Programmers Test - Cross Platform Application Launcher

OpEzee Private Limited,
Sua House, 26/1 Kasturba Cross Road
Bangalore 560001
India.

Welcome to the full stack programmer test!

This test is in 4 parts. You will need to complete Part 1, Part 2 and Part 3 in order before moving onto Part 4.

Part 1,2,3 are mandatory but part 4 is optional.

But please note that a good submission will give you the best opportunity to stand out from the rest of the applicants and we would encourage you to attempt part 4 as well.

This test will require you to have access to a Windows PC and different development environments setup to compile and execute the code.

As a rough guide, Part 1, Part 2 and Part 3 shouldn't take more than 4 days. The length of time spent on Part 4 will depend on your choice changes.

You should spend no longer than 5 working days in total on this test. Please make a note of the time you spend on each part of the test and include these times when sending in your submission – we will take into account the amount of time you have been able to spend on the test when evaluating your submission.

Best of luck.

Part 1 – Server setup

Create a windows application/service/server that:

1. Runs in background
2. Hosts a control UI, that can be opened by a browser on any Windows/Mac/Android/iPhone device on the network, though a link (ex. http://192.168.0.147:2354/Launcher). More on this in part 2.
3. Handles any required data storing and loading from the UI. Ex. List of applications and parameters.
4. Can launch any application present in the local system with given parameters.

Ex. 1

Input:

Application: "C:\Program Files\Google\Chrome\Application\chrome.exe"

Parameter: google.com

Command for testing: "C:\Program Files\Google\Chrome\Application\chrome.exe" google.com

Output:

Should open chrome window and automatically load google.com

Ex. 2

Input:

Application: "C:\Program Files\Google\Chrome\Application\chrome.exe"

Parameter: youtube.com

Command for testing: "C:\Program Files\Google\Chrome\Application\chrome.exe" youtube.com

Output:

Should open chrome window and automatically load youtube.com

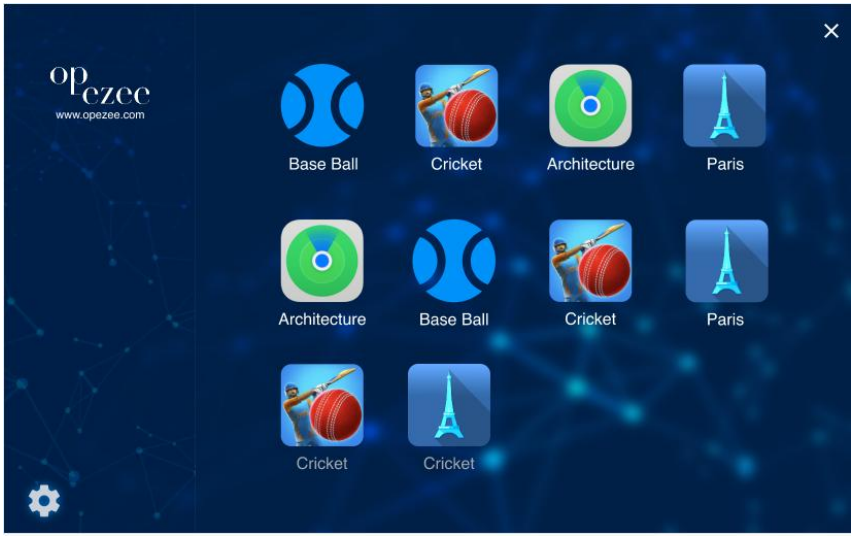
Part 2 – UI/UX

Create a browser based UI that can be opened by any Windows/Mac/Android/iPhone device on the network, though a link (ex. http://192.168.0.147:2354/Launcher).

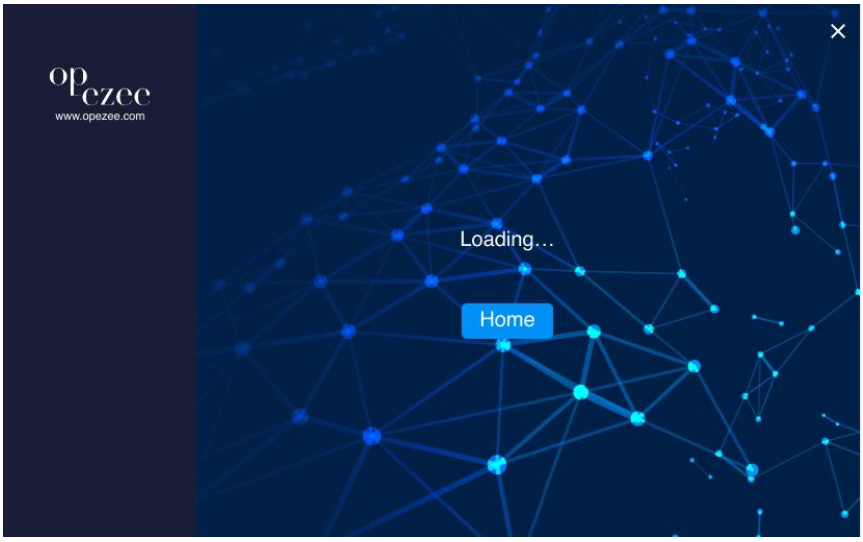
The UI should be connected to a specific server.

When opened on a client (ex. **on Android**) device:

1. Get list of all apps from server.
2. Load icons for all applications from the list.
3. Generate the UI as shown in the example. (Home Screen)



4. Once you tap the icon, it should launch the application that the icon represents on server device (**on Windows**).
5. Once the application is launched, show loading screen on the UI



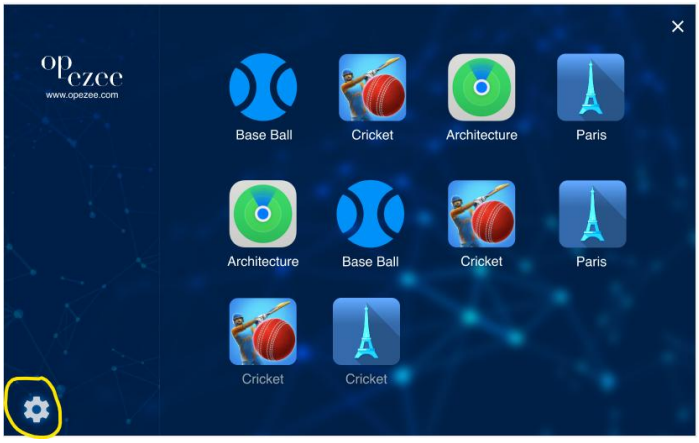
6. If home is pressed quit the application launched in step 4 (**on Windows**) and go back to the Home Screen (**on Android**) (step 3) of the launcher.

For functionality/ UX expectations, you can refer to stock Android/ iPhone launcher apps.

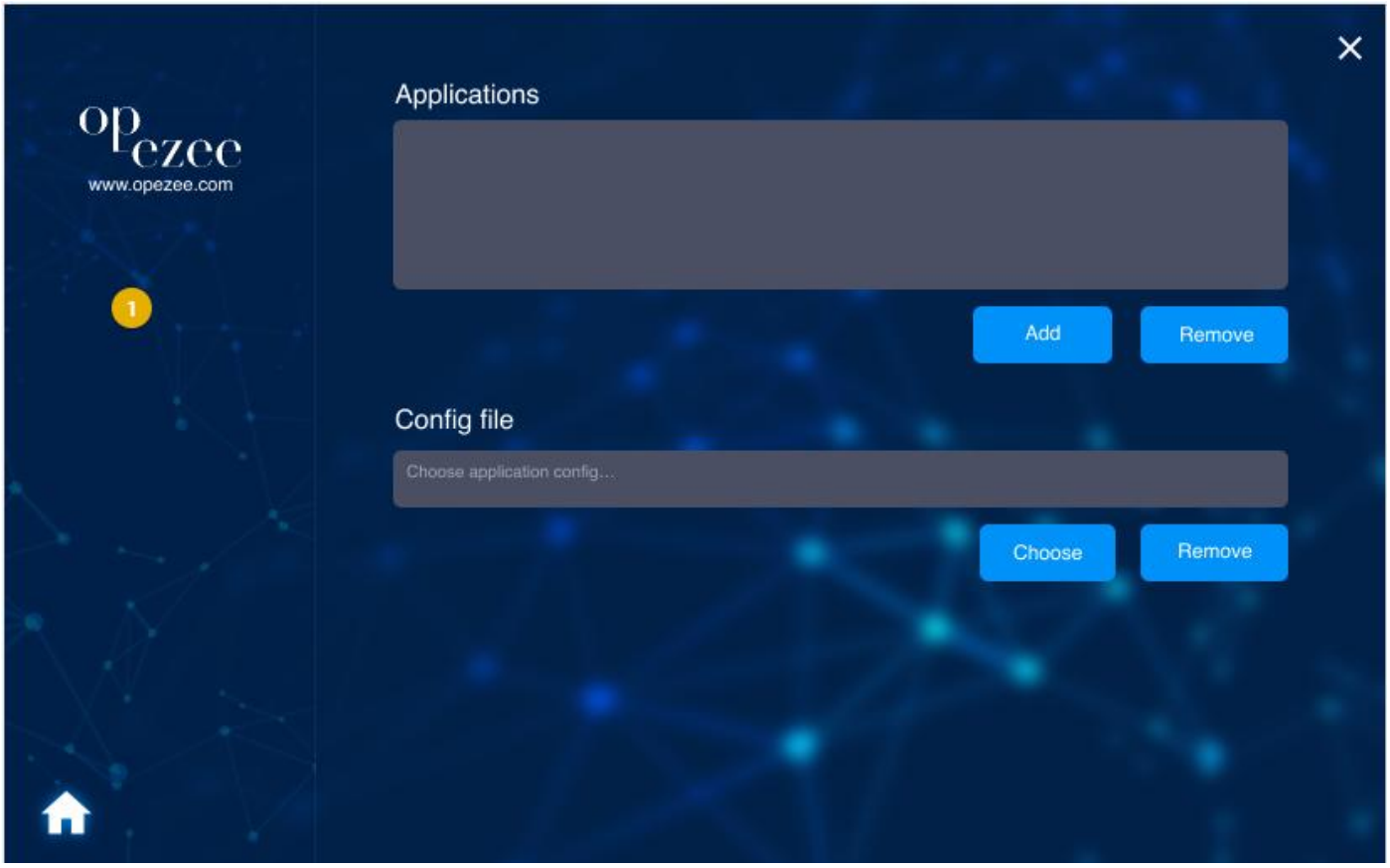
Part 3 – Settings

On Windows only:

1. The Home Screen of the launcher should show a settings button.



2. When this button is tapped, open a new Settings page :



3. From here you should be able to manipulate the Application/Parameter list store on the server. i.e.
   a) Select a application file using Windows file selection dialog.
   b) Add selected exe to the applications list.
   c) Remove previously stored application.
4. Go back to home screen when Home button is pressed.

Part 4 – Showing Off

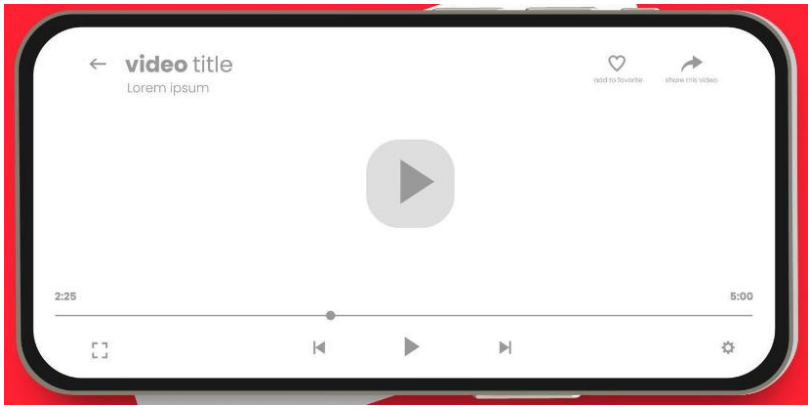
Do something that will blow our socks off!

Pick a change that will improve the app in some way and make that change.

Here are some suggestions if you are feeling stuck:

- Media player with remote controls:
    - Create a web app that connects to a windows media player (vlc, mpv **on Windows**) using TCP/UDP and is able to control its playback (seek, play/ pause, etc).
    - Launch the media player **on Windows** using the Server/UI from a android device (Created in Part 1-2)
    - In android ui, instead of showing just the loading screen in part 5 , load and show the web app.
    - On Windows (full screen media):

    - On Android (web app automatically launched by the launcher):



The above points should be just an outline, we **value** out of the box thinking and ingenuity in our candidates. Express yourself through this test and let the code speak for you.

Sending in your submission

Write a short description about your process for building these features. Include any details of the implementation that you think are important.

Packaging:

- Zip up your code, solution, project files.
- Write and include a clear and concise document explaining how to build the projects.
- In the document, please mention the time spent on each part of the test.
- Include packaged files for the target platforms (exe, apk).
- Include a short video demonstrating the functionality of the launcher.
- Include batch file (windows) to compile/build the components that are not using Android studio/Visual studio.
- Do not include any compiled binaries, intermediate files, temp data that can be generated from code, etc.

Please test that the code included in your zip file is complete and will compile when unzipped onto another machine.

Thank you.