# Ülesanne

Luua *Single Page Application* (SPA) rakendus, mis võimaldab filme otsida pealkirja ja kategooria järgi ning filmide detailinfot vaadata.

## Tehnoloogiad

Backend: ASP.NET MVC

Front-end: kasutada kaasaegset javascript raamistikku nt. Angular, Vue.js, React vms. ning CSS raamistikku nt. Bootstrap, Foundation vms.

## Lahenduse käik

Ülesanne on jagatud mitmeks etapiks. Ei ole kohustuslik lahendada kõiki etappe. Alusta algusest ning tee ära endale jõukohased etapid.

### Etapp 1

1. Kuva filmide list. Listis kuva filmi pealkiri, kategooria, aasta ja hinnang.

### Etapp 2

1. Lisa otsing otsimaks filme pealkirjade kaupa;
2. Täienda otsingut *autocomplete* funktsionaalsusega.

### Etapp 3

1. Lisa filmide kategooriate kaupa filtreerimise võimalus;
2. Täienda filtreerimise funktsionaalsust nii, et kasutaja saaks valida korraga mitu kategooriat.

### Etapp 4

1. Lisa filmi detailinfo vaatamise funktsionaalsus. Kuva detailinfos filmi pealkiri, aasta, hinnang ja kirjeldus.

Kui etappides 1-4 kirjeldatu on lahendatud ja sul on aega ning ideid, siis võid rakendust täiendada. Siinkohal võid mõelda ka ühiktestide lisamise peale.

## Andmed ja äriloogika

1. Loo klassid: Movie [id, title, year, description, rating, category id], Movies list ja Category (id, name);
2. Loo filmide Repository klass:
   * meetod, mis tagastab fikseeritud (hard coded) listi filmidest;
   * meetod, mis tagastab filmi fikseeritud listist id järgi.
3. Loo filmide service klass:
   * list of movies (tagastab filmide listi Repository klassi meetodist);
   * valitud filmi detailinfo (tagastab id järgi filmi Repository klassist).
4. Loo MVC Controller, mis tagastab filmide listi ja filmi detailinfo (kasutades filmi service klassi) json formaadis:
   * API meetod, mis tagastab filmide listi;
   * API meetod, mis tagastab filmi detailinfo.

Repository ja service klasside loomine võimaldab hilisemates iteratsioonides repository implementatsiooni lihtsasti muuta. Lisaks võimaldab see hoida äriloogika andmetest sõltumatuna.

## Dokumentatsioon

Dokumenteeri proovitöö lahendamise käik. Pane kirja tööks kulunud aeg ja tee märkmeid selle kohta, mis oli lihtne ja mis oli raske. Kui jääd mõne probleemiga jänni, siis kirjuta, kust said abi ja kuidas probleemi lahendasid. Kui probleem jääb koodis lahendamata, siis kirjelda, kuidas sinu arvates seda probleemi võiks lahendada. Kui ülesande püstituses on sinu jaoks mitmeti mõistetavaid kohti, siis kirjelda neid ja märgi ära eeldused, mida oled ülesande lahendamisel teinud.

Juhul, kui taaskasutasid lahenduses näidisprojektidest või näiteks Stackoverflow’st kopeeritud pikemaid koodijuppe, siis palun viita neile dokumentatsioonis ja kommentaaridega koodis, et oleks võimalik aru saada, milline oli sinu panus.

Kirjuta ka milliseid arendusvahendeid kasutasid ja kirjelda paari sõnaga oma valikut ning kuidas neist sulle abi oli.

Proovitöös hindame rakenduse vastavust püstitatud ülesandele, tehnilist lahendust ja lahenduse dokumentatsiooni.

## Lahenduse esitamine

**Tähtaeg: 22.aprill kell 20.00.**

**Lahenduse esitamine:** Ülesande kõikide tulemite üleandmiseks kasutada avalikku repositooriumi – Github, Bitbucket vms.

**Kinnituskiri viitega saada kindlasti ka emailile** [**hr.ee@cgi.com**](mailto:hr.ee@cgi.com)