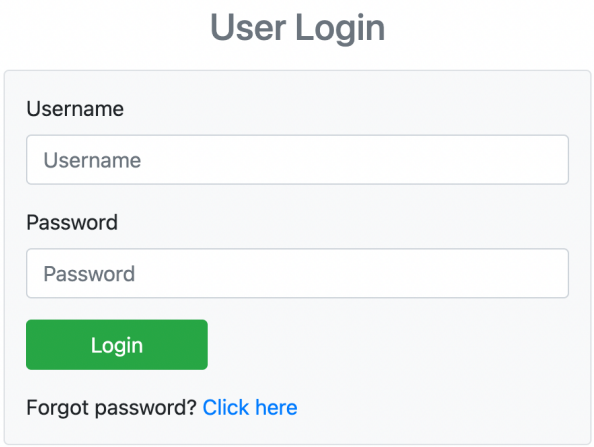
JAVA TECHNOLOGY

(503111)

LAB 4

EXERCISE 1

Given the login.html file containing a html login form as below.



- Create a java servlet project, using the given login.html file as the homepage (index.jsp).
- Create a servlet and name it LoginServlert. Modify the index.jsp file so that the html login form will post to the LoginServlet servlet when the user submits the form.
- The LoginServlet needs to provide two request handler methods, one for the get method and the other for the post method. When the user submits the login form, the information will be passed to the servlet using the post method. When the user accesses the servlet using the get method, the user will be redirected back to the index.jsp page.
- When submitting the form, the login information will be validated. The LoginServlet receives the username/password from the client and then checks it in a HashMap containing the list of pre-built accounts inside the servlet's constructor or init() method.

If the account exists, the servlet will output "Name/Password match", otherwise the servlet will output "Name/Password does not match". At that point the browser url remains in place of the LoginServlet.

- Need to make sure that the output from the servlet must have the format (content-type header) as html, if the content-type is in another format, correct it.

EXERCISE 2

Methods in a servlet like doGet(), doPost() are typically set up to return textual data like text, html or json to the client side. In fact, they can also return to the client binary data in many different content types such as: image, video, document or even raw data (application/octet-stream). In the java web servlet, these binary data files can be placed in the resources folder if the project is organized by maven tools. Watch the attached video demonstration to understand the requirements.

Create a java servlet project containing 3 servlets named respectively: ImageServlet1, ImageServlet2 and DownloadServlet. These servlets only response to get requests from the doGet() method. Fulfill the following requirements:

1. By default the servlets will have the same url pattern as their name, please change their url to: */image1*, */image2* and */download* respectively. For example *http://localhost:8080/download* will be matched with the servlet named DownloadServlet.
2. When accessing the url /image1, the server will response an image to the client and the browser **must display** this image directly at a browser tab (you can choose an arbitrary image).
3. When accessing the url /image2, the server will also response an image to the client but it **must** force the client to **download** the image instead of display it at a browser tab. The downloaded file name must be the same as the file name being saved on the server side.
4. When accessing /download, if the query string is not passed, the server will output the message "File not found". To download a specific file on the server side, you need to use a url of the form */download?file=data.zip*. Then the data.zip file (if available at the server) will be

downloaded in a way similar to how /image2 works. For example, if the user accesses http://localhost:8080/download?file=music.mp3, the music.mp3 file will be downloaded to the user's computer. If the user accesses a valid url but the file is not available on the server, the server should output an appropriate error message.

5. Advanced request: add file download speed limit feature by adding the speed query parameter to get maximum allowed speed. For example, when a user accesses the following url */download?file=movie.mp4&speed=500*, the movie.mp4 file will be downloaded at a maximum speed of 500 kb/s.

EXERCISE 3

An exercise on restful web services. In this exercise students need to create an api endpoint (represented by a servlet) to provide basic CRUD functionality on the product object. This exercise does not require the use of jsp and database, product data only needs to be stored in a static List<Product> initialized in the constructor or init() method of the servlet.

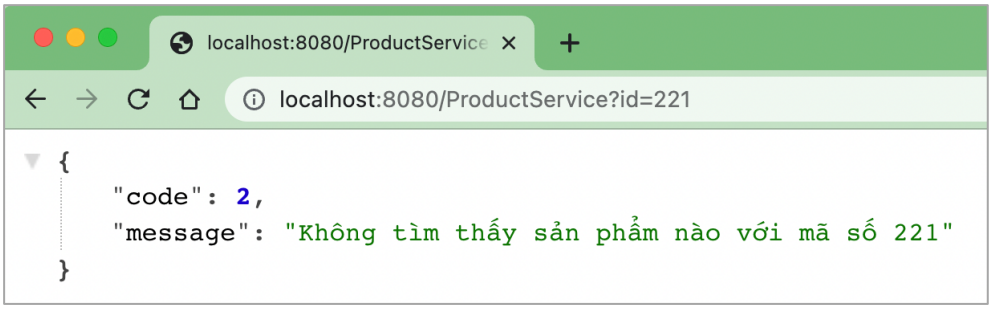
Create a servlet and name it ProductServlet, this servlet provides basic operations on the product object such as: read product information; add, remove and update products. The specific requirements are as follows:

- Create class Product containing product information including: id, name, price.
- Create ProductServlet and implement methods: doGet(), doPost(), doPut(), doDelete() to handle requests sent from client by get, post, put and delete methods.
- The results returned from the above methods need to be in json format, supporting Vietnamese. The returned result consists of 3 fields: id represents the error code, message represents the message and data (if any) represents the data returned from the server. See example image below for better understanding.
- The doGet() method returns a list of all products:
  - When the request parameter is not used, all products are returned
  - When using the request parameter id, only information about the product corresponding to that id is returned. If the id does not exist, return the appropriate error message.
- The doPost() method handles adding a new product:

- When adding a new product, the user needs to provide: id, name and price
- id and price must be numeric values, id cannot be the same as an existing product

- The doPut() method handles updating a product's information
  - When updating a product, the user needs to pass: id, name and price
  - If the id does not exist on the server, the update will fail
- The doDelete() method handles updating a product's information
  - When deleting a product the user only needs to pass the product id
  - If the id does not exist, the product deletion will fail
- For each of the above methods, a valid input check mechanism is required. In case the user provides a missing or incorrect format of the input information, the corresponding error message should be returned.

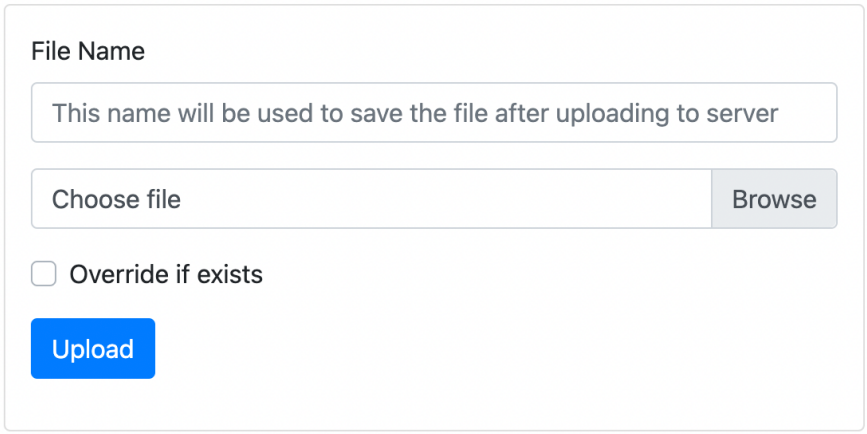
An example of output when sending a GET request to /ProductService



An example of how to return an error message (json) when reading a product that does not exist

EXERCISE 4

Given file upload.html contains the html user interface for uploading a file, in which the textbox contains the name that will be used to save the file after it is uploaded to the server.



- Create a servlet UploadServlet to handle the upload function for the above html form.
- Uploaded files need to be saved in the uploads folder, which is located inside the project's root folder. After uploading the file successfully, the message "File has been uploaded" will be displayed.
- The website only accepts files with the following extensions: txt, doc, docx, img, pdf, rar, zip. If the user uploads a file with an unsupported format, the message "Unsupported file extension" will be displayed.
- When uploading a file, it is necessary to check if a file with that name already exists on the system. If the file already exists but the "override if exists" check box is not selected, the message "File already exists" is displayed. If the check box is selected, it will still save the file and display the message "File has been overriden".

- If the file upload is successful, the server needs to return an html that says "File uploaded. Click **here** to visite file". Where the word "here" contains a hyperlink pointing to the link to download the uploaded file.

EXERCISE 5

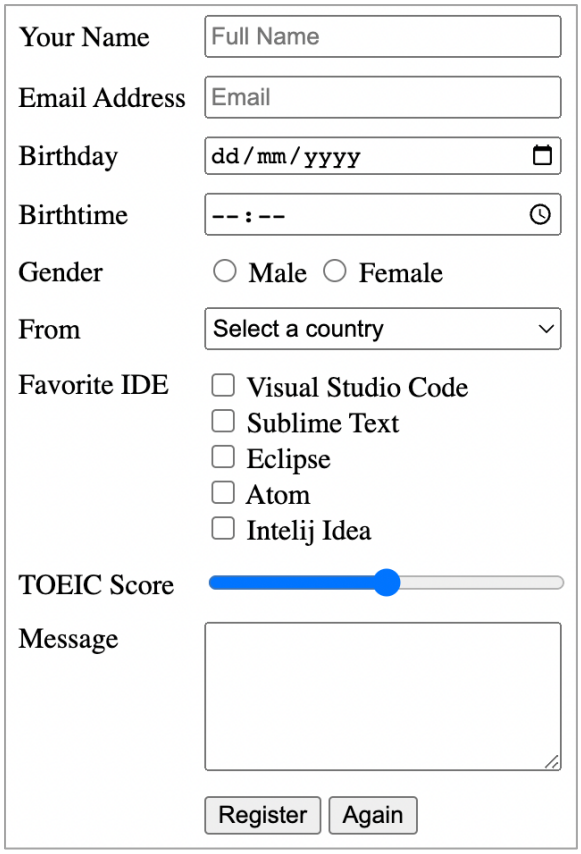
This exercise requires students to organize a web application using the mvc model in which a servlet (HomeServlet) acts as a controller (router) in charge of directing requests to one of three views including about.jsp, contact. jsp and help.jsp. Watch the attached video to understand the requirements of the exercise.

HomeServlet will rely on the query string to determine which view content to select and display accordingly. For example, when a user accesses the url http://localhost:8080/?page=**contact**, the HomeServlet will receive the query string page=contact, then it will be analyzed and determined to select the contact.jsp file. The tasks to be done in this exercise can be summarized as follows:

- Create the HomeServlet act as controller (router)
- Create the files about.jsp, contact.jsp and help.jsp contain arbitrary html UI .
- HomeServlet accepts request, read and parse the query string to determine the page the user wants to visit (home, help, or contact) then use the RequestDispatcher feature to direct the user to the requested page.
- If done correctly, when the jsp pages are displayed, the browser url shows one of the following three cases: /?page=contact, /?page=help or /?page=about
- In case the user does not pass the query string or passes an invalid query string, the HomeServlet needs to output "Welcome to our website" directly at the servlet.

EXERCISE 6

Similar to exercise 1, given register.html file contains the interface of an html form that collects user information. Create a servlet named RegisterServlet, receive information from the html form and display the information that the user has selected to the screen. In case the user does not enter information or enters missing information, the appropriate message should be displayed.



The given html form

Họ tên	Mai Van Manh
Email	mvmanh@gmail.com
Ngày sinh	2020-12-12
Giờ sinh	11:45
Giới tính	male
Quốc gia	Vietnam
IDE Yêu thích Hãy đảm bảo rằng các IDE bạn chọn đã hiển thị đủ	Visual Studio Code Sublime Text Eclipse Atom Intelij Idea
Điểm TOEIC	990
Giới thiệu bản thân	Hello

Example of output displayed by Register Servlet