



samen sterk voor werk

A small icon of a green leaf with a brown stem and a small black dot at the base, positioned above the letter 'i' in the word 'Spring'.

Spring fundamentals takenbundel

Deze cursus is eigendom van de VDAB©

Inhoudsopgave

1	TAKEN	5
1.1	Frituur Frida.....	5
1.2	Sluitingsdagen	5
1.3	Sluitingsdagen 2	5
1.4	Sluitingsdagen 3	5
1.5	Adres	5
1.6	Sauzen	5
1.7	Voorkeurtaal.....	6
1.8	Bezocht.....	6
1.9	Unit test controller	6
1.10	Sauzen.csv	6
1.11	Sauzen.properties	7
1.12	Application.properties.....	7
1.13	DataSource	7
1.14	Repositories.....	7
1.15	Services.....	8
1.16	Begin naam.....	8
1.17	Client sided validatie	8
1.18	Snack wijzigen	8
1.19	Zoek de friet	8
1.19.1	Tip	9
1.20	Saus raden	9
1.21	Custom tags.....	10
1.22	Gastenboek	10
1.23	Gastenboekbeheer	11
2	VOORBEELDOPLOSSINGEN.....	12

2.1	Frituur Frida	12
2.1.1	index.html	12
2.1.2	frituurfrida.css	12
2.2	Sluitingsdagen	12
2.2.1	IndexController	12
2.3	Sluitingsdagen 2	12
2.3.1	Pom.xml	12
2.3.2	IndexController	13
2.3.3	index.jsp	13
2.3.4	application.properties	13
2.4	Sluitingsdagen 3	13
2.5	Adres	13
2.5.1	Gemeente	13
2.5.2	Adres	13
2.5.3	IndexController: gewijzigd return statement	14
2.5.4	index.jsp	14
2.6	Sauzen	14
2.6.1	head.jsp	14
2.6.2	index.jsp	14
2.6.3	Saus	14
2.6.4	SausController	15
2.6.5	sauzen.jsp	15
2.7	Voorkeurtaal	15
2.7.1	VoorkeurTaalController	15
2.7.2	voorkeurtaal.jsp	16
2.8	Bezocht	16
2.8.1	IndexController	16
2.8.2	index.jsp	16
2.9	Unit test controller	16
2.10	Sauzen.csv	17
2.10.1	SausRepository	17
2.10.2	SausRepositoryException	17
2.10.3	CSVSausRepository	17
2.10.4	SausService	18
2.10.5	DefaultSausService	18
2.10.6	SausController	18
2.10.7	SausControllerTest	18
2.11	Sauzen.properties	19
2.11.1	CSVSausRepository	19
2.11.2	PropertiesSausRepository	19
2.11.3	DefaultSausService	20

2.12	Application.properties.....	20
2.12.1	application.properties.....	20
2.12.2	CSVsausRepository	20
2.12.3	PropertiessausRepository	20
2.13	DataSource	20
2.13.1	pom.xml	20
2.13.2	application.properties.....	20
2.13.3	DataSourceTest	21
2.14	Repositories.....	21
2.14.1	Snack	21
2.14.2	SnackRepository.....	21
2.14.3	SnackNietGevondenException	21
2.14.4	JdbcSnackRepository.....	21
2.14.5	insertSnack.sql	22
2.14.6	JdbcSnackRepositoryTest.....	22
2.14.7	application.properties.....	23
2.15	Services.....	23
2.15.1	SnackService.....	23
2.15.2	DefaultSnackService.....	23
2.15.3	SnackController	23
2.15.4	alfabet.jsp	24
2.16	Begin naam.....	24
2.16.1	BeginNaamForm	24
2.16.2	SnackController.....	25
2.16.3	messages.properties	25
2.16.4	ValidationMessages.properties	25
2.16.5	beginnaam.jsp.....	25
2.16.6	Unit test	26
2.17	Client sided validatie	26
2.17.1	beginnaam.jsp.....	26
2.18	Snack wijzigen	26
2.18.1	alfabet.jsp en beginnaam.jsp	26
2.18.2	Snack	26
2.18.3	SnackController	26
2.18.4	snackwijzigen.jsp.....	27
2.18.5	messages.properties	27
2.18.6	snacknietgevonden.jsp	28
2.19	Zoek de friet	28
2.19.1	Deur	28
2.19.2	ZoekDeFrietSpel	28
2.19.3	DefaultZoekDeFrietSpel	28
2.19.4	FrietController.....	29
2.19.5	zoekdefriet.jsp	29
2.19.6	application.properties.....	30

2.20	Saus raden	30
2.20.1	SausRadenSpel	30
2.20.2	DefaultSausRadenSpel	30
2.20.3	SausRadenForm	31
2.20.4	SausController	31
2.20.5	SausControllerTest	32
2.20.6	sausraden.jsp	32
2.21	Custom tags	33
2.21.1	head.tag	33
2.21.2	menu.tag	33
2.21.3	vdab.tld	33
2.21.4	frituurfrida.css	34
2.21.5	JSP's	34
2.22	Gastenboek	34
2.22.1	Nieuwe table in de database	34
2.22.2	Rechten in de database	34
2.22.3	GastenBoekEntry	34
2.22.4	GastenBoekRepository	34
2.22.5	JdbcGastenBoekRepository	35
2.22.6	insertGastenBoek.sql	35
2.22.7	JdbcGastenBoekRepositoryTest	35
2.22.8	GastenBoekService	36
2.22.9	DefaultGastenBoekService	36
2.22.10	GastenBoekController	36
2.22.11	gastenboek.jsp	37
2.22.12	menu.tag	37
2.22.13	frituurfrida.css	38
2.23	Gastenboekbeheer	38
2.23.1	Rechten in database	38
2.23.2	GastenBoekRepository	38
2.23.3	JdbcGastenBoekRepository	38
2.23.4	JdbcGastenBoekRepositoryTest	38
2.23.5	GastenBoekService	39
2.23.6	DefaultGastenBoekService	39
2.23.7	GastenBoekController	39
2.23.8	gastenboek.jsp	39
3	COLOFON	40

1 TAKEN

1.1 Frituur Frida

Je maakt een project voor een website met een statische HTML welkompagina.

- Deze pagina bevat een `<h1>` element met de tekst Frituur Frida
- Je geeft deze tekst een kleur met een CSS bestand
- Deze pagina gebruikt `frituur.ico` uit het takenmateriaal

1.2 Sluitingsdagen

Frituur Frida is gesloten op maandag en donderdag.

Je voegt een controller toe aan de website die requests verwerkt naar de welkompagina.

Als je naar de website surft op maandag of donderdag, toon je de tekst gesloten.

Op andere dagen toon je de tekst open.

1.3 Sluitingsdagen 2

Frituur Frida is gesloten op maandag en donderdag.

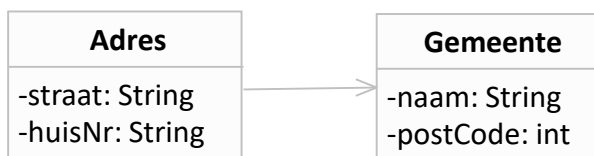
Je verwerkt een request naar de welkompagina eerst in een controller. Deze maakt een String met gesloten (op maandag en donderdag) of open (op andere dagen). De controller geeft deze String als data door aan een JSP. Deze JSP gebruikt EL om met de doorgekregen data de tekst Vandaag zijn we gesloten of Vandaag zijn we open te tonen.

1.4 Sluitingsdagen 3

- Op de dagen dat frituur Frida gesloten is, toon je onder de tekst Vandaag zijn we gesloten de afbeelding `gesloten.png` (uit het materiaal bij deze cursus.)
- Op andere dagen toon je onder de tekst Vandaag zijn we open de afbeelding `open.png` (uit het materiaal bij deze cursus).

1.5 Adres

Je voegt aan het project voor Frituur Frida volgende classes toe:



In de controller die hoort bij de welkompagina

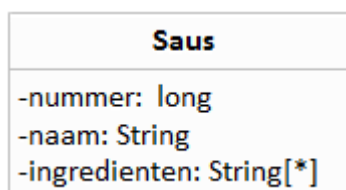
- Maak je een Adres object en een bijbehorend Gemeente object
- Vul je de attributen van deze objecten met zelf verzonden adresgegevens van frituur Frida
- Geef je dit adres object door aan de JSP.

In de JSP toon je het volledige adres van frituur Frida aan de hand van deze data.

1.6 Sauzen

Je maakt een include file `head.jsp` (zoals in de theorie) en past die toe.

Je voegt een class Saus toe:



Je voegt een controller toe die requests verwerkt naar de URL /sauzen

Bij een GET request doe je het volgende:


- Je maakt een `List<Saus>` en je vult die met sauzen met volgende namen: cocktail, mayonaise, mosterd, tartare, vinaigrette.
Je verzint zelf ids en ingrediënten van deze sauzen.
- Je geeft deze verzameling sauzen door aan een JSP.
- Je toont in deze JSP de naam en de ingrediënten van deze sauzen.
Als een saus meerdere ingrediënten heeft, scheid je deze ingrediënten met een komma.
- Je toont bij iedere saus ook een afbeelding van die saus.
Je vindt deze afbeeldingen in het materiaal bij deze cursus.

1.7 Voorkeurtaal

De browser stuurt bij iedere request een header `accept-language` mee. Deze header bevat de taal en eventueel het land van de gebruiker. Als deze header bijvoorbeeld `nl` bevat, betekent dit dat de gebruiker Nederlands spreekt. Als deze header bijvoorbeeld `nl-be` bevat, betekent dit dat de gebruiker Nederlands spreekt en in België woont. Deze header kan ook meerdere taal en taal-land combinaties bevatten. De meest geprefereerde taal of taal-land is als eerste vermeld.

Voorbeeld: `nl-be, en-us`.

De gebruiker bepaalt in de browser instellingen de inhoud van de header. Bij Firefox:

1. Je kiest rechts boven in Firefox .
2. Je kiest Options.
3. Je kiest Choose bij Choose your preferred language for displaying pages.
4. Je kunt taal-land combinaties toevoegen met Select a language to add en Add.
5. Je kunt de volgorde instellen met Move Up en Move Down.

Je toont gebaseerd op deze header een pagina met de tekst `Je voorkeurtaal is Nederlands` of de tekst `Je voorkeurtaal is geen Nederlands`.

1.8 Bezocht

Als een gebruiker voor een tweede, of derde, of ... keer terugkomt op de website toon je in de welkompagina de tekst `Welkom terug`. Je toont deze tekst niet als de gebruiker de eerste keer de website bezoekt.

1.9 Unit test controller

Je schrijft een unit test voor `SausController`.

1.10 Sauzen.csv

Het takenmateriaal bevat een bestand `sauzen.csv`:

```
1,cocktail,mayonaise, ketchup,cognac
2,mayonaise,ei,mosterd
3,mosterd,mosterd,azijn,witte wijn
4,tartare,mayonaise,augurk
5,vinaigrette,olijfolie,mosterd,azijn
```

CSV is een afkorting van comma separated values. Het duidt een tekstbestand aan waarbij de gegevens op een regel gescheiden worden door een komma.

Je maakt een directory `data` in de root van de hard disk en je kopieert het bestand in die directory.

Je maakt een package `be.vdab.frituurfrida.repositories`. Je maakt daarin een interface `SausRepository` met een method `List<Saus> findAll()`. Je implementeert deze interface in een class `CSVSausRepository`. Je maakt van deze class een Spring bean.

Je maakt een package `be.vdab.frituurfrida.services`. Je maakt daarin een interface `SausService` met een method `List<Saus> findAll()`. Je implementeert deze interface in een class `DefaultSausService`. Je maakt van deze class een Spring bean. De code in deze class roept de `CSVSausRepository` bean op om de sauzen uit het CSV bestand te lezen.

Je injecteert de DefaultSausService bean in de SausController. Je roept in de method sauzen de diensten op van deze bean, om de sauzen uit het CSV bestand aan de JSP door te geven.

Je zorgt er voor dat je unit test van SausController terug werkt.

Tip: je kan volgend code fragment gebruiken om een tekstbestand regel per regel te lezen:

```
try (BufferedReader reader =
    Files.newBufferedReader(Paths.get("/data/sauzen.csv"))) {
    for (String regel; (regel = reader.readLine()) != null;) {
        // verwerk de huidige gelezen regel in de variabele regel
    }
} catch (IOException ex) {
    ...
}
```

1.11 Sauzen.properties

Het takenmateriaal bevat een bestand sauzen.properties:

```
1:cocktail,mayonaise, ketchup,cognac
2:mayonaise,ei,mosterd
3:mosterd,mosterd,azijn,witte wijn
4:tartare,mayonaise, augurk
5:vinaigrette,olijfolie,mosterd,azijn
```

Je kopieert dit bestand in de directory data in de root van de harde schijf.

Je maakt in de package be.vdab.frituurfrida.repositories een class PropertiesSausRepository die ook de interface SausRepository implementeert. Je maakt van deze class een Spring bean.

Je gebruikt PropertiesSausRepository als dependency in DefaultSausService.

1.12 Application.properties

Je neemt het pad van sauzen.csv op in application.properties.

Je neemt het pad van sauzen.properties op in application.properties.

Je gebruikt deze instellingen in CSVSausRepository en PropertiesSausRepository.

1.13 DataSource

Je voert het script frituurfrida.sql uit (bij het takenmateriaal). Dit script maakt een database frituurfrida met een table snacks:

snacks	
id	INT
naam	VARCHAR(50)
prijs	DECIMAL(10,2)

Het script geeft aan de gebruiker cursist de nodige rechten op deze table.

Je definieert de verbinding naar deze database in application.properties.

Je schrijft een integration test voor de DataSource.

1.14 Repositories

Je maakt in de package be.vdab.frituurfrida.entities een class Snack.

Je maakt in de package be.vdab.frituurfrida.repositories een interface SnackRepository met 3 methods:

- Optional<Snack> read(long id)
- void update(Snack snack)
- List<Snack> findByBeginNaam(String beginNaam)

Je implementeert deze interface in de class JdbcSnackRepository.

Je schrijft een integration test voor deze class.

1.15 Services

Je maakt in de package `be.vdab.frituurfrida.services` een interface `SnackService` met 3 methods:

- `Optional<Snack> read(long id)`
- `void update(Snack snack)`
- `List<Snack> findByBeginNaam(String beginNaam)`

Je implementeert deze interface in de class `DefaultSnackService`.

Je maakt een pagina. Je toont in die pagina de letters van het alfabet. Iedere letter is een hyperlink. Als de gebruiker zo'n hyperlink aanklikt, toon je terug de pagina. Deze bevat opnieuw de letters van het alfabet als hyperlinks. Je toont daaronder de namen van de snacks waarvan de naam begint met de aangeklikte letter.

1.16 Begin naam

Je toont een form aan de gebruiker. Deze form bevat één invoervak.

De gebruiker tikt in dit invoervak de beginletters van een snack. Hij klikt daarna op een knop Zoeken. Als hij het invoervak leeg gelaten heeft toon je een foutmelding.

Anders toon je de namen van de snacks waarvan de naam begint met de ingetikte beginletters.

Je maakt ook een unit test. Je controleert daar of je de correcte validation annotations gebruikte.

1.17 Client sided validatie

Je past client sided validatie toe in de form van de vorige oefening.

1.18 Snack wijzigen

Je maakt hyperlinks van de namen van snacks

- in de pagina met het alfabet
- in de pagina met het invoervak met beginletters

Als de gebruiker zo'n hyperlink aanklikt,

komt hij op een pagina waar hij de naam en de prijs van de snack kan wijzigen.

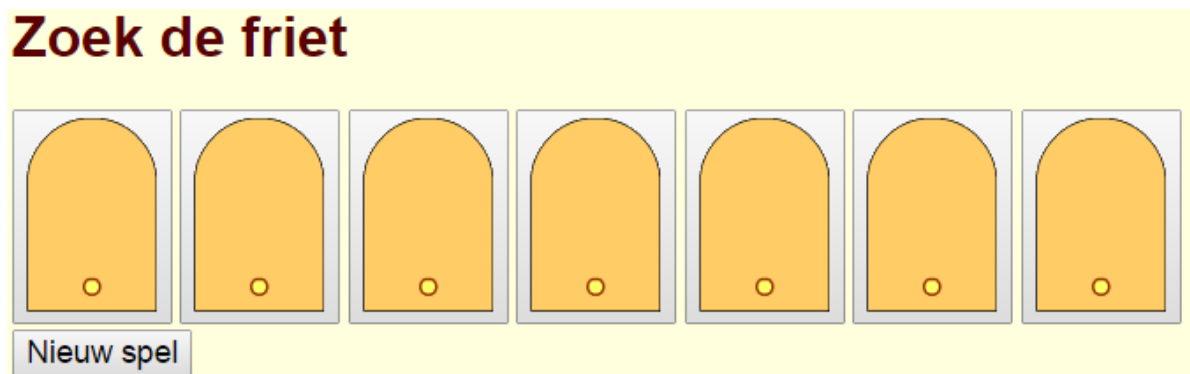
- de naam mag niet leeg zijn.
- de prijs mag niet leeg zijn en moet minstens 0 zijn.

Nadat de gebruiker correcte waarden intikt en op de knop Opslaan klikt, toon je de welkompagina.

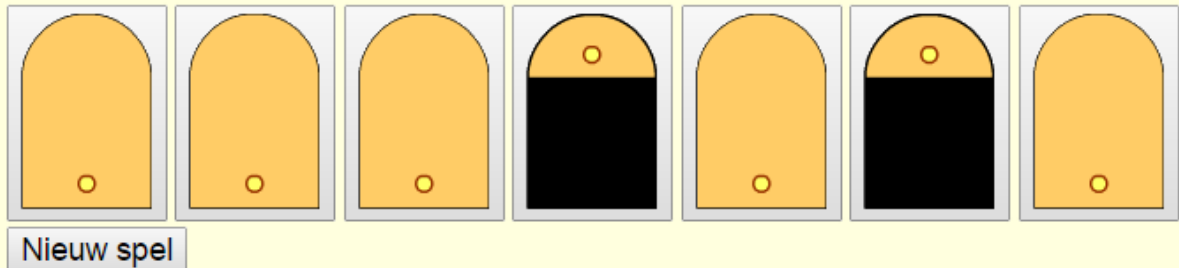
1.19 Zoek de friet

De gebruiker ziet in een pagina zeven gesloten deuren.

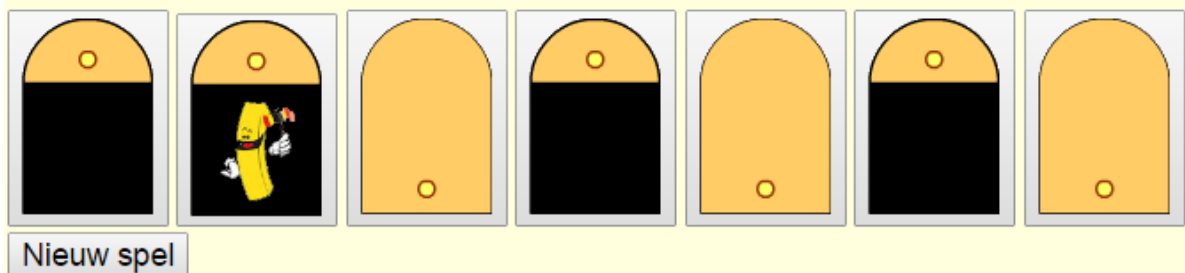
Achter één van de deuren bevindt zich een friet. De gebruiker zoekt deze deur. De gebruiker kan de deuren één per één openen door op een deur te klikken. Eenmaal geopend blijft de deur open.



Zoek de friet



Zoek de friet



Als de gebruiker Nieuw spel kiest, maak je een nieuw spel met zeven gesloten deuren.

Je gebruikt de afbeeldingen `deuropen.png`, `deurtoe.png` en `gevonden.png` bij deze cursus.

1.19.1 Tip

Je maakt met volgende HTML code een afbeelding die een request parameter met de naam `index` en de waarde 3 doorgeeft bij het submitten van de form die de afbeelding bevat:

```
<button type='submit' name='index' value='3'>
  <img src='<c:url value="/images/deurtoe.png"/>' alt='deur toe'>
</button>
```

1.20 Saus raden

Je maakt een nieuwe pagina waarin de gebruiker de naam van één van de sauzen van frituur Frida raadt.

- Je toont naast Te raden saus: evenveel puntjes als de saus die de gebruiker moet raden. Je kiest de saus willekeurig uit de in-memory sauzen database.
- De gebruiker kan naast letter: een letter intikken en op Raden klikken. De gebruiker ziet daarna dezelfde pagina.
 - Als de ingetikte letter voorkomt in de te raden saus, toon je op de plaats van die letter geen puntje, maar de letter. Als de gebruiker bijvoorbeeld de letter *t* tikte en de te raden saus is *cocktail*, toon je Te raden saus:t...

- Als de ingetikte letter niet voorkomt in de te raden saus, verhoog je een teller met het aantal verkeerde pogingen. Je toont deze teller aan de gebruiker via één van de afbeeldingen 0.png ... 9.png bij deze cursus.
Als de gebruiker drie verkeerde pogingen deed, toon je bijvoorbeeld 3.png:

- Als de gebruiker 10 verkeerde pogingen deed, verliest hij het spel en toon je volgende tekst: U bent verloren, de saus was cocktail
- Als de gebruiker de saus raadt en nog geen 10 verkeerde pogingen deed, wint hij het spel en toon je volgende tekst: U bent gewonnen, de saus was cocktail
- De gebruiker kan het spel herbeginnen met de knop Nieuw spel
Jij kiest dan terug een willekeurige saus voor dit nieuwe spel.

1.21 Custom tags

Je maakt een custom tag die de menu voorstelt van de website Frituur Frida.

Je maakt ook een custom tag die de include file head.jsp vervangt.

Je neemt deze custom tags op in elke pagina van de website.

1.22 Gastenboek

Je voegt aan de website voor frituur Frida een gastenboek toe.

Als de gebruiker de pagina opent, ziet hij de berichten die gebruikers al aan het gastenboek toevoegden, in omgekeerde chronologische volgorde:

Je voegt zelf een table toe aan de database om de gastenboekdata bij te houden.

Als de gebruiker Toevoegen kiest, ziet hij de pagina opnieuw, maar kan hij een eigen bericht toevoegen. Hierbij zijn beide vakken verplicht in te tikken.

1.23 Gastenboekbeheer

De gebruiker ziet in het gastenboek bij ieder bericht een checkbox.

De gebruiker ziet in het gastenboek ook een knop Verwijderen.

Als hij deze knop aanklikt, verwijder jij alle aangevinkte berichten.

Tip: je maakt met volgend code fragment de form met checkboxes in de JSP:

```
<c:if test='${not empty gastenboek}'>
  <c:url value='/gastenboek/verwijderen' var='url' />
  <form action='${url}' method='post'>
    <dl>
      <c:forEach var='entry' items='${gastenboek}'>
        <dt><spring:eval expression='entry.datumTijd' /> ${entry.naam}
        <input type='checkbox' name='verwijderid' value='${entry.id}' />
        </dt>
        <dd>${entry.bericht}</dd>
      </c:forEach>
    </dl>
    <input type='submit' value='Verwijderen' />
  </form>
</c:if>
```

Je verwerkt het submitten van de request in volgende controller method:

```
@PostMapping("verwijderen")
String delete(long[] verwijderid) {
  ...
}
```

De parameter verwijderid bevat null als geen enkel vinkje aangeklikt is.

Anders bevat hij de ids van de aangevinkte berichten.

2 VOORBEELDOPLOSSINGEN

2.1 Frituur Frida

2.1.1 index.html

```
<!doctype html>
<html lang='nl'>
  <head>
    <meta charset='UTF-8'>
    <title>Frituur Frida</title>
    <link rel='icon' href='images/frida.ico'>
    <meta name='viewport' content='width=device-width,initial-scale=1'>
    <link rel='stylesheet' href='css/frituurfrida.css'>
  </head>
  <body>
    <h1>Frituur Frida</h1>
  </body>
</html>
```

2.1.2 frituurfrida.css

```
body {
  background-color: #C0BD92;
  font-family: sans-serif;
}
h1 {
  color: red;
}
```

2.2 Sluitingsdagen

2.2.1 IndexController

```
package be.vdab.frituurfrida.web;
// enkele imports
@RestController
@RequestMapping("/")
class IndexController {
  @GetMapping
  String index() {
    LocalDate vandaag = LocalDate.now();
    DayOfWeek weekdag = vandaag.getDayOfWeek();
    return weekdag == DayOfWeek.MONDAY || weekdag == DayOfWeek.THURSDAY ?
      "gesloten" : "open";
  }
}
```

2.3 Sluitingsdagen 2

2.3.1 Pom.xml

```
<dependency>
  <groupId>javax.servlet.jsp</groupId>
  <artifactId>jsp-api</artifactId>
  <version>[2.2,]</version>
  <scope>provided</scope>
</dependency>
<dependency>
  <groupId>jstl</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
  <scope>runtime</scope>
</dependency>
```

```
<dependency>
  <groupId>org.apache.tomcat.embed</groupId>
  <artifactId>tomcat-embed-jasper</artifactId>
  <scope>provided</scope>
</dependency>
```

2.3.2 IndexController

```
package be.vdab.frituurfrida.web;
// enkele imports
@Controller
@RequestMapping("/")
class IndexController {
  @GetMapping
  ModelAndView index() {
    LocalDate vandaag = LocalDate.now();
    DayOfWeek weekdag = vandaag.getDayOfWeek();
    String openGesloten = weekdag == DayOfWeek.MONDAY ||
      weekdag == DayOfWeek.THURSDAY ? "gesloten" : "open";
    return new ModelAndView("index", "openGesloten", openGesloten);
  }
}
```

2.3.3 index.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<!doctype html>
<html lang="nl">
  <head>
    <title>Frituur Frida</title>
    <link rel='icon' href='images/frida.ico'>
    <meta name='viewport' content='width=device-width,initial-scale=1'>
    <link rel='stylesheet' href='css/frituurfrida.css'>
  </head>
  <body>
    <h1>Vandaag zijn we ${openGesloten}</h1>
  </body>
</html>
```

2.3.4 application.properties

```
spring.mvc.view.prefix:/WEB-INF/JSP/
spring.mvc.view.suffix:.jsp
```

2.4 Sluitingsdagen 3

Nieuwe regel voor </body>

```

```

2.5 Adres

2.5.1 Gemeente

```
package be.vdab.frituurfrida.valueobjects;
public class Gemeente {
  private String naam;
  private int postcode;
  // Je maakt zelf getters en setters voor deze private variabelen met Eclipse
  // Je maakt zelf een default en een geparametriseerde constructor met Eclipse
}
```

2.5.2 Adres

```
package be.vdab.frituurfrida.valueobjects;
public class Adres {
  private String straat;
  private String huisNr;
  private Gemeente gemeente;
```

```
// Je maakt zelf getters en setters voor deze private variabelen met Eclipse
// Je maakt zelf een default en een geparametriseerde constructor met Eclipse
}
```

2.5.3 IndexController: gewijzigd return statement

```
return new ModelAndView("index", "openGesloten", openGesloten)
    .addObject(new Adres("Grote markt", "7", new Gemeente("Brussel", 1000)));
```

2.5.4 index.jsp

```
<h2>Adres</h2>
${adres.straat} ${adres.huisNr}<br>
${adres.gemeente.postcode} ${adres.gemeente.naam}
```

2.6 Sauzen

2.6.1 head.jsp

Zelfde als in theorie (maar pizza.ico wordt frida.ico en pizzaluigi.css wordt frituurfrida.css)

2.6.2 index.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<!doctype html>
<html lang="nl">
<head>
<c:import url='/WEB-INF/JSP/head.jsp'>
    <c:param name='title' value='Frituur Frida'>
</c:import>
</head>
...
```

2.6.3 Saus

```
package be.vdab.frituurfrida.entities;
// enkele imports ...
public class Saus {
    private long id;
    private String naam;
    private List<String> ingredienten = new ArrayList<>();
    public Saus(long id, String naam) {
        this.id = id;
        this.naam = naam;
    }
    public Saus(long id, String naam, List<String> ingredienten) {
        this(id, naam);
        this.ingredienten.addAll(ingredienten);
    }
    public long getId() {
        return id;
    }
    public String getNaam() {
        return naam;
    }
    public List<String> getIngredienten() {
        return ingredienten;
    }
    public void addIngredient(String ingredient) {
        ingredienten.add(ingredient);
    }
}
```

2.6.4 SausController

```
package be.vdab.frituurfrida.web;
// enkele imports
@Controller
@RequestMapping("sauzen")
class SausController {
    private List<Saus> sauzen = Arrays.asList(
        new Saus(3L, "cocktail", Arrays.asList("mayonaise", "ketchup", "cognac")),
        new Saus(6L, "mayonaise", Arrays.asList("ei", "mosterd")),
        new Saus(7L, "mosterd", Arrays.asList("mosterd", "azijn", "witte wijn")),
        new Saus(12L, "tartare", Arrays.asList("mayonaise", "augurk", "tabasco")),
        new Saus(44L, "vinaigrette", Arrays.asList("olijfolie", "mosterd", "azijn")));
    private static final String SAUZEN_VIEW = "sauzen";
    @GetMapping
    ModelAndView sauzen() {
        return new ModelAndView(SAUZEN_VIEW, "sauzen", sauzen);
    }
}
```

2.6.5 sauzen.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<!doctype html>
<html lang='nl'>
    <head>
        <c:import url='/WEB-INF/JSP/head.jsp'>
            <c:param name='title' value='Sauzen'/>
        </c:import>
    </head>
    <body>
        <h1>Sauzen</h1>
        <c:forEach var='saus' items='${sauzen}'>
            <h2>${saus.naam}</h2>
            <img src='images/${saus.naam}.png' alt='${saus.naam}'> ingrediënten:
            <c:forEach var='ingredient' items='${saus.ingredienten}'
                varStatus='status'>
                ${ingredient}<c:if test='${not status.last}'>, </c:if>
            </c:forEach>
        </c:forEach>
    </body>
</html>
```

2.7 Voorkeurtaal

2.7.1 VoorkeurTaalController

```
package be.vdab.frituurfrida.web;
// enkele imports
@Controller
@RequestMapping("voorkeurtaal")
class VoorkeurTaalController {
    private static final String VIEW = "voorkeurtaal";
    @GetMapping
    ModelAndView voorkeurTaal(
        @RequestHeader("accept-language") String acceptLanguage) {
        boolean nederlands = acceptLanguage.startsWith("nl-") ||
            acceptLanguage.startsWith("nl,");
        return new ModelAndView(VIEW, "nederlands", nederlands);
    }
}
```


2.7.2 voorkeurtaal.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<!doctype html>
<html lang="nl">
<head>
<c:import url='/WEB-INF/JSP/head.jsp'>
  <c:param name='title' value='Voorkeurtaal' />
</c:import>
</head>
<body>
  <p>
    Je voorkeurtaal is ${nederlands ? 'nederlands' : 'geen nederlands' }.
  </p>
</body>
</html>
```

2.8 Bezoekt

2.8.1 IndexController

De method index:

```
@GetMapping
ModelAndView index(@CookieValue(name = "reedsBezocht", required = false)
String reedsBezocht, HttpServletResponse response) {
    LocalDate vandaag = LocalDate.now();
    DayOfWeek weekdag = vandaag.getDayOfWeek();
    String openGesloten = weekdag == DayOfWeek.MONDAY
        || weekdag == DayOfWeek.THURSDAY ? "gesloten" : "open";
    Cookie cookie = new Cookie("reedsBezocht", "ja");
    cookie.setMaxAge(31_536_000);
    response.addCookie(cookie);
    ModelAndView modelAndView = new ModelAndView("index",
        "openGesloten", openGesloten)
        .addObject(new Adres("Grote markt", "7", new Gemeente("Brussel", 1000)));
    if (reedsBezocht != null) {
        modelAndView.addObject("reedsBezocht", true);
    }
    return modelAndView;
}
```

2.8.2 index.jsp

```
<c:if test='${reedsBezocht}'>
  <h2>Welkom terug</h2>
</c:if>
```

2.9 Unit test controller

```
package be.vdab.frituurfrida.web;
import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertTrue;
// enkele imports
public class SausControllerTest {
    private SausController controller;
    @Before
    public void before() {
        controller = new SausController();
    }
    @Test
    public void sauzenWerktSamenMetDeJspSauzen() {
        assertEquals("sauzen", controller.sauzen().getViewName());
    }
}
```

```

@Test
public void sauzenGeeftSauzenDoor() {
    assertTrue(controller.sauzen().getModel().get("sauzen") instanceof List);
}
}

```

2.10 Sauzen.csv

2.10.1 SausRepository

```

package be.vdab.frituurfrida.repositories;
// enkele imports
public interface SausRepository {
    List<Saus> findAll();
}

```

2.10.2 SausRepositoryException

```

package be.vdab.frituurfrida.exceptions;
public class SausRepositoryException extends RuntimeException {
    private static final long serialVersionUID = 1L;
    public SausRepositoryException(String message) {
        super(message);
    }
}

```

2.10.3 CSVSausRepository

```

package be.vdab.frituurfrida.repositories;
// enkele imports
@Component
class CSVSausRepository implements SausRepository {
    private static final Path PAD = Paths.get("/data/sauzen.csv");
    private static final Logger LOGGER =
        LoggerFactory.getLogger(CSVSausRepository.class);
    @Override
    public List<Saus> findAll() {
        List<Saus> sauzen = new ArrayList<>();
        try (BufferedReader reader = Files.newBufferedReader(PAD)) {
            for (String regel; (regel = reader.readLine()) != null;) {
                if (!regel.isEmpty()) { // blanco regel overslaan
                    sauzen.add(maakSaus(regel));
                }
            }
        } catch (IOException ex) {
            String fout = "Fout bij lezen " + PAD;
            LOGGER.error(fout, ex);
            throw new SausRepositoryException(fout);
        }
        return sauzen;
    }
    private Saus maakSaus(String regel) {
        String[] onderdelen = regel.split(",");
        if (onderdelen.length < 2) {
            String fout = pad + ":" + regel + " bevat minder dan 2 onderdelen";
            LOGGER.error(fout);
            throw new SausRepositoryException(fout);
        }
        try {
            Saus saus = new Saus(Long.parseLong(onderdelen[0]), onderdelen[1]);
            for (int index = 2; index < onderdelen.length; index++) {
                saus.addIngredient(onderdelen[index]);
            }
            return saus;
        }
    }
}

```

```

    } catch (NumberFormatException ex) {
        String fout = pad + ":" + regel + " bevat verkeerde id";
        LOGGER.error(fout, ex);
        throw new SausRepositoryException(fout);
    }
}

```

2.10.4 SausService

```

package be.vdab.frituurfrida.services;
// enkele imports
public interface SausService {
    List<Saus> findAll();
}

```

2.10.5 DefaultSausService

```

package be.vdab.frituurfrida.services;
// enkele imports
@Service
class DefaultSausService implements SausService {
    private final SausRepository sausRepository;
    DefaultSausService(SausRepository sausRepository) {
        this.sausRepository = sausRepository;
    }
    @Override
    public List<Saus> findAll() {
        return sausRepository.findAll();
    }
}

```

2.10.6 SausController

```

package be.vdab.frituurfrida.web;
// enkele imports
@Controller
@RequestMapping("sauzen")
class SausController {
    private static final String SAUZEN_VIEW = "sauzen";
    private final SausService sausService;
    SausController(SausService sausService) {
        this.sausService = sausService;
    }
    @GetMapping
    ModelAndView sauzen() {
        return new ModelAndView(SAUZEN_VIEW, "sauzen", sausService.findAll());
    }
}

```

2.10.7 SausControllerTest

Voor de class:

```
@RunWith(MockitoJUnitRunner.class)
```

Een extra variabele:

```
@Mock
private SausService sausService;
```

De method before:

```
@Before
public void before() {
    controller = new SausController(sausService);
}

```

De method `sauzenGeeftJuisteDataAanJSP`:

```
@Test
public void sauzenGeeftJuisteDataAanJSP() {
    assertTrue(controller.sauzen().getModel().containsKey("sauzen"));
    verify(sausService).findAll();
}
```

2.11 Sauzen.properties

2.11.1 CSVSausRepository

```
package be.vdab.frituurfrida.repositories;
// enkele imports
@Component
@Qualifier("CSV")
class CSVSausRepository implements SausRepository {
    ...
}
```

2.11.2 PropertiesSausRepository

```
package be.vdab.frituurfrida.repositories;
// enkele imports
@Component
@Qualifier("properties")
class PropertiesSausRepository implements SausRepository {
    private static final Path PAD = Paths.get("/data/sauzen.properties");
    private static final Logger LOGGER =
        LoggerFactory.getLogger(PropertiesSausRepository.class);
    @Override
    public List<Saus> findAll() {
        List<Saus> sauzen = new ArrayList<>();
        try (BufferedReader reader = Files.newBufferedReader(PAD)) {
            for (String regel; (regel = reader.readLine()) != null;) {
                if (!regel.isEmpty()) { // blanco regel overslaan
                    sauzen.add(maakSaus(regel));
                }
            }
        } catch (IOException ex) {
            String fout = "Fout bij lezen " + PAD;
            LOGGER.error(fout, ex);
            throw new SausRepositoryException(fout);
        }
        return sauzen;
    }
    private Saus maakSaus(String regel) {
        String[] onderdelen = regel.split(":");
        if (onderdelen.length < 2) {
            String fout = pad + ":" + regel + " bevat minder dan 2 onderdelen";
            LOGGER.error(fout);
            throw new SausRepositoryException(fout);
        }
        try {
            String[] onderdelen2 = onderdelen[1].split(",");
            Saus saus = new Saus(Long.parseLong(onderdelen[0]), onderdelen2[0]);
            for (int index = 1; index < onderdelen2.length; index++) {
                saus.addIngredient(onderdelen2[index]);
            }
            return saus;
        } catch (NumberFormatException ex) {
            String fout = pad + ":" + regel + " bevat verkeerde id";
            LOGGER.error(fout, ex);
            throw new SausRepositoryException(fout);
        }
    }
}
```

2.11.3 DefaultSausService

Constructor:

```
DefaultSausService(@Qualifier("properties") SausRepository sausRepository) {
    this.sausRepository = sausRepository;
}
```

2.12 Application.properties

2.12.1 application.properties

Extra regels:

CSV=/data/sauzen.csv

properties=/data/sauzen.properties

2.12.2 CSVSausRepository

```
package be.vdab.frituurfrida.repositories;
// enkele imports
@Component
@Qualifier("CSV")
class CSVSausRepository implements SausRepository {
    private static final Logger LOGGER =
        LoggerFactory.getLogger(CSVSausRepository.class);
    private final Path pad;
    CSVSausRepository(@Value("${CSV}") Path pad) {
        this.pad = pad;
    }
    ...
}
```

2.12.3 PropertiesSausRepository

```
package be.vdab.frituurfrida.repositories;
// enkele imports
@Component
@Qualifier("properties")
class PropertiesSausRepository implements SausRepository {
    private static final Logger LOGGER =
        LoggerFactory.getLogger(PropertiesSausRepository.class);
    private final Path pad;
    public PropertiesSausRepository(@Value("${properties}") Path pad) {
        this.pad = pad;
    }
    ...
}
```

2.13 DataSource

2.13.1 pom.xml

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <scope>runtime</scope>
</dependency>
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>
```

2.13.2 application.properties

```
spring.datasource.url=jdbc:mysql://localhost/frituurfrida?useSSL=false
spring.datasource.username=cursist
spring.datasource.password=cursist
```

2.13.3 DataSourceTest

Zelfde als in theorie

2.14 Repositories

2.14.1 Snack

```
package be.vdab.frituurfrida.entities;
import java.math.BigDecimal;
public class Snack {
    private long id;
    private String naam;
    private BigDecimal prijs;
    // je maakt getters en setters voor alle private variabelen
    // je maakt een default constructor en een geparametriseerde constructor
}
```

2.14.2 SnackRepository

```
package be.vdab.frituurfrida.repositories;
// enkele imports
public interface SnackRepository {
    Optional<Snack> read(long id);
    void update(Snack snack);
    List<Snack> findByBeginNaam(String beginNaam);
}
```

2.14.3 SnackNietGevondenException

```
package be.vdab.frituurfrida.exceptions;
public class SnackNietGevondenException extends RuntimeException {
    private static final long serialVersionUID = 1L;
}
```

2.14.4 JdbcSnackRepository

```
package be.vdab.frituurfrida.repositories;
// enkele imports
@Repository
class JdbcSnackRepository implements SnackRepository {
    private final JdbcTemplate template;
    JdbcSnackRepository(JdbcTemplate template) {
        this.template = template;
    }
    private static final String UPDATE_SNACK =
        "update snacks set naam=?, prijs=? where id=?";
    @Override
    public void update(Snack snack) {
        if (template.update(UPDATE_SNACK, snack.getNaam(), snack.getPrijs(),
            snack.getId()) == 0) {
            throw new SnackNietGevondenException();
        }
    }
    private static final String SELECT_BY_BEGIN_LETTER =
        "select id, naam, prijs from snacks where naam like ?";
    private final RowMapper<Snack> snackRowMapper = (resultSet, rowNum) ->
        new Snack(resultSet.getLong("id"), resultSet.getString("naam"),
            resultSet.getBigDecimal("prijs"));
    @Override
    public List<Snack> findByBeginNaam(String beginNaam) {
        return template.query(SELECT_BY_BEGIN_LETTER, snackRowMapper, beginNaam+'%');
    }
    private static final String READ =
        "select id, naam, prijs from snacks where id=?";
}
```

```

@Override
public Optional<Snack> read(long id) {
    try {
        return Optional.of(template.queryForObject(READ, snackRowMapper, id));
    } catch (IncorrectResultSizeDataAccessException ex) {
        return Optional.empty();
    }
}
}

```

2.14.5 insertSnack.sql

In src/test/resources:

```
insert into snacks(naam,prijs) values('test', 10);
```

2.14.6 JdbcSnackRepositoryTest

```

package be.vdab.frituurfrida.repositories;
// enkele imports
@RunWith(SpringRunner.class)
@JdbcTest
@AutoConfigureTestDatabase(replace = Replace.NONE)
@Import(JdbcSnackRepository.class)
@Sql("/insertSnack.sql")
public class JdbcSnackRepositoryTest
    extends AbstractTransactionalJUnit4SpringContextTests {
    @Autowired
    private JdbcSnackRepository repository;
    private long idVanTestSnack() {
        return super.jdbcTemplate.queryForObject(
            "select id from snacks where naam='test'", Long.class);
    }
    @Test
    public void update() {
        long id = idVanTestSnack();
        Snack snack = new Snack(id, "test", BigDecimal.ONE);
        repository.update(snack);
        assertEquals(0, BigDecimal.ONE.compareTo(
            super.jdbcTemplate.queryForObject("select prijs from snacks where id=?",
                BigDecimal.class, id)));
    }
    @Test(expected = SnackNietGevondenException.class)
    public void updateOnbestaandeSnack() {
        repository.update(new Snack(-1, "test", BigDecimal.ONE));
    }
    @Test
    public void read() {
        assertEquals("test", repository.read(idVanTestSnack()).get().getNaam());
    }
    @Test
    public void readOnbestaandeSnack() {
        assertFalse(repository.read(-1).isPresent());
    }
    @Test
    public void findByBeginNaam() {
        List<Snack> snacks = repository.findByBeginNaam("t");
        String vorigeNaam = "";
        for (Snack snack : snacks) {
            assertTrue(snack.getNaam().toLowerCase().startsWith("t"));
            assertTrue(vorigeNaam.compareToIgnoreCase(snack.getNaam()) <= 0);
            vorigeNaam = snack.getNaam();
        }
    }
}

```

```

        long aantalSnacks = super.jdbcTemplate.queryForObject(
            "select count(*) from snacks where naam like 't'", Long.class);
        assertEquals(aantalSnacks, snacks.size());
    }
}

```

2.14.7 application.properties

```

logging.level.org.springframework.jdbc.core.JdbcTemplate=DEBUG
logging.level.org.springframework.jdbc.core.simple.SimpleJdbcInsert=DEBUG
logging.level.org.springframework.jdbc.core.StatementCreatorUtils=TRACE

```

2.15 Services

2.15.1 SnackService

```

package be.vdab.frituurfrida.services;
// enkele imports
public interface SnackService {
    Optional<Snack> read(long id);
    void update(Snack snack);
    List<Snack> findByBeginNaam(String beginNaam);
}

```

2.15.2 DefaultSnackService

```

package be.vdab.frituurfrida.services;
// enkele imports
@Service
@Transactional(readOnly = true, isolation = Isolation.READ_COMMITTED)
class DefaultSnackService implements SnackService {
    private final SnackRepository snackRepository;
    public DefaultSnackService(SnackRepository snackRepository) {
        this.snackRepository = snackRepository;
    }
    @Override
    @Transactional(readOnly = false, isolation = Isolation.READ_COMMITTED)
    public void update(Snack snack) {
        snackRepository.update(snack);
    }
    @Override
    public List<Snack> findByBeginNaam(String beginNaam) {
        return snackRepository.findByBeginNaam(beginNaam);
    }
    @Override
    public Optional<Snack> read(long id) {
        return snackRepository.read(id);
    }
}

```

2.15.3 SnackController

```

package be.vdab.frituurfrida.web;
// enkele imports
@Controller
@RequestMapping("snacks")
class SnackController {
    private static final char[] ALFABET =
        "ABCDEFGHIJKLMNOPQRSTUVWXYZ".toCharArray();
    private final SnackService snackService;
    SnackController(SnackService snackService) {
        this.snackService = snackService;
    }
}

```



```

private static final String ALFABET_VIEW = "alfabet";
@GetMapping("alfabet")
ModelAndView alfabet() {
    return new ModelAndView(ALFABET_VIEW, "alfabet", ALFABET);
}
@GetMapping(params = "beginletter")
ModelAndView findByBeginletter(char beginletter) {
    return new ModelAndView(ALFABET_VIEW, "alfabet", ALFABET)
        .addObject("snacks",
            snackService.findByBeginNaam(String.valueOf(beginletter)));
}
}

```

2.15.4 alfabet.jsp

```

<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<!doctype html>
<html lang='nl'>
<head>
    <c:import url='/WEB-INF/JSP/head.jsp'>
        <c:param name='title' value='Snacks (alfabet)'/>
    </c:import>
    <style>
        #alfabet {
            list-style-type: none;
        }
        #alfabet li {
            display: inline;
        }
    </style>
</head>
<body>
    <h1>Snacks (alfabet)</h1>
    <ul id='alfabet'>
    <c:forEach var='letter' items='${alfabet}'>
        <c:url value='/snacks' var='url'>
            <c:param name='beginletter' value='${letter}' />
        </c:url>
        <li><a href='${url}'>${letter}</a></li>
    </c:forEach>
    </ul>
    <c:if test='${not empty snacks}'>
        <ul>
            <c:forEach var='snack' items='${snacks}'>
                <li>${snack.naam}</li>
            </c:forEach>
        </ul>
    </c:if>
</body>
</html>

```

2.16 Begin naam

2.16.1 BeginNaamForm

```

package be.vdab.frituurfrida.web;
// enkele imports
class BeginNaamForm {
    @NotBlank
    private String beginnaam;
    // je maakt een getter en een setter
}

```

2.16.2 SnackController

```
private static final String BEGIN_NAAM_VIEW = "beginnaam";
@GetMapping("beginnaam")
ModelAndView beginNaam() {
    return new ModelAndView(BEGIN_NAAM_VIEW).addObject(new BeginNaamForm());
}
@GetMapping(params = "beginnaam")
ModelAndView beginNaam(@Valid BeginNaamForm beginNaamForm,
    BindingResult bindingResult) {
    ModelAndView modelAndView = new ModelAndView(BEGIN_NAAM_VIEW);
    if (bindingResult.hasErrors()) {
        return modelAndView;
    }
    List<Snack> snacks =
        snackService.findByBeginNaam(beginNaamForm.getBeginnaam());
    if (snacks.isEmpty()) {
        bindingResult.reject("geenSnacks");
    } else {
        modelAndView.addObject("snacks", snacks);
    }
    return modelAndView;
}
```

2.16.3 messages.properties

geenSnacks=geen snacks gevonden

2.16.4 ValidationMessages.properties

Zelfde als in theorie

2.16.5 beginnaam.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<%@taglib prefix='form' uri='http://www.springframework.org/tags/form'%>
<!doctype html>
<html lang='nl'>
    <head>
        <c:import url='/WEB-INF/JSP/head.jsp'>
            <c:param name='title' value='Snacks (begin naam)'/>
        </c:import>
    </head>
    <body>
        <h1>Snacks (begin naam)</h1>
        <c:url value='/snacks' var='url'/>
        <form:form action='${url}' modelAttribute='beginNaamForm' method='get'>
            <form:label path='beginnaam'>Begin naam:
            <form:errors path='beginnaam'/></form:label>
            <form:input path='beginnaam' autofocus='autofocus'/>
            <input type='submit' value='Zoeken'>
            <form:errors/>
        </form:form>
        <c:if test='${not empty snacks}'>
            <ul>
                <c:forEach items='${snacks}' var='snack'>
                    <li>${snack.naam}</li>
                </c:forEach>
            </ul>
        </c:if>
    </body>
</html>
```

2.16.6 Unit test

```
package be.vdab.frituurfrida.web;
// enkele imports
public class BeginNaamFormTest {
    private Validator validator;
    @Before
    public void before() {
        ValidatorFactory factory = Validation.buildDefaultValidatorFactory();
        validator = factory.getValidator();
    }
    @Test
    public void beginNaamOk() {
        assertTrue(validator.validateValue(
            BeginNaamForm.class, "beginnaam", "d").isEmpty());
    }
    @Test
    public void beginNaamMoetVerschillenVanNull() {
        assertFalse(validator.validateValue(
            BeginNaamForm.class, "beginnaam", null).isEmpty());
    }
    @Test
    public void vanMoetMinstensEénTekenBevatten() {
        assertFalse(validator.validateValue(
            BeginNaamForm.class, "beginnaam", "").isEmpty());
    }
}
```

2.17 Client sided validatie

2.17.1 beginnaam.jsp

```
<form:input path='beginnaam' required="required" autofocus='autofocus' />
```

2.18 Snack wijzigen

2.18.1 alfabet.jsp en beginnaam.jsp

```
...
<%@taglib prefix='spring' uri="http://www.springframework.org/tags"%>...
...
<c:forEach var='snack' items='${snacks}'>
    <spring:url value='/snacks/{id}/wijzigen' var='url'>
        <spring:param name='id' value='${snack.id}' />
    </spring:url>
    <li><a href='${url}'>${snack.naam}</a></li>
</c:forEach>
```

2.18.2 Snack

```
...
@NotBlank
private String naam;
@NotNull
@Min(0)
private BigDecimal prijs;
...
```

2.18.3 SnackController

```
private static final String WIJZIGEN_VIEW = "snackwijzigen";
@GetMapping("/{id}/wijzigen")
ModelAndView wijzigen(@PathVariable long id) {
    ModelAndView modelAndView = new ModelAndView(WIJZIGEN_VIEW);
    snackService.read(id).ifPresent(snack -> modelAndView.addObject(snack));
}
```

```

    return modelAndView;
}
private static final String REDIRECT_URL_NA_WIJZIGEN = "redirect:/";
private static final String REDIRECT_URL_BIJ_SNACK_NIET_GEVONDEN =
    "snacknietgevonden";
@PostMapping("/{id}/wijzigen")
String wijzigen(@Valid Snack snack, BindingResult bindingResult) {
    if (bindingResult.hasErrors()) {
        return WIJZIGEN_VIEW;
    }
    try {
        snackService.update(snack);
        return REDIRECT_URL_NA_WIJZIGEN;
    } catch (SnackNietGevondenException ex) {
        return REDIRECT_URL_BIJ_SNACK_NIET_GEVONDEN;
    }
}

```

❶

- (1) Spring maakt een Snack object en vult dit aan de hand van de invoervakken in de form en de path variabele id.

2.18.4 snackwijzigen.jsp

```

<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<%@taglib prefix='spring' uri='http://www.springframework.org/tags'%>
<%@taglib prefix='form' uri='http://www.springframework.org/tags/form'%>
<!doctype html>
<html lang='nl'>
    <head>
        <c:import url='/WEB-INF/JSP/head.jsp'>
            <c:param name='title' value='${snack.naam} wijzigen' />
        </c:import>
    </head>
    <body>
        <h1>${snack.naam} wijzigen</h1>
        <spring:url value='/snacks/{id}/wijzigen' var='url'>
            <spring:param name='id' value='${snack.id}' />
        </spring:url>
        <form:form action='${url}' modelAttribute='snack' method='post'
            id='wijzigenForm'>
            <form:label path='naam'>Naam: <form:errors path='naam' /></form:label>
            <form:input path='naam' required="required" autofocus='autofocus' />
            <form:label path='prijs'>Prijs: <form:errors path='prijs' /></form:label>
            <form:input path='prijs' type='number' required="required" min='0'
                step='0.01' />
            <input type='submit' value='Opslaan' id='opslaanKnop'>
        </form:form>
        <script>
            document.getElementById('wijzigenForm').onsubmit = function() {
                document.getElementById('opslaanKnop').disabled = true;
            };
        </script>
    </body>
</html>

```

2.18.5 messages.properties

typeMismatch.java.math.BigDecimal=tik een bedrag

2.18.6 snacknietgevonden.jsp

```
<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<!doctype html>
<html lang='nl'>
  <head>
    <c:import url='/WEB-INF/JSP/head.jsp'>
      <c:param name='title' value='Snack niet gevonden' />
    </c:import>
  </head>
  <body>
    <h1>Snack niet gevonden</h1>
  </body>
</html>
```

2.19 Zoek de friet

2.19.1 Deur

```
package be.vdab.frituurfrida.web;
import java.io.Serializable;
class Deur implements Serializable {
  private static final long serialVersionUID = 1L;
  private boolean open;
  private boolean metFriet;
  Deur(boolean metFriet) {
    this.metFriet = metFriet;
  }
  void open() {
    open = true;
  }
  public boolean isOpen() {
    return open;
  }
  public boolean isMetFriet() {
    return metFriet;
  }
}
```

2.19.2 ZoekDeFrietSpel

```
package be.vdab.frituurfrida.web;
interface ZoekDeFrietSpel {
  void openDeur(int index);
  Deur[] getDeuren();
  void resetDeuren();
}
```

2.19.3 DefaultZoekDeFrietSpel

```
package be.vdab.frituurfrida.web;
// enkele imports
@Component
@SessionScope
class DefaultZoekDeFrietSpel implements Serializable, ZoekDeFrietSpel {
  private static final long serialVersionUID = 1L;
  private static final int AANTAL_DEUREN = 7;
  private final Deur[] deuren = new Deur[AANTAL_DEUREN];
  DefaultZoekDeFrietSpel() {
    resetDeuren();
  }
}
```

```

@Override
public void openDeur(int index) {
    deuren[index].open();
}
@Override
public Deur[] getDeuren() {
    return deuren;
}
@Override
public void resetDeuren() {
    int indexMetFriet = ThreadLocalRandom.current().nextInt(AANTAL_DEUREN);
    for (int index = 0; index != AANTAL_DEUREN; index++) {
        deuren[index] = new Deur(index == indexMetFriet);
    }
}
}

```

2.19.4 FrietController

```

package be.vdab.frituurfrida.web;
// enkele imports
@Controller
@RequestMapping("friet")
class FrietController {
    private ZoekDeFrietSpel zoekDeFrietSpel;
    FrietController(ZoekDeFrietSpel zoekDeFrietSpel) {
        this.zoekDeFrietSpel = zoekDeFrietSpel;
    }
    private static final String ZOEK_DE_FRIET_VIEW = "zoekdefriet";
    @GetMapping("zoekdefriet")
    ModelAndView zoekDeFriet() {
        return new ModelAndView(ZOEK_DE_FRIET_VIEW, "spel", zoekDeFrietSpel);
    }
    private static final String REDIRECT_NA_RESET_DEUREN =
        "redirect:/friet/zoekdefriet";
    @PostMapping("zoekdefriet/nieuwspel")
    String nieuwSpel() {
        zoekDeFrietSpel.resetDeuren();
        return REDIRECT_NA_RESET_DEUREN;
    }
    private static final String REDIRECT_NA_OPEN_DEUR =
        "redirect:/friet/zoekdefriet";
    @PostMapping(value = "zoekdefriet", params = "index")
    String openDeur(int index) {
        zoekDeFrietSpel.openDeur(index);
        return REDIRECT_NA_OPEN_DEUR;
    }
}

```

2.19.5 zoekdefriet.jsp

```

<%@page contentType='text/html' pageEncoding='UTF-8' session="false"%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<!doctype html>
<html lang='nl'>
<head>
<c:import url='/WEB-INF/JSP/head.jsp'>
    <c:param name='title' value='Zoek de friet'/>
</c:import>
</head>
<body>
    <h1>Zoek de friet</h1>

```

```

<c:url value='/frietten/zoekdefriet' var='url'/>
<form action='${url}' method='post'>
  <c:forEach items='${spel.deuren}' var='deur' varStatus="status">
    <button type='submit' name='index' value='${status.index}'>
      <c:choose>
        <c:when test='${deur.open}'>
          <c:choose>
            <c:when test='${deur.metFriet}'>
              <img src='<c:url value="/images/gevonden.png"/>' alt='gevonden'>
            </c:when>
            <c:otherwise>
              <img src='<c:url value="/images/deuopen.png"/>' alt='deur open'>
            </c:otherwise>
          </c:choose>
        </c:when>
        <c:otherwise>
          <img src='<c:url value="/images/deurtoe.png"/>' alt='deur toe'>
        </c:otherwise>
      </c:choose>
    </button>
  </c:forEach>
</form>
<c:url value='/frietten/zoekdefriet/nieuwspel' var='url'/>
<form method='post' action='${url}'>
  <input type='submit' value='Nieuw spel'>
</form>
</body>
</html>

```

2.19.6 application.properties

server.session.tracking-modes=cookie

2.20 Saus raden

2.20.1 SausRadenSpel

```

package be.vdab.frituurfrida.web;
interface SausRadenSpel {
    void reset(String saus);
    void doeGok(char letter);
    String getSausMetPuntjes();
    String getSaus();
    int getVerkeerdeBeurten();
    boolean isGewonnen();
    boolean isVerloren();
}

```

2.20.2 DefaultSausRadenSpel

```

package be.vdab.frituurfrida.web;
// enkele imports ...
@Component
@SessionScope
class DefaultSausRadenSpel implements Serializable, SausRadenSpel {
    private static final long serialVersionUID = 1L;
    private static final int MAX_BEURTEN = 10;
    private String saus;
    private StringBuilder sausMetPuntjes;
    private int verkeerdeBeurten;
    @Override
    public void reset(String saus) {
        this.saus = saus;
        sausMetPuntjes = new StringBuilder(saus.length());
    }
}

```

```

        IntStream.rangeClosed(1, saus.length()).forEach(
            teller -> sausMetPuntjes.append('.'));
        verkeerdeBeurten = 0;
    }
    @Override
    public void doeGok(char letter) {
        int letterIndex = saus.indexOf(letter);
        if (letterIndex == -1) {
            verkeerdeBeurten++;
        } else {
            do {
                sausMetPuntjes.setCharAt(letterIndex, letter);
                letterIndex = saus.indexOf(letter, letterIndex + 1);
            } while (letterIndex != -1);
        }
    }
    @Override
    public String getSausMetPuntjes() {
        return sausMetPuntjes.toString();
    }
    @Override
    public String getSaus() {
        return saus;
    }
    @Override
    public int getVerkeerdeBeurten() {
        return verkeerdeBeurten;
    }
    @Override
    public boolean isGewonnen() {
        return sausMetPuntjes.indexOf(".") == -1;
    }
    @Override
    public boolean isVerloren() {
        return verkeerdeBeurten == MAX_BEURTEN;
    }
}

```

2.20.3 SausRadenForm

```

package be.vdab.frituurfrida.web;
class SausRadenForm {
    @NotNull
    private Character letter;
    // je maakt een getter en een setter
}

```

2.20.4 SausController

```

package be.vdab.frituurfrida.web;
// enkele imports
@Controller
@RequestMapping("sauzen")
class SausController {
    private static final String SAUZEN_VIEW = "sauzen";
    private final SausService sausService;
    private final SausRadenSpel sausRadenSpel;
    SausController(SausService sausService, SausRadenSpel sausRadenSpel) {
        this.sausService = sausService;
        this.sausRadenSpel = sausRadenSpel;
    }
}

```



```

@GetMapping
ModelAndView sauzen() {
    return new ModelAndView(SAUZEN_VIEW, "sauzen", sausService.findAll());
}
private String randomSaus() {
    List<Saus> sauzen = sausService.findAll();
    return sauzen.get(
        ThreadLocalRandom.current().nextInt(sauzen.size())).getNaam();
}
private static final String RADEN_VIEW = "sausraden";
@GetMapping("raden")
ModelAndView raden() {
    sausRadenSpel.reset(randomSaus());
    return new ModelAndView(RADEN_VIEW, "spel", sausRadenSpel)
        .addObject(new SausRadenForm());
}
private static final String REDIRECT_NA_NIEUW_SPEL_RADEN =
    "redirect:/sauzen/raden";
@PostMapping("raden/nieuwspel")
String radenNieuwSpel() {
    sausRadenSpel.reset(randomSaus());
    return REDIRECT_NA_NIEUW_SPEL_RADEN;
}
private static final String REDIRECT_NA_LETTER_RADEN =
    "redirect:/sauzen/raden/volgendegok";
@PostMapping(value = "raden", params = "letter")
ModelAndView raden(@Valid SausRadenForm form, BindingResult bindingResult) {
    if (bindingResult.hasErrors()) {
        return new ModelAndView(RADEN_VIEW, "spel", sausRadenSpel);
    }
    sausRadenSpel.doeGok(form.getLetter());
    return new ModelAndView(REDIRECT_NA_LETTER_RADEN);
}
@GetMapping("volgendegok")
ModelAndView volgendeGok() {
    return new ModelAndView(RADEN_VIEW, "spel", sausRadenSpel)
        .addObject(new SausRadenForm());
}
}

```

2.20.5 SausControllerTest

Extra variabele:

```

@Mock
private SausRadenSpel sausRadenSpel;

```

Gewijzigde opdracht in before:

```

controller = new SausController(sausService, sausRadenSpel);

```

2.20.6 sausraden.jsp

```

<%@page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<%@taglib prefix='form' uri='http://www.springframework.org/tags/form'%>
<!doctype html>
<html lang='nl'>
    <head>
        <c:import url='/WEB-INF/JSP/head.jsp'>
            <c:param name='title' value='Saus raden' />
        </c:import>
    </head>

```

```

<body>
  <h1>Saus raden</h1>
  <c:choose>
    <c:when test="${spel.verloren}">U bent verloren, de saus was ${spel.saus}.
    </c:when>
    <c:when test="${spel.gewonnen}">U bent gewonnen, de saus was ${spel.saus}.
    </c:when>
    <c:otherwise>
      Te raden saus: ${spel.sausMetPuntjes}
      <c:url value='/sauzen/raden' var='url' />
      <form:form action='${url}' method='post' id='radenform'
        modelAttribute="sausRadenForm">
        <form:label path='Letter'>letter:
          <form:errors path='Letter' /></form:label>
          <form:input path='Letter' size='1' maxlength='1'
            autofocus='autofocus' required='required' />
          <input type='submit' value='Raden' id='radenknop'>
        </form:form>
      <script>
        document.getElementById('radenform').onsubmit = function() {
          document.getElementById('radenknop').disabled = true;
        };
      </script>
    </c:otherwise>
  </c:choose>
  <c:url value='/sauzen/raden/nieuwspel' var='url' />
  <form method='post' action='${url}'>
    <input type='submit' value='Nieuw spel'>
  </form>
  
</body>
</html>

```

2.21 Custom tags

2.21.1 head.tag

Zelfde als in theorie (maar pizza.ico wordt frida.ico
en pizzaluigi.css wordt frituurfrida.css)

2.21.2 menu.tag

```

<%@tag description='website menu' pageEncoding='UTF-8'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<header>
  <nav>
    <ul>
      <li><a href="<c:url value='/' />">Welkom</a></li>
      <li><a href="<c:url value='/snacks/alfabet' />">Snacks (alfabet)</a></li>
      <li><a href="<c:url value='/snacks/beginnaam' />">Snacks (begin naam)
        </a></li>
      <li><a href="<c:url value='/sauzen/raden' />">Saus raden</a></li>
      <li><a href="<c:url value='/sauzen' />">Sauzen</a></li>
      <li><a href="<c:url value='/voorkeurtaal' />">Voorkeurtaal</a></li>
      <li><a href="<c:url value='/frietten/zoekdefriet' />">Zoek de friet</a></li>
    </ul>
  </nav>
</header>

```

2.21.3 vdab.tld

Zelfde als in theorie.

2.21.4 frituurfrida.css

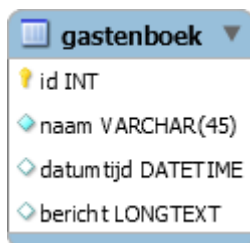
```
nav ul {
    list-style:none;
    padding:0;
}
nav ul li {
    display:inline;
}
}
```

2.21.5 JSP's

```
...
<%@taglib prefix='vdab' uri="http://vdab.be/tags" %>...
    <vdab:head title='...' />
...
<vdab:menu/>
...
```

2.22 Gastenboek

2.22.1 Nieuwe table in de database



2.22.2 Rechten in de database

```
use frituurfrida;
grant insert,select on gastenboek to cursist
```

2.22.3 GastenBoekEntry

```
package be.vdab.frituurfrida.entities;
// enkele imports
public class GastenBoekEntry {
    private long id;
    @NotBlank
    private String naam;
    @NotNull @DateTimeFormat(style = "S-")
    private LocalDateTime datumTijd = LocalDateTime.now();
    @NotBlank
    private String bericht;
    // geparametriseerde constructor
    // geparametriseerde constructor zonder id
    // getters voor alle private variabelen
    // setter voor id
}
```

2.22.4 GastenBoekRepository

```
package be.vdab.frituurfrida.repositories;
// enkele imports
public interface GastenBoekRepository {
    void create(GastenBoekEntry entry);
    List<GastenBoekEntry> findAll();
}
```

2.22.5 JdbcGastenBoekRepository

```
package be.vdab.frituurfrida.repositories;
// enkele imports
@Repository
class JdbcGastenBoekRepository implements GastenBoekRepository {
    private final SimpleJdbcInsert insert;
    private final JdbcTemplate template;
    JdbcGastenBoekRepository(JdbcTemplate template) {
        this.template = template;
        this.insert = new SimpleJdbcInsert(template);
        insert.withTableName("gastenboek");
        insert.usingGeneratedKeyColumns("id");
    }
    @Override
    public void create(GastenBoekEntry entry) {
        Map<String, Object> kolomWaarden = new HashMap<>();
        kolomWaarden.put("naam", entry.getNaam());
        kolomWaarden.put("datumtijd", entry.getDatumTijd());
        kolomWaarden.put("bericht", entry.getBericht());
        Number id = insert.executeAndReturnKey(kolomWaarden);
        entry.setId(id.longValue());
    }
    private static final String SQL_SELECT_ALL=
        "select id,naam,datumtijd,bericht from gastenboek order by datumtijd desc";
    private final RowMapper<GastenBoekEntry> entryRowMapper =
        (resultSet, rowNum) -> new GastenBoekEntry(resultSet.getLong("id"),
            resultSet.getString("naam"),
            resultSet.getTimestamp("datumtijd").toLocalDateTime() ,
            resultSet.getString("bericht"));
    @Override
    public List<GastenBoekEntry> findAll() {
        return template.query(SQL_SELECT_ALL, entryRowMapper);
    }
}
```

2.22.6 insertGastenBoek.sql

```
insert into gastenboek(naam,datumtijd,bericht)
values ('test','2018-01-01 20:00', 'test');
```

2.22.7 JdbcGastenBoekRepositoryTest

```
package be.vdab.frituurfrida.repositories;
// enkele imports
@RunWith(SpringRunner.class)
@JdbcTest
@AutoConfigureTestDatabase(replace = Replace.NONE)
@Import(JdbcGastenBoekRepository.class)
@Sql("/insertGastenBoek.sql")
public class JdbcGastenBoekRepositoryTest extends
AbstractTransactionalJUnit4SpringContextTests {
    private static final String GASTENBOEK = "gastenboek";
    @Autowired
    private JdbcGastenBoekRepository repository;
    @Test
    public void create() {
        int aantalEntries = super.countRowsInTable(GASTENBOEK);
        GastenBoekEntry entry =
            new GastenBoekEntry("test2", LocalDateTime.now(), "test");
        repository.create(entry);
        assertEquals(0, entry.getId());
    }
}
```

```

    assertEquals(aantalEntries + 1, this.countRowsInTable(GASTENBOEK));
    assertEquals(1, super.countRowsInTableWhere(GASTENBOEK,
        "id=" + entry.getId()));
}
@Test
public void findAll() {
    List<GastenBoekEntry> entries = repository.findAll();
    assertEquals(super.countRowsInTable(GASTENBOEK), entries.size());
    LocalDateTime vorige = LocalDateTime.MIN;
    for (GastenBoekEntry entry : entries) {
        assertTrue(entry.getDatumTijd().compareTo(vorige) >= 0);
        vorige = entry.getDatumTijd();
    }
}
}

```

2.22.8 GastenBoekService

```

package be.vdab.frituurfrida.services;
// enkele imports
public interface GastenBoekService {
    void create(GastenBoekEntry entry);
    List<GastenBoekEntry> findAll();
}

```

2.22.9 DefaultGastenBoekService

```

package be.vdab.frituurfrida.services;
// enkele imports
@Service
@Transactional(readonly = true, isolation = Isolation.READ_COMMITTED)
class DefaultGastenBoekService implements GastenBoekService {
    private final GastenBoekRepository gastenBoekRepository;
    DefaultGastenBoekService(GastenBoekRepository gastenBoekRepository) {
        this.gastenBoekRepository = gastenBoekRepository;
    }
    @Override
    @Transactional(readonly = false, isolation = Isolation.READ_COMMITTED)
    public void create(GastenBoekEntry entry) {
        gastenBoekRepository.create(entry);
    }
    @Override
    public List<GastenBoekEntry> findAll() {
        return gastenBoekRepository.findAll();
    }
}

```

2.22.10 GastenBoekController

```

package be.vdab.frituurfrida.web;
// enkele imports
@Controller
@RequestMapping("gastenboek")
public class GastenBoekController {
    private final GastenBoekService gastenBoekService;
    GastenBoekController(GastenBoekService gastenBoekService) {
        this.gastenBoekService = gastenBoekService;
    }
    private static final String VIEW = "gastenboek";
    @GetMapping
    ModelAndView findAll() {
        return new ModelAndView(VIEW, "gastenboek", gastenBoekService.findAll());
    }
}

```

```

@GetMapping("toevoegen")
ModelAndView findAllMetToevoegOnderdeel() {
    return new ModelAndView(VIEW, "gastenboek",
        gastenBoekService.findAll()).addObject(new GastenBoekEntry());
}
private static final String REDIRECT_NA_CREATE = "redirect:/gastenboek";
@PostMapping
ModelAndView create(@Valid GastenBoekEntry entry, BindingResult bindingresult){
    if (bindingresult.hasErrors()) {
        return new ModelAndView(VIEW, "gastenboek", gastenBoekService.findAll());
    }
    gastenBoekService.create(entry);
    return new ModelAndView(REDIRECT_NA_CREATE);
}
}

```

2.22.11 gastenboek.jsp

```

<%@ page contentType='text/html' pageEncoding='UTF-8' session='false'%>
<%@taglib prefix='c' uri='http://java.sun.com/jsp/jstl/core'%>
<%@taglib prefix='spring' uri='http://www.springframework.org/tags' %>
<%@taglib prefix='form' uri='http://www.springframework.org/tags/form' %>
<%@taglib prefix='vdab' uri='http://vdab.be/tags' %>
<!doctype html>
<html lang='nl'>
<head>
<vdab:head title='Gastenboek'/>
</head>
<body>
<vdab:menu/>
<h1>Gastenboek</h1>
<c:choose>
    <c:when test='${empty gastenBoekEntry}'>
        <c:url value='/gastenboek/toevoegen' var='url' />
        <a href='${url}'>Toevoegen</a>
    </c:when>
    <c:otherwise>
        <c:url value='/gastenboek' var='url' />
        <form:form action='${url}' method='post' modelAttribute='gastenBoekEntry'>
            <form:label path='naam'>Naam:<form:errors path='naam' /></form:label>
            <form:input path='naam' autofocus='autofocus' required='required' />
            <form:label path='bericht'>Bericht:<form:errors path='bericht' />
            </form:label>
            <form:textarea path='bericht' required='required' rows='5' cols='80' />
            <input type='submit' value='Toevoegen' />
        </form:form>
    </c:otherwise>
</c:choose>
<c:if test='${not empty gastenboek}'>
    <dl>
        <c:forEach var='entry' items='${gastenboek}'>
            <dt><spring:eval expression='entry.datumTijd' /> ${entry.naam}</dt>
            <dd>${entry.bericht}</dd>
        </c:forEach>
    </dl>
</c:if>
</body>
</html>

```

2.22.12 menu.tag

```

<li><a href="<c:url value='/gastenboek' />">Gastenboek</a>

```

2.22.13 frituurfrida.css

```
label span {
    background-color: red;
    border-bottom-right-radius: 0.5em;
    border-top-left-radius: 0.5em;
    border-top-right-radius: 0.5em;
    box-shadow: 5px 5px #777777;
    margin-left: 0.5em;
    padding: 0.5em;
    color: white;
}
label span:empty {
    display: none;
}
input,textarea {
    display: block;
    margin-top: 0.2em;
    margin-bottom: 1em;
    font-size: 1.1em;
}
input[type='checkbox'] {
    display: inline;
}
input:focus,textarea:focus {
    background-color: #FFFFD6
}
dt {
    margin-top: 1em;
}
dd {
    font-weight: bold;
    margin-left: 0;
}
dt input {
    margin-bottom: 0;
}
```

2.23 Gastenboekbeheer

2.23.1 Rechten in database

```
use frituurfrida;
grant delete on gastenboek to cursist
```

2.23.2 GastenBoekRepository

```
void delete(long id);
```

2.23.3 JdbcGastenBoekRepository

```
private static final String SQL_DELETE = "delete from gastenboek where id=?";
@Override
public void delete(long id) {
    template.update(SQL_DELETE, id);
}
```

2.23.4 JdbcGastenBoekRepositoryTest

```
private long idVanTestEntry() {
    return super.jdbcTemplate.queryForObject(
        "select id from gastenboek where naam='test'", Long.class);
}
@Test
public void delete() {
    long id = idVanTestEntry();
    int aantalEntries = super.countRowsInTable(GASTENBOEK);
    repository.delete(id);
    assertEquals(aantalEntries - 1, super.countRowsInTable(GASTENBOEK));
}
```

```
    assertEquals(0, super.countRowsInTableWhere(GASTENBOEK, "id=" + id));
}
```

2.23.5 GastenBoekService

```
void delete(long[] ids);
```

2.23.6 DefaultGastenBoekService

```
@Override
@Transactional(readOnly = false, isolation = Isolation.READ_COMMITTED)
public void delete(long[] ids) {
    Arrays.stream(ids).forEach(id -> gastenBoekRepository.delete(id));
}
```

2.23.7 GastenBoekController

```
private static final String REDIRECT_NA_DELETE = "redirect:/gastenboek";
@PostMapping(params = "verwijderid")
String delete(long[] verwijderid) {
    if (verwijderid != null) {
        gastenBoekService.delete(verwijderid);
    }
    return REDIRECT_NA_DELETE;
}
```

2.23.8 gastenboek.jsp

```
<c:if test='${not empty gastenboek}'>
    <c:url value='/gastenboek/verwijderen' var='url'/>
    <form action='${url}' method='post'>
    <dl>
    <c:forEach var='entry' items='${gastenboek}'>
        <dt><spring:eval expression='entry.datumTijd'> ${entry.naam}
        <input type='checkbox' name='verwijderid' value='${entry.id}'>
        </dt>
        <dd>${entry.bericht}</dd>
    </c:forEach>
    </dl>
    <input type='submit' value='Verwijderen'>
    </form>
</c:if>
```


3 COLOFON

Domeinexpertisemanager:	Jean Smits
Moduleverantwoordelijke:	Jean Smits
Medewerkers:	Hans Desmet
Versie:	1/10/2018
Nummer dotatielijst:	