Osa 4

Sovelluksen siirtäminen pilvipalveluun

Oppimistavoitteet

- Tiedät mitä muutoksia sovellukseen tulee tehdä, jotta se toimii pilvipalvelussa (tässä Heroku).
- Osaat siirtää sovelluksesi pilvipalveluun kaikkien nähtäville.

Tutustutaan seuraavaksi sovelluksen siirtämiseen verkkoon kaikkien saataville. Käytämme tässä Herokua, joka on rajoitetun (ja ilmaisen) sijoituspaikan sovelluksille tarjoava pilvipalvelu. Herokun ilmaisversiossa sovellus voi olla käynnissä joitakin tunteja päivässä, jonka lisäksi tietokannassa voi olla kerrallaan korkeintaan 10000 riviä. Tämä on riittävä kurssimme puitteissa.

Heroku tarjoaa oppaan Spring Boot -sovelluksen käyttöönottoon Herokussa. Käy opas läpi nyt.

Tarkastellaan seuraavaksi miten kurssin tehtävän voi lisätä Herokuun. Lisäämme Herokuun neljännen osan ensimmäisen tehtävän "Jokes". Tehtävän lisääminen verkkoon sisältää muutaman askeleen.

1. Herokun käyttämän tietokannanhallintajärjestelmän ajurin määrittely.
2. Tuotantokonfiguraation määrittely.
3. Sovelluksen käynnistämiseen käytetyn ohjetiedoston määrittely.
4. Sovelluksen lisäämisen versionhallintaan.
5. Heroku-projektin luomisen sekä sovelluksen lähettämisen herokuun.

Tietokannanhallintajärjestelmän ajurin määrittely

Sovelluksen käytössä olevaan tiedostojärjestelmään mahdollisesti tehtävät muutokset eivät säily sovelluksen uudelleenkäynnistyksen yhteydessä, joten tähän asti käyttämämme tiedostopohjainen H2-tietokannanhallintajärjestelmä ei sovellu sovellukseemme. Heroku käyttää oletuksena PostgreSQL-tietokannanhallintajärjestelmää, jonka se tarjoaa käyttöömme erillisenä palveluna.

Jotta voimme käyttää PostgreSQL-tietokannanhallintajärjestelmää, tarvitsemme sovellukseemme PostgreSQL-ajurin. Ajurin saa käyttöön lisäämällä projektin `pom.xml`-tiedostoon seuraavan riippuvuuden.

```
<dependency>
    <groupId>org.postgresql</groupId>
    <artifactId>postgresql</artifactId>
</dependency>
```

Kun yllä oleva riippuvuus on määritelty projektiin, on ajuri käytössämme. Mikäli ajuria ei ole aiemmin ladattu koneelle, NetBeansissa joutuu klikkaamaan projektiin liittyvää `Dependencies`-kohtaa ja valitsemaan `Download Declared Dependencies`.

Tuotantokonfiguraation määrittely

Luodaan projektille seuraavaksi tuotantopalvelimen asetukset sisältävä konfiguraatiotiedosto. Käytämme tuotantoprofiilin nimenä merkkijonoa `production`, joten luodaan sovelluksemme kansioon `src/main/resources` tiedosto `application-production.properties`. Oletuksena konfiguraatio peritään `application.properties`-tiedostosta. Tehtäväpohjan tiedostossa määritellään mm. Hibernaten käyttämä kieli, joka on asetettu noudattamaan H2-tietokannanhallintajärjestelmän käytänteitä. Tämä tulee muuttaa.

Heroku syöttää käyttöömme merkittävän osan tietokanta-asetuksista valmiiksi, joten pääsemme konfiguraatiossa melko kevyellä. Ainoa oleellinen konfiguraatio on PostgreSQL-tietokannanhallintajärjestelmän käytänteiden seuraaminen. Konfiguraatiotiedoston sisällöstä tulee seuraava.

```
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialec
```

Sovelluksen käynnistämiseen käytetty ohjetiedosto

Heroku käynnistää sovelluksen sovelluksessa olevan `Procfile`-tiedost perusteella. Lisätään sovelluksen juurikansioon tiedosto `Procfile` ja

määritellään siihen ohjeet sovelluksen käynnistämiseen. `Procfile`-tiedoston sisällöksi tulee seuraava:

```
web: java $JAVA_OPTS -Dspring.profiles.active=production -Dserver.port=$PORT -
```

Yllä oleva ohjeistus kertoo Herokulle miten sovellus tule käynnistää ja mitä muuttujia sovellukselle tulee antaa käynnistyksen yhteydessä. Yllä sovellukselle annetaan parametrina profiili 'production` sekä portti, johon palvelin käynnistyy. Tätä käytetään sen takia, että yksittäisellä Herokun palvelimella voi olla useita sovelluksia samaan aikaan käynnissä — tiettyyn osoitteeseen tulevat pyynnöt ohjataan aina tiettyyn porttiin.

Mikäli yllä olevaa konfiguraatiotiedostoa ei määritellä, Heroku ei käytä `production`-profiilia. Tällöin mm. PostgreSQL-tietokantaa ei saada käyttöön.

Sovelluksen lisääminen git-versionhallintaan

Tässä oletamme, että projektia ei ole lisätty versionhallintaan. Alustetaan projektin juureen git-versionhallinta. Heroku käyttää myös git-versionhallintaa, joten voimme myöhemmin lähettää gitin avulla sovelluksemme Herokuun. Huom! Mikäli projektin juuressa on `target`-kansio, suorita komento `mvn clean` ennen seuraavia askeleita.

```
user@kone:~projekti$ git init
user@kone:~projekti$ git add .
user@kone:~projekti$ git commit -m "ensimmäinen versio"
```

Nyt sovellus on näennäisesti versionhallinnassa (vain paikallisella koneellamme).

Heroku-projektin luominen ja sovelluksen siirtäminen herokuun

Sovelluksen lisääminen herokuun vaatii sen, että luomme ensin Herokuun uuden (tyhjän) sovelluksen. Tämä onnistuu Herokun komentorivisovelluksella seuraavasti.

```
user@kone:~projekti$ heroku create
```

Komento luo Herokuun paikan sovelluksellemme sekä määrittelee versionhallintaamme etäosoitteen `heroku`, johon sovelluksemme voi lähettää. Sovelluksen lähetys Herokuun onnistuu nyt gitin avulla seuraavasti.

```
user@kone:~projekti$ git push heroku master
```

Kun sovellus lähetetään Herokuun, Heroku lataa sovellukseen liittyvät riippuvuudet ja käynnistää sen. Tämän jälkeen sovellus on verkossa kaikkien tarkasteltavana. Voit avata sovelluksen selaimeen komennolla `heroku open`.

```
user@kone:~projekti$ heroku open
```

Sovellus on nyt tarkasteltavissa verkossa. Tätä materiaalia kirjoittaessa Heroku loi sovellukselle osoitteen https://infinite-citadel-61352.herokuapp.com. Mikäli sovellus ei vastaa tai se vastaa hitaasti, kyse on siitä, että sovellusta ei ole käytetty vähään aikaan — Heroku sammuttaa sovelluksen silloin kun se ei ole käynnissä, ja käynnistää sovelluksen kun sitä tarvitaan.

Kun sovellus on Herokussa, voi sen toimintaa tarkastella lokeista. Komento `heroku logs` on erittäin hyödyllinen virheiden etsinnässä. Tulostettavien rivien määrää voi kasvattaa parametrilla `-n` — alla olevassa esimerkissä pyydetään viimeisimmät 500 riviä.

```
user@kone:~projekti$ heroku logs -n 500
```

Muutosten tekeminen sovellukseen

Kun sovellus on Herokussa, muutosten lisääminen sovellukseen on suoraviivaista. Kun lähdekoodia on muokattu ja sovellus toimii paikallisesti, uuden version lähettäminen herokuun onnistuu lisäämällä muutokset gitiin, committaamalla muutokset, ja lähettämällä muutokset Herokuun. Tätä ennen kannattaa ajaa komento `mvn clean`, joka poistaa `target`-kansion.

```
user@kone:~projekti$ mvn clean
user@kone:~projekti$ git add .
user@kone:~projekti$ git commit -m "seuraava versio"
user@kone:~projekti$ git push heroku master
```

Heroku ja PostgreSQL

Jos Herokun PostgreSQL ei lähde edellä kuvatulla esimerkillä käyntiin, tarkasta vielä, että sovellukseen on lisätty Herokussa tietokannanhallintajärjestelmä. Tämä löytyy Herokusta sovelluksen Resources-välilehdeltä kohdasta Add-ons. Tutustumme hieman myöhemmin tietokantamigraatioihin eli tietokannan rakenteen muuntamiseen tarvittaessa — mikäli haluat, että tietokanta alustetaan alusta, Herokun verkkopalvelusta löytyy toiminnallisuus tietokannan resetointiin.

Kysely:

Sovelluksen lisääminen Herokuun

Pisteitä:

1/1

Lisää neljännen osan kolmas tehtävä Herokuun. Kirjoita palautuksen ensimmäiselle riville sovelluksesi osoite Herokussa ja sitä seuraaville riveille sovelluksesi Herokuun lisäämiseen liittyneet mahdolliset ongelmat ja ratkaisut.

Vastauksesi:

https://peaceful-gorge-10517.herokuapp.com/jokes. Sotkeennuin Herokun ja Githubin lukuisiin ohjeisiin, minkä vuoksi ähelsin tämän taskin kanssa useamman tunnin. Lopulta seurailin vain yo.

kurssimateriaalin ohjetta ja homma onnistui helposti. Ongelma taisi olla lähinnä näppäimistöni ja tuolini välissä. Hyvänä puolena mainittokoon, että tuli tutustuttua mm. Githubiin ja myös Herokuun enemmän kuin tämän tehtävän kannalta olisi ollut tarvetta.

Pääsit aliluvun loppuun! Jatka tästä seuraavaan osaan:

→ 7. Yhteenveto

Tässä osassa:

Materiaalin lähdekoodi

Kurssin on tehnyt Helsingin yliopiston Agile Education Research -tutkimusryhmä.

Kiitokset ja materiaalista.

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

MASSIIVISET AVOIMET VERKKOKURSSIT
MASSIVE OPEN ONLINE COURSES · MOOC.FI