Application.java

```
package org.example.context;

import com.fasterxml.jackson.databind.ObjectMapper;
import org.example.service.InvoiceService;
import org.example.service.UserService;

public class Application {
    public static final UserService userService = new
UserService();
    public static final InvoiceService invoiceService =
new InvoiceService(userService);
    public static final ObjectMapper objectMapper = new
ObjectMapper();

}
```

UserService

```
package org.example.service;

import org.example.model.User;
import java.util.UUID;

public class UserService {

    public User findById(String id) {
        String randomName = UUID.randomUUID().toString();
        // always "finds" the user, every user has a
random name
        return new User(id,randomName);
    }
}
```

InvoiceService

```
package org.example.service;

import org.example.context.Application;
import org.example.model.Invoice;
```

```java
import org.example.model.User;

import java.util.List;
import java.util.concurrent.CopyOnWriteArrayList;

public class InvoiceService {

    private final UserService userService;

    private List<Invoice> invoices = new
CopyOnWriteArrayList<>(); // (1)

    public InvoiceService(UserService userService) {
        this.userService = userService;
    }

    public List<Invoice> findAll() {
        // Debugging : System.out.println("in findAll");
        return invoices;
    }
    public Invoice create(String userId, Integer amount){
        User user = userService.findById(userId);
        if (user == null) {
            throw new IllegalStateException();
        }
    //  TODO real pdf creation and storing it on network
server

        Invoice invoice = new
Invoice(userId,amount,"http://www.africau.edu/images/
default/sample.pdf");

        /* Debugging old style
        System.out.print("Create
org.example.model.Invoice: ");
        System.out.println(invoice);
        */
        invoices.add(invoice); // hier wird Invoice
gecloned (für CopyOnWrite..)
        return invoice;
    }
}
```

Invoice

```java
package org.example.model;

import com.fasterxml.jackson.annotation.JsonProperty;

import java.util.UUID;

public class Invoice {
    private String id;

    @JsonProperty("user_id")
    private String userId;

    @JsonProperty("pdf_url")
    private    String pdfUrl;

    private Integer amount;

    public Invoice(){
    }

    public Invoice(String userId, Integer amount, String
pdfUrl){
        this.id = UUID.randomUUID().toString();
        this.userId = userId;
        this.pdfUrl = pdfUrl;
        this.amount = amount;
    }

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }

    public String getUserId() {
        return userId;
    }
    public void setUserId(String userId) {
        this.userId = userId;
    }
```

```
    public String getPdfUrl(){
        return pdfUrl;
    }
    public void setPdfUrl(String pdfUrl){
        this.pdfUrl = pdfUrl;
    }

    public Integer getAmount() {
        return amount;
    }
    public void setAmount(Integer amount){
        this.amount = amount;
    }
}
```

User

```
package org.example.model;

public class User {

    private String id;
    private String name;

    public User(){
    }

    public User(String id, String name) {
        this.id = id;
        this.name =name;
    }
    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
```

MyInvoiceServlet

```
package org.example.web;

import com.fasterxml.jackson.databind.ObjectMapper;
import org.example.context.Application;
import org.example.model.Invoice;
import org.example.service.InvoiceService;

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.util.List;

public class MyInvoiceServlet extends HttpServlet {

    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response) throws IOException{
        if (request.getRequestURI().equalsIgnoreCase("/
invoices")) {
            String userId =
request.getParameter("user_id");
            Integer amount =
Integer.valueOf(request.getParameter("amount"));

            Invoice invoice =
Application.invoiceService.create(userId, amount);

            response.setContentType("application/json;
charset=UTF-8");
            String json = new
ObjectMapper().writeValueAsString(invoice);
            response.getWriter().print(json);
        }
        else {

response.setStatus(HttpServletResponse.SC_NOT_FOUND);
        }
    }
    @Override
    protected void doGet(HttpServletRequest
request,HttpServletResponse response) throws IOException
{
```

```
        if
(request.getRequestURI().equalsIgnoreCase("/")) {
            response.setContentType("text/html;
charset=UTF-8");
            response.getWriter().print(
                    "<html>\n" +
                            "<body>\n" +
                            "<h1>Hello Barbara</h1>\n" +
                            "<p>This is my first,
embedded Tomcat HTML Page!</p>\n" +
                            "</body>\n" +
                            "</html>\n");
        }
        else if
(request.getRequestURI().equalsIgnoreCase("/invoices"))
        {
            response.setContentType("application/json;
charset=UTF-8");
            List<Invoice> invoices =
Application.invoiceService.findAll();

response.getWriter().print(Application.objectMapper.write
ValueAsString(invoices));
        }
    }
}
```

ApplicationLauncher

```
package org.example;

import org.apache.catalina.Context;
import org.apache.catalina.LifecycleException;
import org.apache.catalina.Wrapper;
import org.apache.catalina.startup.Tomcat;
import org.example.web.MyInvoiceServlet;

public class ApplicationLauncher {
    public static void main(String[] args) throws
LifecycleException {
        Tomcat tomcat = new Tomcat();
        tomcat.setPort(8080);
```

```
        tomcat.getConnector();

        Context ctx = tomcat.addContext("", null);
        Wrapper servlet = Tomcat.addServlet(ctx,
"org.example.web.MyInvoiceServlet", new
MyInvoiceServlet());
        servlet.setLoadOnStartup(1);
        servlet.addMapping("/*");

        tomcat.start();
    }
}
```

Pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"
         xsi:schemaLocation="http://maven.apache.org/POM/
4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>MyInvoice</artifactId>
    <version>1.0-SNAPSHOT</version>

    <packaging>jar</packaging>

    <properties>
        <maven.compiler.source>15</maven.compiler.source>
        <maven.compiler.target>15</maven.compiler.target>
        <project.build.sourceEncodeing>UTF-8</
project.build.sourceEncodeing>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.apache.tomcat.embed</groupId>
            <artifactId>tomcat-embed-core</artifactId>
            <version>9.0.36</version>
        </dependency>
```

```
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-databind</artifactId>
            <version>2.11.3</version>
        </dependency>
    </dependencies>

    <build>
             <plugins>
                 <plugin>
                    <groupId>org.apache.maven.plugins</
groupId>
                    <artifactId>maven-compiler-plugin</
artifactId>
                    <version>3.8.1</version>
                </plugin>

            <!-- required tag:: shade to add external
libraries to the jar -->
                <plugin>
                    <groupId>org.apache.maven.plugins</
groupId>
                    <artifactId>maven-shade-plugin</
artifactId>
                    <version>3.2.4</version>
                    <executions>
                        <execution>
                            <phase>package</phase>
                            <goals>
                                <goal>shade</goal>
                            </goals>
                            <configuration>
                                <transformers>
                                <transformer
implementation="org.apache.maven.plugins.shade.resource.M
anifestResourceTransformer">

<mainClass>org.example.ApplicationLauncher</mainClass>
                                </transformer>
                                </transformers>
                            </configuration>
                        </execution>
                    </executions>
                </plugin>
```

```
            </plugins>
        </build>

</project>
```