code.sprint MT

TASK BOOKLET

Open Category

2023

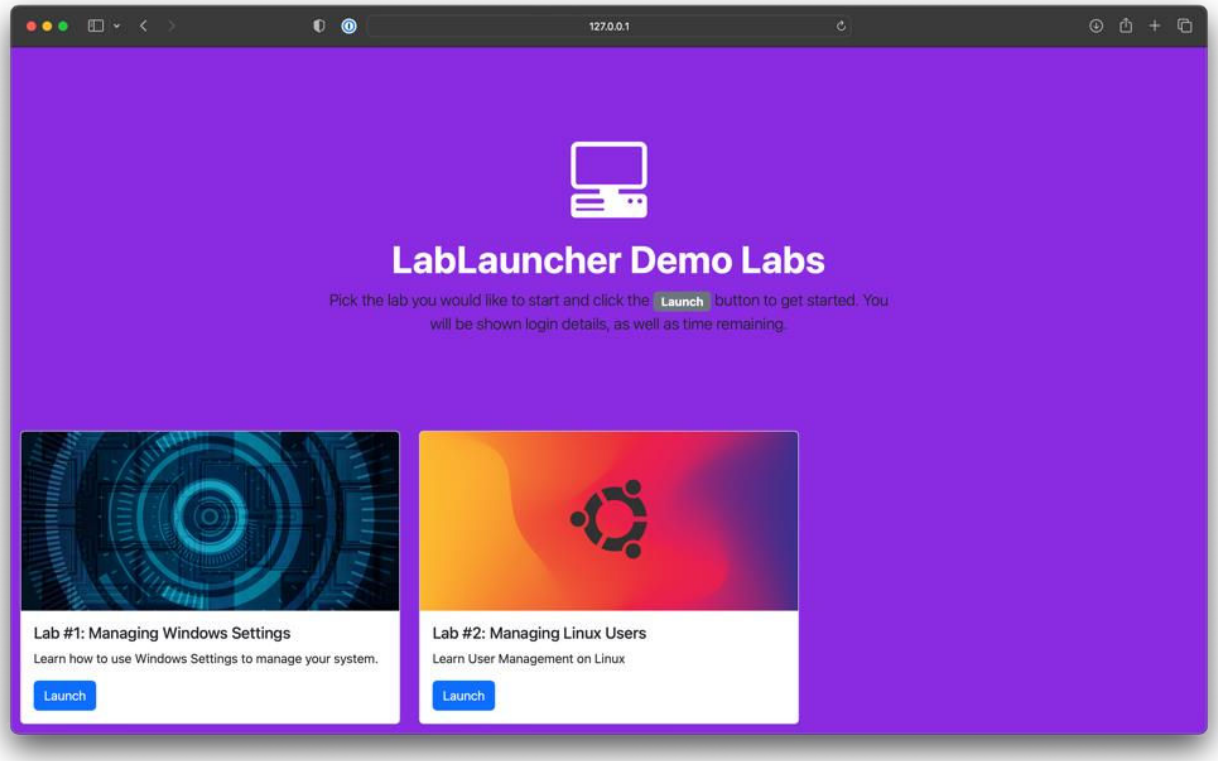
ICE Malta

GOVERNMENT OF MALTA
MINISTRY FOR EDUCATION, SPORT, YOUTH, RESEARCH AND INNOVATION
DIRECTORATE FOR LEARNING AND ASSESSMENT PROGRAMMES

LabLauncher

Hello!

Welcome to the CODESPRINT 2023 competition. Today you'll be flexing your coding, infrastructure and UI/UX design muscles to create a practical lab learning environment.



1. Design Brief

In recent times, it has become more and more common for learning to happen remotely. Using tools such as videoconferencing applications, remote whiteboards and polling systems, educators strive to create an online experience which recreates the interactivity of an in-person classroom. One area where more progress is needed, however, is the implementation of virtual lab environments, particularly for students to be able to use software or practice skills without using their own machine.

The Ministry for Education and Employment wishes to develop an application to distribute freely in all local schools. The application should allow educators to add details about virtual lab environments they have created, which students can then use to launch virtual labs.

At this stage, the Ministry would like a proof-of-concept to be developed. They want to see what the system will look like from a user perspective. A user should be able to login to the system, view available labs, and launch a lab. They will then be given connection details (via RDP or VNC) to connect to their lab environment, which is provisioned for them after they launch the lab. For this proof-of-concept, no back-end UI is required – meaning details available lab details can be hardcoded or directly retrieved from a database or file.

The Ministry has developed a table of the core functionality required in the app, which is shown below.

Functionality		Notes
1	**Ability to view available labs**	After opening or logging into the labs system, the students will be able to view which labs are available. Certain labs may be disabled (as the educator oversees when labs should be made available to students). Disabled labs should still be shown but should be marked as such and the student should not be able to launch these labs.
2	**Ability to Launch a Lab**	After selecting a lab, a student should be able to launch it. When a lab is launched: • The virtual machine is provisioned for this student. This can be running in the cloud (ex: AWS, Azure etc.), or any server. • The system must monitor the status of the machine and identify when this has launched. • The system should have a delay to ensure the machine is ready to be connected to before giving the student access details. • The host/IP address, username and password to connect to the machine should be displayed. It is critical that the remote system is only created once the student has launched a lab, to avoid unnecessary cost.
3	**View Lab Guide**	Each lab has an associated lab guide. After launching a lab, the student should be able to view this guide to follow instructions or complete tasks.
4	**Generate Connection File**	Although the student will be given connection details once their lab has been provisioned, it would be easier if a connection file is generated, which the student can download. This would allow the student to open the file, optionally be prompted for a password, and then automatically login to the remote system. The system should be able to generate connection files for both RDP and VNC.
5	**End Lab**	Each lab is timed, and the educator decides how much time students will be given to complete the lab. The student should be shown a countdown timer showing how much time they have left. Once the countdown timer expires, the remote machine should automatically be terminated. Additionally, the student should be able to end their lab at any point. This can help save costs, as the student will be instructed in lab guides to end their lab once they are done.

2. Technical Guidelines

The method you choose to implement the app is up to you. However, the following technical guidelines are intended to help ensure you stay on the right track.

2.1 Reference Implementation

The judging panel has created a reference implementation of this app in one working day. This is to ensure that the task given is possible within the timeframe allocated. A video of this app in operation is available below. We highly recommend that you watch this video carefully, to get an idea of the functionality and level of polish the judging panel is expecting.

https://icepublicvids.s3.eu-south-1.amazonaws.com/lablauncher.mp4

2.2 Platform

Your app can run on any platform you choose (Windows, macOS, Linux, iOS, Android, Web). We suggest creating a web application to reduce the barrier of entry for students, however this is ultimately up to you.

2.3 Development Environment

You are free to use **ANY** programming language you wish to create your solution. However, do remember that the solution must run on the judge's computers, and that you must provide both a binary/executable solution, as well as source code.

2.4 Tips

Here are a few tips and resources to consider.

- Due to time constraints, you may want to seriously consider building a simple web application for this proof-of-concept, rather than a mobile or desktop application.
- As you will be provisioning resources programmatically, it may be a good idea to use a cloud environment such as AWS or Azure to simplify this aspect of the system.

2.5 Resources

- RDP File Format: https://www.donkz.nl/overview-rdp-file-settings/
- VNC File Format: https://www.reddit.com/r/sysadmin/comments/1qh9fk/batch_creation_of_vnc_files

Here are a few resources to consider should you go with the cloud route:

AWS

- Launch Templates: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/create-launch-template.html https://docs.aws.amazon.com/autoscaling/ec2/userguide/launch-templates.html
- EC2 Instance: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html
- API Gateway https://docs.aws.amazon.com/apigateway/latest/developerguide/getting-started.html
- Amazon Machine Images (AMIs) https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html

Azure

- ARM Templates: https://learn.microsoft.com/en-us/azure/azure-resource-manager/templates/overview
- Azure Virtual Machines: https://learn.microsoft.com/en-gb/training/modules/create-windows-virtual-machine-in-azure/
- Azure API Management: https://learn.microsoft.com/en-GB/shows/Hello-World/Hello-World-API-Management#time=04m05s?culture=en-gb&country=gb
- VM Images: https://learn.microsoft.com/en-us/azure/virtual-machines/capture-image-portal

2.6 Name

LabLauncher is a sample name – you are free to call your app whatever you want 😀

3. Judgement Criteria

Your submission will be given a maximum of 185 points. The criteria by which points are awarded are detailed below. **Note that you do not need to achieve all the criteria**, however, the more criteria you achieve, the greater your chances of winning! The numbers in **[brackets]** refer to the functionality in the design brief.

Criterion	Notes	Maximum Points
Core Functionality		
[1] Ability to view available labs	Will be awarded based on the ability to view labs made available to the student, and disabled labs being shown as such.	**10**
[2] Ability to launch a lab	The system should provision the lab environment when launched by the student. The system should inform the student when the lab is launching and is ready to be used. The system should provide the user with connection details.	**30**
[3] Ability to View the Lab Guide	The student should be able to view a lab guide once they have launched a lab.	**10**
[4] Generate connection file	The system should be able to generate an RDP or VNC connection file once the remote machine is ready.	**20**
[5] End Lab	The system should display a timer showing the student how much time they have left. The lab should be ended when the time expires. The student should be able to end the lab at any point.	**20**
UI/UX		
Neat/Aesthetically pleasant user interface	Rather than 'flair', we are looking for a neat, organized and functional UI	**10**
App is easy to use	The user should not need a manual to use the app	**5**
Experience	The User Experience is up to you, however, make sure your app: • Gives the user feedback as to the status of their lab provisioning. • Allows them to view the lab guide without losing access to the connection details. • Ensures the remote machine is ready to go before giving access details.	**15**
Code Quality		
Code is organized into packages/modules/units etc.		**5**
Separation between presentation and logic layers	For example, using a REST API model	**10**
Consistent and correct use of a programming paradigm	Such as OOP, AOP, functional etc.	**5**
Function cohesion	Functions should be kept small, and do one thing, without being too dependent on other functions	**5**
Inline documentation	i.e. comments	**5**
Maintainable code	Ex: use of abstract classes, interfaces, function prototypes etc. Depending on the programming paradigm chosen	**5**

Additional Functionality/Features		
Sample data is persistently stored	Labs available should be stored in a file or database rather than being hardcoded.	**5**
Copy to Clipboard	Students should be able to copy usernames, passwords, and host details by clicking on a 'copy to clipboard' button.	**10**
Additional features	Additional features over and above the design brief will be graded, up to a maximum of 15 points	**15**

Submission Criteria

At the end of the time allocated to this competition, you must submit your code to the judging panel. The code, including all assets and other resources, must be submitted as a folder or compressed archive.

You will also be required to demonstrate your application running.

NOTES


eSkills
eSkills Malta Foundation


LeoVegas


me direct