# Practical Project Assignment for Java EE Fundamentals course

# Design and implement a Web-based JEE application, e.g. blog / forum / photo album / listings site / other.

## Technologies

The **technology stack** for your project development must include the following, e.g.

* Java, JSF 2x and Prime Faces, **EJB 3x,** JPA and Hibernate, MySQL, Wildfly, HTML, CSS

You are allowed to use in addition other **technologies** like Bootstrap, SASS, LESS. You are allowed to use **development tools**, **libraries** and **resources** like Web design templates, Java Maven artefacts, Code generators and others.

## Project Scope

Your project should implement **at least** the following **functionality**:

* User **registration**, **login** and **logout**.
* **View** some content (e.g. blog articles, listings, photos, issues, publications, NOT users).
* **Create/Edit** new content (e.g. post new blog article, post new listing, upload new photo, create new issue).
* **Optionally** implement more functionality.

Your project should keep its data in a **relational** **database**

* Use at least **2 tables** (**collections**) with a relationship, e.g. users and blog posts.
* Use a **database** (like MySQL).

Your project should implement **at least 4 pages** (views).

You are allowed even to take the project developed during the course and extend it for your needs.

## Forbidden Techniques and Tools

* Your project should be created by **you only**.
* You are **not allowed to copy a project from Internet** and present it as your development.
* You can use external libraries, frameworks and tools, but **not to clone a project** and present it as yours.

## Deliverables and Deadline

* The project should be submitted as homework on the “**Exam**” part of the course
* All projects should be submitted not later than **21 November 2016**.

## Public Project Defense

Each student will have to deliver a **public defense** of its work in front of the trainer.

The student will have **only ~15 minutes** for the following:

* **Demonstrate** the application’s functionality (very shortly).
* Show the **source code** and explain briefly how it works.
* If requested by the trainer, make a small change on the source code.

Hints for better presentation:

* Be **well prepared** for presenting maximum of your work for minimum time.
* **Open all project assets** beforehand to **save time**: open your site in the browser, login and open the user / admin panel in another browser, etc.

## Assessment Criteria

* **Functionality** – **0…70**
  + **What** is implemented? Does it work correctly? Does it have intuitive UI?
  + How much **effort** you have put in this project?
  + Is the functionality **enough** according to the project requirements?
  + What portion of the work is **own code written by you** and what is ready-to-use framework?
* **Project/Source quality** – **0…30**
  + **Is the source code well formatted**?
  + **Are code standards followed**?
  + **Is JEE project structure correct?**
  + **Is MVC standard followed?**
* **Bonus** – **0...10**
  + Bonus point are given for implementing more than expected.

## Sample Projects

The below described projects are **sample**, just to give you some **ideas**. You could work on your own project.

### Issue Tracker

Design and implement a simple **issue tracking system** (bug tracker).

**Required** functionalities:

* User **registration** (and optionally user profiles) / **login** / **logout**.
* **View** **all issues** (optionally with paging), without login.
* **View issue** details, without login.
* Create **new issue** (after login). Issues have title, description, author, state and submission date and time. States are: New, Open, Fixed and Closed.

**Optional** functionalities:

* **Edit issue** (after login). Can change only title, description and state.
* Add **new comment** for existing issue by visitors – each visitor must fill out his name and comment text.
* Implement a sidebar holding a **list of all issues states**. Clicking at issue state shows all issues matching this state.
* Functionality for **searching** by title (as substring).
* **Admin panel**: add / edit / delete issues and comments, etc.

### Blog

Design and implement a simple **blog system**.

**Required** functionalities:

* User **registration** (and optionally user profiles) / **login** / **logout**.
* **View all posts** (optionally with paging), without login.
* Create **new post** by the blog owner (after login). Optionally, each post may have **tags**.

**Optional** functionalities:

* Add **comments** for every post by visitors – each visitor must fill out his name, email (optionally) and comment text.
* Implement a sidebar holding a **list of posts** sorted by month / year / etc. and a list of the **most popular tags**.
* **Counter of visits** for each post.
* Functionality for **searching** by tags.
* **Admin panel**: add / edit / delete posts, comments, tags, etc.

### Forum

Design and implement a simple **forum** (discussion board) system.

**Required** functionalities:

* User **registration** (and optionally user profiles) / **login** / **logout**.
* **View** all questions (optionally by category, optionally with paging), without a login.
* Ask a **new question** by the forum users (after login). Optionally question may have **tags** and **category**.

**Optional** functionalities:

* Implement **tags** for the forum questions.
* Implement **categories** for the forum questions.
* Implement adding **answers** to the questions by the forum visitors – each visitor must fill out his name, email (optionally) and comment text.
* **Counter** for visits for each question.
* Functionality for **searching** by question, answer and tags.
* Implement **ranking** according to user activity.
* **Admin panel**: add /edit / delete forum posts, tags, answers, categories.

### Photo Gallery

Design and implement a simple **photo gallery** (photo album).

**Required** functionalities:

* User **registration** (and optionally user profiles) / **login** / **logout**.
* **Browse albums** and **photos** (and optionally categories, optionally with paging), without a login.
* **Upload photos** (after login, optionally validate image size and type) / **download** photos.

**Optional** functionalities:

* **Create new album** in a category.
* Add **comments** to photos and albums.
* Implement album's **ranking system** (e.g. vote from 1 to 10 or like / dislike).
* Show the most **highly ranked** albums in a special section at the main page.
* Functionality for **searching** by album name / category.
* **Admin panel**: add / edit / delete albums, photos, comments.

### Ads Listing Site

Design and implement a simple **ads listing site** where users publish their ads and visitors can browse and view them.

**Required** functionalities:

* User **registration** (and optionally user profiles) / **login** / **logout**.
* **View** all ads (optionally by category and town, optionally with paging), without a login.
* **Post a new ad** (optionally with a photo), after login.

**Optional** functionalities:

* Implement **towns** and **categories** for the ads. Implement browse by category / town.
* Post new ads in state “**Waiting approval**”. Administrators can see and approve them.
* Implement a **mini-photo gallery**: each ad could hold a set of **photos**. Users can upload / delete photos. Visitors can view the photos for each ad. One of the photos is designated as **primary**.
* **My ads panel**: view / edit / delete own ads (posted by the current user).
* **Admin panel**: add / edit / delete users, ads, categories, towns.