ANGULAR SPRINGBOOT REST EXAMPLE DOCUMENTATION

SUBMITTED BY ONE STOP CODING

Wednesday, October 25, 2023

Prepared by: Daniel Demesmaecker

Pastoor De Meerleerstraat 103/5

9100 Sint-Niklaas

Phone: 0496/83.28.51

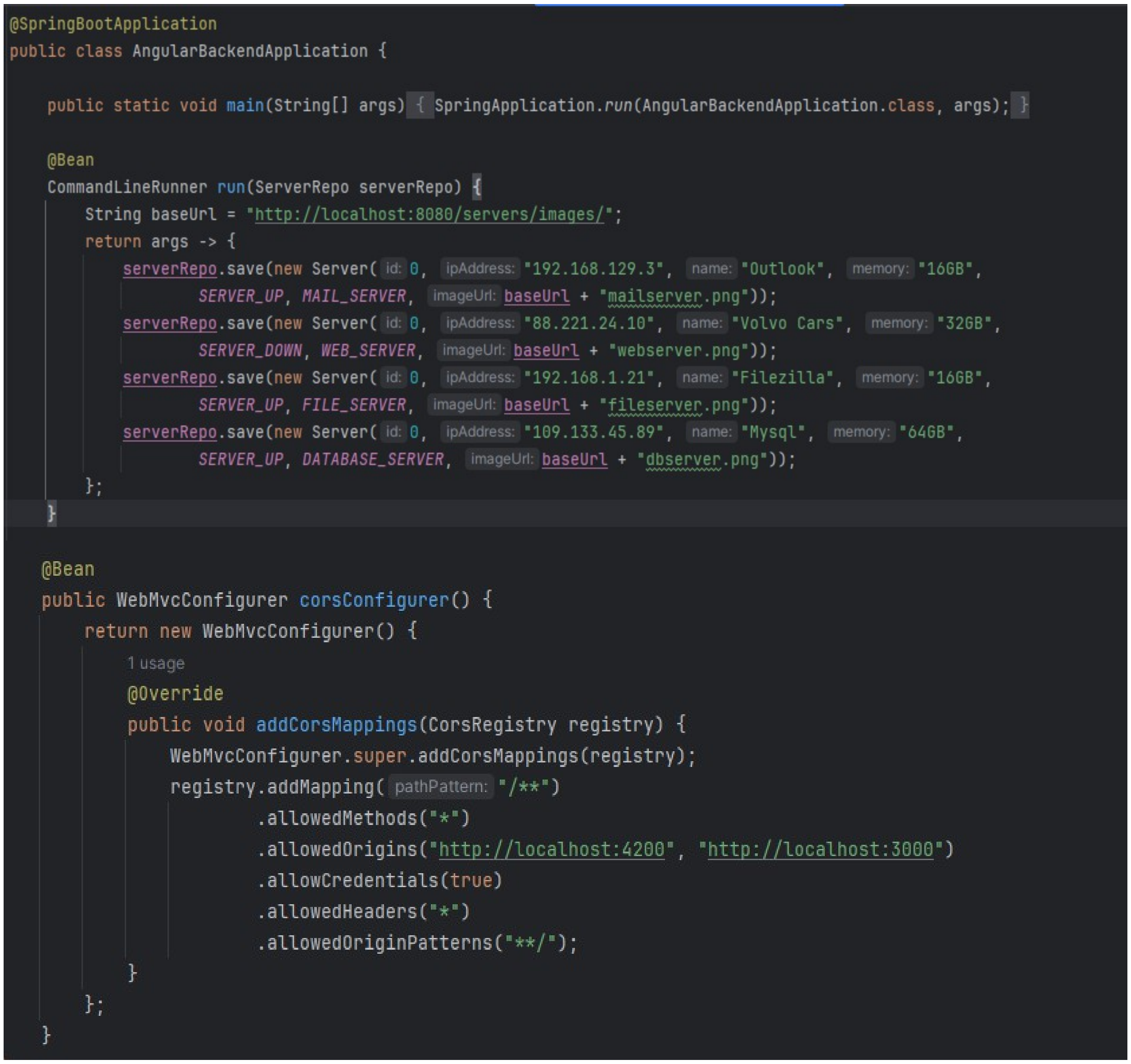
Email: daniel@onestopcoding.com

SERVER SIDE

This is a Server Managing application meant as an simple showcase example. On the backend I'm using a SpringBoot RestApi and mysql database. To make it easier on my self I have choosen to use lombok en hibernate

APPLICATIONCLASS

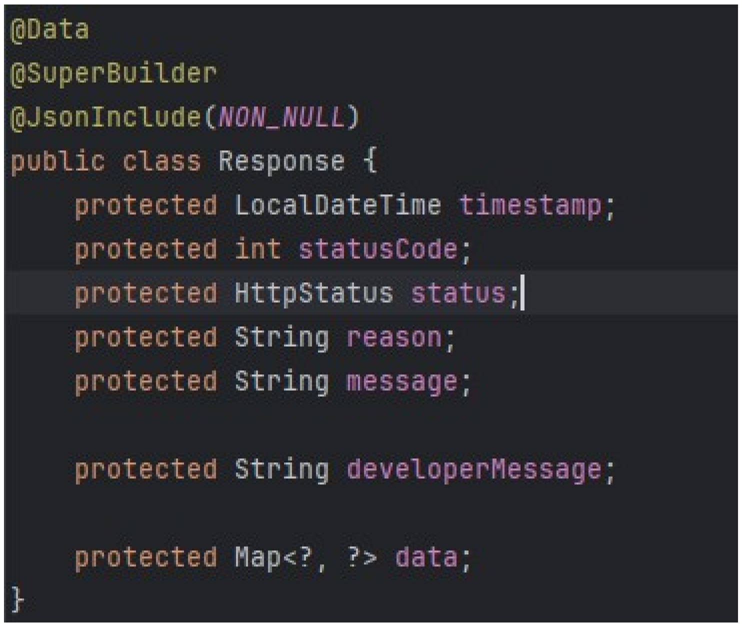
As said it's a simple springbootapplication. I create 2 beans one to populate the database with some basic entries and a second for the cors-configuration:



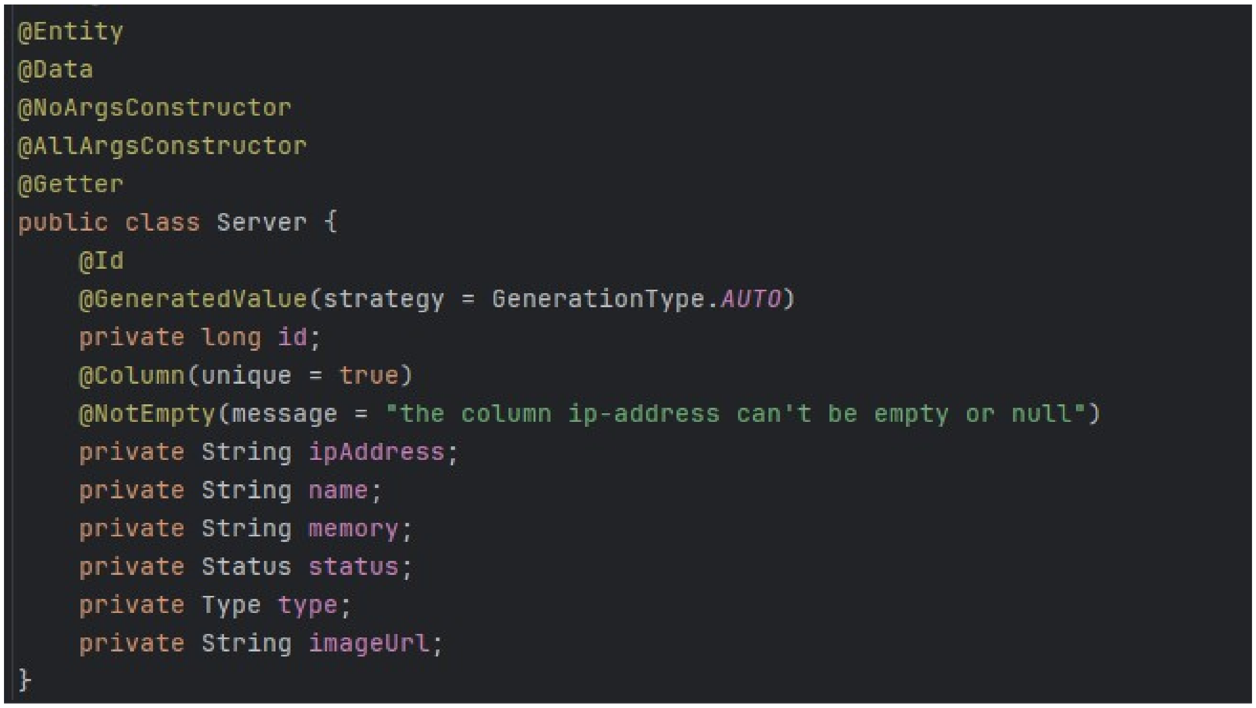
```
@SpringBootApplication
public class AngularBackendApplication {

    public static void main(String[] args) { SpringApplication.run(AngularBackendApplication.class, args); }

    @Bean
    CommandLineRunner run(ServerRepo serverRepo) {
        String baseUrl = "http://localhost:8080/servers/images/";
        return args -> {
            serverRepo.save(new Server( id: 0,  ipAddress: "192.168.129.3",  name: "Outlook",  memory: "16GB",
                    SERVER_UP, MAIL_SERVER,  imageUrl: baseUrl + "mailserver.png"));
            serverRepo.save(new Server( id: 0,  ipAddress: "88.221.24.10",  name: "Volvo Cars",  memory: "32GB",
                    SERVER_DOWN, WEB_SERVER,  imageUrl: baseUrl + "webserver.png"));
            serverRepo.save(new Server( id: 0,  ipAddress: "192.168.1.21",  name: "Filezilla",  memory: "16GB",
                    SERVER_UP, FILE_SERVER,  imageUrl: baseUrl + "fileserver.png"));
            serverRepo.save(new Server( id: 0,  ipAddress: "109.133.45.89",  name: "Mysql",  memory: "64GB",
                    SERVER_UP, DATABASE_SERVER,  imageUrl: baseUrl + "dbserver.png"));
        };
    }

   @Bean
   public WebMvcConfigurer corsConfigurer() {
       return new WebMvcConfigurer() {
           1 usage
           @Override
           public void addCorsMappings(CorsRegistry registry) {
               WebMvcConfigurer.super.addCorsMappings(registry);
               registry.addMapping( pathPattern: "/**")
                       .allowedMethods("*")
                       .allowedOrigins("http://localhost:4200", "http://localhost:3000")
                       .allowCredentials(true)
                       .allowedHeaders("*")
                       .allowedOriginPatterns("**/");
           }
       };
   }
```

Models & databasestructure

The serverside contains 2 models one to build a response to return when the api is called. The other is the databasetable containing all the servers:



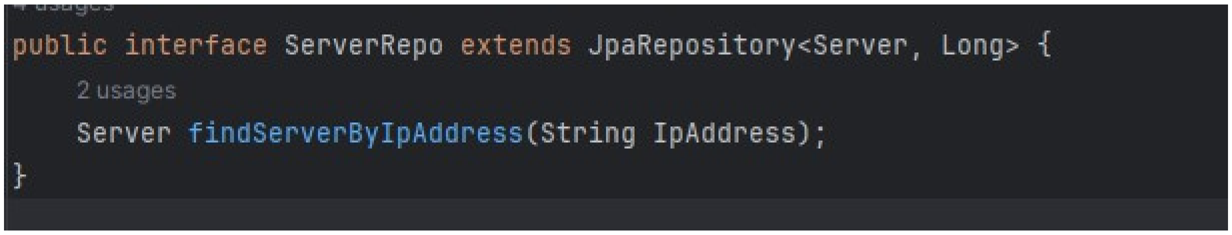
```java
@Data
@SuperBuilder
@JsonInclude(NON_NULL)
public class Response {
    protected LocalDateTime timestamp;
    protected int statusCode;
    protected HttpStatus status;
    protected String reason;
    protected String message;

    protected String developerMessage;

    protected Map<?, ?> data;
}
```



```java
@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
@Getter
public class Server {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private long id;
    @Column(unique = true)
    @NotEmpty(message = "the column ip-address can't be empty or null")
    private String ipAddress;
    private String name;
    private String memory;
    private Status status;
    private Type type;
    private String imageUrl;
}
```

Repository & Service

The repository is a JPA-Repository, for this example I had to only implement one custommethod for the search by ip-functionality.

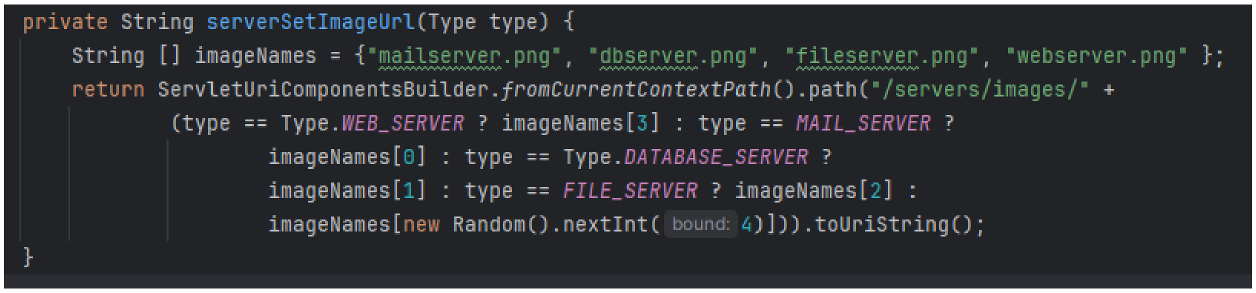


```
public interface ServerRepo extends JpaRepository<Server, Long> {
    2 usages
    Server findServerByIpAddress(String IpAddress);
}
```

The service provides following method:

- Server create(Server server);
- Server ping(String ipAddress) throws IOException;
- List<Server> list(int limit);
- Server get(Long id);
- Server update(Server server);
- Boolean delete(Long id);
- Server findByIp(String ip);

furthermore it contains the method to set the url on the image and is programmed against an interface:



```
private String serverSetImageUrl(Type type) {
    String [] imageNames = {"mailserver.png", "dbserver.png", "fileserver.png", "webserver.png" };
    return ServletUriComponentsBuilder.fromCurrentContextPath().path("/servers/images/" +
            (type == Type.WEB_SERVER ? imageNames[3] : type == MAIL_SERVER ?
                    imageNames[0] : type == Type.DATABASE_SERVER ?
                    imageNames[1] : type == FILE_SERVER ? imageNames[2] :
                    imageNames[new Random().nextInt( bound: 4)])).toUriString();
}
```

RESOURCES

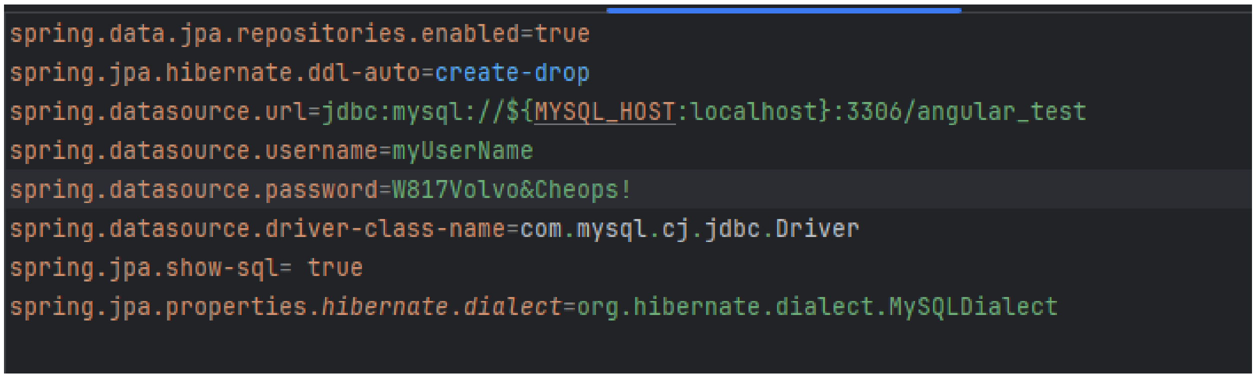
The Serverresource exposes following endpoints:

- servers/list: GET: returns a response containing a list of 30 servers
- servers/ping/{ipAddress}: GET : returns response with message
- servers/{id}: GET: returns response containing requested server
- servers/images/{filename}: GET : Produces: IMAGE_PNG_VALUE
- servers/save: POST: return response with created server
- servers/delete/{id}: DELETE: returns response with message

Following Http.Statusses are being sent:

- 200: OK
- 201: CREATED
- 404: PAGE NOT FOUND
- 500: Internal server error

The configuration is done in application.properties. For this example I didn' use any security or profile so it only contains the database configuration:



```
spring.data.jpa.repositories.enabled=true
spring.jpa.hibernate.ddl-auto=create-drop
spring.datasource.url=jdbc:mysql://${MYSQL_HOST:localhost}:3306/angular_test
spring.datasource.username=myUserName
spring.datasource.password=W817Volvo&Cheops!
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.show-sql= true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
```

FRONT END

The frontend is written in angular and uses bootstrap and npm.

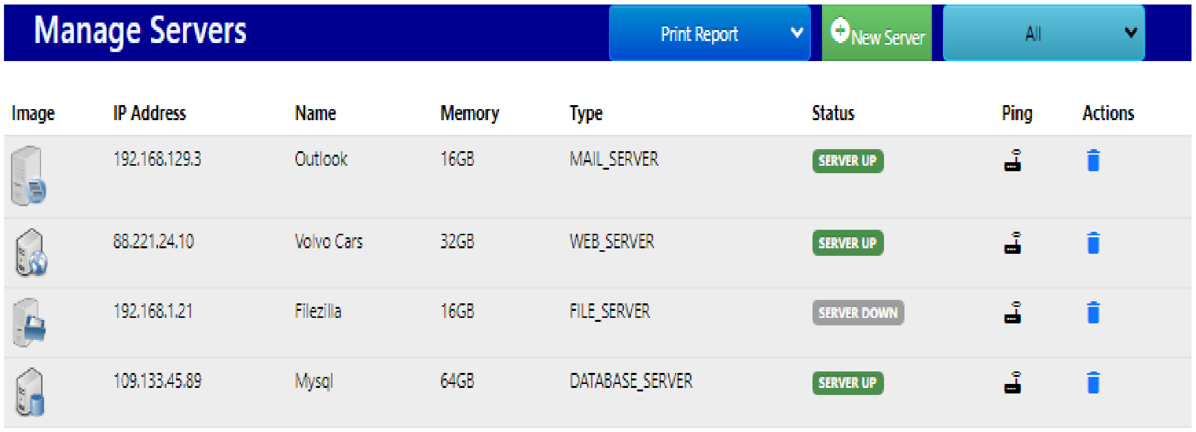


Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER UP		
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER DOWN		
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		

PRINT REPORT

When printing a report the user can choose to download the excel format or save it as an PDF

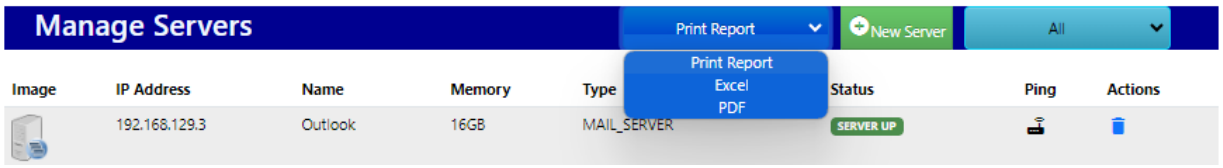


Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		

When choosen to download the excel the excel is downloaded and placed in the downloadfolder. A notification is shown to the user that the action was successful.

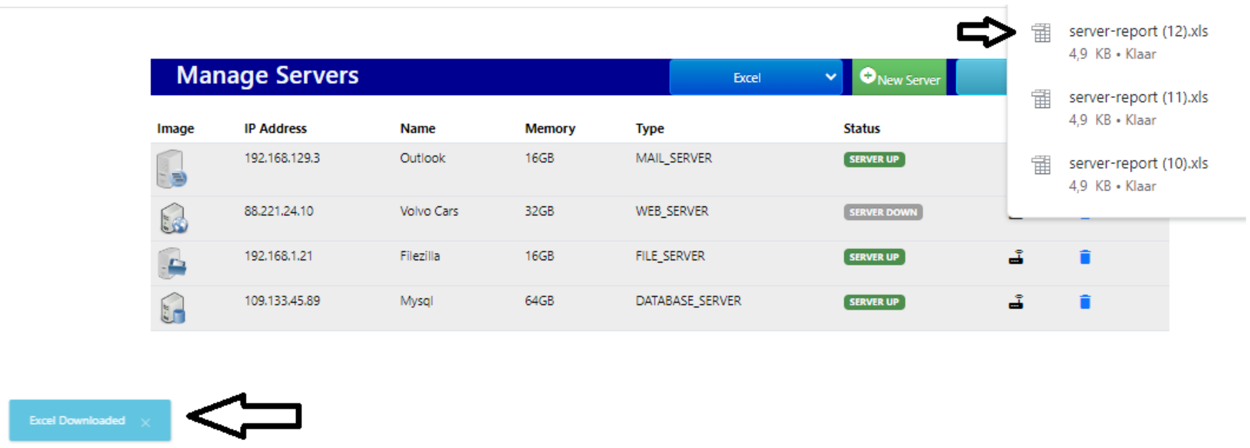
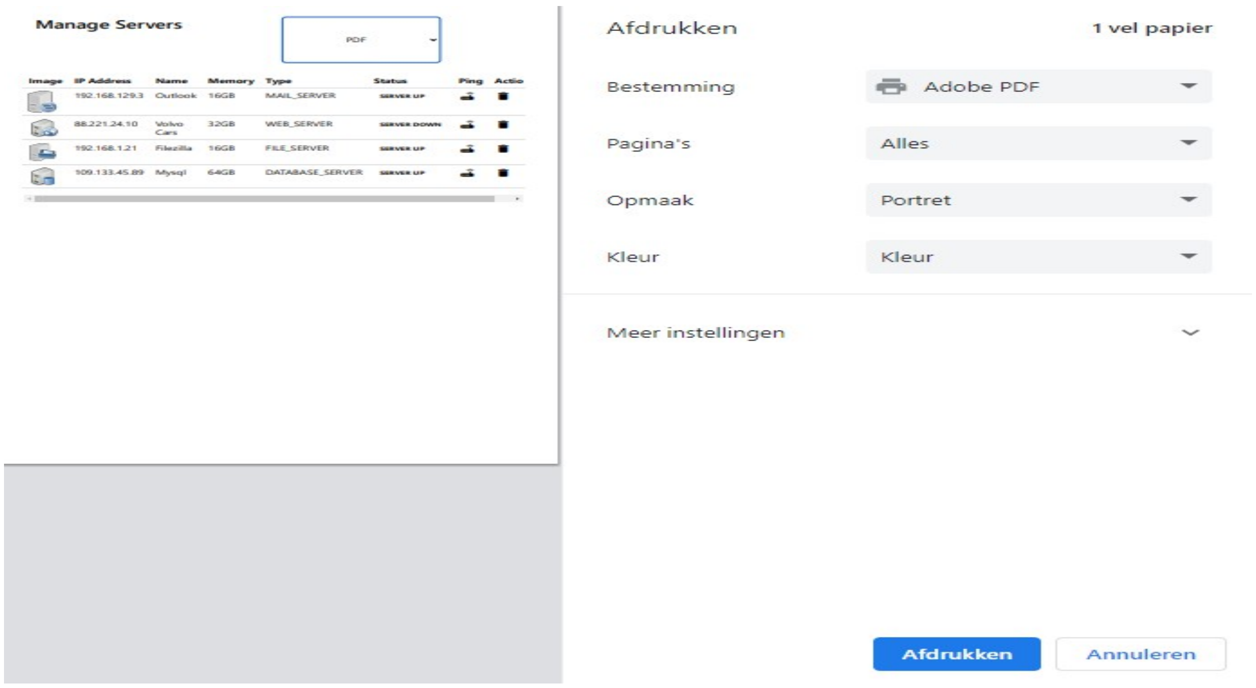


Image	IP Address	Name	Memory	Type	Status
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER DOWN
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP

When you choose to save the pdg the printer window is opened so the user can store it as pdf and then a notification is shown:



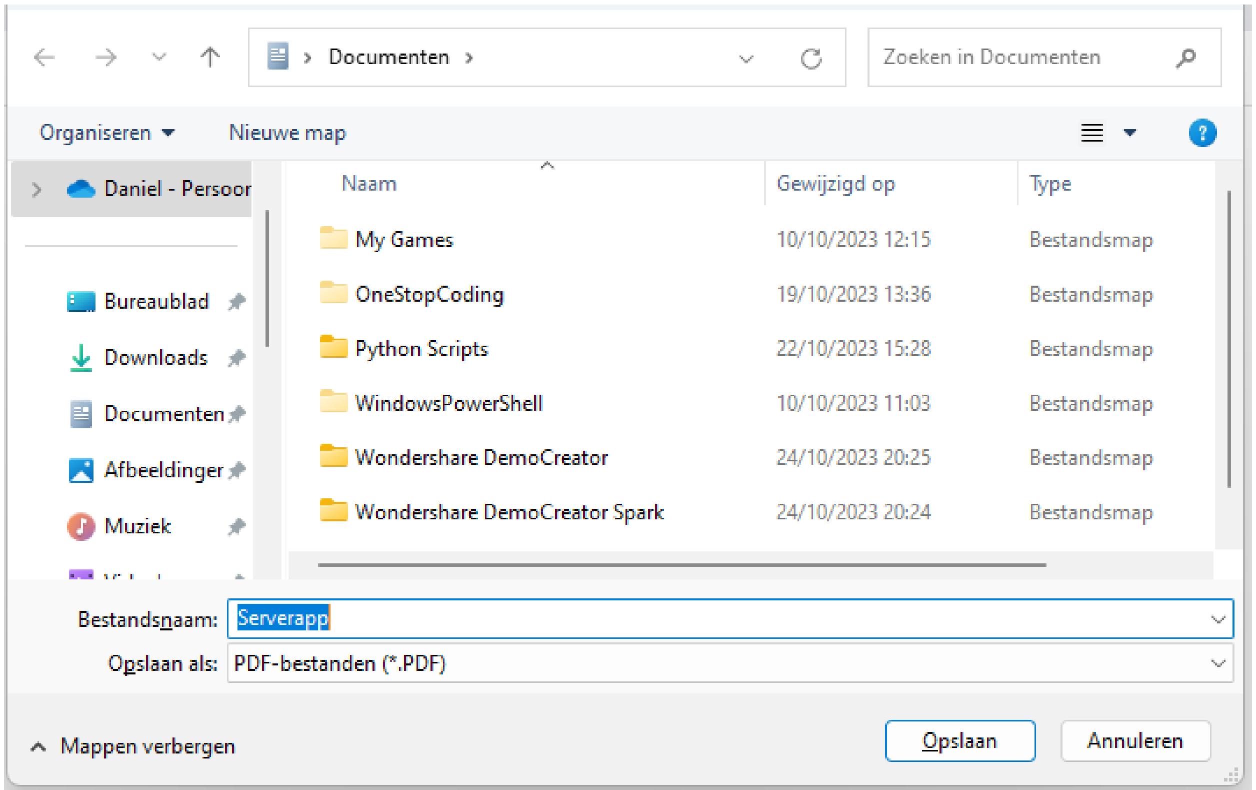
Manage Servers

PDF

Image	IP Address	Name	Memory	Type	Status	Ping	Actio
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER DOWN		
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP		
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		

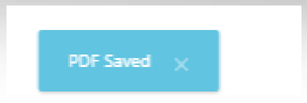
Afdrukken — 1 vel papier

Bestemming	Adobe PDF
Pagina's	Alles
Opmaak	Portret
Kleur	Kleur



Naam	Gewijzigd op	Type
My Games	10/10/2023 12:15	Bestandsmap
OneStopCoding	19/10/2023 13:36	Bestandsmap
Python Scripts	22/10/2023 15:28	Bestandsmap
WindowsPowerShell	10/10/2023 11:03	Bestandsmap
Wondershare DemoCreator	24/10/2023 20:25	Bestandsmap
Wondershare DemoCreator Spark	24/10/2023 20:24	Bestandsmap

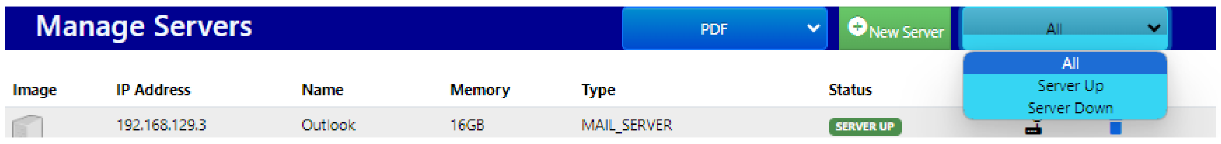
Bestandsnaam: Serverapp
Opslaan als: PDF-bestanden (*.PDF)



PDF Saved

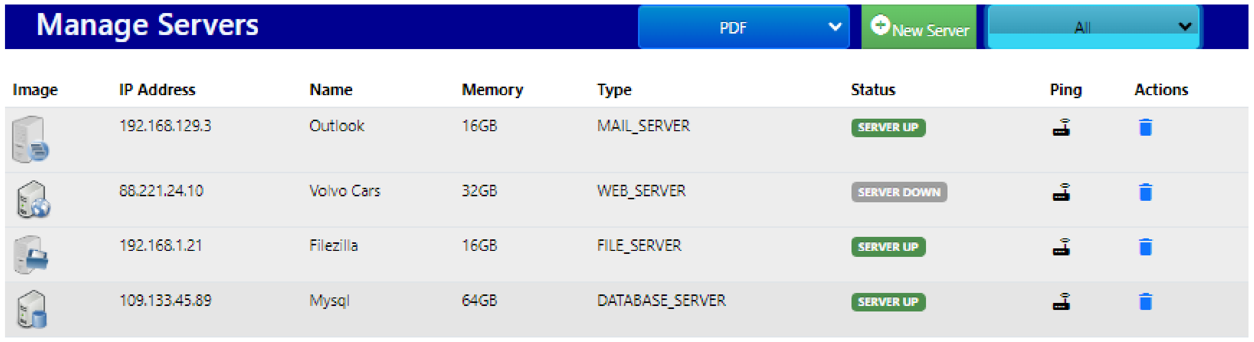
Filtering the result

To filter the result the user can use the dropdown/ Options are All, Server Down and Server Up.



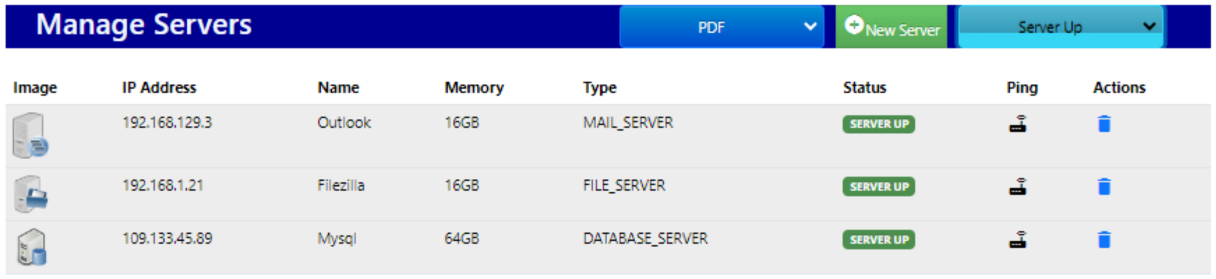
Manage Servers

Image	IP Address	Name	Memory	Type	Status		
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		🗑



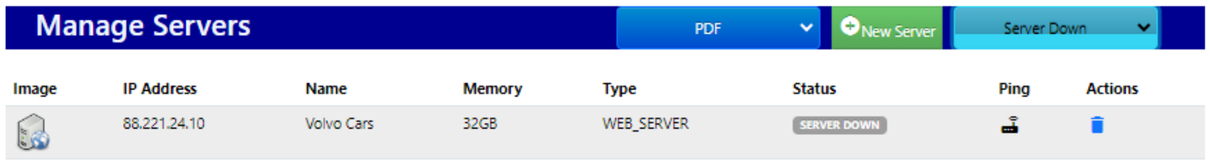
Manage Servers

Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		🗑
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER DOWN		🗑
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP		🗑
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		🗑



Manage Servers

Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		🗑
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP		🗑
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		🗑



Manage Servers

Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER DOWN		🗑

Adding a sever

When the user tries to add an server a modalform is opened. The add button shows a spinner as long the form is invallid and the use is unable to submit it.

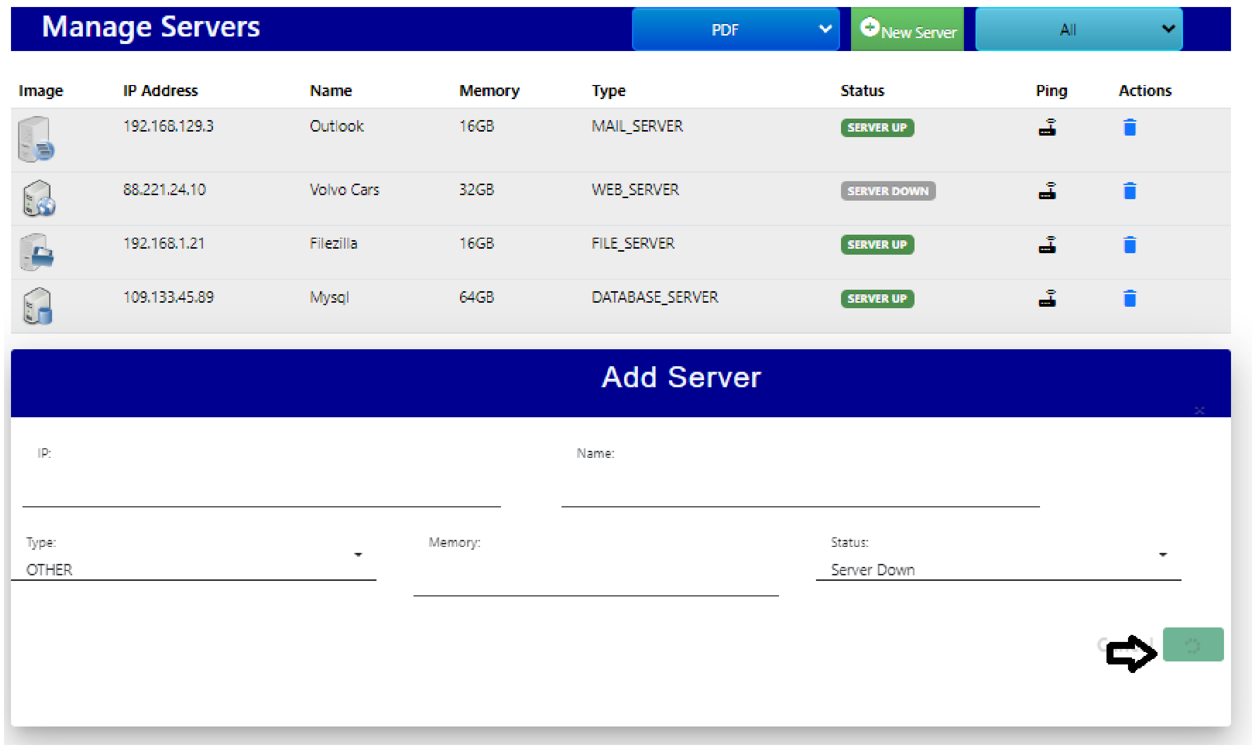


Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER DOWN		
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP		
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		

IP:

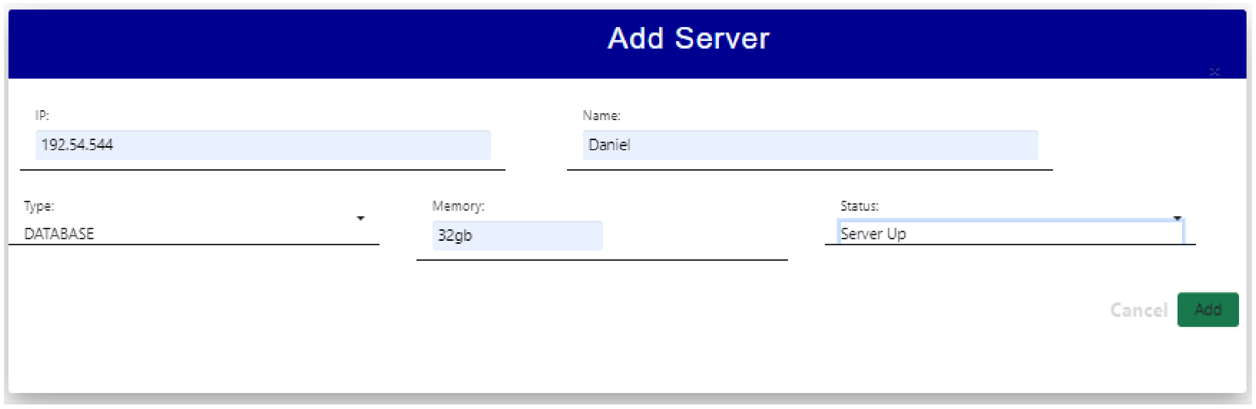
Name:

Type: OTHER

Memory:

Status: Server Down

After formvaludation the user can add the server, the server is added to the table and a notification is shown:



IP: 192.54.544

Name: Daniel

Type: DATABASE

Memory: 32gb

Status: Server Up

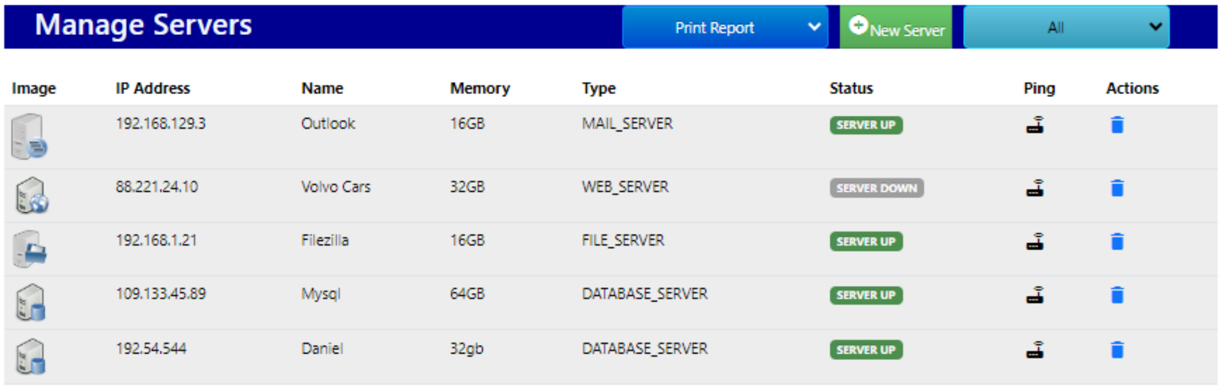


Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER DOWN		
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP		
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		
	192.54.544	Daniel	32gb	DATABASE_SERVER	SERVER UP		

PINGING A SERVER

When pinging a server the method is disabled and a spinner is shown:

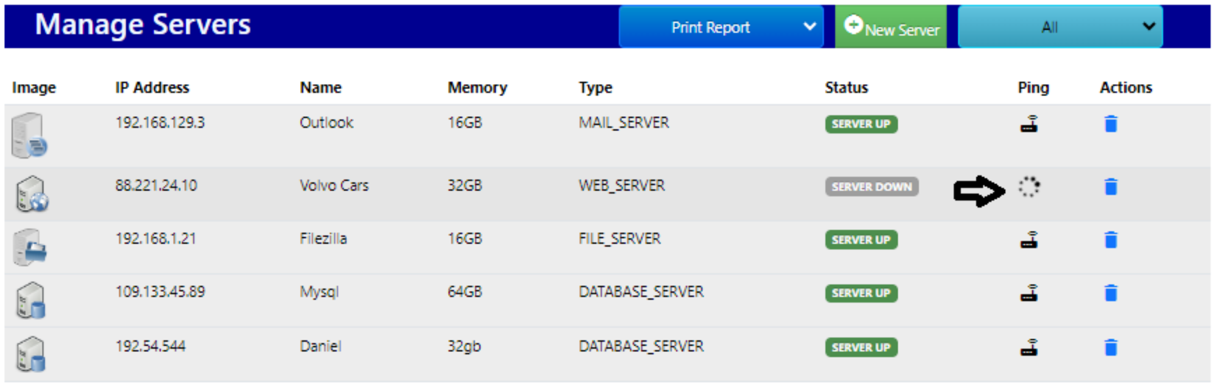


Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER DOWN		
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP		
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		
	192.54.544	Daniel	32gb	DATABASE_SERVER	SERVER UP		

After the ping is completed a notification is shown whether the ping was successful or not and if needed the status is updated.

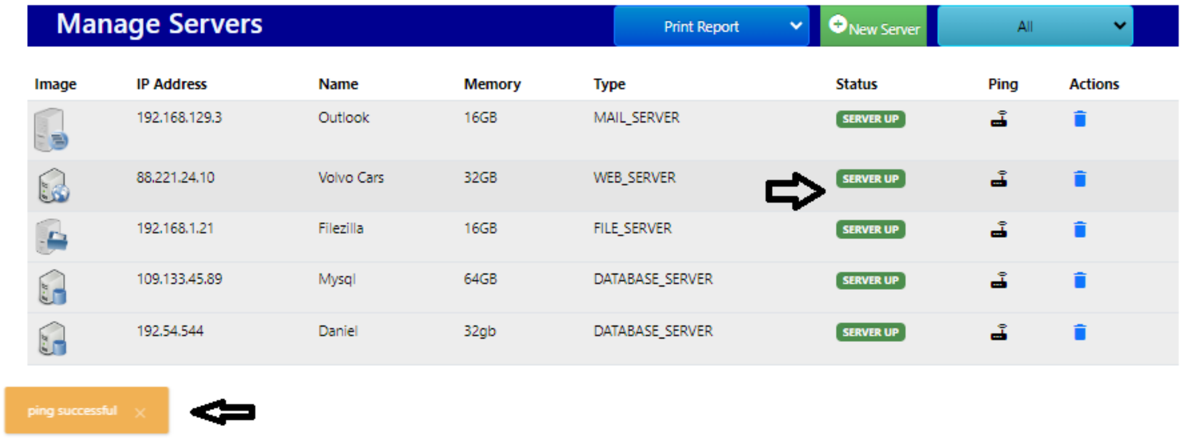


Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER UP		
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP		
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		
	192.54.544	Daniel	32gb	DATABASE_SERVER	SERVER UP		

Deleting a server

When pressing the delte button, the server is removed and a notification is shown:

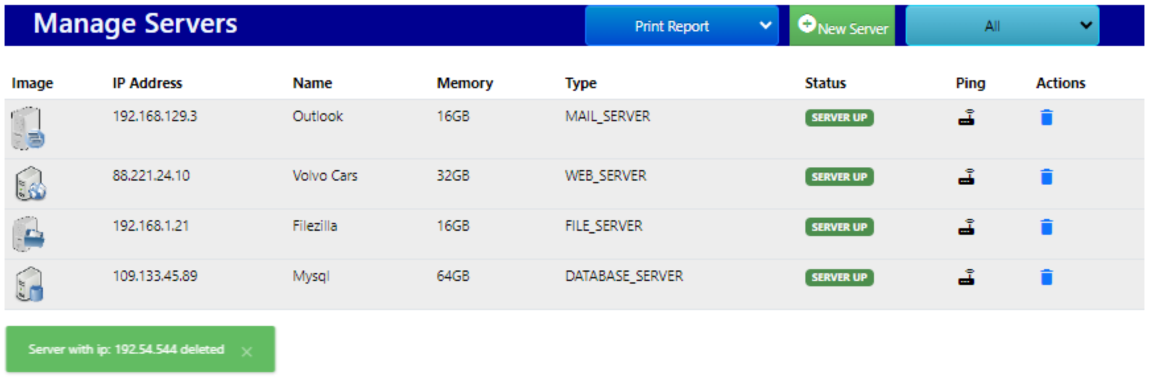


Image	IP Address	Name	Memory	Type	Status	Ping	Actions
	192.168.129.3	Outlook	16GB	MAIL_SERVER	SERVER UP		
	88.221.24.10	Volvo Cars	32GB	WEB_SERVER	SERVER UP		
	192.168.1.21	Filezilla	16GB	FILE_SERVER	SERVER UP		
	109.133.45.89	Mysql	64GB	DATABASE_SERVER	SERVER UP		

Server with ip: 192.54.544 deleted

Sourcecode at [Github]()