Savonia-ammattikorkeakoulun logo


Muu raportti - Ammattikorkeakoulututkinto

Tekniikan ja liikenteen ala

Eka VAADIN

Java-web ohjelmoinnin kurssin harjoitustyö

TEKIJÄ Viljami Rossinen

SISÄLTÖ

[1 OHJELMAN YLEISKUVAUS 3](#_Toc196571716)

[2 SOVELLUKSEN KÄYTTÖOHJE 4](#_Toc196571717)

[2.1 Aloitussivu (Event view) 4](#_Toc196571718)

[2.2 Kirjautuminen 4](#_Toc196571719)

[2.3 Tapahtumien hallintasivu (Event View) 5](#_Toc196571720)

[2.4 Locations sivu 6](#_Toc196571721)

[2.5 Organizers sivu 7](#_Toc196571722)

[2.6 Participants sivu 8](#_Toc196571723)

[2.7 Admin sivu ja käyttöoikeudet 8](#_Toc196571724)

[3 SOVELLUKSEN TÄRKEIMMÄT TIEDOSTOT 8](#_Toc196571725)

[3.1 Käyttöliittymänäkymät 8](#_Toc196571726)

[3.2 Kooditiedostot ja luokat 9](#_Toc196571727)

[4 itsearviointi JA TOTEUTETUT OMINAISUUDET 10](#_Toc196571728)

[4.1 Itsearviointi 10](#_Toc196571729)

[4.2 Toteutetut ominaisuudet (arviointimatriisin mukaan): 10](#_Toc196571730)

[4.2.1 Data ja entiteetit (5/5 pistettä): 10](#_Toc196571731)

[4.2.2 Suodattimet (5/5 pistettä): 10](#_Toc196571732)

[4.2.3 Tyylit (5/5 pistettä): 10](#_Toc196571733)

[4.2.4 Ulkoasu (5/5 pistettä): 11](#_Toc196571734)

[4.2.5 Autentikointi ja tietoturva (4/5 pistettä): 11](#_Toc196571735)

[4.2.6 Lisätoiminnallisuudet (3 pistettä): 11](#_Toc196571736)

# OHJELMAN YLEISKUVAUS

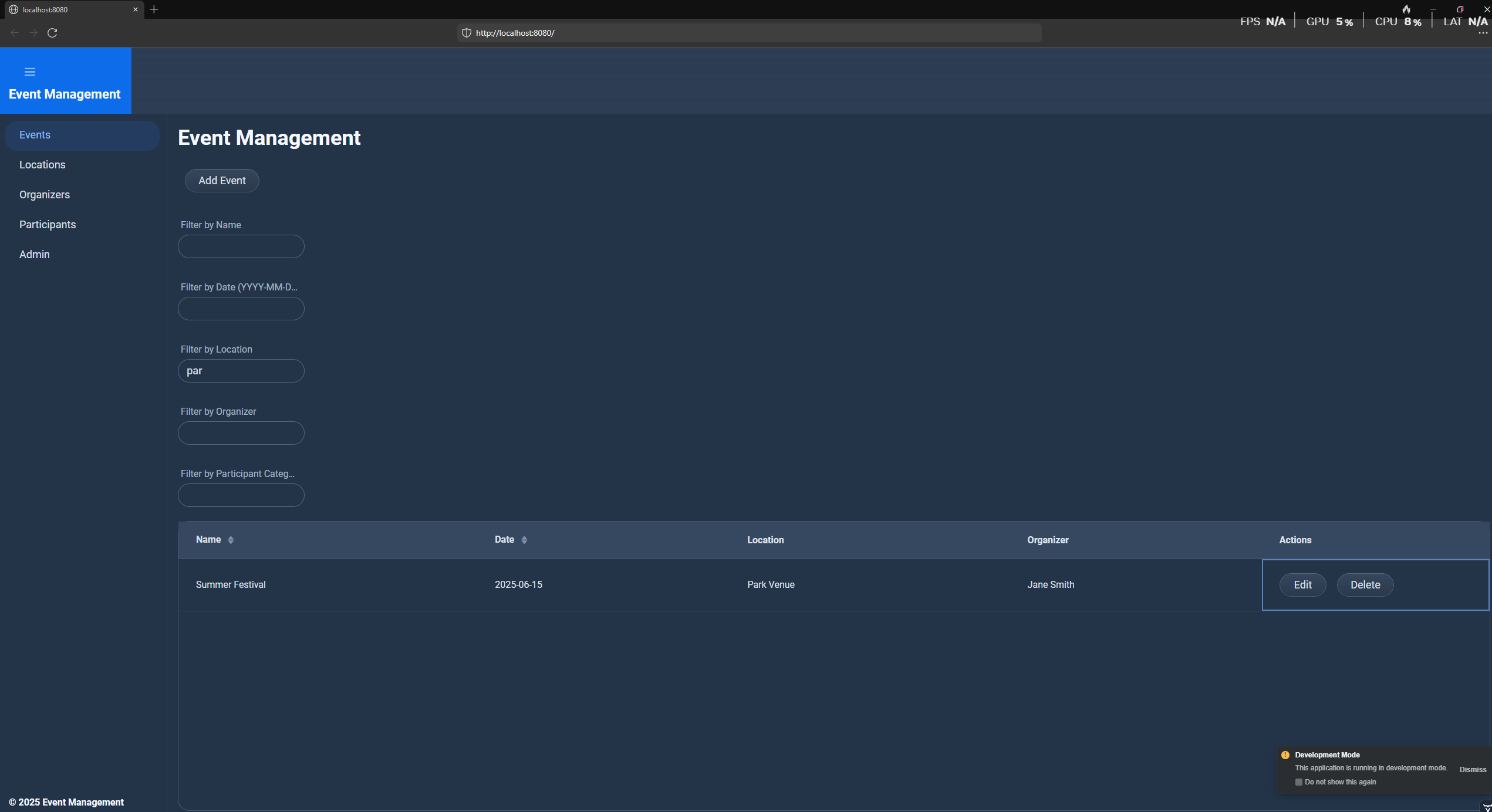
Eka Vaadin on web-sovellus, joka on suunniteltu tapahtumien hallintaan ja tilastointiin. Sovellus tukee tapahtumien, paikkojen, järjestäjien ja osallistujien hallintaa sekä tarjoaa käyttöoikeuksien hallinnan eri käyttäjärooleille (USER ja ADMIN). Sovellus on rakennettu Java-pohjaisella Spring Boot -kehyksellä ja Vaadin Flow -käyttöliittymäkirjastolla, ja se käyttää relaatiotietokantaa (H2) tietojen tallentamiseen.

Sovelluksessa on mahdollista luoda, muokata ja poistaa tapahtumia, paikkoja, järjestäjiä ja osallistujia. Tapahtumien hallintasivulla (EventView) voi suodattaa tapahtumia esimerkiksi nimen, päivämäärän, paikan, järjestäjän ja osallistujien kategorian perusteella. Sovellus tukee relaatiomallinnusta: tapahtumilla on yksi paikka (1-to-1), yksi järjestäjä (1-to-M) ja useita osallistujia (M-to-M). Käyttöoikeuksien hallinta mahdollistaa sen, että vain ADMIN-roolin käyttäjät pääsevät admin-sivulle, kun taas USER-roolin käyttäjät voivat hallinnoida tapahtumia, paikkoja, järjestäjiä ja osallistujia.

# SOVELLUKSEN KÄYTTÖOHJE

## Aloitussivu (Event view)

Tapahtumasivu on avoin kaikille ilman kirjautumista ja toimii aloitussivuna sovellukselle. Tapahtumia voi selata ja suodattaa, mutta niitä ei voi lisätä, muokata tai poistaa ilman kirjautumista. (katso KUVA 1)



KUVA 1. Sovelluksen aloitussivu (Event view).

## Kirjautuminen

Voidakseen kirjautua käyttäjän tulee valita sivupaneelista jokin toinen sivu esim. Locations. Jos käyttäjä ei ole kirjautunut avautuu kirjautumissivu (Login View). Käyttäjän on kirjauduttava sisään voidakseen käyttää sovellusta. Sovelluksessa on kaksi esimerkkikäyttäjää:

* User: Käyttäjätunnus: user, **salasana: userpass** (rooli: USER)
* Admin: Käyttäjätunnus: admin, **salasana: adminpass** (rooli: ADMIN)

Kirjautumissivulla syötetään käyttäjätunnus ja salasana, ja painetaan "Login"-painiketta. Onnistuneen kirjautumisen jälkeen käyttäjä ohjataan aiemmin valikosta valitulle sivulle (esim. Locations View). Jos kirjautuminen epäonnistuu, näytetään virheviesti. Kirjautumissivu on kaikkien käyttäjien nähtävissä, myös kirjautumattomien. Kirjautumisen jälkeen kaikki toiminnot ovat käytettävissä mm. tapahtumien muokkaus ja poisto.

A screenshot of a login screen

AI-generated content may be incorrect.

KUVA 2. Kirjautumissivu.

## Tapahtumien hallintasivu (Event View)

Tapahtumien hallintasivu (EventView) on sovelluksen päänäkymä, jonka toiminnallisuudet avautuvat kirjautumisen jälkeen. Sivulla näytetään lista kaikista tapahtumista Grid-komponentissa, ja tapahtumia voi suodattaa seuraavilla kriteereillä:

* Nimi (nameFilter)
* Päivämäärä (dateFilter)
* Paikka (locationFilter)
* Järjestäjä (organizerFilter)
* Osallistujien kategoria (participantFilter)

Uusi tapahtuma lisätään painamalla "Add Event" -painiketta, joka avaa ikkunan. Ikkunassa syötetään tai valitaan tapahtuman tiedot (nimi (Name), päivämäärä (Date), paikka (Location), järjestäjä (Organizer) ja osallistujat (Participants)), ja tallennetaan painamalla "Save". Location, organizer ja Participants valitaan olemassa olevista vaihtoehdoista. Uusia vaihtoehtoja voi lisätä kyseisten sivujen toimintojen kautta esim. Location sivulla voidaan lisätä uusi paikka tapahtumille, jolloin se tulee valittavaksi myös event sivulle. Tapahtumia voi muokata ja poistaa Grid-komponentin "Edit"- ja "Delete"-painikkeilla. Kaikki muutokset tallennetaan automaattisesti tietokantaan (H2).

Navigointipalkki (sivupalkki) sisältää linkit muihin näkymiin: "Events", "Locations", "Organizers", "Participants" ja "Admin". Footerissa on "Logout"-painike, jolla käyttäjä voi kirjautua ulos ja palata kirjautumissivulle.

A screenshot of a computer

AI-generated content may be incorrect.

KUVA 3. Event view ja uuden tapahtuman lisäys-ikkuna.

## Locations sivu

Locations sivulla voidaan hakea tapahtumien järjestyspaikkoja nimen perusteella. Lisäksi voidaan lisätä uusi paikka Add Location -painikkeesta. Tästä avatuu ikkuna jossa voidaan syöttää uuden paikan tiedot. Nimi (name) kenttä on pakollinen ja osoite (address) vapaaehtoinen.

A screenshot of a computer

AI-generated content may be incorrect.

KUVA 4. Location sivu ja uuden paikan lisäys-ikkuna.

## Organizers sivu

Organizer sivulla voidaan hakea tapahtumien järjestäjiä nimen perusteella. Lisäksi voidaan lisätä uusia järjestäjiä Add Organizer -painikkeella. Tästä avautuu ikkuna jossa voidaan syöttää uuden järjestäjän tiedot. Nimi (name) on pakollinen tieto ja yhteystieto (contact) on vapaaehtoinen.

A screenshot of a computer

AI-generated content may be incorrect.

KUVA 5. Organizers sivu ja uuden järjestäjän lisäys-ikkuna.

## Participants sivu

Participants sivulla voidaan hakea tapahtumien osallistujia nimen ja kategorian perusteella. Lisäksi voidaan lisätä uusi osallistuja Add Participant -painikkeella. Tästä avautuu ikkuna jossa voidaan syöttää uuden osallistujan tiedot. Nimi (name) on pakollinen ja email (s-posti) sekä category (kategoria, esim. speaker = puhuja tilaisuudessa) ovat vapaaehtoisia.

A screenshot of a computer

AI-generated content may be incorrect.

KUVA 6. Participants sivu ja uuden osallistujan lisäys-ikkuna.

## Admin sivu ja käyttöoikeudet

Admin-sivu (AdminView) on vain ADMIN-roolin käyttäjille tarkoitettu näkymä. Sivulla näytetään yksinkertainen teksti: "Admin Dashboard" ja "This page is only accessible to admins." (lokalisoitu suomeksi: "Admin-hallintapaneeli" ja "Tämä sivu on vain ylläpitäjien käytettävissä."). Huomio: sivulle pääsee vain jos on kirjautunut sisään tunnuksella admin (salasana: adminpass).

# SOVELLUKSEN TÄRKEIMMÄT TIEDOSTOT

## Käyttöliittymänäkymät

Sovelluksen käyttöliittymä on rakennettu Vaadin Flow -kirjaston avulla ja tärkeimmät näkymät ovat:

* LoginView.java: Kirjautumissivu, joka käyttää LoginOverlay-komponenttia.
* EventView.java: Päänäkymä tapahtumien hallintaan, sisältää Grid-komponentin, suodattimet ja dialogin.
* LocationView.java, OrganizerView.java, ParticipantView.java: Näkymät paikkojen, järjestäjien ja osallistujien hallintaan.
* AdminView.java: Admin-sivu, joka on suojattu @RolesAllowed("ADMIN")-annotaatiolla ja lokalisoitu.
* ErrorView.java: Virhesivu, joka näytetään, kun käyttöoikeus evätään. Huomio: navigointia tänne ei saatu toimimaan ja näkyviin tulee vain Vaatimen oletusvirheilmoitus.
* MainLayout.java: Pääasettelu, joka sisältää navigointipalkin (SideNav), headerin ja footerin.

Näkymät on tyylitelty styles.css, main-layout.css ja event-view.css-tiedostoilla, ja ne hyödyntävät Lumo-teemaa sekä kustomoituja tyylejä.

## Kooditiedostot ja luokat

Sovelluksen toiminnallisuus on toteutettu Java-luokissa, jotka on jaettu useisiin paketteihin:

* Entiteetit (com.example.application.data):
* Event, Location, Organizer, Participant, User: JPA-entiteetit, jotka mallintavat tietokantatauluja. Relaatiot on määritelty annotaatioilla (@OneToOne, @ManyToOne, @ManyToMany).
* Repositoryt (com.example.application.data):
* EventRepository, LocationRepository, OrganizerRepository, ParticipantRepository, UserRepository: Käyttöliittymän ja tietokannan välinen rajapinta, joka mahdollistaa CRUD-operaatiot.
* Servicet (com.example.application.data):
* EventService, LocationService, OrganizerService, ParticipantService, UserService: Liiketoimintalogiikka, joka käsittelee entiteettien tallennuksen, haun, muokkauksen ja poiston.

Turvallisuus (com.example.application.security):

* SecurityConfig: Määrittää Spring Securityn asetukset, kuten käyttöoikeudet ja kirjautumissivun.
* CustomAccessDeniedHandler: Käsittelee AccessDeniedException-poikkeukset ja ohjaa /error-reittiin. Huomio: ei saatu toimimaan kunnolla.
* AuthenticatedUser: Hallinnoi kirjautuneen käyttäjän tietoja.

Lokalisointitiedostot:

* translations.properties ja translation\_fi.properties: Sisältävät lokalisoituja tekstejä AdminView-sivulle.

# itsearviointi JA TOTEUTETUT OMINAISUUDET

## Itsearviointi

Vaadin ei ollut entuudestaan tuttu juttu, mutta luentomateriaalin ja Vaatimen omien sivujen kautta aloin tutustua aiheeseen. Juuri ennen tätä harjoitustyötä olin tehnyt (ryhmässä) Ohjelmistotuotanto 2 kurssin web-sovelluksen ruokaresepteille ja aluksi ajattelin, että tämä tulisi olemaan kevyempi versio siitä. Tämä piti osittain paikkaansa, mutta Vaatimen käyttäminen oli mielestäni hankalampaa kuin aiemmin käyttämäni tekniikat esimerkiksi React ja Node.js. Tein kuitenkin Vaatimen generaattorilla perusluurangon, jonka päälle aloin kasaamaan (tekoälyn avustamana) vaadittua sovellusta. Peruspohjan ja useiden ominaisuuksien tekeminen sujui melko nopeasti, mutta tietyt asiat osoittautuvat hieman hankalaksi toteuttaa. Projektin loppuvaiheessa ilmenneet navigointiongelmat (esim. ErrorView-ohjauksen kanssa) hidastivat työskentelyä ja lopulta luovuin joistakin vaatimuksista saadakseni sovelluksen valmiiksi. Sovellus kuitenkin toimii vaaditun mukaisesti ja kyseessä olevat puutteet ovat enemmän kauneusvirhetyyppisiä.

## Toteutetut ominaisuudet (arviointimatriisin mukaan):

### Data ja entiteetit (5/5 pistettä):

Toteutin neljä entiteettiä (Event, Location, Organizer, Participant) ja niiden relaatiot:

* 1-to-1: Event ja Location
* 1-to-M: Event ja Organizer
* M-to-M: Event ja Participant

Kaikille entiteeteille on toteutettu CRUD-operaatiot (lisäys, haku, muokkaus, poisto) käyttöliittymältä tietokantaan.

### Suodattimet (5/5 pistettä):

* Toteutin viisi suodatinta EventView-näkymässä:
  + Nimi, päivämäärä, paikka, järjestäjä ja osallistujien kategoria.
  + Suodattimet toimivat Grid-komponentin kanssa ja mahdollistavat relaatioiden (esim. Location, Organizer) suodattamisen.

### Tyylit (5/5 pistettä):

* Muutin globaaleja tyylejä (styles.css), käytin Lumo Utility -luokkia ja lisäsin kustomoituja tyylejä (event-view.css, main-layout.css).
* Tyylien muokkaus tehtiin sekä näkymä- että komponenttitasolla CSS-luokilla.

### Ulkoasu (5/5 pistettä):

* Sovellus on SPA (Single Page Application), jossa on päänäkymä (EventView), header, navigointipalkki, footer ja viisi sisältösivua (EventView, LocationView, OrganizerView, ParticipantView, AdminView).

### Autentikointi ja tietoturva (4/5 pistettä):

* Käytin Spring Securityä (SecurityConfig), loin sisäänkirjautumissivun (LoginView) ja määritin roolit (User ja Admin).
* Kaikki käyttäjät näkevät päänäkymän, USER ja ADMIN näkevät hallintasivut, ja vain ADMIN näkee AdminView-sivun.
* Lisäsin kustomoidun virheviestin (ErrorView), joka näytetään, kun USER yrittää päästä Admin-sivulle. Tämä on luotu, mutta navigointia ei toimi.

### Lisätoiminnallisuudet (3 pistettä):

* GIT-julkaisu (1 piste): Projekti on julkaistu GIT:issä.
* Salasanojen salaus (1 piste): Käytin BCryptPasswordEncoder-luokkaa salasanojen salaamiseen.
* Lokalisointi (1 piste): Toteutin lokalisoinnin AdminView-sivulla (translation.properties ja translation\_fi.properties).

Server push, Docker-image ja Docker Compose jäivät toteuttamatta.