Projektdokumentation

**Handbuch Softwareentwicklung – Webapplikation**

**Titel:** FHJ-WebService

|  |  |  |
| --- | --- | --- |
| **Auftraggeber** |  | Dr. Baptiste Alcalde, [Baptiste.Alcalde@fh-joanneum.at](mailto:Baptiste.Alcalde@fh-joanneum.at) |
| **Autoren des Dokuments** |  | Andreas Mossier BSc, [Andreas.Mossier@edu.fh-joanneum.at](mailto:Andreas.Mossier@edu.fh-joanneum.at)  Mina Shokrollahi BSc, [Mina.Shokrollahi@edu.fh-joanneum.at](mailto:Mina.Shokrollahi@edu.fh-joanneum.at)  Romana Ausim BSc, [Romana.Ausim@edu.fh-joanneum.at](mailto:Romana.Ausim@edu.fh-joanneum.at) |
| **Version des Dokuments**  **Dateiname** |  | Versionsnummer 1.0 |
| **Ort, Datum** |  | Graz, 16. Jänner 2017 |
| **Klassifizierung** |  | Benutzerhandbuch |
| **Projektkürzel** |  | FHJ-WS |

**Dokumentinformationen**

**Allgemeine Informationen**

|  |  |
| --- | --- |
| **Beschreibung** | Beschreibung der Maßnahmen und etwaiger Vorkehrungen für den Betrieb des Software-Produkts FHJ-WS.  Detaillierte Beschreibung von Installationen notwendiger .jar-Bibliotheken, Serverkonfigurationen, Datenbankverbindung und Einbindung externer Stylesheets für den erfolgreichen Betrieb der Webapplikation (für Testzwecke). |
| **Empfänger / Nutzer** | Dr. Baptiste Alcalde |
| **Zugehörige Dokumentationen** | Beschreibung der einzelnen Funktionen, Klassen, HTML-Seiten, CSS-Styles direkt beim Programmcode als Kommentare hinterlegt. |

**Änderungsnachweis**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Status** | **Änderung** | **durch** | **gültig ab** |
| 1.0 | Entwurf | Ersterstellung | Andreas Mossier | 28. November 2016 |
|  |  |  |  |  |

**Verantwortlichkeit**

|  |  |  |  |
| --- | --- | --- | --- |
| **Rolle** (Autor, Prüfer) | **Name** | **Unterschrift** | **Datum** |
| Autor | Andreas Mossier |  |  |
| Autorin | Mina Shokrollahi |  |  |
| Autorin | Romana Ausim |  |  |
| Prüfer | Baptiste Alcalde |  |  |

**Inhaltsverzeichnis**

[Entwicklungs- und Testumgebung 4](#_Toc472358214)

[Webserver Apache Tomcat Setup 4](#_Toc472358215)

[Server in NetBeans hinzufügen 4](#_Toc472358216)

[Projektzuweisung des Servers 4](#_Toc472358217)

[Start des Projektes 5](#_Toc472358218)

[Projekt 7](#_Toc472358219)

[Backend 7](#_Toc472358220)

[Webserver: Apache Tomcat 7](#_Toc472358221)

[Datenbank: JDBC + JPA 7](#_Toc472358222)

[Frontend 8](#_Toc472358223)

[Java Server Pages (JSP) 8](#_Toc472358224)

[Servlets 8](#_Toc472358225)

[Beans 8](#_Toc472358226)

[HTML 9](#_Toc472358227)

[CSS 9](#_Toc472358228)

[Java 9](#_Toc472358229)

[JavaScript 9](#_Toc472358230)

[Verwendete Muster 9](#_Toc472358231)

[Abbildungs-/Tabellenverzeichnis 10](#_Toc472358232)

# Entwicklungs- und Testumgebung

An die Entwicklungs- und Testumgebung sind folgende minimale Anforderungen gestellt (Gewisse Abweichungen sind möglich):

Tabelle 1: Minimale Anforderungen für die Entwicklungs- und Testumgebung

|  |  |
| --- | --- |
|  |  |
| Java SE Development Kit | jdk1.8.0\_101 |
| Entwicklungsumgebung | NetBeans IDE Download Bundle  Java EE (Enterprise Edition) |
| NetBeans Plugins (Tools 🡪 Plugins) | JPA Modeler Core 🡪 ERD-Entwicklung und Entitätengenerierung  Java Persistence API 🡪 JDBC Datenbank (lokal) |
| NetBeans Java .jar-files (Rechtsklick auf Projekt 🡪 Properties 🡪 Libraries) | Java DB Driver:  Derby.jar  Derbyclient.jar  Derbynet.jar  Derbyrun.jar  Eclipselink-2.2.0.jar  Javax.persistence.jar  Befinden sich unter anderem auch im Projektordner:  \FHJ-WebService\src\java\libraries |
| Webserver apache tomcat | 8.5.8  Core: [32-bit/64-bit Windows Service Installer](http://www-eu.apache.org/dist/tomcat/tomcat-8/v8.5.8/bin/apache-tomcat-8.5.8.exe) |
|  |  |

## Webserver Apache Tomcat Setup

Ausführen der heruntergeladenen Datei: apache-tomcat-8.5.8.exe

Klick auf **Next** 🡪 **I Agree** 🡪 Select the type of install: Full – **Next** 🡪 Ports: nichts zu ändern – Tomcat Administrator Login: User Name: admin – Password: admin **Next** 🡪 Pfad zu Java SE hinzufügen: “C:\Program Files\Java\jre….” **Next** 🡪 Installationsort auswählen und **Finish**

### Server in NetBeans hinzufügen

Tools 🡪 Servers 🡪 Add Server 🡪 Apache Tomcat or TomEE – Next 🡪 Server Location: Installationsort vom Webserver Apache Tomcat | User Name: admin | Password: admin | Finish

Unter Pfad: C:\Users\......\AppData\Roaming\NetBeans\8.2 🡪 tomcat80.properties ändern:

tomcat.home=C:\\Program Files\\Apache Software Foundation\\Tomcat 8.5

tomcat.url=http://localhost:8080

tomcat.username=admin

### Projektzuweisung des Servers

(falls der Server noch nicht zugewiesen wurde)

Rechtsklick auf Projekt 🡪 Properties 🡪 Run 🡪 Server auswählen - OK

## Start des Projektes

1. Datenbank hinzufügen:

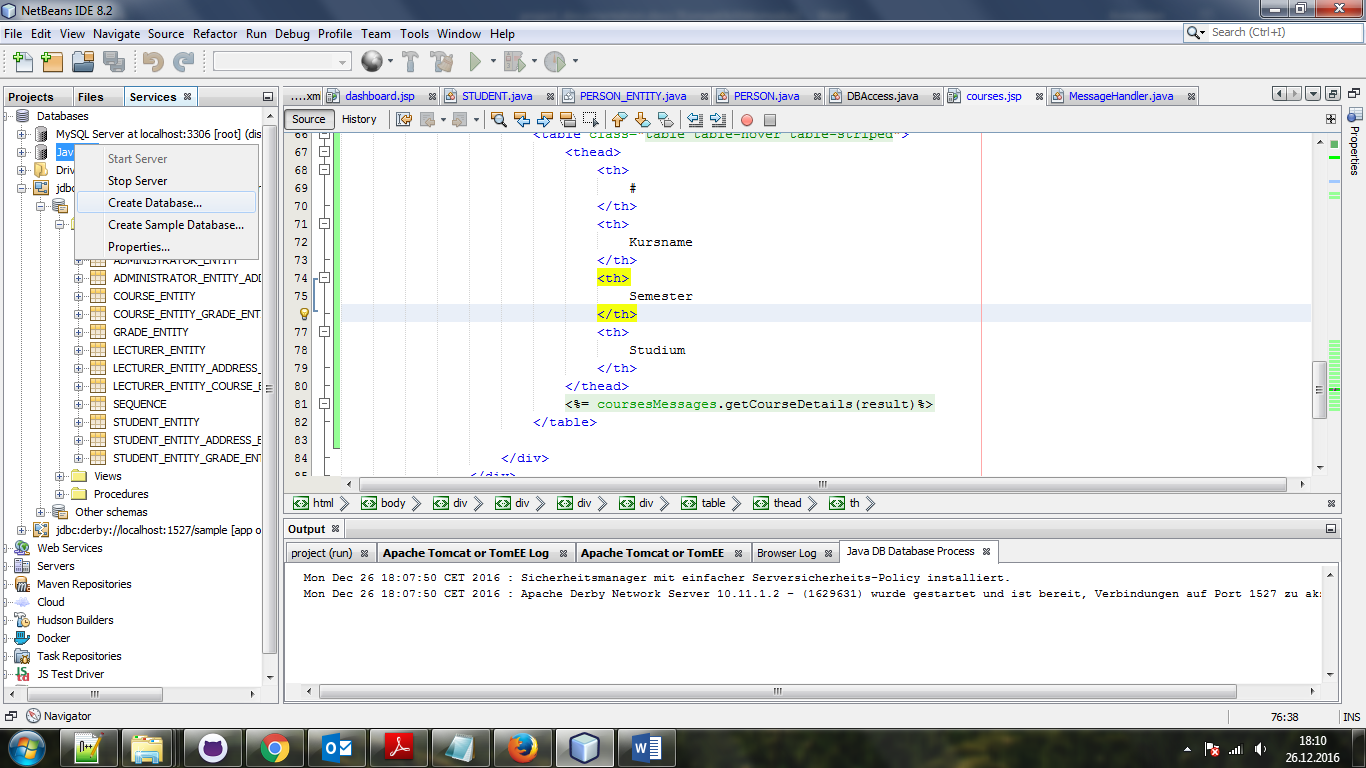


Abbildung 1: Projektstart - Datenbank

1. Folgende Konfigurationen vornehmen (Passwort: 123)

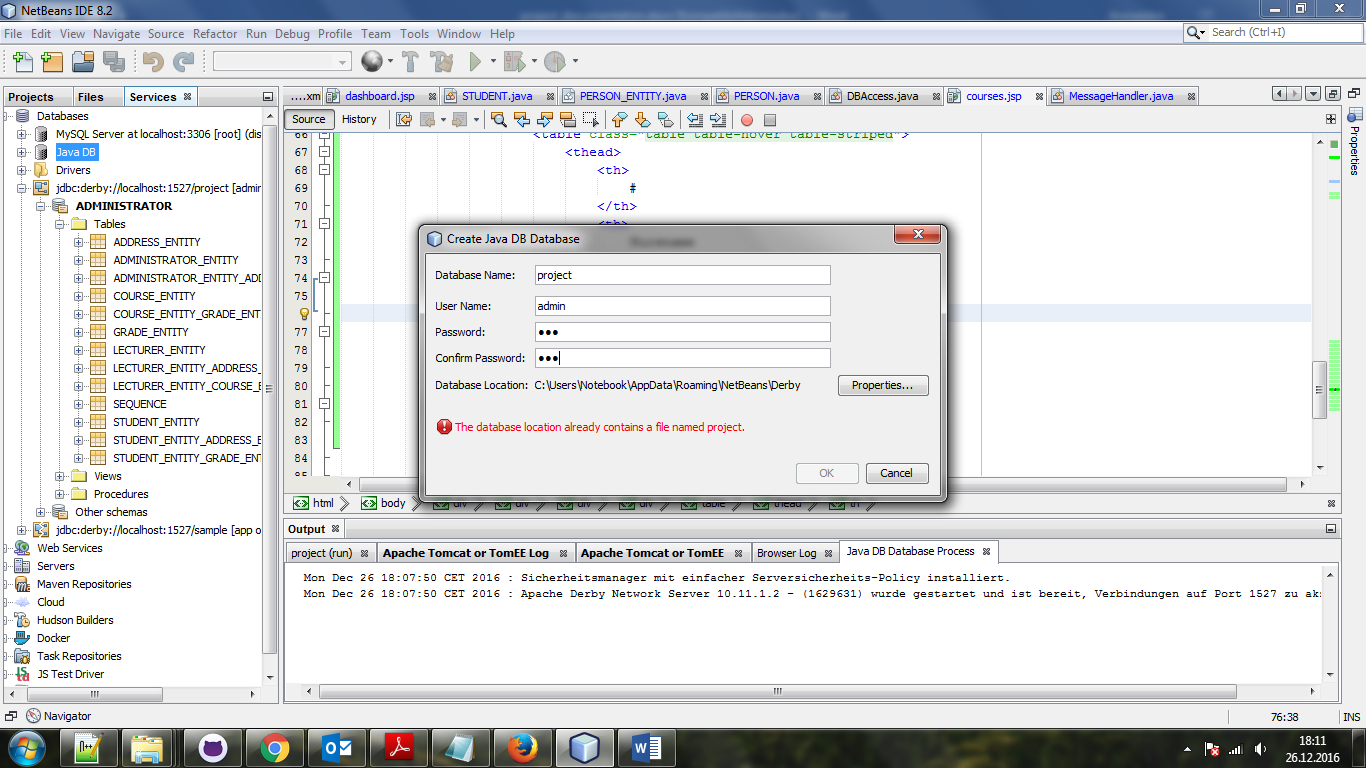


Abbildung 2: Projektstart - Anmeldedaten

1. Beim ersten Start bei neu erstellter Datenbank auf Connect klicken

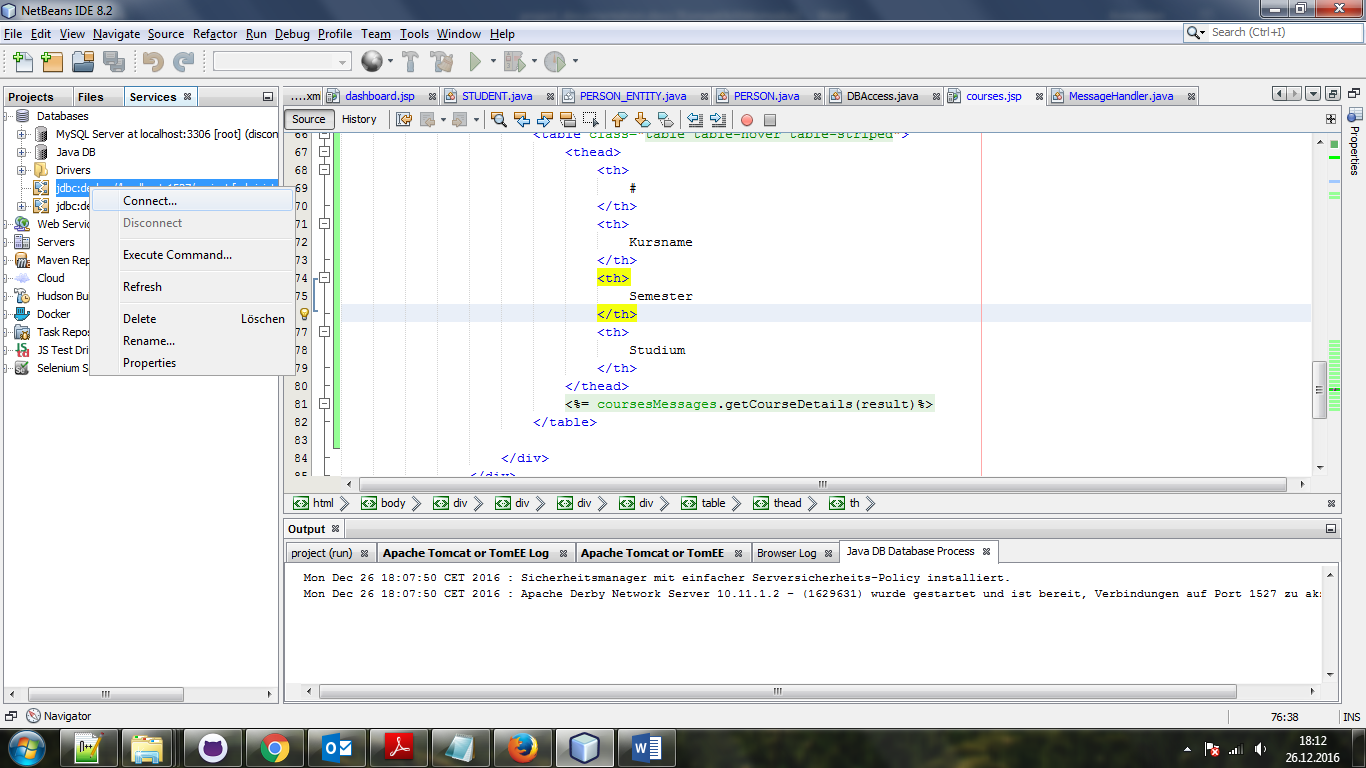


Abbildung 3: Projektstart - Datenbankverbindung

1. In der Datei: „persistence.xml“ die folgende Zeile ändern:

<property name="eclipselink.ddl-generation" value="none"/>

**Beim ersten Start des Projektes mit dieser Zeile überschreiben (für die Generierung der Entitäten)**

<property name="eclipselink.ddl-generation" value="drop-and-create-tables"/>

1. In der Datei: index.jsp die folgenden Zeilen auskommentieren

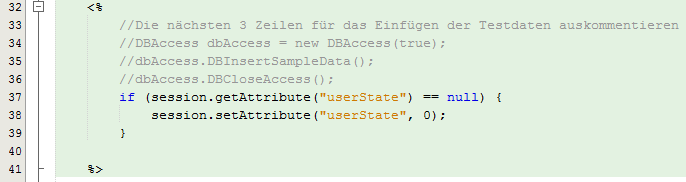


Abbildung 4: Projektstart - Konfiguration1

1. Auf „Run Project“ klicken

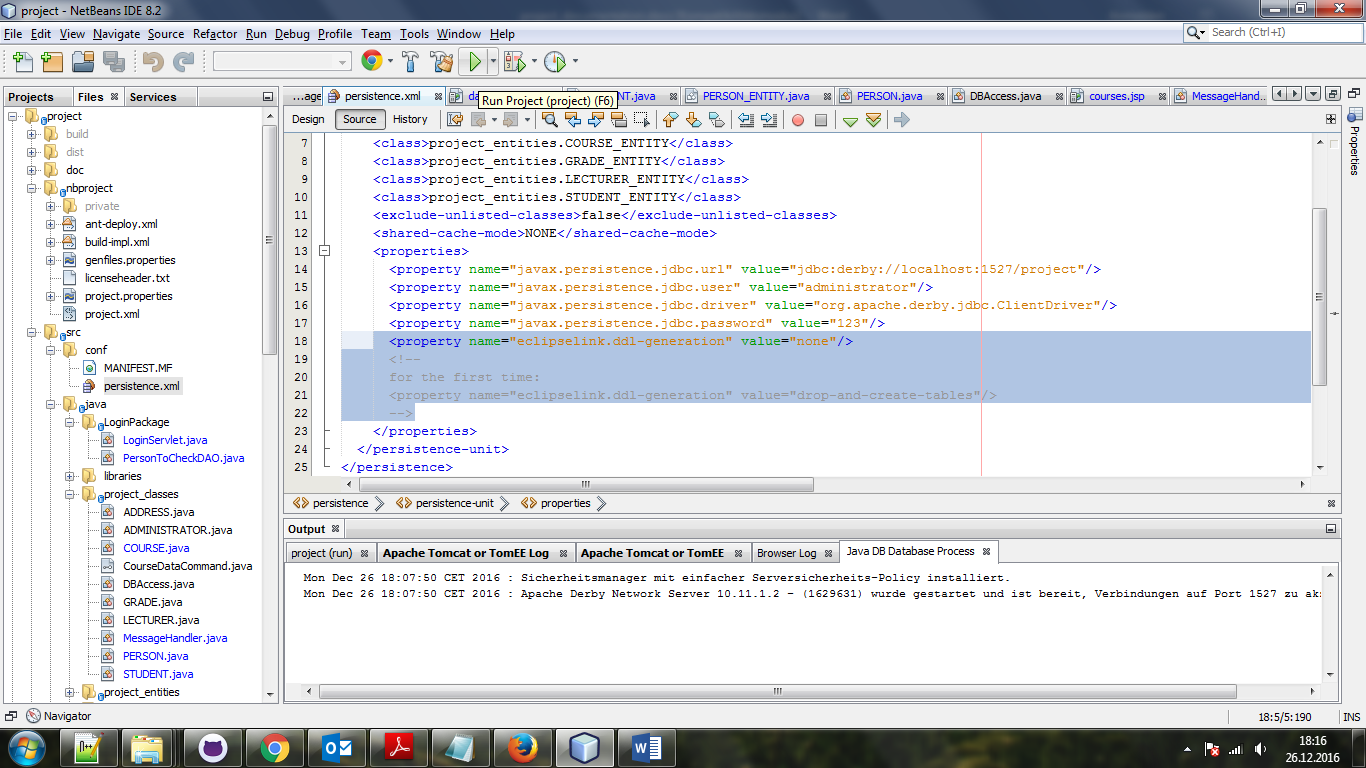


Abbildung 5: Projektstart - Konfiguration2

1. Beim ersten Start ist der Benutzername und das Passwort für den Apache Server einzugeben:
   1. Benutzername: admin
   2. Passwort: admin
2. Webseite schließen und Schritt 5 und 4 wieder rückgängig machen

Nun sind die Testdaten vorhanden und das Projekt kann getestet werden.

**Beispiel für Student-Login:**

Benutzername: mossiera16

Passwort: test

**Beispiel für Vortragender-Login:**

Benutzername: alcaldebaptiste

Passwort: test

**Beispiel für Administrator-Login:**

Benutzername: mossieraadmin

Passwort: test

# Projekt

Das Projekt umfasst einen Webservice mit dem Vortragende Lehrveranstaltungen/Kurse und Noten/Ergebnisse verwalten können und Studenten ihre Noten einsehen und sich für Kurse eintragen lassen können. Zudem sollen Administratoren Studenten, Vortragende und Kurse editieren, hinzufügen und entfernen können.

Das Projekt besteht aus 2 großen Bereichen:

* **Backend:** 
  + Webserver: Apache Tomcat
  + Java DataBase Connectivity (JDBC) + Java Persistency API (JPA) 🡪 Datenbank
* **Frontend:**
  + JSP (Java Server Pages) 🡪 Verbindung von HTML, CSS, Java und JavaScript
  + Servlets
  + Beans
  + HTML 🡪 Layout
  + CSS 🡪 Design
  + Java 🡪 serverseitige Logik und Formularüberprüfungen auf Server-Ebene
  + JavaScript 🡪 clientseitige Logik und Formularüberprüfungen auf Client-Ebene

## Backend

Für den Endbenutzer nicht sichtbar sind der Webserver und die Datenbank.

### Webserver: Apache Tomcat

Stellt eine Umgebung zur Ausführung von Java-Code auf Webservern bereit, die im Rahmen des Jakarta-Projekts der Apache Software Foundation entwickelt wird. Es handelt sich um einen in Java geschriebenen Servlet-Container, der mit Hilfe des JSP-Compilers Jasper auch JavaServer Pages in Servlets übersetzen und ausführen kann. Dazu kommt ein kompletter HTTP-Server. Der HTTP-Server von Tomcat wird vor allem zur Entwicklung eingesetzt, während in Produktion zumeist ein Apache Web-Server vor Tomcat geschaltet wird. Dazu wird in Apache ein Plugin eingebunden, das Requests für dynamische Inhalte an Tomcat weiterleitet. Das Plugin spricht Tomcat dann normalerweise über das Apache JServ Protocol an.

### Datenbank: JDBC + JPA

**Java Persistence API** ist eine Sammlung von Klassen und Methoden zu beharrlich speichern die riesigen Datenmengen in eine Datenbank, die von der Oracle Corporation bereitgestellt wird.

**JDBC** ist die Implementierung des SQL/CLI (SQL Call-Level Interface) für die Programmiersprache Java. JDBC ist also eine Klassenbibliothek für den Zugriff auf SQL-Datenbanken in Java-Programmen

## Frontend

Das Frontend wurde hauptsächlich mithilfe von JSP in Verbindung mit HTML, Java, CSS und JavaScript realisiert.

### Java Server Pages (JSP)

Java Server Pages sind im Grunde HTML-Seiten, in denen Java-Code eingebettet ist.

In unserem Projekt haben wir Java-Code in HTML eingebettet, wenn es absolut notwendig war. Der größte Programmieraufwand in Verbindung mit HTML wurde bei uns mithilfe von Beans bzw. eigenen Java-Klassen realisiert.

Unter dem Punkt „Neues Projekt“ wurde Java Web 🡪 Web Application ausgewählt

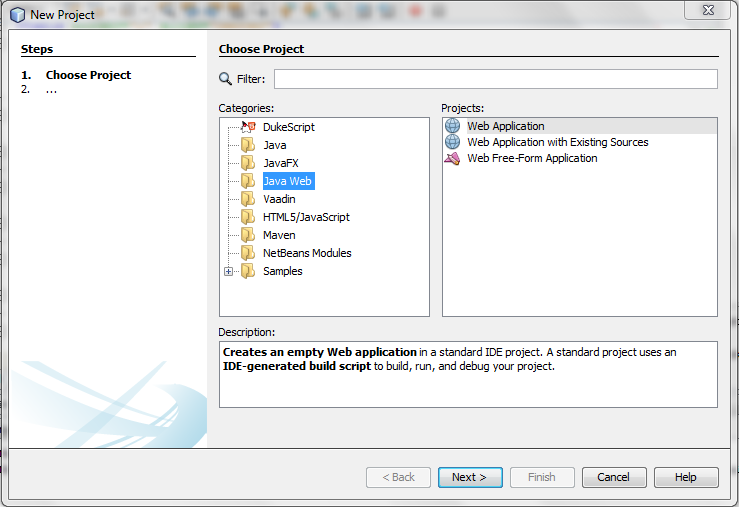


Abbildung 6: NetBeans: „Neues Projekt“ 🡪 Java Web Application = JSP

### Servlets

Wir haben ein Servlet zur Überprüfung der Logindaten erstellt. Im Ordner LoginPackage befinden sich die Dateien LoginServlet.java und PersonToCheckDAO.java (DAO 🡪 Data Access Object). In diesen Dateien wird die Eingabe des Loginformulars mithilfe einer Abfrage auf die dementsprechende Tabelle in der Datenbank überprüft. Bei erfolgreicher Anmeldung wird eine Session erzeugt und notwendige Session-Parameter zur Identifikation des angemeldeten Benutzers gesetzt.

### Beans

Beans wurden bei unserem Projekt unter anderem für die Ausgabe von HTML-Text benötigt.

Z. B. die Datei: Data2HTMLConverterBean.java 🡪 Verarbeitet sämtliche Daten, die aus unserer Datenbank abgefragt werden zu HTML.

### HTML

Mithilfe von HTML wurde bei unserem Projekt das Layout entwickelt. Dafür haben wir Tabellen, Listen und div-Container verwendet.

### CSS

Für das Design wurde eine CSS/JavaScript-OpenSource-Bibliothek heruntergeladen. Bootstrap war für unsere Zwecke als solche Bibliothek ausreichend.

Herunterzuladen unter: <http://getbootstrap.com/>

Überschreibungen bestimmter Styles wurde mithilfe unserer dashboard.css übernommen.

Für manche Styles wird eine Internetverbindung benötigt.

Z.B. results.jsp/87-89:

"language": {

"url": "//cdn.datatables.net/plug-ins/9dcbecd42ad/i18n/German.json"

}

### Java

Die serverseitige Logik und Formularüberprüfungen auf Server-Ebene wurden mithilfe von Java unter Verwendung von eigenen Java-Klassen, Beans und Servlets realisiert.

Nähere Beschreibungen der einzelnen Funktionen finden sich als Kommentare direkt bei den Funktionen.

### JavaScript

Mithilfe von JavaScript wurden sämtliche Veränderungen von Daten als Query-String in der URL an den Server übermittelt, um dort weiterverarbeitet zu werden.

Des Weiteren wurde mithilfe von JavaScript die Datentabellen filter- und sortierbar dargestellt. Dies wurde ermöglicht durch eine externe JavaScript-Bibliothek „DataTables“.

Weitere **Funktionen** von JavaScript in unserem Projekt:

* clientseitige Logik
* Formularüberprüfungen auf Client-Ebene
* Laufzeitdynamisches Setzen bestimmter CSS-Styles

## Verwendete Muster

Tabelle 2: Auflistung der verwendeten Entwurfsmuster

|  |  |  |
| --- | --- | --- |
| Muster | Art der Verwendung | Ort der Verwendung |
| Singleton | Datenbankverbindung | DBAccess.java |
| Prototype | Einfügen von Testdaten (STUDENT und LECTURER Clones) | DBAccess.java |

# Abbildungs-/Tabellenverzeichnis

[Abbildung 1: Projektstart - Datenbank 5](#_Toc472357670)

[Abbildung 2: Projektstart - Anmeldedaten 5](#_Toc472357671)

[Abbildung 3: Projektstart - Datenbankverbindung 5](#_Toc472357672)

[Abbildung 4: Projektstart - Konfiguration1 6](#_Toc472357673)

[Abbildung 5:Projektstart - Konfiguration2 6](#_Toc472357674)

[Abbildung 6: NetBeans: „Neues Projekt“ 🡪 Java Web Application = JSP 8](#_Toc472357675)

[Tabelle 1: Minimale Anforderungen für die Entwicklungs- und Testumgebung 4](#_Toc472358173)

[Tabelle 2: Auflistung der verwendeten Entwurfsmuster 9](#_Toc472358174)