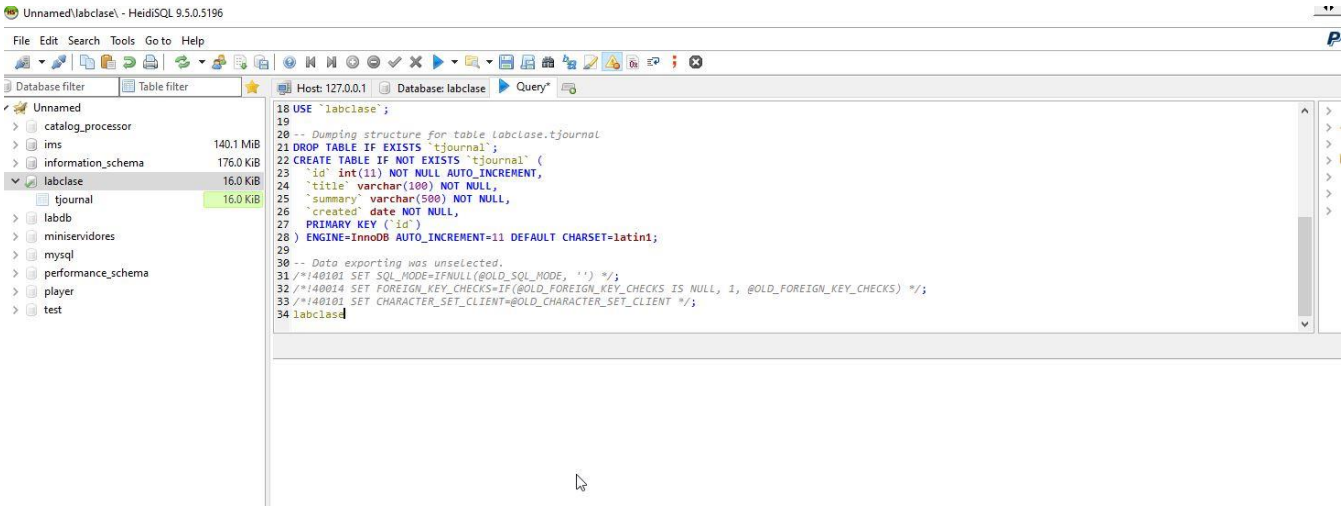
Laboratorio 6

BAJE LOS ARCHIVOS EXTRA DEL LABORATORIO. AHÍ ESTÁ EL SCRIPT DE LA BD.

1. En MariaDB, Creamos la BD llamada labclase. Con una tabla, llamada tjournal. (ver script).

Recomiendo usar la aplicación HEIDISQL para crear más fácilmente la BD usando el script.



2. Ahora en STS, iniciamos nuevo proyecto. File → New → Spring Starter project

3. Completamos el dialogo con la siguiente informacion
a. **Name**: journal
b. **Group**: com.cenfotec
c. **Description**: aplicacion inicial con spring boot
d. **Package**: debe quedar com.cenfotec.journal

Asegurese que el tipo sea Maven, JDK 1.8, packaging jar.

4. Seleccione NEXT

5. En la siguiente pantalla seleccione JPA (como en el laboratorio anterior), Spring Web (bajo Web), Thymeleaf(bajo Template Engine).

6. Dele Finish.

7. En el pom.xml agregue la siguiente dependencia.

```
<dependency>
	<groupId>org.mariadb.jdbc</groupId>
	<artifactId>mariadb-java-client</artifactId>
</dependency>
```

8. Haga un nuevo paquete: com.cenfotec.journal.domain. Agregue una clase llamada Journal. Escriba antes de public class… lo siguiente

(OJO LOS IMPORTS ESTÁN EN EL PASO 10)

```
@Entity
@Table(name = "TJournal")
public class J….
```

9. Agregue los siguientes campos a la clase:

```
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;

@Column(name="title")
private String title;

@Column(name="created")
private Date created;

@Column(name="summary")
private String summary;

@Transient
private SimpleDateFormat format = new SimpleDateFormat("MM/dd/yyyy");
public String getCreatedAsShort() {
return format.format(created);
}

public Journal(String title, String summary, String date) throws
ParseException {
this.title = title;
this.summary = summary;
this.created = format.parse(date);
}

Journal() {
}
```

Agreguen los set y get

10. Los imports deben quedar así:

```
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;
import javax.persistence.Transient;
```

11. Vea como cada campo es mapeado a una columna de la BD (recuerde ver el SCRIPT). La anotación @*Transient* hace que un campo NO se mappee a la BD.

12. Agregue un nuevo paquete: : com.cenfotec.journal.repository, en ese paquete, agreguen UNA INTERFAZ: JournalRepository.java.

a. Esta interfaz es todo lo que se necesita para tener montada practicamente toda la persistencia.
b. Observe la referencia a la clase y al ID de la misma para definir el JpaRepository.

Todo el codigo de la interfaz es este:

```
public interface JournalRepository extends JpaRepository<Journal,
Long> {
}
```

Los imports del caso, son:

```
import org.springframework.data.jpa.repository.JpaRepository;
import com.cenfotec.journal.domain.Journal;
```

13. Ahora haga un nuevo paquete: com.cenfotec.journal.service.

14. En el nuevo paquete haga una nueva INTERFAZ llamada JournalService.

15. Dentro de la interfaz llamada JournalService escriba el siguiente código

```
    public void saveJournal(Journal newJournal);
    public List<Journal> getAllJournals();
```

Los imports son:

```
import java.util.List;
import com.cenfotec.journal.domain.Journal;
```

16. En el mismo paquete, haga una nueva CLASE llamada JournalServiceImpl

En el nueva clase JournalServiceImpl el código es:

```
public class JournalServiceImpl implements JournalService {

	@Autowired
	JournalRepository journalRepo;

	@Override
	public void saveJournal(Journal newJournal) {
		journalRepo.save(newJournal);
	}

	@Override
	public List<Journal> getAllJournals() {
		return journalRepo.findAll();
	}

}
```

17. Los imports de la clase son:

```
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.cenfotec.journal.domain.Journal;
import com.cenfotec.journal.repository.JournalRepository;
```

18. Ahora agregue un nuevo paquete: nuevo paquete com.cenfotec.journal.web

Y en el, agregue una nueva clase: JournalController.java
Esta nueva clase queda asi

```
@Controller
public class JournalController {
	@Autowired
	JournalService journalService;

	@RequestMapping("/")
	public String index(Model model) throws ParseException {
		model.addAttribute("journal",
journalService.getAllJournals());
		Journal newEntry = new Journal("Hola Mundo", "un saludo",
"07/15/2017");
		journalService.saveJournal(newEntry);
```

```
            return "index";
        }
}
```

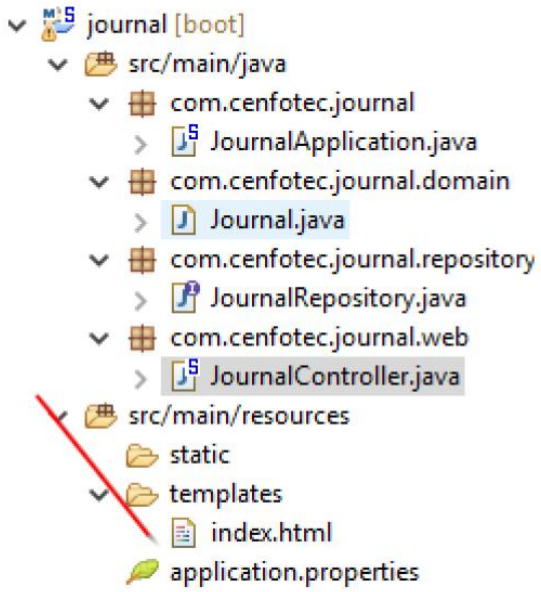
LOS IMPORTS QUEDAN ASÍ

```
import java.text.ParseException;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import com.cenfotec.journal.domain.Journal;
import com.cenfotec.journal.service.JournalService;
```

19. En el folder src/main/resources/template
    Pueden crear index.html y copiar el contenido del archivo suministrado

    ó, puede copiar el archivo proporcionado en ese directorio.
→USE EL archivo index.html en el subfolder de archivos extra



20. Del mismo subfolder, copy paste el contenido de application.properties a su application.properties.

21. En el Package Explorer pongase sobre el proyecto, dele botón derecho y dele RUN AS seleccionando Spring Boot App.
Intente ejecutarla.
Qué pasa? Obtiene un error? INVESTÍGUELO.

22. Para solucionar el problema del paso 21, vaya a la clase JournalServiceImpl y agregue la anotación

@Service

Antes de la declaración de la clase. Quedará así:

```
@Service
public class JournalServiceImpl implements JournalService {
```

Ahora entra a http://localhost:8080/journal y cada vez que se cargue la página, insertará a la BD y de paso se mostrará el listado actualizado.

Reto:

1. usando como referencia https://spring.io/guides/gs/handling-form-submission/ (puede implementar dicho proyecto), ó la respuesta en https://stackoverflow.com/questions/24551915/how-to-get-form-data-as-a-map-in-spring-mvc-controller

2. monte un formulario que envie los datos del Diario y los guarde en la DB
3. Investigue los annotation @PostMapping y @GetMapping. Vea si hay otros parecidos.
4a. Implemente otra página donde se listan todos las entradas al journal (diario) hechas y hay links para ver cada una no necesarimente editar).