GSOC 2020 Project Proposal at RedHenLab

Nitesh Mahawar

March 30, 2020

About Me

I am a 19-year-old, Chemical Engineering sophomore at Indian Institute of Technology, Roorkee. I developed a passion for programming, and machine learning in particular in my freshman year and have been working with various technologies, languages, and frameworks since then. I've read and implemented a considerable number of research papers on Machine learning, computer vision, and sequence/language modeling. For the last few months, I've worked on a project gave me by the principal scientist of CSIR-CBRI, Roorkee **(I'm currently all done with this project).**

I was looking at organizations accepted for GSoC, and Red Hen immediately sparked my interest as most of their projects are in line with my previous projects. I choose multimodal TV show segmentation as my project as I'm very interested in working with a video dataset of such large size. While going through this project, I also developed a curiosity about the benefits of considering multiple modes/cues in concluding something.

Availability

I currently do not have any other commitments for summer and do not plan on having any if I get accepted for GSoC. My vacations extend from the first week of May until 12th July. I can easily dedicate 50-60 hours a week on the project during my vacation and up to 40 hours a week after that. I have flexible timings for communication via email, skype, discord, slack, or any other form.
Available for Skype: 5:00 AM - 10:00 PM UTC / 10:00 PM - 3:00 PM PDT

Programming knowledge:

1. **Programming Languages -** Fluent in Python, C/C++, Java, HTML, CSS, Javascript.

2. **Data analysis -** Fluent in NumPy, Pandas, Matplotlib.
3. **Machine learning -** Fluent in PyTorch, Tensorflow, scikit-learn. Familiar with OpenCV, Keras and face-recognition (the Python API).
4. **Linux -** Have been using Ubuntu 16.04 for over 3 years now. Fluent in bash commands and the Linux directory structure.

Summary of the Proposal

- **Background information -** We can find boundaries between the shows finished through the Custom clustering algorithm and filtering algorithm, which is written by Sasi Kiran(GSoCer '19). We also identify anchors but limited by the availability of the Anchor in the MSCeleb dataset.
- **Proposed approach -** I am working on **Sasi's work** and only adding extra features to the in-production code and enhance the previous work. The main research question or problem is to split the videos into named show and dated show. Identify anchor/show names or recognizes what show it is. Using the IMDb dataset (Will discuss in method section) to identify show names from anchor names (identified using MSCeleb). We Split the redhen's video dataset into smaller segments, one for each TV show, with annotations of show title, channel, time, date. Currently, the most time-consuming process in the program is that of going frame by frame and extracting faces. Last year's face recognition library also supported by batch processing. Batch processing can be 3x faster then processing single images at a time. So I think we use batch processing for multi-threading.

Background

- **Previous work is done in the project by Sasi Kiran(GSoC'19) -**

  He created an algorithm that can automatically find boundaries between TV shows in unannotated recordings. The final method uses face recognition to see all the faces present in the video. It clusters them to group the faces by persons. We then separate anchors/hosts of shows from those persons that are not anchors/hosts using a few rules. All anchors found using this method are named using our custom classifier built on the MS-Celeb data set.
- **What we don't know about the topic -**

We don't know if all anchors are present in the dataset and identifiable by facial recognition. The facial recognition system is a technology capable of identifying or verifying a person from a digital image or a video frame from a video source. But in general, it works by comparing selected facial features from a given picture with faces within a database.

- **What are the challenges -**

  The other most significant challenge for us that the show names and the channel names are hard to detect from the video's frames because it's costly and time-consuming to build a recognition model. So in this approach, we try to infer show name and channel name on the information which we can collect that is anchor name instead of video's frames.

- **What the unknowns that I am going to address -**

  The unknowns are Anchor, Channel, and Show names which adequately explained in methodology.

  Currently, there is one thing which I'm going to address in this project is the IMDb dataset.

  I will explain this point in detail in the methods section

Goal and Objectives

- **The goal of this research is to split the videos with the show name, dated shows and to find "What the show it is" in the redhenlab dataset videos -**

  1. To identify Show name.
  2. To find the show channel and Show date.
  3. In the current implementation performing face-recognition iteratively on the video's frames is the most time-consuming process. I aim to address this using multi-threading so we can parallelly process multiple frames on a GPU.
  4. Finally split the videos into named and dated shows.

The Goals briefly discussed in the Methods section

Methods

- **IMDb Dataset -** IMDb Dataset is an online database of information related to films, television programs, home videos, video games, and streaming content online – including cast, production crew, and personal biographies. It provides us the list of Anchor's name and channel's name. IMDb has approximately 6.5 million titles (including shows) and 10.4 million personalities in its database.

- **Subtitle Matching -** To explore subtitles because subtitles are the only sources that contain a lot of important and explicit information, but the problem is how to extract or track the information about the show like anchor name for each segment of videos. So I have one idea by using the IMDb dataset; we will try to find anchor names which unable to find by face recognition. So by matching all anchors from the channel name to subtitles, we get anchor name.

- **How to tackle those problems and Challenges briefly discussed here with reason -**

  How to find the Channel Name -

  For the channel name, we already know about anchor name(face recognition). By using the IMDb dataset for a given anchor, we can get a list of all the channels that he/she worked for. But if we want to know the correct channel name, then we use co-anchors or some anchors which is come for a short time in the show, later with the help of both the main Anchor or co-anchor works in the same channel we easily extract correct channel name.

  How to find the Show Name -

  With the combination of Anchor name and Channel name, by using IMDb dataset for a given anchor, we can get a list of all the shows that he/she worked now filter the exact show we can use filters like channel name and airing date.

  Example - If the Anchor is X, Channel is Y, and the video is from July 1977. What the show is?

  Solution - We should get 3 or 4 shows with the combination of (X, Y), But we should get less show mostly even one show with the combination of (X, Y, July 1977).

- The additional resources (IMDb data sets, pre-implemented methods or algorithms by Sasi Kiran(GSoCer' 19))

- The result of my current approach is to split the more shows with the show-name and date. The script of breaking the videos already written.

1. After my work, we get mostly show name and date.
2. Then finally, I have to need to edit the splitting script then we get the split's shows with the show name and date.
3. We achieve all the information regarding the shows recognizes "what show it is?" with annotations of show title, channel.

My implementation in brief with in-complete information -

There are three unknowns that we need to address (anchor name, channel name, show name). Currently, my approach revolves around filtering out any of these unknowns when we know at least one or two of the other unknowns. For getting show name using (anchor name, channel name) and for getting channel name using (anchor name, show name). I am going to explain about these unknowns with the suitable briefly example -

Currently, we have three **dependent** variables (Anchor, show, channel) which we are going to find out by following approaches -

1. **Getting channels from group of anchors -**
   In an eight-hour-long video, we get seven shows by using last year's segmentation algorithm. There should be seven anchors present in that long video, but we get only five anchors because it is not completely accurate. We don't need to know about all the Anchor's names, but we have to know at least 2,3 anchors' names so that we can accurately filter the channel name.

   Example - Let's suppose we get five associate anchors(A1, A2, A3, A4, A5) instead of 7 anchors then how we can filter more accurate channel name.
   A1 = C1, C2, C3, C4
   A2 = C5, C2 ,C6 ,C7
   A3 = C8, C2 ,C9 ,C10
   A4 = C11, C2 ,C12 ,C13
   A5 = C14, C2 ,C15 ,C16
   So C2 is the only channel where all five anchors work for

2. **Getting show from (anchor, channel) -**
   We can easily filter the show name from the combination of (Anchor, channel name), so basically firstly, we get all shows from the Anchor worked for and also got all shows from the particular channel. Then we compare

both the shows and using filters like (date, duration) get the correct show.

Example - Let's suppose we get
Ten shows(S1, S2, S3, S4, S5, S6, S7, S8, S9, S10) form the given Anchor and get
12 shows(S1, s2, S3, s4, S5, s6, S7, s8, S9, s10, s11, s12) from the given channels after comparing both the shows and get only
5 Shows(S1, S3, S5, S7, S9), then using more filter(date, duration), we should get fewer shows, maybe one show.

These three approaches are direct

3. **Getting channel from anchor -**
   In this approach, we are going to catch directly channel name from the anchor name because sometimes an anchor has only worked with one channel in his whole life. Then we easily get the channel name, and we further work on the channel name and get other information like shows and other anchors.

   Example - By using my code(will explain in the next section), we get the channel from a particular anchor like **Courtney Friel** has only worked for KTLA-TV throughout her career.

4. **Getting channel from show -**
   In this approach, we are going to catch directly channel name from the show name because of mostly a single show telecast on a particular channel. Then we easily get the channel name, and we further work on the channel name and get other information like channels and other anchors.

   Example - In the 99 percent cases, the particular show telecast on a single channel, so we get channel names very easily.

5. **Getting show name from anchor -** In this approach, we are going to catch directly show name from the anchor name because sometimes an anchor has only worked with one show in his whole life. Then we easily get the show name, and we further work on show name and get other information like channels and other anchors.

Pseudo-Code

IMDb dataset contains these files
names.tsv, title-basics.tsv, title-crew.tsv, title-principals.tsv

1 Find shows using information like anchor,channel,date,duration

nconst = constant value associate with the particular anchor
tconst = constant value associate with the particular show

1. **Get nconst from name.basics dataset**

```
# Here I'm going to read all anchors name from imdb dataset and store it
into a namebasics

namebasics = read_table('name.basics')

# Here I'm comparing anchor name(face recognition) with IMDb(anchors)
and store it into a nConst

nConst = (namebasics[namebasics['primaryName'] == 'Dana Bash']
['nconst']).item()
```

2. **Get tconst from the title.principal dataset with the help of nconst**

```
# Here I'm going to read all shows name from IMDb dataset and store
it into a titleprincipal

titleprincipal = read_table('title.principals')

# Getting all shows(tconst) by using anchorname(nConst)

for i in enumerate((titleprincipal)):

    # here we are comparing both the anchor name(by face recognition
    or by dataset) to get shows

    req_shows = titleprincipal[titleprincipal['nconst'] == nConst]
    all_shows = all_shows.append(req_shows, ignore_index = True)
```

```
# this the list of some shortlisted shows

all_shows
```

3. **get show names from title.basics with the help of (nconst,tconst)**

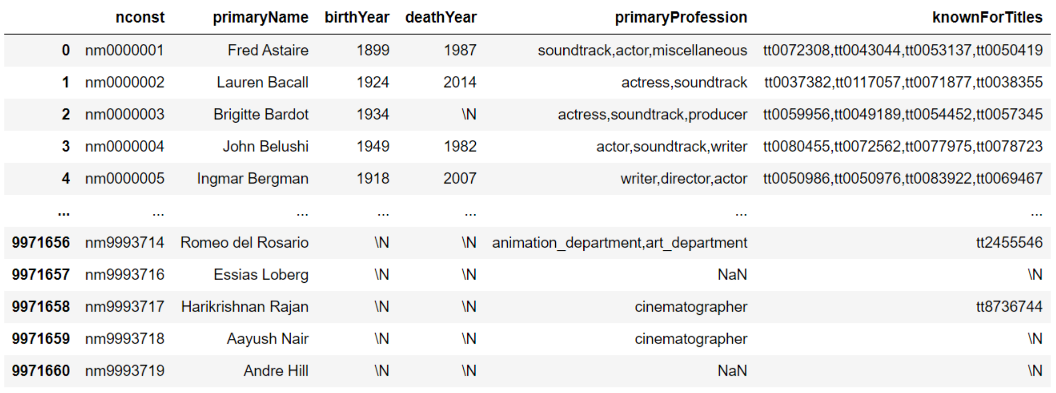
```
titlebasics = read_table('title.basics')

# store all slected shows in show_tConst
show_tConsts = set(all_shows['tconst'])

for i in enumerate(titlebasics):
    show_names = titlebasics[titlebasics['tconst'].isin(show_tConsts)]
    Show_Names = Show_Names.append(show_names, ignore_index = True)

Show_Names

# then to find the single show on a particular date

tConst = (Show_Names[Show_Names['primaryTitle'] == 'Episode dated
5 October 2004']['tconst']).item()
```

**** This is only pseudo-code and is subject to change based on the actual implementation.**

Results

I have already tested my current approach and get results given below-

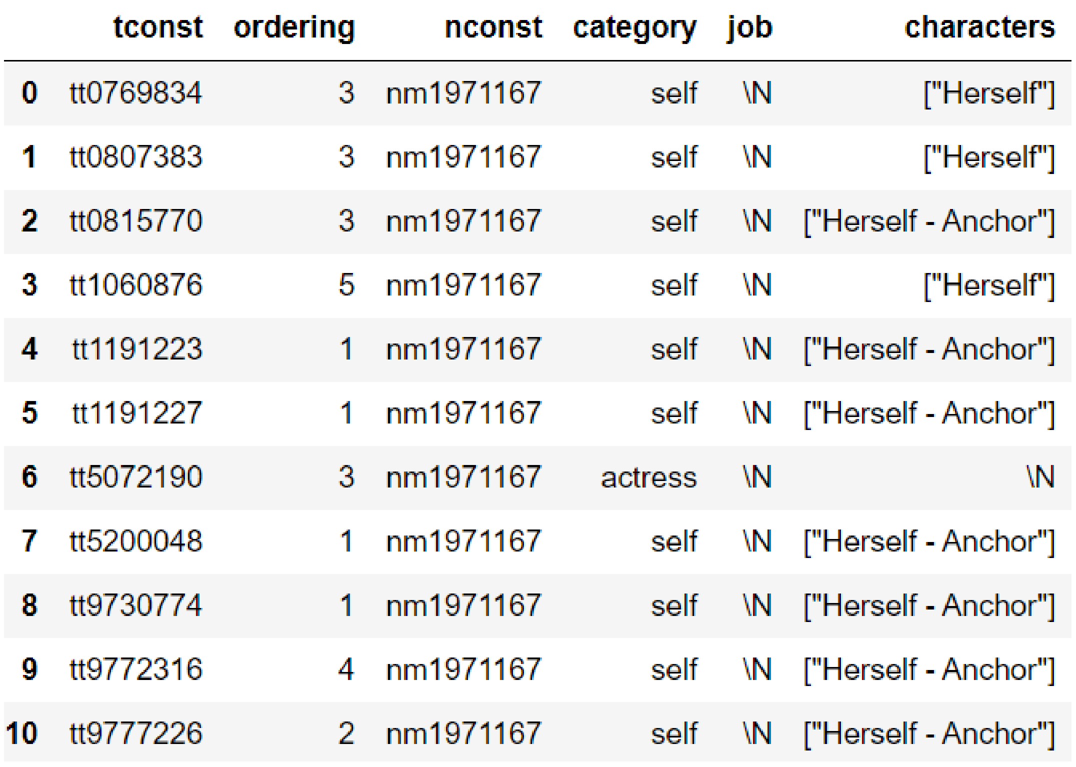
1. I took an anchor named **"Heidi Collins"** and compare this name to IMDb dataset to get constant value of this particular anchor is listed below-



	nconst	primaryName	birthYear	deathYear	primaryProfession	knownForTitles
0	nm0000001	Fred Astaire	1899	1987	soundtrack,actor,miscellaneous	tt0072308,tt0043044,tt0053137,tt0050419
1	nm0000002	Lauren Bacall	1924	2014	actress,soundtrack	tt0037382,tt0117057,tt0071877,tt0038355
2	nm0000003	Brigitte Bardot	1934	\N	actress,soundtrack,producer	tt0059956,tt0049189,tt0054452,tt0057345
3	nm0000004	John Belushi	1949	1982	actor,soundtrack,writer	tt0080455,tt0072562,tt0077975,tt0078723
4	nm0000005	Ingmar Bergman	1918	2007	writer,director,actor	tt0050986,tt0050976,tt0083922,tt0069467
...	...	...	...	...	...	...
9971656	nm9993714	Romeo del Rosario	\N	\N	animation_department,art_department	tt2455546
9971657	nm9993716	Essias Loberg	\N	\N	NaN	\N
9971658	nm9993717	Harikrishnan Rajan	\N	\N	cinematographer	tt8736744
9971659	nm9993718	Aayush Nair	\N	\N	cinematographer	\N
9971660	nm9993719	Andre Hill	\N	\N	NaN	\N

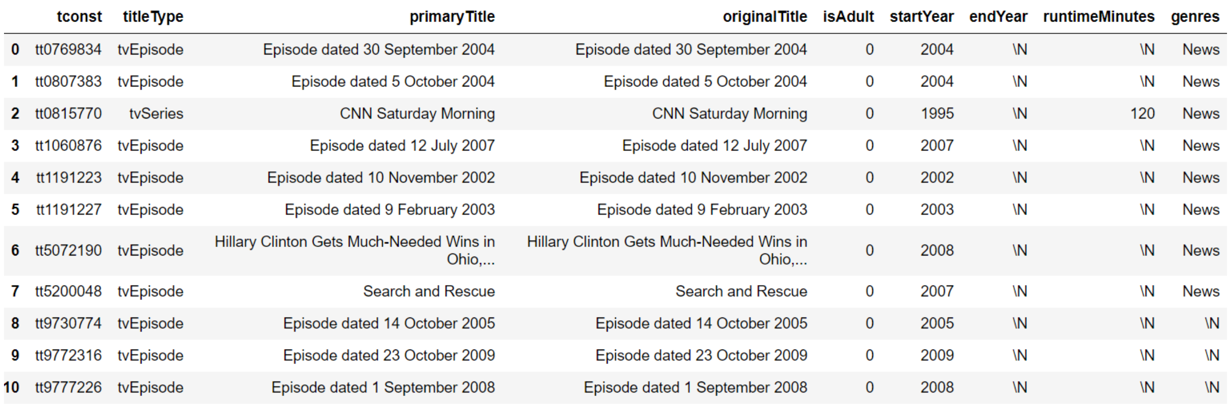
9971661 rows × 6 columns

2. By using a constant value of show(tconst) I get some shows she worked for are listed below -



	tconst	ordering	nconst	category	job	characters
0	tt0769834	3	nm1971167	self	\N	["Herself"]
1	tt0807383	3	nm1971167	self	\N	["Herself"]
2	tt0815770	3	nm1971167	self	\N	["Herself - Anchor"]
3	tt1060876	5	nm1971167	self	\N	["Herself"]
4	tt1191223	1	nm1971167	self	\N	["Herself - Anchor"]
5	tt1191227	1	nm1971167	self	\N	["Herself - Anchor"]
6	tt5072190	3	nm1971167	actress	\N	\N
7	tt5200048	1	nm1971167	self	\N	["Herself - Anchor"]
8	tt9730774	1	nm1971167	self	\N	["Herself - Anchor"]
9	tt9772316	4	nm1971167	self	\N	["Herself - Anchor"]
10	tt9777226	2	nm1971167	self	\N	["Herself - Anchor"]

3. By using constant value of shows(tconst), we get shows name listed below-

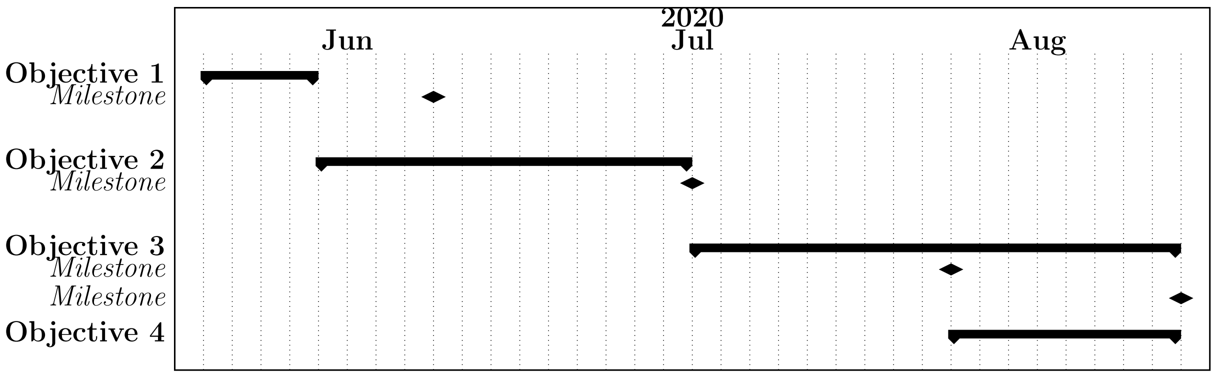


	tconst	titleType	primaryTitle	originalTitle	isAdult	startYear	endYear	runtimeMinutes	genres
0	tt0769834	tvEpisode	Episode dated 30 September 2004	Episode dated 30 September 2004	0	2004	\N	\N	News
1	tt0807383	tvEpisode	Episode dated 5 October 2004	Episode dated 5 October 2004	0	2004	\N	\N	News
2	tt0815770	tvSeries	CNN Saturday Morning	CNN Saturday Morning	0	1995	\N	120	News
3	tt1060876	tvEpisode	Episode dated 12 July 2007	Episode dated 12 July 2007	0	2007	\N	\N	News
4	tt1191223	tvEpisode	Episode dated 10 November 2002	Episode dated 10 November 2002	0	2002	\N	\N	News
5	tt1191227	tvEpisode	Episode dated 9 February 2003	Episode dated 9 February 2003	0	2003	\N	\N	News
6	tt5072190	tvEpisode	Hillary Clinton Gets Much-Needed Wins in Ohio,...	Hillary Clinton Gets Much-Needed Wins in Ohio,...	0	2008	\N	\N	News
7	tt5200048	tvEpisode	Search and Rescue	Search and Rescue	0	2007	\N	\N	News
8	tt9730774	tvEpisode	Episode dated 14 October 2005	Episode dated 14 October 2005	0	2005	\N	\N	\N
9	tt9772316	tvEpisode	Episode dated 23 October 2009	Episode dated 23 October 2009	0	2009	\N	\N	\N
10	tt9777226	tvEpisode	Episode dated 1 September 2008	Episode dated 1 September 2008	0	2008	\N	\N	\N

4. Then to get a single show, then use date which already given with .MP4 file. Like I took date **24 August 2005** then comparing this data to listed channels output, and we get show name constant **tt0512208**

I have already mailed some examples in pdf with code and output also to professor Francis.
Our actual task is to find the shows name and dated show.

Tentative Timeline



I have designed the timeline to cover the primary goals first and then move on to secondary goals. This priority order can change upon discussion with the mentors.

27th April - 18th May Community Bonding

- Wrap up at college and travel back home.
- Go through the dataset of videos to know how many shows manually annotated.
- Talk with mentors about the dataset, the approach, and discuss priorities in the approach.

18th May - 2nd June

- Go through 2019 GSoC and Sasi Kiran's code and make a rough list of what functions of the code reused.
- Collect all the given information about the show like duration and date of the show.

3rd June - 19th June

- After face-recognition we get anchor name then collect information like channel name, show name from IMDb dataset.
- Work on co-anchors identification by Sasi's method.
- We get (anchor + co-anchor) of the show then we easily get other information like show name and anchor name.

19th June Phase 1 evaluation

20th June - 29th June

- By the duration time method we can get show name.
- From this we collect all the information of channel name, anchor name of the show.

30th June - 9th July

- Working on multi-threading because the current method is the most time-consuming. So we have to parallelly process multiple frames on a GPU.

10th July - 16th July (will be packing up and traveling back to college for 3-4 days in between)

- After getting most of anchor, show, channel then Work on the splitting script.
- Finally split the redhen's video dataset into shows which contains show name, channel name and date also.
- Refactor and optimize the current code for any possible speed-ups.

17th July **Phase 2 evaluation**

18th July - 10th August

- Discuss any new ideas that might have come up in the two months of GSoC.
- If no feasible ideas, start working on the part of grouping shows with respective channels.
- If time permits, write a blog post detailing the approach is taken and the work done in the past three months.
- Prepare for final evaluation.

10th August - 17th August **Final evaluation**

Nitesh Mahawar

Github: https://github.com/EdOates84
Linkedin: https://www.linkedin.com/in/niteshmahawar9984/
Skype ID: live:.cid.757324467cf3b61d
Email: nmahawar@ch.iitr.ac.in

EDUCATION

7.2018 - dato	**Bachelor of Technology, Chemical Engineering** at Indian Institute of Technology,Roorkee *Passion for Programming, Machine Learning, Android* *Expected graduation in June of 2022*
7.2016 - 6.2017	**Class 12th** at New Model School(RBSE)
7.2014 - 6.2015	**Class 10th** at New Model School(RBSE)

WORK EXPERIENCE

9.2019 - 1.2020	**Season of Code** at MDG IIT Roorkee
11.2019 - dato	**Rotary Care** Under the president of Rotary Club,Roorkee

SKILLS AND QUALIFICATIONS

Programming Languages

Advanced skills	Python, Java, C/C++, HTML, CSS, JavaScript

Languages

Native	Hindi
Advanced	English, Hindi
Basics	Telugu

PROJECTS

Dec 2018 **Chat Room** Link to project website: https://github.com/EdOates84/chat-room

- This is my first project and I used Python, Django and SQL Database
- Login with the username and create or add a chat room.
- Another user can use the same name of the chat room and talk similar to group chat.

Oct'19-Jan'20 **LifeLine** Link to project website: https://github.com/EdOates84/lifeline

- LifeLine is an Android app for the institute hospital that is helpful for our campus Students as well as faculties.
- This is basically to use to make an appointment and check the current status of the ongoing Token number in the institute hospital.
- Required skills Java, Android Studio, Google Firebase-Database.

Dec 2019 **Rotary Care** Link to project website: https://github.com/EdOates84/RotCare2.0

- Rotary Care is also an Android app where Members and Volunteer can raise their problems and generate token no.
- Show, all problems (Requests) on the app and any-one can accept the request and solve the problem.
- Required skills Java, Android Studio, Google Firebase-Database.

RECOMMENDATION

Dr. Achal Mittal
Sr. Principal Scientist
CSIR-CBRI, Roorkee-India