

## Spring initializer - start.spring.io



**Project**  
☒ Maven Project ☐ Gradle Project

**Language**  
☒ Java ☐ Kotlin ☐ Groovy

**Spring Boot**  
☐ 2.6.0 (SNAPSHOT) ☐ 2.5.3 (SNAPSHOT) ☒ 2.5.2 ☐ 2.4.9 (SNAPSHOT)  
☐ 2.4.8 ☐ 2.3.12

**Project Metadata**

Group

Artifact

Name

Description

Package name

Packaging ☒ Jar ☐ War

Java ☐ 16 ☐ 11 ☒ 8

### Dependencies

ADD DEPENDENCIES... CTRL + B

#### Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

#### Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

#### MySQL Driver SQL

MySQL JDBC and R2DBC driver.

#### Lombok DEVELOPER TOOLS

Java annotation library which helps to reduce boilerplate code.

#### Spring Boot DevTools DEVELOPER TOOLS

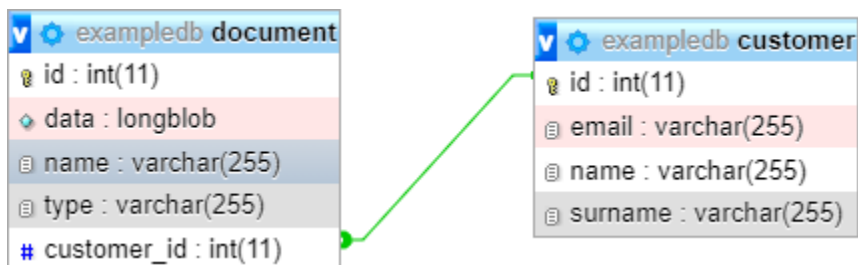
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Some settings: under resources: application.properties

```
BackendApplication.java x application.properties x Customer.java x DocumentRepository.java x CustomerS
1
2 spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
3
4 spring.datasource.username=root
5 spring.datasource.password=
6 spring.datasource.url=jdbc:mysql://localhost:3306/exampledb
7
8 spring.jpa.show-sql=true
9 spring.jpa.hibernate.ddl-auto= update
10 spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
11
12 logging.level.org.hibernate.SQL=DEBUG
13 logging.level.org.hibernate.type=TRACE
14
15 spring.servlet.multipart.file-size-threshold=2KB
16 spring.servlet.multipart.max-file-size=200MB
17 spring.servlet.multipart.max-request-size=200MB
18
19 server.port=8070
20
```

### Created the database

User table is not created because of time limitation. Backend part is completed with customer and document tables.



**Add more than one customer:**

POST POST/api/addCustomers

post new customers

<http://localhost:8070/api/addCustomers>

POST	▼	http://localhost:8070/api/addCustomers		
Params	Authorization	Headers (9)	Body ●	Pre-request
<input type="radio"/> none	<input type="radio"/> form-data	<input type="radio"/> x-www-form-urlencoded	<input checked="" type="radio"/> raw	<input type="radio"/>
<pre>1  [ 2    { 3      "name": "example", 4      "surname": "example_Surname", 5      "email": "example@example.com" 6    }, 7    { 8      "name": "avakodo", 9      "surname": "avakodoSurname", 10     "email": "avakodo@avakodo.com" 11   }, 12   { 13     "name": "rakun", 14     "surname": "Şakacı", 15     "email": "rakun@rakun.com" 16   }, 17   ] 18     "name": "future", 19     "surname": "future_Surname", 20     "email": "future@future.com" 21   ] 22 ]</pre>				

**Add customer:**

POST POST/api/addCustomer

post a new customer

<http://localhost:8070/api/addCustomer>

The screenshot displays a REST client interface. At the top, a dropdown menu is set to 'POST' and the URL 'http://localhost:8070/api/addCustomer' is entered. Below this, tabs for 'Params', 'Authorization', 'Headers (9)', 'Body', and 'Pre-requests' are visible. The 'Body' tab is selected, showing a JSON payload in a text editor with line numbers 1 through 5. The payload is: 

```
{ 1: { 2:   "name": "exampleOne", 3:   "surname": "exampleOne_Surname", 4:   "email": "exampleOne@exampleOne.com" 5: }
```

 Below the editor, there are radio buttons for 'none', 'form-data', 'x-www-form-urlencoded', and 'raw', with 'raw' being selected. At the bottom, tabs for 'Body', 'Cookies', 'Headers (8)', and 'Test Results' are shown, with 'Body' selected. Below these tabs are buttons for 'Pretty', 'Raw', 'Preview', and 'Visualize', followed by a 'JSON' dropdown and a menu icon. The response is displayed in a text editor with line numbers 1 through 6: 

```
1 { 2:   "id": 5, 3:   "name": "exampleOne", 4:   "surname": "exampleOne_Surname", 5:   "email": "exampleOne@exampleOne.com" 6: }
```

**Get all customers:**

GET GET/api/customers

get all customers

<http://localhost:8070/api/customers>

The screenshot shows a REST client interface with a GET request to `http://localhost:8070/api/customers`. The response is displayed in JSON format, showing a list of five customer objects. The interface includes tabs for Params, Authorization, Headers (9), Body, and Pre-requests. The response body is shown in the 'Body' tab, with a 'Pretty' view selected. The JSON response is as follows:

```
1  [
2    {
3      "id": 1,
4      "name": "example",
5      "surname": "example_Surname",
6      "email": "example@example.com"
7    },
8    {
9      "id": 2,
10     "name": "avakodo",
11     "surname": "avakodoSurname",
12     "email": "avakodo@avakodo.com"
13   },
14   {
15     "id": 3,
16     "name": "rakun",
17     "surname": "Şakacı",
18     "email": "rakun@rakun.com"
19   },
20   {
21     "id": 4,
22     "name": "future",
23     "surname": "future_Surname",
24     "email": "future@future.com"
25   },
26   {
27     "id": 5,
28     "name": "exampleOne",
29     "surname": "exampleOne_Surname",
30     "email": "exampleOne@exampleOne.com"
31   }
32 ]
```

**Get one customer by id:**

GET GET/api/customer/{id}

get a customer

http://localhost:8070/api/addCustomer/{id}

GET ▼ http://localhost:8070/api/customers/2

Params Authorization Headers (9) Body ● Pre-re

body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ▼

```

1  {
2    "id": 2,
3    "name": "avakodo",
4    "surname": "avakodoSurname",
5    "email": "avakodo@avakodo.com"
6  }

```

+ Options

				id	email	name	surname
<input type="checkbox"/>	Edit	Copy	Delete	1	example@example.com	example	example_Surname
<input type="checkbox"/>	Edit	Copy	Delete	2	avakodo@avakodo.com	avakodo	avakodoSurname
<input type="checkbox"/>	Edit	Copy	Delete	3	rakun@rakun.com	rakun	Şakacı
<input type="checkbox"/>	Edit	Copy	Delete	4	future@future.com	future	future_Surname
<input type="checkbox"/>	Edit	Copy	Delete	5	exampleOne@exampleOne.com	exampleOne	exampleOne_Surname

☐ Check all
 With selected:
 Edit
 Copy
 Delete
 Export

## Update the customer

PUT PUT/api/update

update a customer

<http://localhost:8070/api/update>

The screenshot shows the REST Client interface with a PUT request to `http://localhost:8070/api/update`. The request body is a JSON object: `{ "id": 2, "name": "avakodo", "surname": "avakodoAiles", "email": "avakodo@avakodo.com" }`. The "Body" tab is selected, and the "JSON" format is chosen. The response area is empty.

+ Options				id	email	name	surname
<input type="checkbox"/>				1	example@example.com	example	example_Surname
<input type="checkbox"/>				2	avakodo@avakodo.com	avakodo	avakodoAilesi
<input type="checkbox"/>				3	rakun@rakun.com	rakun	Şakacı
<input type="checkbox"/>				4	future@future.com	future	future_Surname
<input type="checkbox"/>				5	exampleOne@exampleOne.com	exampleOne	exampleOne_Surname

**Upload a file for the customer with the id number 3:**

POST POST/api/customers/{id}/uploadFile

upload a new file for a customer

<http://localhost:8070/api/customers/{id}/uploadFile>

POST ▼ http://localhost:8070/api/customers/3/uploadFile

Params Authorization Headers (9) **Body** ● Pre-request Script Tests Settings

● none ● **form-data** ● x-www-form-urlencoded ● raw ● binary ● GraphQL

KEY	VALUE
<input checked="" type="checkbox"/> documents	nishbs-müşteri-arşivleme-sistemi.pdf ×
Key	Value

**After uploading random files, the DB is:**

- Options

Click the drop-down arrow to toggle column's visibility.

	id	data	name	type	customer_id
<input type="checkbox"/>	1	[BLOB - 462.4 KiB]	Assignment4.pdf	application/pdf	3
<input type="checkbox"/> Edit Copy Delete	2	[BLOB - 359.0 KiB]	CNG445_Assignment1_20201.pdf	application/pdf	2
<input type="checkbox"/> Edit Copy Delete	3	[BLOB - 40.6 KiB]	nishbs-müşteri-arşivleme-sistemi.pdf	application/pdf	3
<input type="checkbox"/> Edit Copy Delete	4	[BLOB - 159.2 KiB]	Vuralcan Asal CV-(eng).pdf	application/pdf	1
<input type="checkbox"/> Edit Copy Delete	5		demoo.txt	text/plain	1

This means, the customers with id number 1, 2 and 3 have some files.



**Download one file by id:**

GET GET/api/downloadFile/{id}

download a file

<http://localhost:8070/api/downloadFile/{id}>

**GET** <http://localhost:8070/api/downloadFile/3>

Params Authorization Headers (9) Body ● Pre-request Script Tests Settings

Body Cookies Headers (10) Test Results 200 OK 46 ms 40

**Müşteri Arşivleme Sistemi:**

Sistem kullanıcısının görevi müşteri ve dosya ekleyip görüntülemektir.

Sistemin temelinde müşteriler vardır, her müşterinin birden fazla dosyası olabilir.

Sistemdeki entity'ler: Kullanıcı, Müşteri, Dosya

Yazacağınız web servisten ve REST API'den beklentilerimiz:

- Müşteri listesinin getirilmesi, müşteri eklenmesi, silinmesi ve güncellenmesi
- Dosya listesinin getirilmesi, dosya eklenmesi, silinmesi ve güncellenmesi

Kullanılacak teknolojiler:

- Java ile Spring boot
- SQLITE, postgre db vs vs kolayınıza hangisi geliyorsa, lokal bir dbde çalışabilirsiniz

Bonus:

- React.js ile yazılmış bir ekranda müşteri ekleme çıkarma yapılabilmesi.

Not: Model, controller ve repository pattern'i gibi bir yapı kullanabilirsiniz.

Çıktılar:

- ER veri modeli (JPEG veya PDF)
- API Dökümantasyonu (PDF veya github README)
- Kodların yüklü olduğu github vs linki.

Deadline: 7 Temmuz 2021 mesai bitimi.

Kodları github, gitlab vs üzerinden paylaşabilirsiniz. Paylaşımınız ufak bir sunum gibi bize anlatmanızı bekleyebiliriz.

Teşekkürler

**Get all files:**

GET GET/api/files

get all files

http://localhost:8070/api/files

GET ⌵ http://localhost:8070/api/files

Params Authorization Headers (9) Body ● Pre

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ⌵

```

1  [
2    {
3      "id": 1,
4      "name": "Assignment4.pdf",
5      "type": "application/pdf",
6      "data":
          "JVBERi0xLjcNCiW1tbW1DQoxIDAg
          XJQcmVmZXJlbnNlcyA2ODkgMCBSPj
          +DQplbmRvYmoNCjMgMCBvYmoNCjw8
          jcgMCBSL0Y5IDMxIDAgUj4+L0V4dE
          +Pi9Qcm9jU2V0Wy9QREYvVGv4dC9J
          ...

```

GET ⌵ http://localhost:8070/api/files

Params Authorization Headers (9) Body ● Pre-request Scrip

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON ⌵ 

```

1  [
2    > { ...
13  },
14  > { ...
25  },
26  > { ...
32  },
33  > { ...
44  },
45  {
46    "id": 5,
47    "name": "demo.txt",
48    "type": "text/plain",
49    "data": "",
50    "customer": 1
51  }
52 ]

```

**GET all files of one customer:**

GET GET/api/customer/{id}/files

get a customer's all files

http://localhost:8070/api/customer/{id}/files

The screenshot shows a REST client interface with the following components:

- Method and URL:** A dropdown menu set to "GET" and a text field containing the URL "http://localhost:8070/api/customer/1/files".
- Request Headers:** A tab labeled "Headers (9)" is selected, showing a list of headers.
- Request Body:** A tab labeled "Body" is selected, showing a JSON response in "Pretty" format.
- Response Body:** The JSON response is displayed in a code editor with line numbers 1 through 21. The response is an array of objects, with the first object expanded to show its details.

```
1 [
2 > { ...
13 },
14 {
15   "id": 5,
16   "name": "demoo.txt",
17   "type": "text/plain",
18   "data": "",
19   "customer": 1
20 }
21 ]
```

**GET one file:**

GET GET/api/files/{id}

get a file

http://localhost:8070/api/files/{id}

The screenshot shows a REST client interface with a GET request to `http://localhost:8070/api/files/4`. The response is displayed in JSON format, showing file details for ID 4.

```

1 {
2   "id": 4,
3   "name": "Vuralcan Asal CV-(eng).pdf",
4   "type": "application/pdf",
5   "data":
      "JVBERi0xLjcNCiW1tbW1DQoxIDAgb2JqDQo8PC9UeX
      mVmZXJlbmNlcyAyMDUgMCBSPj4NCmVuZG9iag0KMiAw
      +Pg0KZW5kb2JqDQozIDAgb2JqDQo8PC9UeXB1L1BhZ2
      1N0YXRlPDwvR1M3IDcgMCBSL0dT0CA4IDAgUj4+L1hP
      +Pi9NZWRpYUJveFsgMCAwIDU5NS4yIDg0MS42XSAvQ2
      +Pg0KZW5kb2JqDQo0IDAgb2JqDQo8PC9GaWx0ZXIvRm
      oKfFTFErvJMKgkHjsV2k23Rb27v7EPfBiR07qG87tt0
  
```

**Before Updating**

- Options

Click the drop-down arrow to toggle column's visibility.

	id	data	name	type	customer_id
<input type="checkbox"/>	1	[BLOB - 462.4 KiB]	Assignment4.pdf	application/pdf	3
<input type="checkbox"/> Edit Copy Delete	2	[BLOB - 359.0 KiB]	CNG445_Assignment1_20201.pdf	application/pdf	2
<input type="checkbox"/> Edit Copy Delete	3	[BLOB - 40.6 KiB]	nishbs-müşteri-arşivleme-sistemi.pdf	application/pdf	3
<input type="checkbox"/> Edit Copy Delete	4	[BLOB - 159.2 KiB]	Vuralcan Asal CV-(eng).pdf	application/pdf	1
<input type="checkbox"/> Edit Copy Delete	5		demoo.txt	text/plain	1

**Update the file:**

PUT PUT/api/file/update/

update a file

http://localhost:8070/api/file/update

The screenshot shows a REST client interface with the following components:

- Method and URL:** PUT http://localhost:8070/api/file/update
- Tabs:** Params, Authorization, Headers (9), Body (selected), Pre-request S
- Body Type:** none, form-data, x-www-form-urlencoded, raw (selected), t
- Body Content (Raw):**

```
1 {
2   "id": 5,
3   "name": "UpdateDemo.txt",
4   "type": "text/plain",
5   "data": "",
6   "customer": {
7     "id": 1,
8     "name": "example",
9     "surname": "example_Surname",
10    "email": "example@example.com"
11  }
12 }
```
- Bottom Tabs:** body (selected), Cookies, Headers (8), Test Results
- Body View Options:** Pretty (selected), Raw, Preview, Visualize, JSON (selected), and a menu icon.
- Body Content (Pretty):**

```
1 {
2   "id": 5,
3   "name": "UpdateDemo.txt",
4   "type": "text/plain",
5   "data": "",
6   "customer": {
7     "id": 1,
8     "name": "example",
9     "surname": "example_Surname",
10    "email": "example@example.com"
11  }
12 }
```

After Updating:

+ Options

 			id	data	name	type	customer_id
<input type="checkbox"/>  Edit  Copy  Delete	1	[BLOB - 462.4 KiB]	Assignment4.pdf	application/pdf	3		
<input type="checkbox"/>  Edit  Copy  Delete	2	[BLOB - 359.0 KiB]	CNG445_Assignment1_20201.pdf	application/pdf	2		
<input type="checkbox"/>  Edit  Copy  Delete	3	[BLOB - 40.6 KiB]	nishbs-müşteri-arşivleme-sistemi.pdf	application/pdf	3		
<input type="checkbox"/>  Edit  Copy  Delete	4	[BLOB - 159.2 KiB]	Vuralcan Asal CV-(eng).pdf	application/pdf	1		
<input type="checkbox"/>  Edit  Copy  Delete	5		UpdateDemo.txt	text/plain	1		

Before Deleting

Customer:

+ Options

				id	email	name	surname
<input type="checkbox"/>		Edit		Copy		Delete	1 example@example.com example example_Surname
<input type="checkbox"/>		Edit		Copy		Delete	2 avakodo@avakodo.com avakodo avakodoAilesi
<input type="checkbox"/>		Edit		Copy		Delete	3 rakun@rakun.com rakun Şakacı
<input type="checkbox"/>		Edit		Copy		Delete	4 future@future.com future future_Surname
<input type="checkbox"/>		Edit		Copy		Delete	5 exampleOne@exampleOne.com exampleOne exampleOne_Surname

Document:

Options								
← T →			id	data	name	type	customer_id	
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	[BLOB - 462.4 KiB]	Assignment4.pdf	application/pdf	3
<input type="checkbox"/>	 Edit	 Copy	 Delete	2	[BLOB - 359.0 KiB]	CNG445_Assignment1_20201.pdf	application/pdf	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	3	[BLOB - 40.6 KiB]	nishbs-müşteri-arşivleme-sistemi.pdf	application/pdf	3
<input type="checkbox"/>	 Edit	 Copy	 Delete	4	[BLOB - 159.2 KiB]	Vuralcan Asal CV-(eng).pdf	application/pdf	1
<input type="checkbox"/>	 Edit	 Copy	 Delete	5		UpdateDemo.txt	text/plain	1

**DELETING PARTS:****Delete the file:**

DELETE DELETE/api/deleteFile/{id}

destroy a file

http://localhost:8070/api/deleteFile/{id}

**DELETE** http://localhost:8070/api/deleteFile/1

Params Authorization Headers (9) Body Pre-request Script T

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize Text

1 The file with id: 1 was deleted!!

+ Options

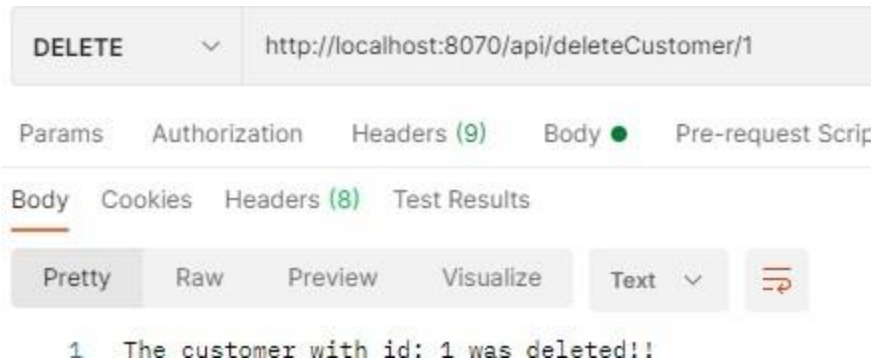
				id	data	name	type	customer_id
<input type="checkbox"/>				2	[BLOB - 359.0 KiB]	CNG445_Assignment1_20201.pdf	application/pdf	2
<input type="checkbox"/>				3	[BLOB - 40.6 KiB]	nishbs-müşteri-arşivleme-sistemi.pdf	application/pdf	3
<input type="checkbox"/>				4	[BLOB - 159.2 KiB]	Vuralcan Asal CV-(eng).pdf	application/pdf	1
<input type="checkbox"/>				5		UpdateDemo.txt	text/plain	1

**Delete the customer:**

DELETE DELETE/api/deleteCustomer/{id}

destroy a customer

http://localhost:8070/api/deleteCustomer/{id}



At the same time, the customer files were deleted.

Customer:

+ Options

				id	email	name	surname
<input type="checkbox"/>	 Edit	 Copy	 Delete	2	avakodo@avakodo.com	avakodo	avakodoAilesi
<input type="checkbox"/>	 Edit	 Copy	 Delete	3	rakun@rakun.com	rakun	Şakacı
<input type="checkbox"/>	 Edit	 Copy	 Delete	4	future@future.com	future	future_Surname
<input type="checkbox"/>	 Edit	 Copy	 Delete	5	exampleOne@exampleOne.com	exampleOne	exampleOne_Surname

Document:

Options

				id	data	name	type	customer_id
<input type="checkbox"/>	 Edit	 Copy	 Delete	2	[BLOB - 359.0 KiB]	CNG445_Assignment1_20201.pdf	application/pdf	2
<input type="checkbox"/>	 Edit	 Copy	 Delete	3	[BLOB - 40.6 KiB]	nishbs-müşteri-arşivleme-sistemi.pdf	application/pdf	3



☐ Check all

With selected:

 Edit

 Copy

 Delete

 Export

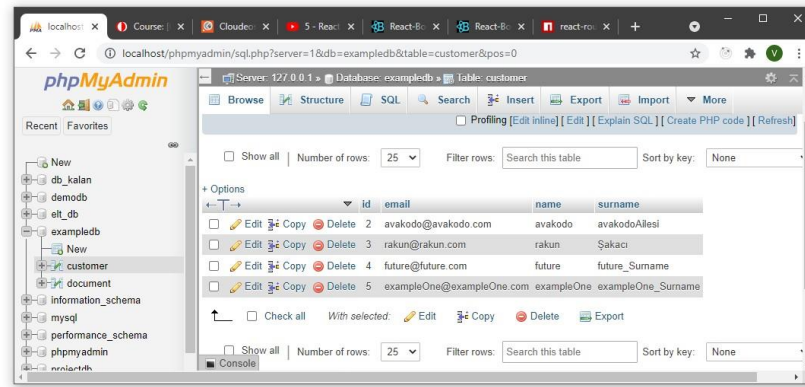


## React Part: Frontend

**Note: The part is not completed. The app has one view which shows the customer data from the backend part.**



ID	Adı	Soyadı	E-mail Adresi	İşlemler
2	avakodo	avakodoAilesi	avakodo@avakodo.com	<button>Dosyalar</button> <button>Güncelle</button> <button>Sil</button>
3	rakun	Şakacı	rakun@rakun.com	<button>Dosyalar</button> <button>Güncelle</button> <button>Sil</button>
4	future	future_Surname	future@future.com	<button>Dosyalar</button> <button>Güncelle</button> <button>Sil</button>
5	exampleOne	exampleOne_Surname	exampleOne@exampleOne.com	<button>Dosyalar</button> <button>Güncelle</button> <button>Sil</button>



id	email	name	surname
2	avakodo@avakodo.com	avakodo	avakodoAilesi
3	rakun@rakun.com	rakun	Şakacı
4	future@future.com	future	future_Surname
5	exampleOne@exampleOne.com	exampleOne	exampleOne_Surname