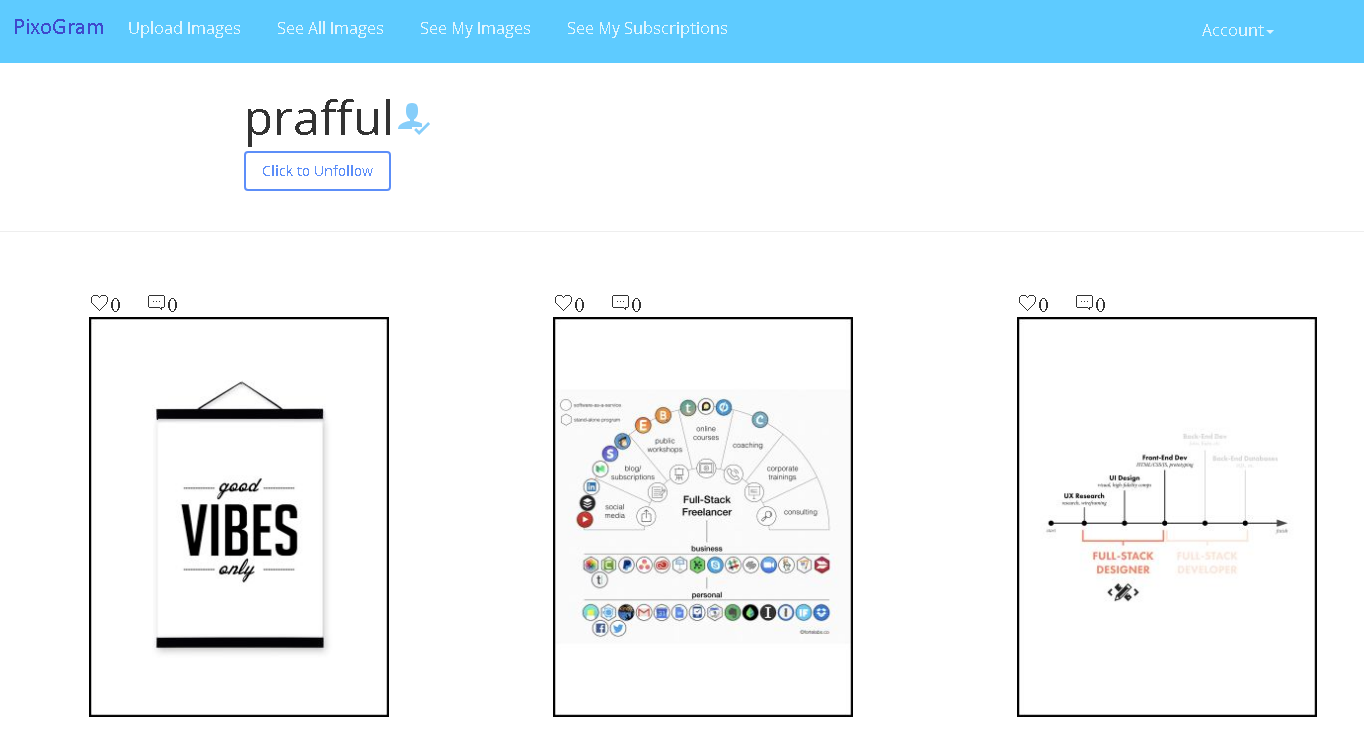
Maven, GIT, Tomcat & Junit Microlayer

Prafful Daga  
IIHT

Micro Credential: Maven, GIT, Tomcat & Junit Microlayer

Duration: 4 to 8 Hour



Maven, GIT, Tomcat & Junit Configuration FOR SOCIAL PICTURE SHARING APPLICATION

This page is intentionally left blank.

Contents

[1 Important Instructions 3](#_Toc529802586)

[2 Business-Requirement: 4](#_Toc529802587)

[2.1 Problem Statement: 4](#_Toc529802588)

[3 Methodology 5](#_Toc529802589)

[3.1 Agile 5](#_Toc529802590)

[3.2 Continuous Integration 5](#_Toc529802591)

[3.2.1 As you code the solution: 5](#_Toc529802592)

[3.2.2 On conclusion of application development (before going to production) 5](#_Toc529802593)

[4 Technical Specification – Maven, GIT, Tomcat & Junit Microlayer Development Environment 6](#_Toc529802594)

[4.1 Spring Boot Layer 6](#_Toc529802595)

[4.2 Editors 6](#_Toc529802596)

[5 Development Workflow 7](#_Toc529802597)

[6 Important Instructions 8](#_Toc529802598)

[7 Assessment Deliverables 8](#_Toc529802599)

[8 Other Full Stack Layers 9](#_Toc529802600)

[8.1 UI Layer (Not Applicable for Present Case Study) 9](#_Toc529802601)

[8.2 UX Layer (Not Applicable for Present Case Study) 9](#_Toc529802602)

[8.3 Back End Layer (Not Applicable for Present Case Study) 9](#_Toc529802603)

[8.4 Middle Tier Framework Layer (Not Applicable for Present Case Study) 9](#_Toc529802604)

[8.5 ORM & Integration Layer (Not Applicable for Present Case Study) 9](#_Toc529802605)

[8.6 Database Layer (Not Applicable for Present Case Study) 9](#_Toc529802606)

[8.7 Ancillary Layer(Applicable for Present Case Study) 9](#_Toc529802607)

[8.8 Deployment & Infra(Not Applicable for Present Case Study) 9](#_Toc529802608)

# Important Instructions

1. Follow the design specifications mentioned in the case study. You are free to improvise certain specifications mentioned in the case-study. But, for each such improvisation, you should keep the concerned POC informed. **POC will get in touch with concerned team at IIHT.**
2. You should stay **motivated** to initiate such and specific communications as it may have positive influence on the evaluation scores.
3. Please make sure that your code does not have any compilation errors while submitting your case study solution.
4. The final solution **should show the maximum code coverage of 95% plus.**
5. **Use h2 in-memory OR MySQL database where database functionality is required.**
6. Implement the code using best design standards for:
   1. Variable declarations
   2. Class names
   3. Package names
   4. Code Refactoring

# Business-Requirement:

## Problem Statement:

**The PixoGram (Spring Boot Web Application)** allows you to:

1. Register as a user
2. Login as a user
3. Retrieve/Change password
4. Manage your user account
5. Login/Logout to/from your account on PixoGram
6. Add Content
   1. Upload single/multiple pictures, caption and description
   2. Upload single/multiple videos, caption and description
7. Manage Content
   1. Organize Picture in Gallery
   2. Organize Videos in Playlists
   3. Rename Pictures and Videos
   4. Edit Caption, Description, Comment
8. Social Features
   1. Use emojis in comment
   2. Like or Unlike comment, pictures and videos of other users
   3. Follow/Unfollow other users
9. Edit Pictures
   1. Apply effects to pictures (sepia, grayscale, etc.)
10. Hide Pictures/Videos
11. Activity/Newsfeed
    1. View activity log of user-activity on the PixoGram
12. Offline Functionality:
    1. Certain parts of the application should be available in absence of connectivity.
    2. Relevant areas on the screen should display “Connectivity Not Available”

**In this micro layer you will write unit test for the application which you may have developed in either of the microlayer:**

* + - 1. **Spring Core and Spring MVC Microlayer**
      2. **Spring Boot & ORM Microlayer**

**It is recommended that you choose, Spring Boot & ORM solution if it is developed by you in one of the microlayer evaluations.**

**Please note that:**

**After writing the unit test, the code coverage report should be generated. The code coverage should show the maximum coverage of 95% plus.**

**You may do the code coverage with JaCoCo.**

[**https://www.eclemma.org/jacoco/**](https://www.eclemma.org/jacoco/)

**After maximum code coverage is done with unit test, deploy the app by signing-in as manager in Apache Tomcat.**

# Methodology

## Agile

1. Mentor will ask you about daily progress as you start implementing Junit Layer.
2. Communicate with your mentor via email; ideally daily as you develop the Spring framework Layer.
3. Scope of discussion with your mentor:
   1. Q/A
   2. New Ideas and New feature implementations
   3. Any development related challenges
   4. Skill Gaps
   5. Another pointers key to Spring Development

## Continuous Integration

### As you code the solution:

1. Check in your code - ideally every hour in the internal GIT repository at 172.18.2.18

### On conclusion of application development (before going to production)

1. Set up Jenkins on the cloud and integrate with internal GIT repository.
2. Setup Jenkins to automatically run the unit test on each successful build.
3. Save the Jenkins/build report and code coverage report from JaCoCo as a part of final assessment deliverable.

# Technical Specification – Maven, GIT, Tomcat & Junit Microlayer Development Environment

## Spring Boot Layer

|  |  |
| --- | --- |
| **Framework(s)/SDK/Libraries** | **Version** |
| Spring Boot, Hibernate, Spring Data | - |
| Maven | - |
| Junit | - |
| GIT Basics | - |
| Jenkins Basics | - |
| Docker Basics | - |

## Editors

|  |  |
| --- | --- |
| **Name** | **Version** |
| STS | - |

# Development Workflow

1. You must follow following process while creating Unit test cases
   1. Feel free to take approval at each stage:
      1. Approval includes:
         1. Communicating with mentor via email to showcase the progress.
         2. Progress must be shared with mentor as each JUnit layer is crossed.
         3. It is recommended to avoid moving to next stage until feedback from the mentor is received.
         4. Pushing assets/code to GIT repository.

# Important Instructions

1. Follow the design specifications mentioned in the case study. You are free to improvise certain specifications mentioned in the case-study. But, for each such improvisation, you should keep the concerned POC informed. **POC will get in touch with concerned team at IIHT.**
2. You should stay **motivated** to initiate such and specific communications as it may have positive influence on the evaluation scores.
3. Please make sure that your code does not have any compilation errors while submitting your case study solution.
4. The final solution **should show the maximum code coverage of 95% plus.**
5. **Use h2 in-memory OR MySQL database where database functionality is required.**
6. Implement the code using best design standards for:
   1. Variable declarations
   2. Class names
   3. Package names
   4. Code Refactoring

# Assessment Deliverables

1. Dump of command “git log”
2. Jenkins report about number of times it pulled the code from GIT
3. Working POM.xml for the project.
4. Jenkins/build report and code coverage report from JaCoCo

# Other Full Stack Layers

## UI Layer (Not Applicable for Present Case Study)

|  |  |
| --- | --- |
| HTML5 | - |
| CSS3 | - |
| Bootstrap/Material | - |
| Typescript | - |

## UX Layer (Not Applicable for Present Case Study)

|  |  |
| --- | --- |
| Angular | 6 |
| Javascript & JQuery | - |
| Typescript | - |

## Back End Layer (Not Applicable for Present Case Study)

|  |  |
| --- | --- |
| JDK | 1.8 or above |

## Middle Tier Framework Layer (Not Applicable for Present Case Study)

|  |  |  |
| --- | --- | --- |
| **Technology** | **Framework(s)/SDK/Libraries** | **Version** |
| Spring Framework | Spring MVC | 5.0 or above |

## ORM & Integration Layer (Not Applicable for Present Case Study)

|  |  |  |
| --- | --- | --- |
| **Technology** | **Framework(s)/SDK/Libraries** | **Version** |
| Spring | Spring Boot | 2.0 |
| Java JPA | Hibernate | 5.0 or above |
| Spring Data | 2.0 |

## Database Layer (Not Applicable for Present Case Study)

|  |  |  |
| --- | --- | --- |
| MySQL | MySQL | 7.x + |

## Ancillary Layer(Applicable for Present Case Study)

|  |  |  |
| --- | --- | --- |
| **Technology** | **Framework(s)/SDK/Libraries** | **Version** |
| Source Code Management Tool | GIT | 2.18 |
| Build Tool/JAVA Stack | Maven | 3.5.x |
| Testing Tool/JAVA Stack | Junit/Spring Test | 4.x/5.x |
| Javascript Dependency Management Tool | NPM | 6.x.x |

## Deployment & Infra(Not Applicable for Present Case Study)

|  |  |  |
| --- | --- | --- |
| **Technology** | **Framework(s)/SDK/Libraries** | **Version** |
| Docker | - | 17.06.2 |
| Apache Tomcat | - | 9.0 |
| Jenkins | - | 2.121.2 |