# 

# 

# Extension Repository

# Final Project Assignment

Contents

[General Project Requirements 3](#_Toc508906885)

[Development Requirements 5](#_Toc508906886)

[Optional Requirements 5](#_Toc508906887)

[Deliverables 6](#_Toc508906888)

[Public Project Defense 6](#_Toc508906889)

[Final Projects Examples From Previous Years 7](#_Toc508906890)

# **General Project Requirements**

**Java Final Project**

This document describes the **final project assignment** for the **Java** course at Telerik Academy.

**Project Description**

Your task is to design and develop a VSCode-like extension repository system. It should allow extension developer to manage their extensions. The clients of the system must be able to browse the extensions and download them. You may use VSCode Marketplace as a sample Extension Repository - https://marketplace.visualstudio.com.

**Public Part**

The **public part** of your projects **MUST** be **visible without authentication**.

* Application **MUST** have a public homepagе
* Home page **MUST** contain 3 lists of extensions: **Featured** (selected by the admis), most **Popular** (the most downloaded extensions) and **New** (the newest extensions)
* Application **MUST** display extension detail when an extension has been clicked.
* Extension detail page **MUST** display the following information
  + Name
  + Description
  + Version
  + Owner
  + Number of downloads
  + Tags
  + Download link
  + Link to the source repository – the source of the extension should be a github repository.
  + Number of open issues (retrieved from github)
  + Number of pull requests (retrieved from github)
  + Last commit date (retrieved from github)

ADD THE MISSING COLUMNS!

* Application **MUST** have user registration and logon capability.
* Application **MUST** allow listing extensions with filtering by name and sorting by the following fields
  + Name
  + Number of downloads
  + Upload date
  + Last commit date

**Private Part (Users only)**

**Registered users** **MUST** have private area in the web application accessible after **successful login**, where they could see all extensions that are owned by the currently logged user. Additionally the registered user must be able to:

* Delete/update/create extension. The extension must have the following data:
  + Name
  + Description
  + Tags
  + Binary content (the extension itself)
  + Origin – this is the location of the repo hosting the extension. The application must support github as a location.

Once extension is created the it is “pending” state until the administrator approves it. The extension is visible in the **Public** **Part** only if it is approved.

**Administration Part**

**System administrators** should have administrative access to the system and permissions to administer all major information objects in the system.

* Administrators **COULD** approve extension
* Administrators **COULD** delete/edit all extensions
* Administrators **COULD** disable user accounts

# 

# **Development Requirements**

Your Web application should use the following technologies, frameworks and development techniques:

* Use **Java** 1.8 or 1.9
* Use tiered architecture
* Use whatever UI technology suits you better
* Create simple UI with a fast/responsive backend
* Provide REST API for manipulating the data
* Use Hibernate for DB layer
* Unit Tests, ideally with at least 66% coverage

# **Optional Requirements**

* Implement optimistic locking
* Integration with continuous integration server
* Integration tests
* Source data synchronization
  + Periodically refresh source information. The “refresh” interval could be configurable.
  + Administrators could trigger repo source refresh for an extension
* Display information about extension synchronization in the Admin panel, including:
  + Last successful synchronization
  + Last failure
  + Failure details(the error that has occurred).

# **Deliverables**

Put the following in a **Github repo**:

* The **source code**
* The project documentation (**MUST** include build and run instructions, and name-check major components in the application)
* Screenshots of your application
* If hosted online - the URL of the application

# **Public Project Defense**

Each student must make a **public defense** of its work to the trainers, Partner and students (~30-40 minutes). It includes:

* Live **demonstration** of the developed web application (please prepare sample data)
* Explain application structure and its **source code**
* Show the **commit logs** in the source control repository to prove a contribution from all team members

Many projects in the enterprise suffer from degradation of design and increasing complexity over time, leading them to develop defects and gradually become unmanageable.

We value greatly teams who are able to do their job cleanly with a logical and maintainable design, without either unnecessary abstraction or ad hoc hacks.

You need to understand the system you have created. Any defects or incomplete functionality must be properly documented and secured. It’s OK if the proof of concept of your application has flaws or is missing one or two **MUST**’s. What’s not OK is if you don’t know what’s working and what isn’t and if you present an incomplete project as functional.

Some things you need to be able to explain during your project presentation:

* What are the most important things you’ve learned while working on this project?
* What are the worst “hacks” in the project, or where do you think it needs more work?
* What would you do differently if you were implementing the system again?