UNIVERSITÀ DI PISA
SCUOLA DI INGEGNERIA

Dipartimento di Ingegneria dell'Informazione

Industrial Application

Film reproduction prototype

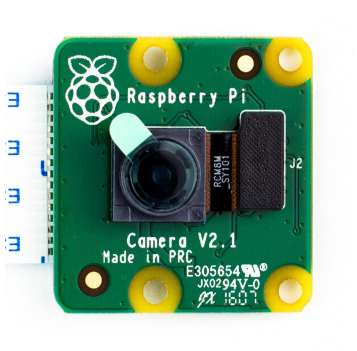
Fabiana Ritorti
Matteo Dessì

PROTOTYPE

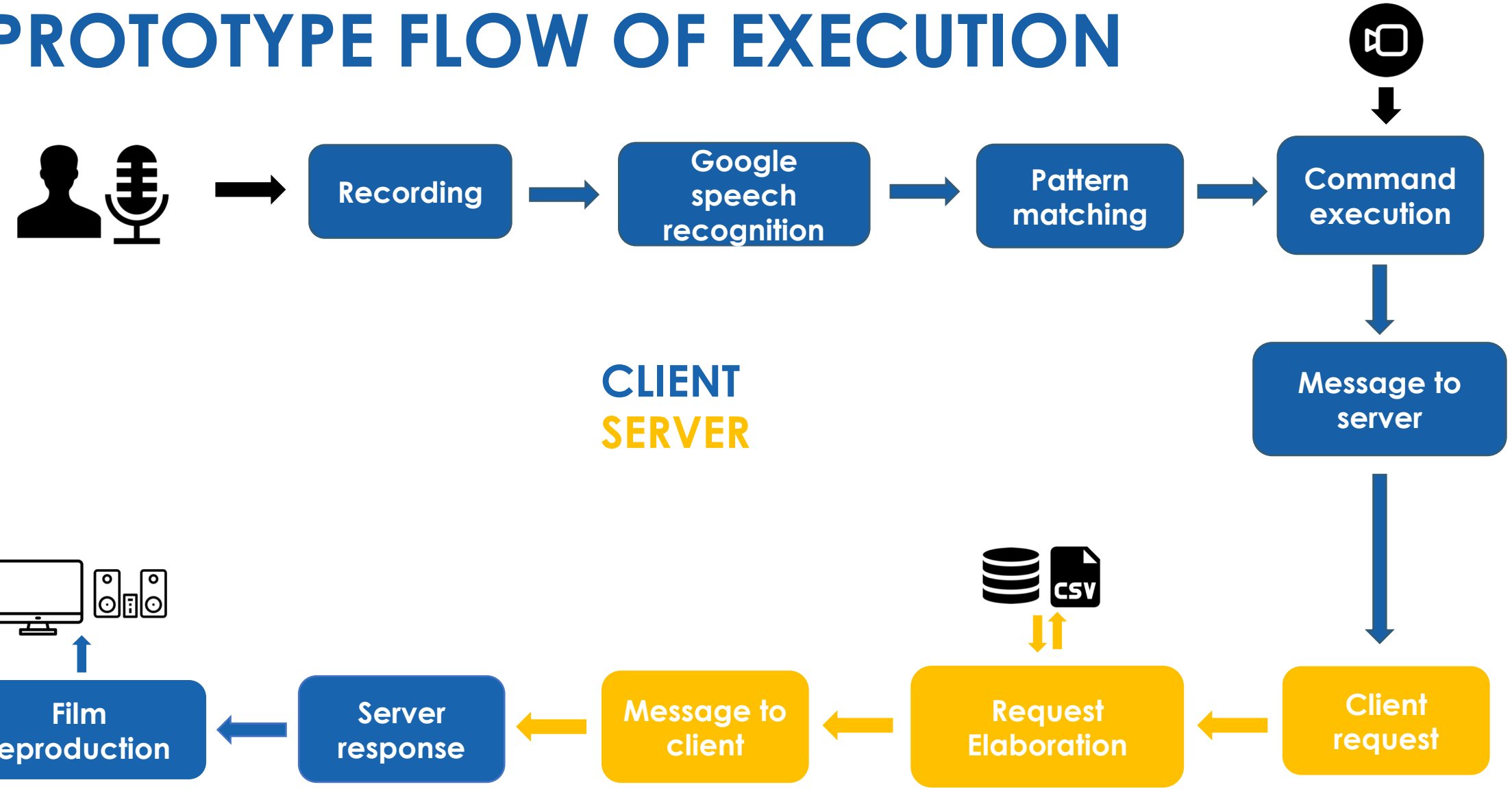
The purpose of this prototype is to reproduce the trailer of the movie chosen by the users among the possible candidates selected by the system.

The selection is based on one of the interests of the users, who can select through a voice command the film they are most interested in.

HARDWARE

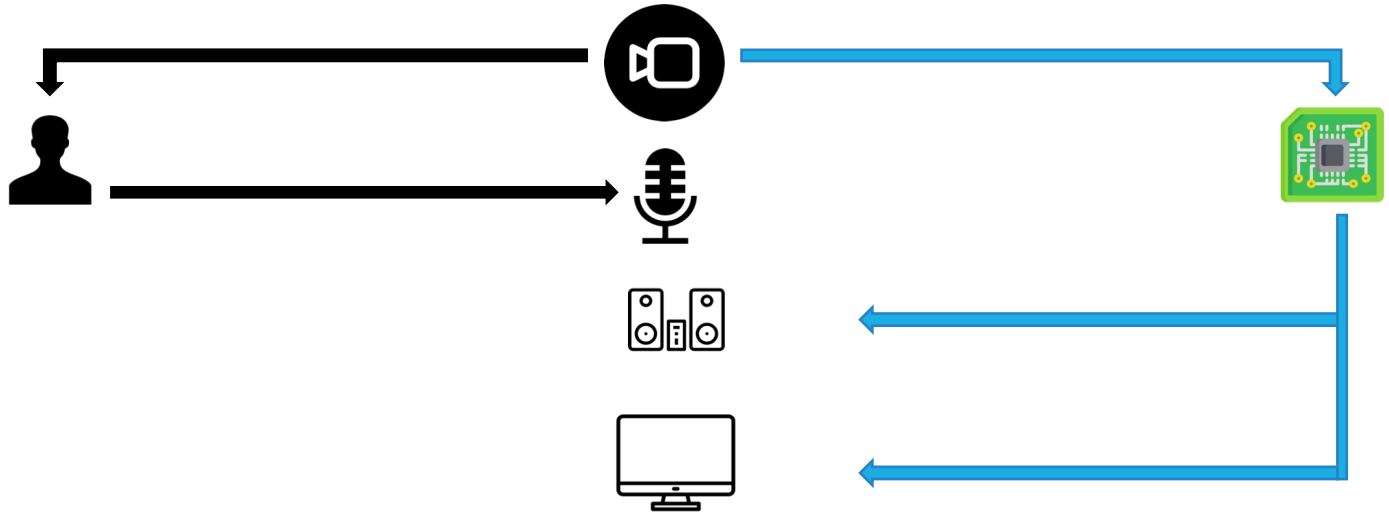


PROTOTYPE FLOW OF EXECUTION



USER-MACHINE INTERACTION

The user-machine interaction adopted in our prototype consists on, from the user side, the use of the microphone to provide commands to the system, and from the machine side, the reproduction of the audio which is the most adapt to respond to the user request.



MICROPHONE

```
import speech_recognition as sr

r = sr.Recognizer()
microphone_index = search_microphone_index()

if(microphone_index==-1):
   print("No microphone,try again!")
   exit
with sr.Microphone(device_index=microphone_index) as  m:
   r.adjust_for_ambient_noise(m,duration = 5)
```

SPEECH RECOGNITION WITH GOOGLE

```
from playsound import playsound
import speech_recognition as sr


def listen_phrase(r,m):
text = ""
audio = ""
firstTime = True
with m as source:
   while(audio==""):
      if (firstTime == False):
         playsound('./audio/unknown_response.mp3')
      else:
         audio = r.listen(source)
         firstTime = False
   try:
      text = r.recognize_google(audio,language="it-IT")
   except Exception as e:
      print(e)
print(text.lower())
return text.lower()
```

PATTERN MATCHING

What is understood by google's speech recognition system is used to check if the users have used any of the available interactions with the system.

```
if(text== "login"):
```

The machine will answer, in the case the pattern matching is correct,with the reproduction of an audio through the connected stereo.

```
from playsound import playsound

playsound('./audio/command.mp3')
```

INITIAL COMMANDS

The machine via audio command tells the user what functions the application provides:

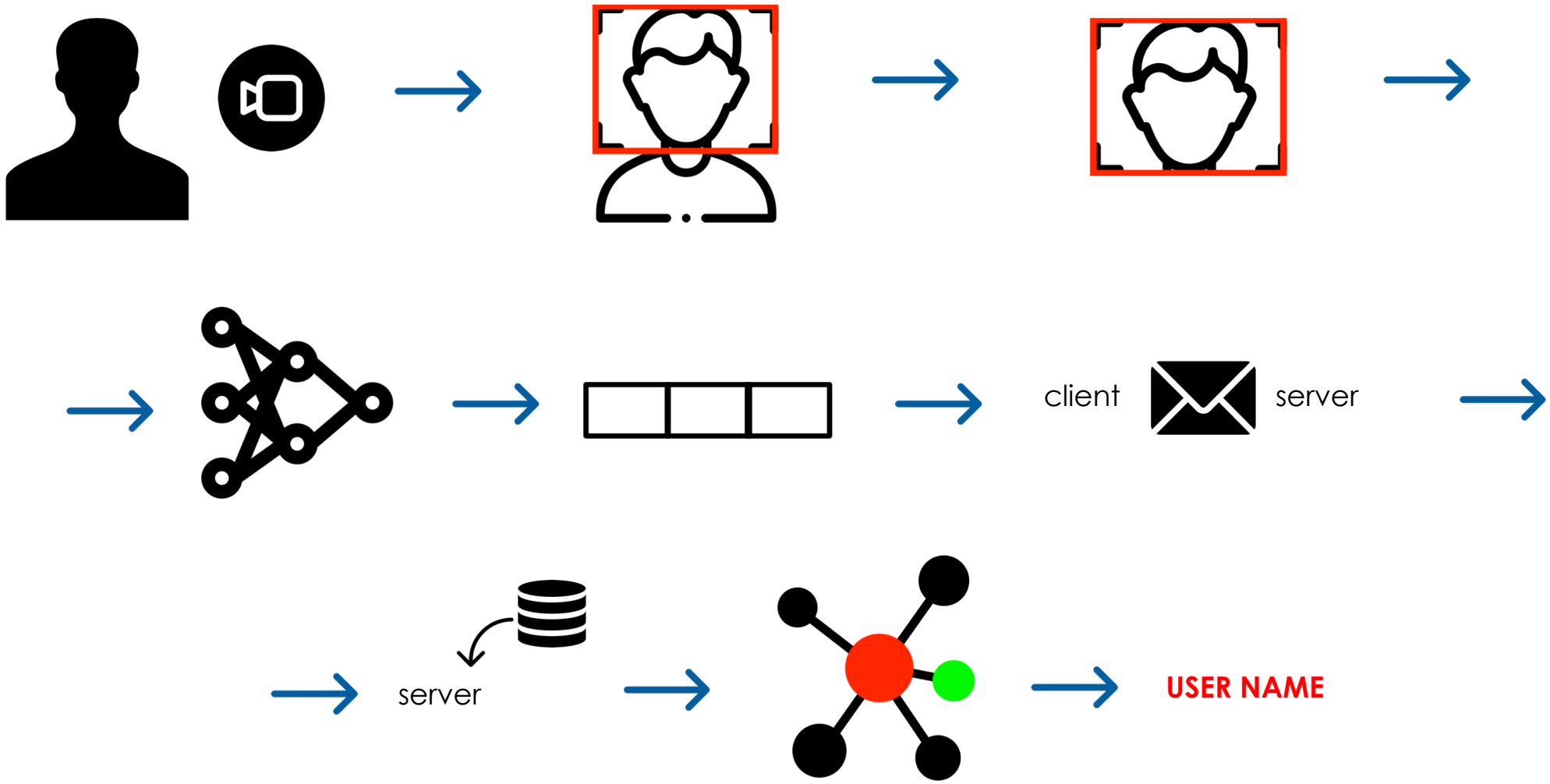
Registration: the users can register themselves with name, surname and face identity;

Login: the users can login to the application through a face identification and access the film reproduction;

Delete: the user can delete his/her profile, after having confirmed his/her identity through face identification.

Exit.

FACE RECOGNITION



FEATURE EXTRACTION TIME

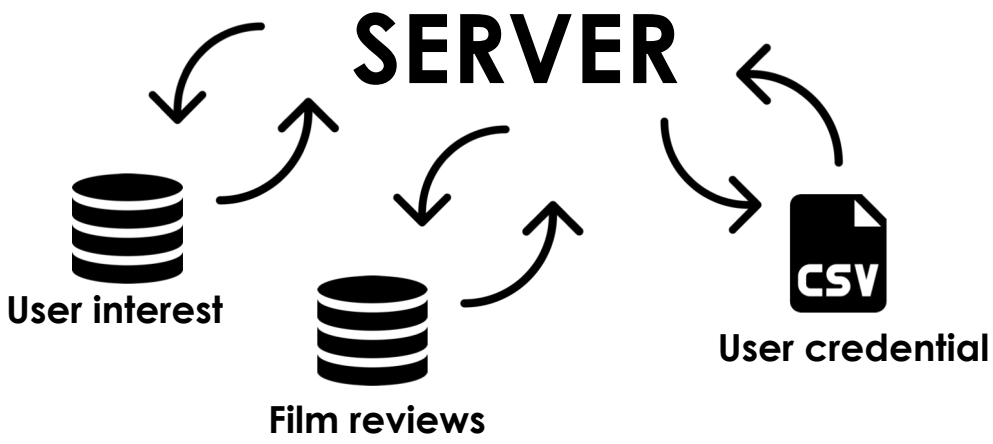
TEST	LOCAL TIME(s)	RASPBERRY TIME(s)
1	11.2	66.3
2	11.1	63.4
3	11.2	65.5
4	11.2	67.3
5	11.5	65.3
6	11.1	68.2
7	11.1	66.1
8	10.9	66.7
9	11.2	66.5
10	11.1	67.2

The average time to do the feature extraction **locally** is: **11.2 s**
The average time to do the feature extraction on **raspberry** is: **66.3 s**

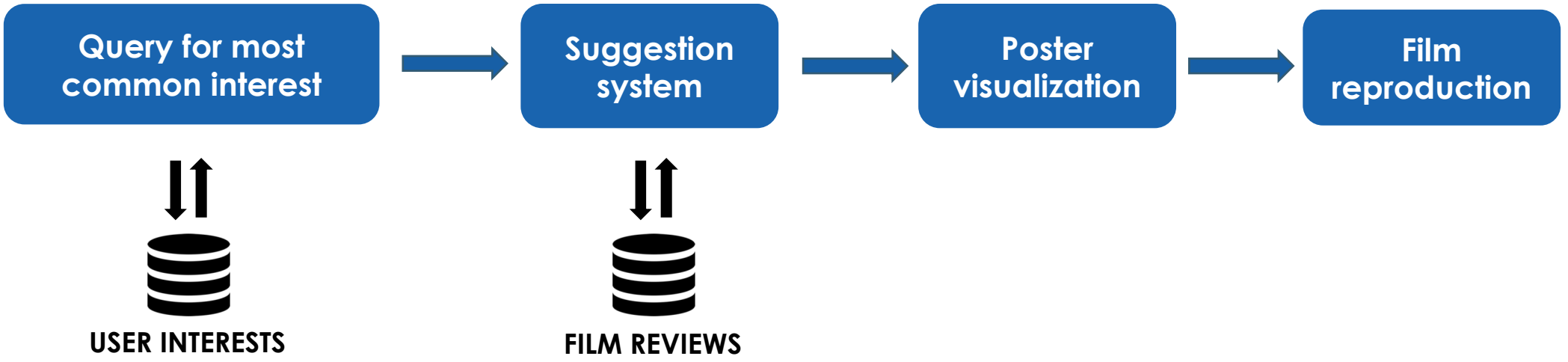
SERVER

```
import socket
import socketserver

def server_program():
    HOST = socket.gethostbyname(socket.gethostname())
    print("host",HOST)
    PORT = 5000
    server = socketserver.TCPServer((HOST, PORT), MyTCPHandler)
    server.serve_forever()
```



FILM TRAILER REPRODUCTION



ACCESS TO MONGO DB AND INTERESTS

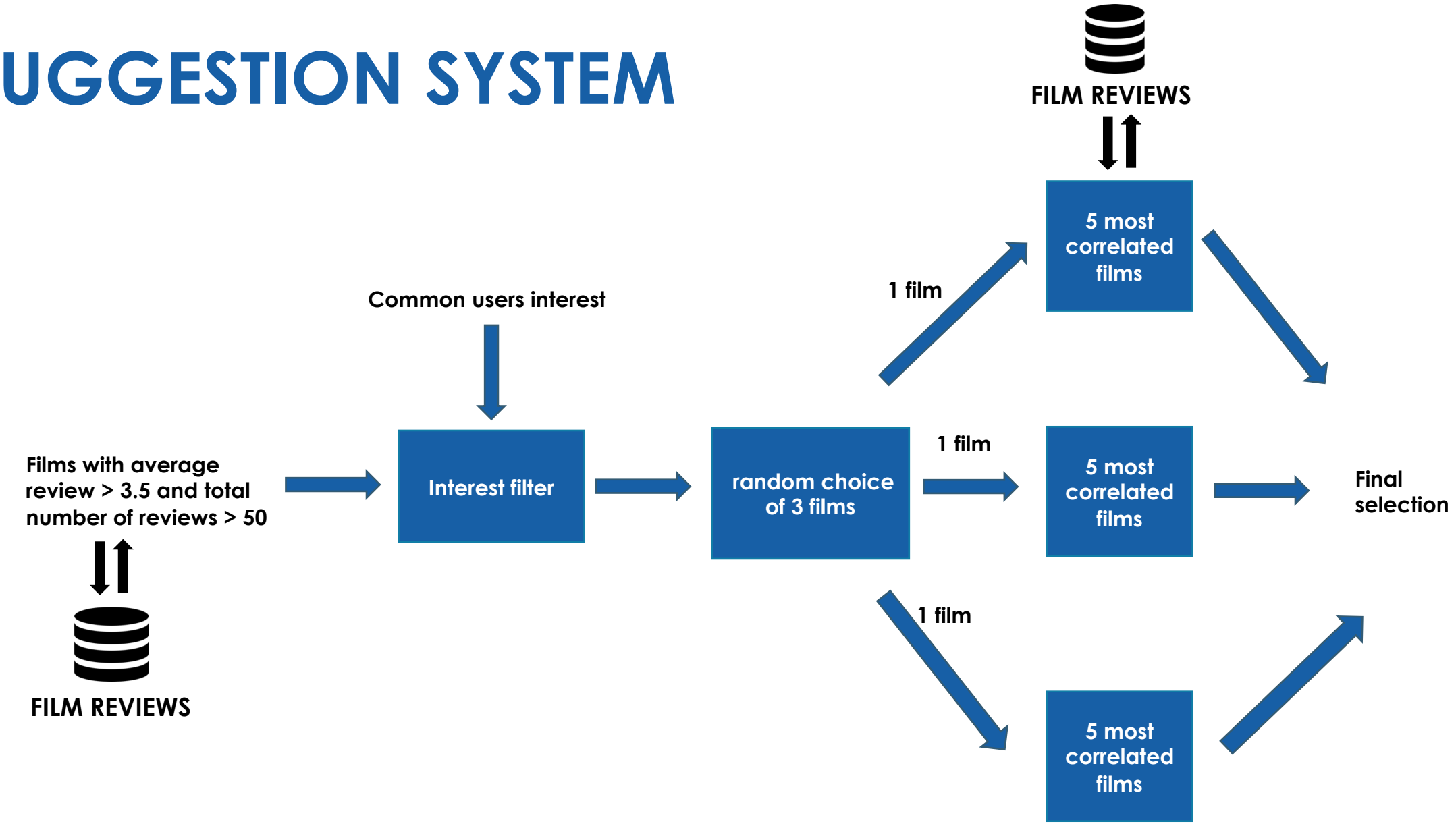
```
{ "_id" : ObjectId("61b5d8d4507ac2ed6383b2b3"), "name" : "ciccio_pasticcio", "film_interest" : [ "War", "Musical",
"Romance" ], "music_interest" : [ "Dance", "Classical", "Jazz" ], "news_interest" : [ "Fashion", "Medical", "Online" ] }

{ "_id" : ObjectId("61b5da7ef60d90c8b1121f8e"), "name" : "fabiana_ritorti", "film_interest" : [ "Documentary", "War",
"Mystery" ], "music_interest" : [ "Disney", "Reggae", "Opera" ], "news_interest" : [ "Online", "Sport", "Western" ] }
```

```
import pymongo
import random


def get_random_best_common_interests(self,names,interest_type):
string = self.interest_string(interest_type)
interests=list(self.db.Data.aggregate([{"$match":{"name":{"$in":names}}},{"$unwind":string},{"$group":{"_id":string,"count
":{"$sum":1}}},{"$sort":{"count":-1}}]))
if not interests:
   return self.get_random_interest(interest_type)
else:
   max_pop = interests[0]["count"]
   selected_interests = []
   for interest in interests:
      if(interest["count"]<max_pop):
         break
      else:
         selected_interests.append(interest["_id"])
return random.choice(selected_interests)
```

SUGGESTION SYSTEM



POSTER VISUALIZATION

Film name

Link searcher

Download Image

SHOW TRAILER

Film name

Link searcher

Request

Download

YouTube