Custom Stripped Tensorflow-Lite Build for your model in Windows

1)Git clone the tensorflow library: https://github.com/tensorflow/tensorflow/
2)Install Android SDK and Android NDK(15 c version)
3)Install Bazel : https://docs.bazel.build/versions/master/install-windows.html
4)Install MSYS2 x86_64 : https://www.msys2.org/#installation

Bazel is the build system here required for stripping the binary. MSYS2 is required for using common Bash and Unix commands.
Other than this, you should have the Java Development kit installed(Preferably Jdk 14) and environment variable set.
Similarly for Bazel, set the environment variable, if you encounter the error as mentioned in the troubleshooting section of the above link.
Before starting the build procedure, kindly run this small example

a) git clone https://github.com/bazelbuild/examples
b) cd java-tutorial.
c) bazel build //:ProjectRunner

Just see if this produces the build without any error. This would be a quick check of your java and bazel environment set up. Now we will move to steps for building.
Steps for building:

1) From the root, run : ./configure
2) Answer all questions as “n”, except for “Download fresh release of clang” and “configure your workspace for Android Builds”, answer “y”.
3) For Android NDK : use the path - C:\Users\[username]\AppData\Local\Android\Sdk\ndk-bundle
   Similarly provide the path for Android SDK.
4) Dont write anything in android sdk and build tools version, it should be done by bazel automatically.
5) Once configuring is done, open file “.tf_configure.bazelrc” and change the “build --action_env ANDROID_NDK_API_LEVEL” to 21.
6) Now open your “.tflite” model in neutron and note down all the blocks used. The blocks are the tensorflow operations used by your model.
7) Go to \tensorflow\lite\kernels\. Open “builtin_op_kernels.h” and comment out all operations except the once used by your model from Step 6. Also open “register.cc” and similarly comment out the unrequired operations.
8) Run
   bazel build //tensorflow/lite/java:tensorflow-lite --cxxopt=--std=c++14 -c opt --fat_apk_cpu=armeabi-v7a --verbose_failures
   Here I have built for armeabi-v7a architecture only. You can add more architectures like

--fat_apk_cpu=armeabi-v7a,arm64-v8a. However armeabi-v7a covers most of android devices.

9) In the very end, a zip error should come saying “Nothing to do”. The --verbose_failures(used in step 8) would list out the commands where the build got failed. Just copy paste them one by one till the command “SET TF_ENABLE_XLA=1”

This should be something on these lines:

```
cd C:/users/admin/_bazel_admin/lvlw6h4e/execroot/org_tensorflow
  SET ANDROID_BUILD_TOOLS_VERSION=30.0.0
    SET ANDROID_NDK_API_LEVEL=21
    SET ANDROID_NDK_HOME=C:/Users/ADMIN/Pictures/android-ndk-r17c
    SET ANDROID_SDK_API_LEVEL=29
    SET ANDROID_SDK_HOME=C:/Users/ADMIN/AppData/Local/Android/Sdk
    SET
PATH=C:\msys64\usr\bin;C:\msys64\bin;C:\WINDOWS;C:\WINDOWS\System32;C:\WINDOWS
\System32\WindowsPowerShell\v1.0;C:\Program Files (x86)\Common
Files\Oracle\Java\javapath;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\W
bem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\WINDOWS\System32\OpenSSH\;C
:\Program Files\OpenVPN\bin;C:\Android;C:\Windows\System32;C:\Program
Files\Git\cmd;C:\Users\ADMIN\AppData\Local\Microsoft\WindowsApps;C:\Program Files
(x86)\Atmel\Flip 3.4.7\bin;;C:\Users\ADMIN\AppData\Local\Programs\Microsoft VS Code\bin
    SET PYTHON_BIN_PATH=C:/Users/ADMIN/anaconda3/python.exe
    SET PYTHON_LIB_PATH=C:/Users/ADMIN/anaconda3/lib/site-packages
    SET RUNFILES_MANIFEST_ONLY=1
    SET TF2_BEHAVIOR=1
    SET TF_CONFIGURE_IOS=0
    SET TF_ENABLE_XLA=1
```

10) The command after this fails mainly because source is not valid command for windows. Further the path set for generating libtensorflow-lite.so is incorrect. Run the following commands one by one.

- C:/msys64/usr/bin/bash.exe -c external/bazel_tools/tools/genrule/genrule-setup.sh;
- cp bazel-out/x64_windows-opt/bin/tensorflow/lite/java/tensorflowlite.aar bazel-out/x64_windows-opt/bin/tensorflow/lite/java/tensorflow-lite.aar
- chmod +w bazel-out/x64_windows-opt/bin/tensorflow/lite/java/tensorflow-lite.aar
- mkdir common

- cd common
- mkdir lib
- mkdir jni
- unzip "../bazel-out/x64_windows-opt/bin/tensorflow/lite/java/tensorflow-lite_dummy_app_for_so_unsigned.apk" -d "lib"
- mv lib/lib/armeabi-v7a jni/
- zip -r ../bazel-out/x64_windows-opt/bin/tensorflow/lite/java/tensorflow-lite.aar jni/armeabi-v7a/*.so
- mkdir headers
- mkdir "headers/tensorflow/lite"
- cp -RL "../tensorflow/lite/builtin_ops.h" "headers/tensorflow/lite/builtin_ops.h"
- mkdir "headers/tensorflow/lite/c/c_api.h"
- cp -RL "../tensorflow/lite/c/c_api.h" "headers/tensorflow/lite/c/c_api.h"
- cp -RL "../tensorflow/lite/c/c_api_experimental.h" "headers/tensorflow/lite/c/c_api_experimental.h"
- cp -RL "../tensorflow/lite/c/common.h" "headers/tensorflow/lite/c/common.h"
- zip -r ../bazel-out/x64_windows-opt/bin/tensorflow/lite/java/tensorflow-lite.aar headers

11) This should finish the build successfully. Final Generated aar should be at : bazel-out/x64_windows-opt/bin/tensorflow/lite/java/tensorflow-lite.aar. You can import this in your android project as putting the following line implementation project(':tensorflow-lite') in build.gradle file after importing the .aar as New module inside the Android Project.

Other than this, a prevalent error occurs related to flatbuffers in bazel.
The following link covers some of the issues however not exhaustive.
https://github.com/tensorflow/tensorflow/issues/38525
The following blog from Harshit Dwivedi was referred for doing the build process.
https://heartbeat.fritz.ai/compiling-a-tensorflow-lite-build-with-custom-operations-cf6330ee30e2