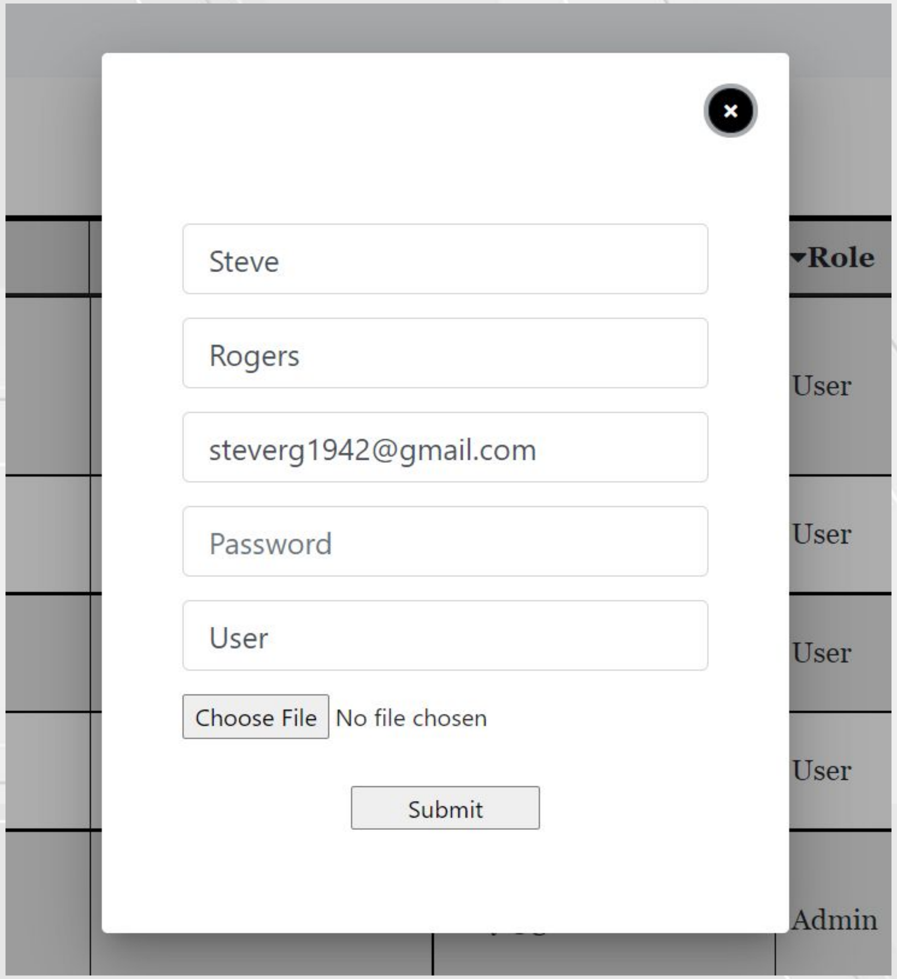
Password

Role

Choose File No file chosen

Submit

Role
User
User
User
User
Admin


Steve

Rogers

steverg1942@gmail.com

Password

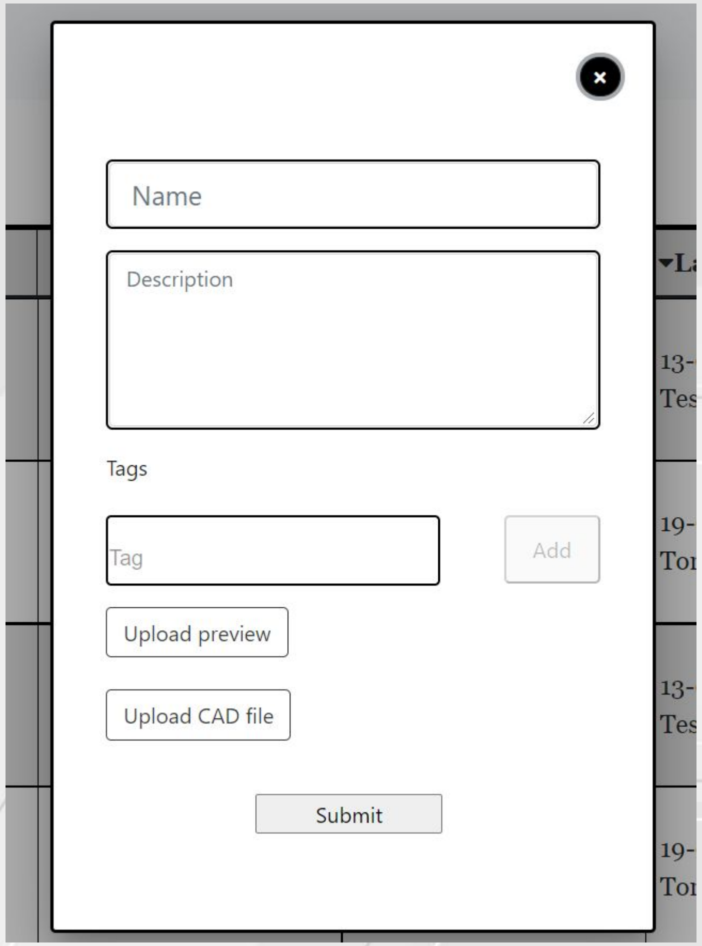
User

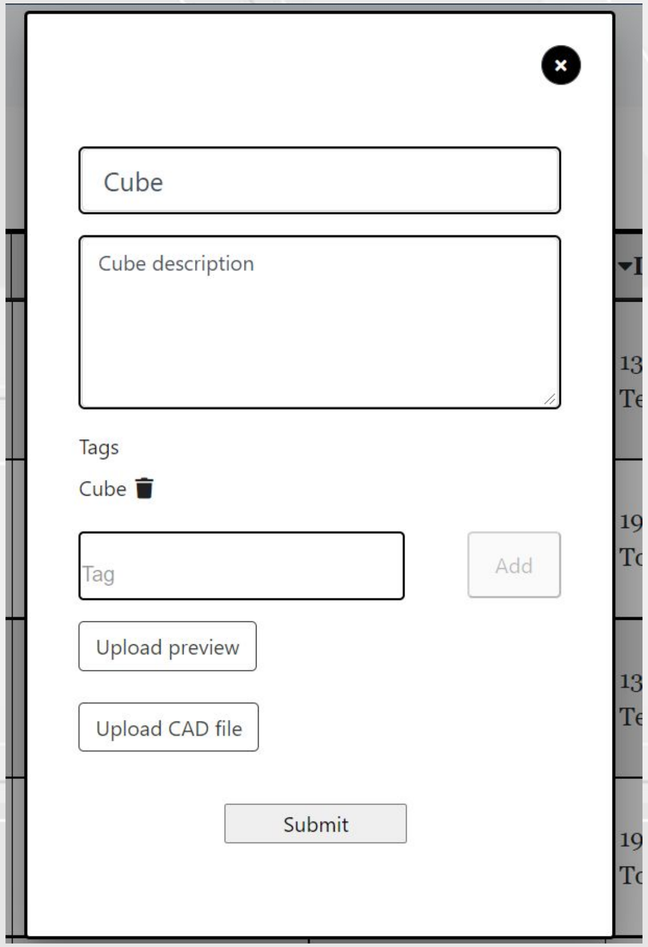
Choose File No file chosen

Submit

Role
User
User
User
User
Admin

Create and update model dialog




Cube

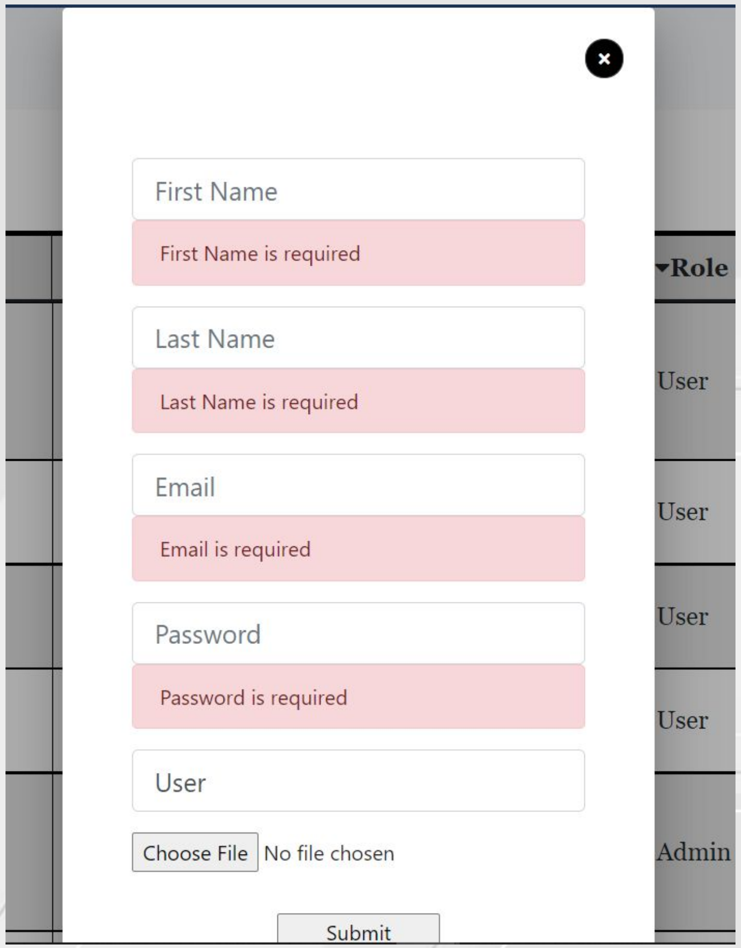
Cube description

Tags

Cube

Input data validation

AMCBRIDGE
Engineering Software Development Made Easy



First Name

First Name is required

Last Name

Last Name is required

Email

Email is required

Password

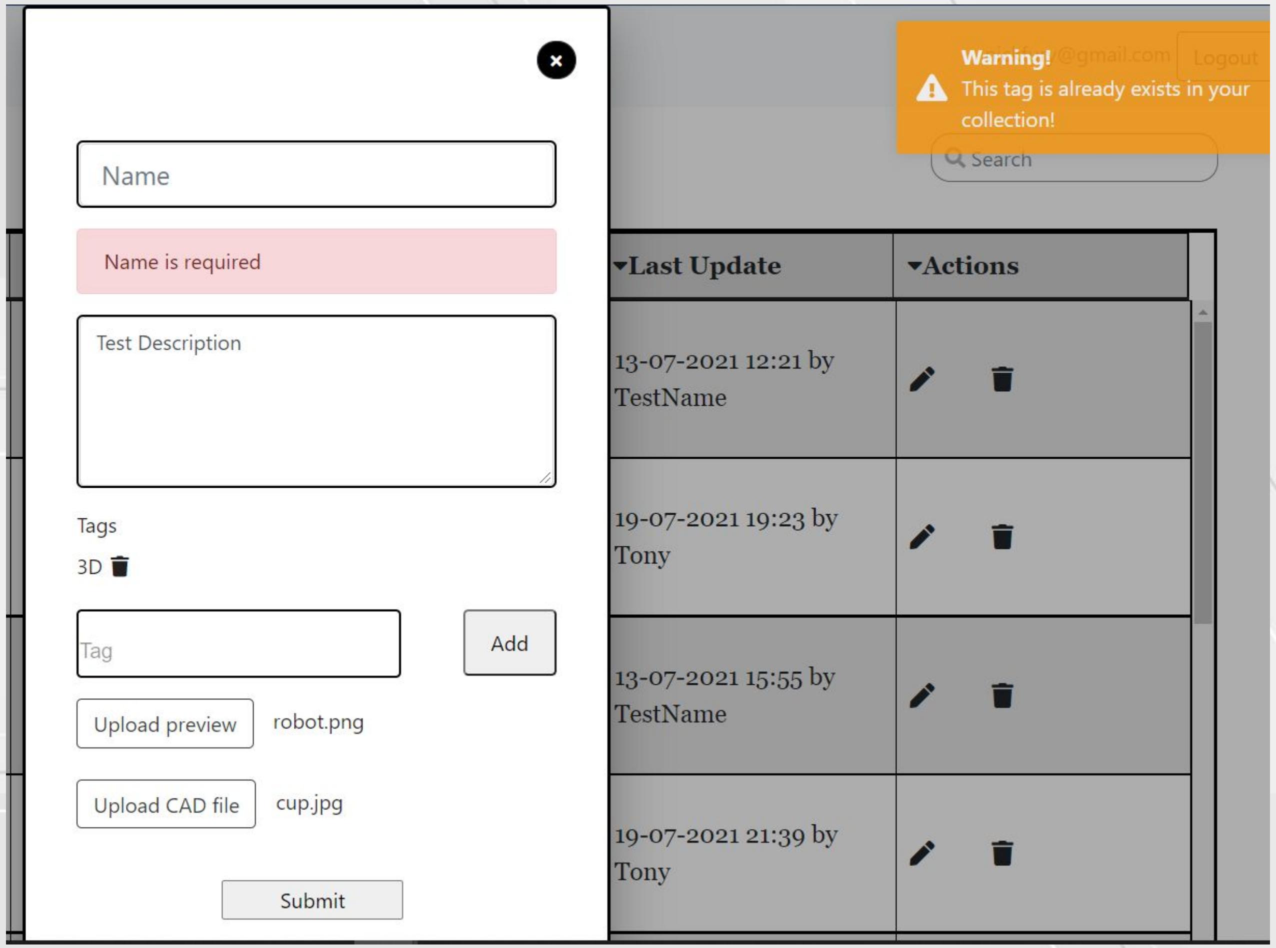
Password is required

User

Choose File No file chosen

Submit

Role
User
User
User
User
Admin



Warning! This tag is already exists in your collection!

Logout

Search

Name

Name is required

Test Description

Tags

3D

Tag

Add

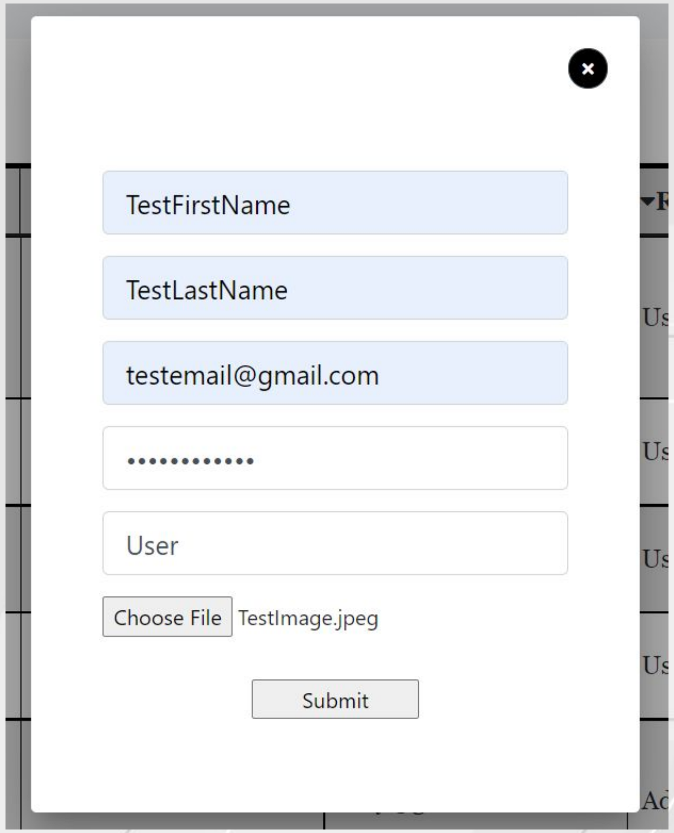
Upload preview robot.png

Upload CAD file cup.jpg

Submit

Last Update	Actions
13-07-2021 12:21 by TestName	
19-07-2021 19:23 by Tony	
13-07-2021 15:55 by TestName	
19-07-2021 21:39 by Tony	

Create new data



TestFirstName

TestLastName

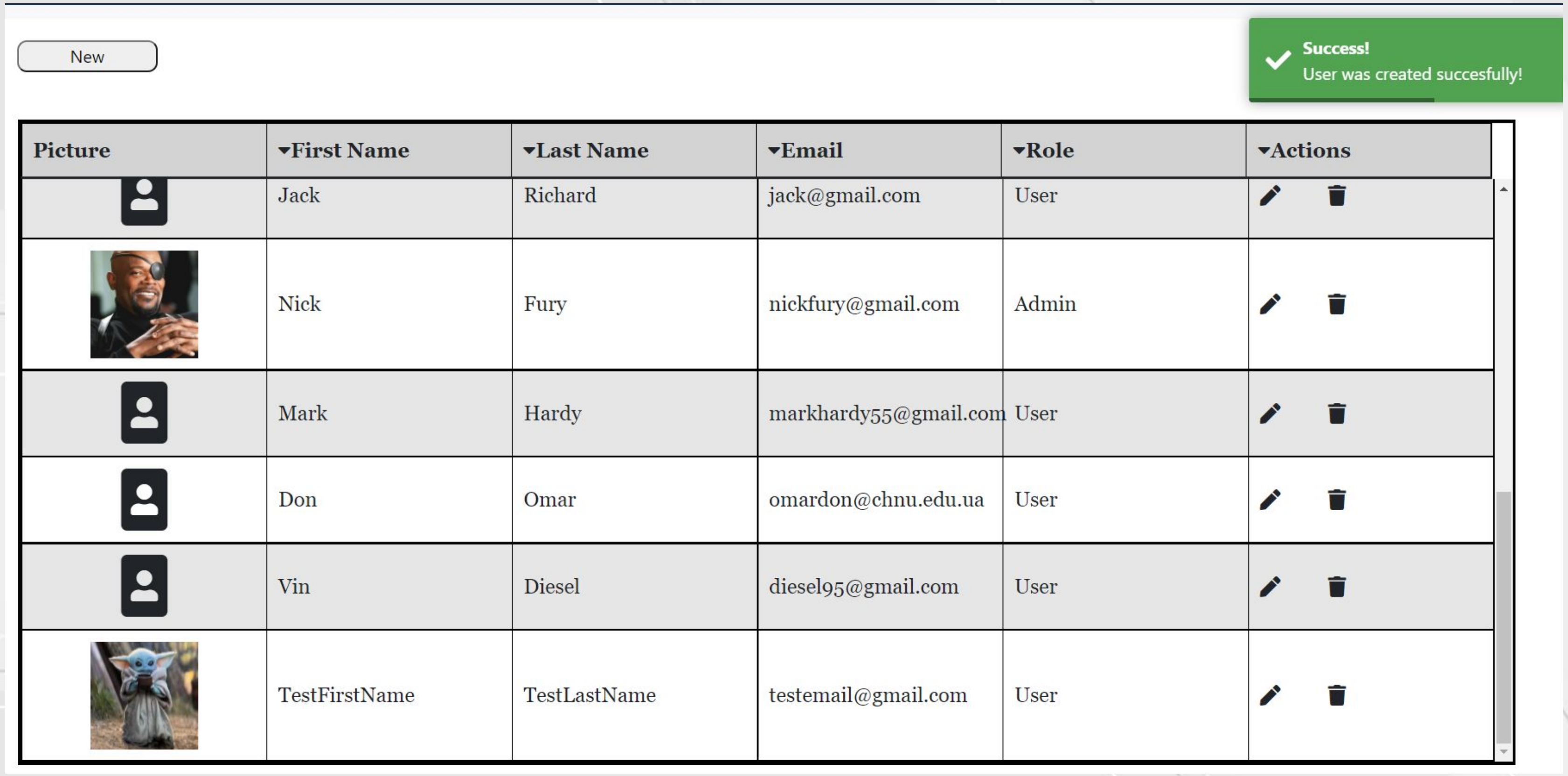
testemail@gmail.com

•••••••••••••

User

Choose File TestImage.jpeg

Submit



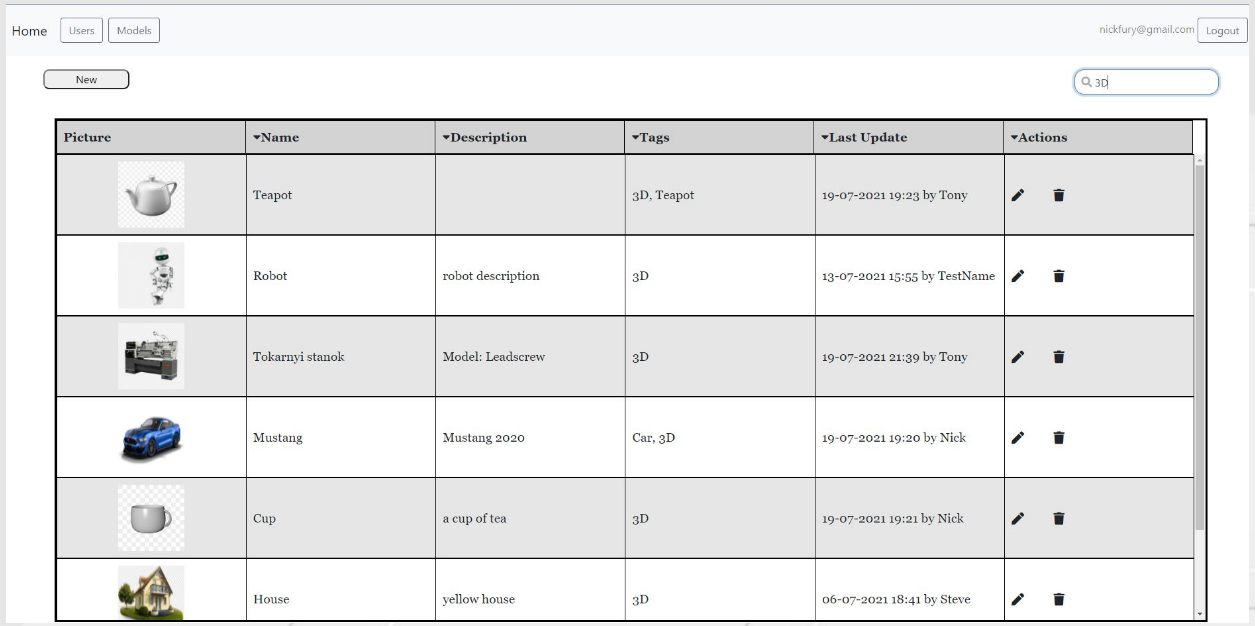
New

Success!
User was created succesfully!

Picture	First Name	Last Name	Email	Role	Actions
	Jack	Richard	jack@gmail.com	User	
	Nick	Fury	nickfury@gmail.com	Admin	
	Mark	Hardy	markhardy55@gmail.com	User	
	Don	Omar	omardon@chnu.edu.ua	User	
	Vin	Diesel	diesel95@gmail.com	User	
	TestFirstName	TestLastName	testemail@gmail.com	User	

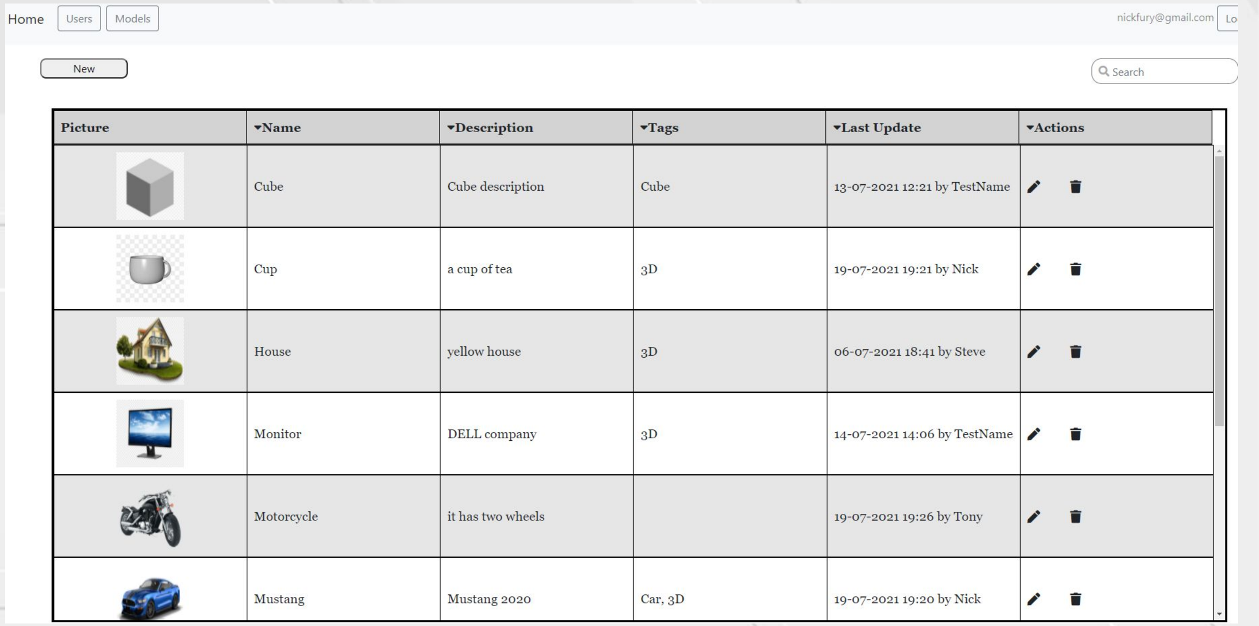
Search and filter data

Search by tag '3D'



Picture	Name	Description	Tags	Last Update	Actions
	Teapot		3D, Teapot	19-07-2021 19:23 by Tony	
	Robot	robot description	3D	13-07-2021 15:55 by TestName	
	Tokarnyi stanok	Model: Leadscrew	3D	19-07-2021 21:39 by Tony	
	Mustang	Mustang 2020	Car, 3D	19-07-2021 19:20 by Nick	
	Cup	a cup of tea	3D	19-07-2021 19:21 by Nick	
	House	yellow house	3D	06-07-2021 18:41 by Steve	

Filter by the field 'Name'



Picture	Name	Description	Tags	Last Update	Actions
	Cube	Cube description	Cube	13-07-2021 12:21 by TestName	
	Cup	a cup of tea	3D	19-07-2021 19:21 by Nick	
	House	yellow house	3D	06-07-2021 18:41 by Steve	
	Monitor	DELL company	3D	14-07-2021 14:06 by TestName	
	Motorcycle	it has two wheels		19-07-2021 19:26 by Tony	
	Mustang	Mustang 2020	Car, 3D	19-07-2021 19:20 by Nick	

Team Members

AMCBRIDGE
Engineering Software Development Made Easy

- Oleksandr Ohorodnik - Team leader
- Mykola Golovach - Developer

THANK YOU!

Question
s
&
Answers