Project #2 Due: 8th December @ 23:59

Reverse auctions are becoming increasingly popular online.

Registered users publish jobs they need done, e.g. fit an Ikea kitchen or install a new bathroom suite, and businesses will bid for those jobs. The lowest bid wins the job.

You are to develop a website with the following features:

* Users can register to for the service using an email, a password and personal details i.e. name and phone number. An address is not needed - it would be communicated to the interested bidders directly by the customer.
* Authenticated users can
  + add a job, giving it a name, and description.  
    The system should store the date on which the job was first published.   
    Suggestion: use LocalDate which has many useful time/date related [methods](https://docs.oracle.com/javase/8/docs/api/java/time/LocalDate.html)including LocalDate.now() for today's date.
  + bid for a job, if the user does not own the job
    - Bids cannot be made to Inactive (closed) jobs.
    - A bid must be lower than the current lowest bid.
  + A job is closed after 20 days.
    - This can be achieved using a [scheduler](https://cit.instructure.com/courses/28509/pages/scheduling-tasks). Every day the system change the status of jobs more than 20 days old to closed.
  + Any user can view jobs to see all the details of each job including the list of bids
  + If a job is closed, the user can view the job's details including who won that job and the winning bid for the job.

You must also provide

* Two REST API endpoints to return the following data in json format to authorised users
  + List all currently active jobs
  + List the bids made by a user, given an id
* An example of consuming these two APIs (requires another Spring project) – nothing fancy is needed here, just another project which has two controllers that send authentication data to the REST APIs and present retrieved data as, for example, a list – no CSS is needed. Do not waste time on making it look good – the focus is on functionality.

**Technical Notes**

Develop a **WebMVC** Spring Boot application.

Use an **in-memory h2** database to create an "out of the box" application. This database must be populated with sample data.

The application must be implemented using be implemented using **JPA**, making use of all that offers. This project essentially has 3 entities: registered user, job and bid; a user can have many jobs, and a job has many bids.

Use the **Security** module for authentication and authorisation. This will require another entity i.e. Role.

I am not interested in visual styling – just make the web site well structured, readable, navigable, etc. Credit any templates (CSS/HTML) if you use them.

Forms must be validated using form binding and suitable error messages should be displayed by the view if the user makes a mistake.

It must be possible to **change** **the** **language** of the website. User input must be validated with suitable (international) error messages. You do not have to translate content. I will accept subtle changes e.g. "Balance" becoming "Balance\_FR" to indicate the language has changed to French.

You **must** use the following:

* Spring Boot
* Spring MVC
* Thymeleaf (not JSP)
* Spring Data JPA
* Spring Security
* H2 for an embedded database
* Maven

Unit tests are not required.

You may use **Project Lombok** if you wish but it is not a requirement.

Provide a **brief document** outlining the high-level design of your system (1 or 2 A4 pages) including but not limited to your database design and class diagrams and the beans which you used.

**Marking**

Consult the rubric for a breakdown of the marks and guidance on what is expected in each category.

**Submission**

You may work in pairs or alone. Either way please sign up for a group even if it is a group consisting only of you. If you work in pairs, you will both get the same mark.

Submit as a 7z or zip file including

* A **brief document** outlining the high-level design of your system (2/3 A4 pages) including but not limited to your database design along with a list of issues (if you have any) e.g. functionality not implemented, limitations in code etc.
* Your **Spring project**as a .7z or .zip file.