Description:

This task is intended to provide impression on overall coding style and capabilities. Please note that only working code is not sufficient enough, modular structure, overall coding style, as well as test coverage will be taken in account during assessment. Delivered work should include only maven project descriptors and java source files, no IDE related files or compiled class files\artifacts should be delivered. Additional frameworks as well as specific build or deploy instructions should be described in README.md file, located near root .pom descriptor.

Please focus on main objectives first and work on optional sections only after main objectives are implemented. Once main objectives are finished, keep working on optional sections if you have time.

Contact for questions: askar.akhmerov@smava.de
Expected duration: 6-10 hours
Recommended technologies:
- maven
- spring
- spring MVC
- junit
- tomcat
- jQuery (any other js library for basic manipulations)

Optional technologies:
- hibernate
- hsql
- Jersey (JAX-RS)
- jaxB + Jaxon
- ActiveMQ

Main objectives

1. Create multi module maven project
2. Create jsp page that displays list of bank accounts, using jstl, provided by backend
   A bank account consists of two fields: **iban (String) and Business Identifier Code (String)**
3. Create rest endpoint to store changes to the bank account list from user
4. Implement logic to store user data
5. Create js based submission of bank account list edited by user towards backend

Optional section 1

1. Implement storage of user data in embedded hsql database using jpa

Optional section 2

1. Add wadl description to the rest endpoint
2. Add profile to generate client based on wadl
3. Add jms producer to flush persistent bank account lists to the consumers
4. Add module with jms consumer that prints messages from queue to the log file

User stories

1. User opens web page and sees default bank account list + values from session storage. Bank account list supports basic CRUD operations which are persisted to session storage through rest endpoint. Page refresh should not create session for user and consequently edited list should be fully displayed. Application is packaged as war file and deployed to standalone application server or web container manually.

2. Application is fully launched from command line during maven build lifecycle, values provided by user are stored in real persistent storage instead of session. User is still identified by session.

3. REST api utilized simulating integration with 3rd party partner system. Simulated integration with internal sub system using jms.