Please make a copy of this document and include this in your GitHub repository for your submission, using the tag #AndroidDevChallenge

Tell us what your idea is: Helper App to detect washing machine is done, iron is idling..

Describe in 250 words what the feature or service will do and how you'll use Machine Learning to push the bar:

The Smart Home Helper app will provide the following features to the user by applying deep learning mechanism:

- *Power metering 'sensor':*
  The app evaluates the power usage of connected appliances via smart plugs, to
  - identify automatically the connected appliance
  (e.g. light, coffee-machine, iron, electric kettle, washing machine, dryer...)
  - detect automatically the programme of the washing machine and when it has finished
  (to inform the user, and automatically switch-off in order to save power)
  - detect that the coffee-machine or electric kettle has finished
  (and to inform user that the coffee is ready or the water is ready for tea)
  - detect idling irons (to alert the user and switch-off in case to prevent fire conditions)

- *User information services:*
  The app informs the user via
  - push notifications on all handsets/wear
  - voice output on Google Home

The appliances are plugged to electric power via smart outlets. Actually cheap smart plugs from Gosund (belonging to Chinese Tuya group) will be used.
In order to have full control and access by the Helper app, the smart plugs are flashed with new open source firmware from the Tasmota project.

The app uses the 'electric metering sense' to classify via trained Tensor Flow model which appliance is connected, what programme is running and when the appliance has finished its task:

- the app requests on a continuous base the metering values from the associated smart-plug(s)
- by using the Tasmato flashed plugs, the app gets in short intervals the current power consumption values

Tell us how you plan on bringing it to life.

Describe where your project is, how you could use Google's help in the endeavor, and how you plan on using On-Device ML technology to bring the concept to life. The best submissions have a great idea combined with a concrete path of where you plan on going, which should include:

- *(1) any potential sample code you've already written,*
- *(2) a list of the ways you could use Google's help,*
- *(3) as well as the timeline on how you plan on bringing it to life by May 1, 2020.*

Status of the project:
- the sensory to get the power metering values from the smart plugs is in place
- the smart plugs from Gosund (SP1-C) are flashed with the open source firmware from Tasmota project
- an app framework is in place which fetches and processes the power metering data from the smart plugs (the app will run on Android handset in home environment, i.e. using same WiFi network as the smart plugs)
- some sample code for the "receiver app" (to inform user) is in place, actually it will be a PWA which gets push notifications about status updates (info that the appliance has started its task, an update which programme is running and the actual 'finished task' notification)
- a Tensor Flow lite model is in place, which was adapted to apply object detection (person detection from connected camera) and shall form the 'base' for the time-series based deep-learning model
- development- and testing environment (Android Studio, root server, tooling etc.) is in place

List of ways to use Google's help:
- support on specific TF (lite) questions on adapting the underlying model to this scenario
- support on best approaches to train model on cloud and transfer trained weights onto app, or to apply local learning mechanism within the app locally
- potentially ways/methods to avoid that the service is being terminated by the OS (as the app will collect continuously metering values)
- to be defined - actually the category of 'to expect the still unexpected specific questions'

Timeline:
November:
- by end of november the phase to collect power metering value series will be reached (to collect as many as possible training data sets, i.e. labelled value series of electric kettle, washing machine and underlying washing programmes, dryer and iron power usage)
- further studies and prototyping on TF models, applicable model, to apply own training data
- organizing applicable boilerplate code, preparing app framework material (e.g. Vue for PWA)
- preparation of detailed work packages

- precise timeline plan

December:
- by end of december (21st December), the TF model shall be ready and based on training- and test-data, an initial evaluation of the results is achieved (to detect reliable the start of a task, the connected appliance and the actual correct prediction of end of the task)
- by end of december the TF app shall be ready as functional PoC (basic UI, foreground running service, components to interface smart plugs and provide input values to TF, trigger to send push notifications).
- by end of december the PWA app and small server aoo shall be ready as functional PoC (service workers, manifest, basic UI, getting push notifications)
- Time from 22nd - 31st December is reserved buffer to finalize above work packages

January:
- Focus is on optimizing the TF model (to improve the prediction rate)

February:
- focus is finalizing the user experience, to optimize the user scenarios (the service shall work as transparent and silently in the background as possible, just informing and in case alerting the user)
- further optimization of TF model, training and test data sets

March::
- Intensive testing of the app and service with friends (beta)
- further usability optimization

April:
- buffer (which typically is highly needed)
- preparation of event, testing all possible (and impossible) scenarios and applicable mitigations

May, 1st:
- ready, everything in place and works without quirks and issues

Tell us about you.

A great idea is just one part of the equation; we also want to learn a bit more about you. Share with us some of your other projects so we can get an idea of how we can assist you with your project.

With my professional background in Giesecke+Devrient, we launched the payment apps and underlying services on server side in mobile payments e.g. for
Banks and Institutes like S-Payment Group:
https://play.google.com/store/apps/details?id=com.s_payment.mobiles_bezahlen&hl=en

Or Rabobank:

https://play.google.com/store/apps/details?id=nl.rabowallet&hl=en

In small team (2 colleagues and myself) participated in hackathons like

2018: https://ahoi.sparkassen-hub.com/en/symbioticon/

2019:
https://www.ecb.europa.eu/pub/conferences/html/20190506_innovative_euro_single_market.en.html

Next steps.

- Be sure to include this cover letter in your GitHub repository
- Your GitHub repository should be tagged #AndroidDevChallenge
- Don’t forget to include other items in your GitHub repository to help us evaluate your submission; you can include prior projects you've worked on, sample code you've already built for this project, or anything else you think could be helpful in evaluating your concept and your ability to build it
- **The final step is to fill out this form to officially submit your proposal.**