

Quotr

Final Year Project Report

DT228

BSc in Computer Science

**Ronan Dillon**

**Fred Mtenzi**

School of Computing

Dublin Institute of Technology

**<Date>**



Abstract

The entertainment industry is at the heart of modern day pop culture. Movie and TV references and quotes have become a part of everyday conversation, so much so, it is sometimes hard to recall where the quote originated.

The objective of this project was to create a cross-platform quote recognition application which allows users to identify the origin of their TV and movie quotes. The users will be provided with information about their resulting TV show or movie, such as the actors, the director and the rating of the TV show or movie. The user will be able to view the most popular media and their search history. The application will also provide the users with recommendations based on the user’s searches.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ronan Dillon

<Date>

Acknowledgements

Body text

Table of Contents

1. Introduction
   1. Project Overview
   2. Project Objectives
   3. Project Challenges
   4. Document Structure
2. Research

2.1 Introduction

2.2 Existing Technology

2.3.1 IMDB

2.3.2 Shazam

2.3.3 Viggle

2.3.4 IntoNow

2.3 User Requirements

2.4 Survey

2.5 Conclusion

* 1. Eg. Research related to identifying the problem that this project solves, research into solution definition

1. Technology Used

3.1 Introduction

3.2 Tier 1

3.3 Tier 2

3.4 Tier 3

3.5 Content Recognition

3.6 Conclusion

* 1. An overview of the technologies evaluated and selected or rejected and the rationale behind the key decisions.

1. Design

4.1 Introduction

4.2 Methodology

4.3 System Architecture

4.4 Diagrams

4.5 Conclusion

* 1. Identification of a design methodology including why it was chosen
  2. Design of each of the project components eg: the UI, Network, Project Demonstration, source code layout
  3. Clearly identifying the list of features and use cases supported within the project.
  4. Overview of the system architecture and a diagram to represent all of the key elements within the architecture.

1. Implementation

5.1 Introduction

5.2 Development Environment

5.3 Web Service Implementation

5.4 Application Implementation

5.5 Deployment

5.6 Conclusion

* 1. Details of each component within the project, problems encountered and resolved, challenges overcome or worked around.
  2. Identify key development components;
  3. Identification/explanation of external APIs used versus own code ; List of classes of your code etc .

1. Testing and Evaluation
   1. Testing
      1. What testing was performed, why it was selected and what are the key use cases within the project.
      2. White Box Testing
      3. Black Box Testing
      4. Usability Testing
   2. Demonstration
      1. Identify what features can be demonstrated and show screen shots or reference a video online to show the project demonstration (for audience not at demo)
   3. Project Plan analysis and review of how it changed from the initial proposal including explanation of what changed and why, and suggestions on how to address this if the project was repeated.

* 1. Future Work

1. Conclusion
   1. Analysis of the projects key elements identify the key learning obtained from the project and recommendations and suggestions for how the work can be improved on continued into the future.
2. Bibliography
3. Appendix

# Chapter 1. Introduction

# **Introduction**

The average person will spend nine years of their life in front of a television and with the increasing popularity of instant streaming sites like Netflix and Hulu more people are “binge watching” television shows. ([r] A.C. Nielsen Co. 2015) Some of these shows and movies provide the viewer with historic moments of entertainment, from The Terminator’s iconic “I’ll be back” to When Harry Met Sally’s hilarious “I’ll have what she’s having”. To a person who hasn’t seen or heard them, they are meaningless, but to the viewers they can be a memory, a souvenir of the show.

* 1. **Project Aim**

The aim of the project is to develop a cross-platform application that can retrieve and display TV show and movie data based off of a user’s quote. The user enters their quote and are given facts about the TV show or movie their quote is from.

The application will provide a simplistic user interface which will allow the user to easily navigate and make use of the application’s features. Based on user’s searches the application will present the user with recommendations of other TV shows and movies that the user might be interested in. The application will contain “Top 20” and “History” pages where the user can view their past searches and also check the most popular searches on the app.

Chapter 2. Research

**2.1 Introduction**

**2.2 Existing Solutions**

Research into existing solutions is an important step at the beginning of a project like this. The goal of this stage of research is to find ways to improve current solutions that relate to the area of the project. At present, there are no apps which allow a user to identify movies and TV shows using quotes from the show or movie and displays info from the shows or movie for the user. IMDB is a popular application which has information on movies, TV shows and actors. Viggle and Shazam are both applications which can identify movies and TV shows. Finally, IntoNow was a former application which was used to identify TV shows and movies but has been taken off the market.

**2.2.1 IMDB – Internet Movie Database**

The Internet Movie Database (IMDB) is one of the most popular online databases relating to movies, television shows and video games. It is available on the web and on multiple mobile operating systems. It contains information about the cast, director, release date, plot synopses and other details relating to different titles. As of 24th of November the IMDB website was an Alexa Top 50 website. ([r] Alexa top 500 global sites, November 2015) On the 17th of October 1990 the company saw its first signs of life, when the founder Col Needham published scripts which could allow people to search through lists of credits. Today, the website welcomes over one hundred million users a month. ([r] IMDb History, 2015) IMDb contains over 3.5 million titles of movies and TV shows, this includes episodes of TV series. It has over 6.9 million personalities in its database, credits over 70 million and has 64 million registered users. ([r] IMDb Database Statistics, November 2015)

IMDB allows its users to search for movies, TV shows and actors to view details of the movie, TV show or person that was searched for. From a movie or TV show’s page, a user can click on an actor and go to that actor’s profile. On an actor’s profile a user can browse details of all of their acting roles, along with other details about them. The user can then click into one of the actors movies and view the details of that particular movie and continue on from there. IMDB supplies its users with trivia and other fun facts about the person, movie or TV show’s page the user is on, along with viewing related titles that the user may find interesting. Using IMDB’s search function users can search for Movies, TV shows and actors using titles or the name of the actor. It does not allow for searches based off of quotes.



**2.2.2 Shazam Entertainment**

July 10th, 2008, the world was first introduced to the Shazam mobile application, one of the first apps available on the AppStore, but the company has been around for much longer than that. When the company first started operating a user would dial 2580 and call the number. They would then hold their phone up to the speaker where the song was playing and after thirty seconds the call would automatically hang up. The user would then receive a text message containing the song title and artist name. A user could pay a monthly fee for the usage of the 2580 number or they could pay per tag. ([r] Cooley and Lim, 2006) Today, the user experience is much different although the fundamental usage remains the same. A user opens the Shazam application and on the home page of the application is the Shazam button. The user presses this button to tag the song they are currently listening to. Shazam uses audio fingerprinting to be able to identify music recorded by a user. A user records a short segment of a song which is fingerprinted by the app and compared to Shazam’s database to return the song the user is listening to. When the result is returned the user can see the song title, the artist, the album the song is from and a link is provided to purchase the song on iTunes or on the Google Play Store depending on whether the user is using iOS or Android. ([r] SHAZAM COMPANY, 2015)

Recently, Shazam Entertainment have brought a new feature to their famous application. The new feature allows users to tag the television show they are watching. With Shazam’s affective use of audio fingerprinting, the app returns details of the show. The users view the cast of the show and the latest tweets about the show. Shazam have partnerships with most of the national television channels in the United States allowing them to fingerprint the shows that are airing live on those channels. This means, that Shazam only returns a match if the show the user is watching is currently being broadcast. The application will not return any results if the show is not being broadcast on one of the channels that have a partnership with Shazam. Shazam don’t currently have any partnerships with television channels outside of the US so unless a user has access to US national television channels they will not be able to use this feature. ([r] Ha, 2012)

Shazam also have a relatively new worldwide feature for advertisement tagging. Shazam Entertainment have made some partnerships with companies. When an advertisement from one of these partnered companies is shown on television, a Shazam logo will appear signifying to viewers to tag this advertisement using Shazam to find out more information on the product being advertised. Companies like Calvin Klein and Coca-Cola have been incentivising the use of Shazam to their customers. In the case of Calvin Klein customers can walk up to one of their branded podiums, take out the Shazam app and record the current song playing. In doing this simple interaction, the customer can receive exclusive content for their mobile or in-store discounts and promotions. ([r] O’Brien, 2011) Coca-Cola recently had an advertisement on television which was released in partnership with Shazam. A user could tag the ad getting themselves a free drink of Coca-Cola in the process. Other companies enter users into draws to win prizes, give away coupons or discounts.

**2.2.3 Viggle**

The Viggle app allows its users to ‘check-in’ to what they are watching. A user collects points when they check-in to a show and then collect more points the longer they watch the show for. Viggle has in-app quizzes about shows currently on air which users can partake in, to win more points. The quiz questions are regarding events that are unfolding throughout the show the user is watching which keeps the user watching the show. The App’s “Whats On” function displays the current listings on the channels that Viggle users have the ability to check in on. Viggle has a marketplace where users can spend their points on gift cards, movies, Viggle merchandise or even a cruise. ([r] Cunningham and Weinel, 2015)

Similar to Shazam’s television feature, the Viggle app is only available in the US and is also limited to certain national channels. Unlike Shazam however, it allows users to check-in to some shows that aired within the last seven days. ([r] *Does Viggle work with recorded TV shows?*, 2015) This is a much better improvement to Shazam’s restriction of live television but, it still doesn’t have the capacity to match television shows which haven’t aired in the last seven days on their selected television channels.



**2.2.4 IntoNow**

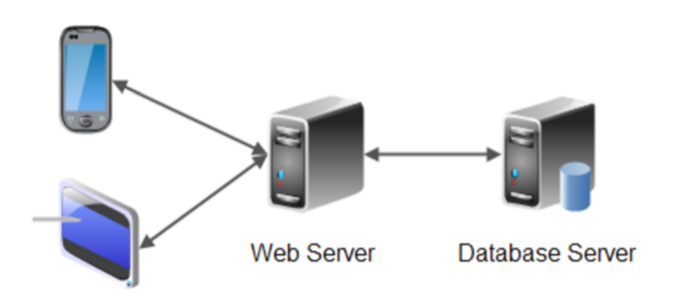
IntoNow was a previous solution to the problem this project is trying to solve. IntoNow was founded in 2010, just twelve weeks after it was released it was purchased by Yahoo for a hefty sum in the region of $20-30 million. The idea of the application was similar to Viggle, a user can tag a show they are watching and sends it to their friends so they know what the user is watching. IntoNow, like Viggle and Shazam used audio fingerprinting to detect the TV show or movie that the user is watching. Unlike Viggle and Shazam however, IntoNow had a database which it was constantly adding to which would allow users to watch recorded TV shows and movies and still have the ability to check-in as watching them. ([r] Siegler, 2011)

Unfortunately for IntoNow users, it did not last. In early 2014 Yahoo took IntoNow off the App Store and the Google Play Store. Users were sent emails to tell them that the app would stop working two months later. Although the application itself is gone, the auto content recognition (ACR) technology lives on in some other Yahoo applications like its smart TV app. ([r] Lawler, 2014)

Chapter 3. Technology Used

**3.1 Introduction**

At the heart of examining technologies to use in a project is choosing an architecture for the project. This examination led to the realisation that a three tier system architecture would be necessary. The figure below shows how a three tier architecture works at the most basic level. The mobile application will be on tier one, tier two will comprise of the server and scripting language and tier three will contain the database.



**3.2 Tier 1**

Tier 1 involves the front end development of the project. With mobile development the choice between developing a native or a hybrid application is first up.

The first option is a native Android application. This would limit the amount of platforms the application would be available to Android alone, which wouldn’t be ideal, but developing an Android application natively would provide a better look and performance. As most applications on the Google Play Store are made specifically for Android, users have an expectation when using an Android app. They have become accustom to the native platform which they would find easy to use as it is what they are familiar with. It is not just Android users, iOS users have a similar experience with applications available on the App Store. They have come to expect a style of application for iOS users. When using a hybrid application it is harder to satisfy users on all platforms. ([r] Korf and Oksman, 2015) Due to lack of experience and not owning an Apple Mac, a native iOS application was ruled out early on.

The hybrid application option is the PhoneGap framework. PhoneGap allows a user to use standards-based Web technologies HTML5, JavaScript and CSS3 to create cross-platform mobile apps. ([r] R.M.de Andrade et al., 2015) The variety of devices supported by PhoneGap are a large pull factor in choosing the framework “PhoneGap supports seven mobile platforms (iOS, Android, HP WebOS, BlackBerry OS, Windows Phone 7, Symbian, Bada); apart from Web apps this is the highest number of different platforms” ([r] Heitkötter, A. Majchrzak, and Hanschke, 2012). Figure 3 below shows the PhoneGap Architecture, PhoneGap uses native API calls to access mobile features like the camera, and the user’s geolocation.

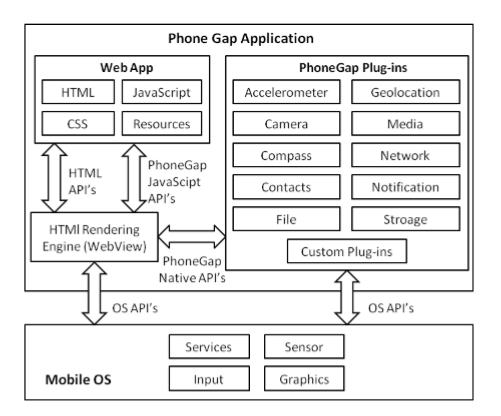


Figure 2 Phone Gap Architecture (Palmieri, Singh, and Cicchetti, 2012)

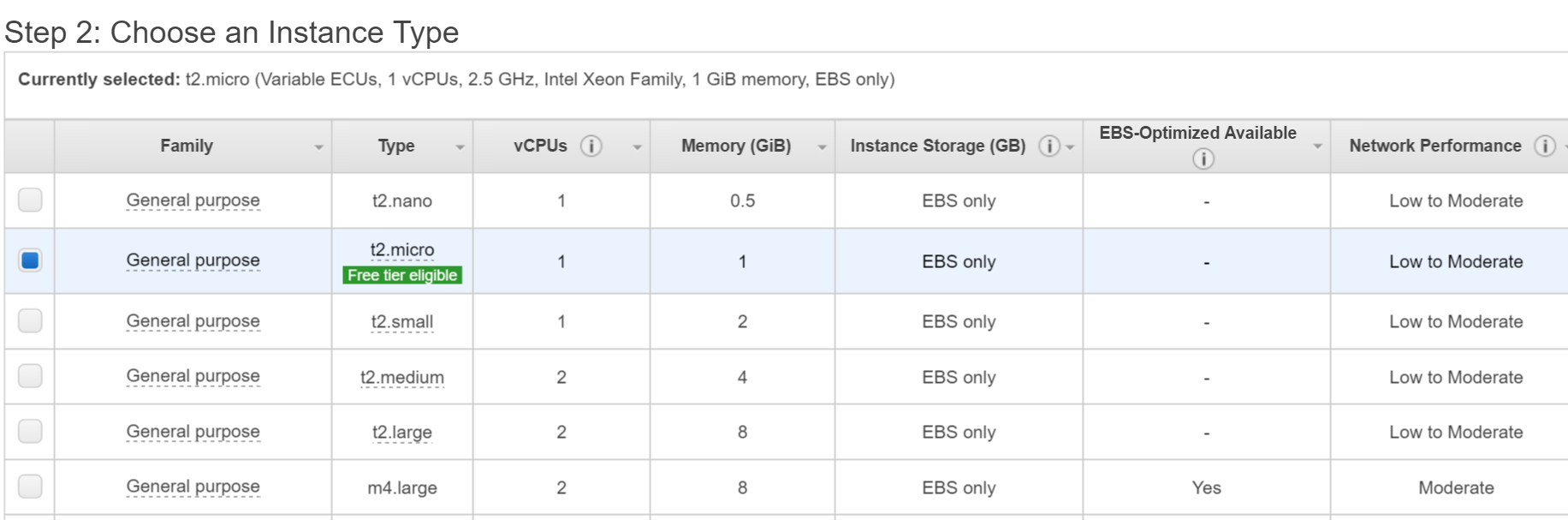
Phonegap was chosen as the framework for the project, mainly due to the want of creating a hybrid application and extensive knowledge of web development. Although the hybrid application may suffer somewhat in performance when compared with a native application, the benefits of using PhoneGap outweigh the negatives. The application will be made using HTML, CSS and JavaScript. For responsiveness and layout Bootstrap and AngularJS will be used.

PhoneGap has a mobile application which can be used to connect to the server that a PhoneGap project is on. This allows the developer to view the results of their changes as soon as they make them. This was a big advantage as it has aided in the progress of the development stage of the project. It was much easier to spot when an error has been made.

**3.3 Tier 2**

Tier 2 contains the server and the scripting languages. The server will need to be accessed every time the app is in use, therefore a reliable server is needed for the project.

“*Amazon Web Services (AWS) is nowadays the leading public Infrastructureas-a-Service (IaaS) cloud provider in terms of number of users, allowing resources in their data centers to be rented on-demand through Elastic Compute Cloud (EC2) service.” ([r] Expósito et al., 2013)*



**Figure 2: Choosing the size of an EC2 Instance**

As can be seen from figure two, when creating an instance AWS gives you the option of the size of your EC2 instance. After this choice has been made a user pays for the size of the instance used. As the size of the instance goes up, so too does the price. If an instance is receiving a lot of traffic it can be scaled up to a bigger size to cater for the load. It can also be scaled down if the traffic across the instance doesn’t require a large instance. EC2 “provides resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.” ([r] Amazon EC2 - Virtual Server Hosting)

EC2 also allows for scaling out automatically which is aimed at systems with fluctuating traffic. Auto Scaling Groups can be used to trigger the launching or termination of EC2 instances upon exceeding defined thresholds. ([r] A Comprehensive Guide to Building a Scalable Web App on Amazon Web Services)

This project will use an Elastic Compute Cloud (EC2) instance on Amazon Web Services, mainly due to its scalability which can be used to increase or decrease the capacity as needed. This means that if the system grows in popularity and usage the capacity can be increased to accommodate for the added users. EC2 also offers a free tier which can be used in development with no charge. The EC2 instance will be located in a Virtual Private Cloud which provides security tools so that the instance is secure. ([r] *Elastic compute cloud (EC2) cloud server & hosting – AWS*, 2010)

Apache was the clear front runner in the choice for webserver, as it is the most popular webserver with plenty of documentation and information about it freely available on the internet. To provide an interface between the Apache webserver and scripting language, the Web Server Gateway Interface (WSGI) was chosen. This also dictated the choice in programming language as WSGI is used for communication between Python’s web frameworks and webservers, which makes it possible to handle HTTP requests using Python code. ([r] OpenStack OCCI interface) The Python web development framework Flask will be used to communicate with the Apache server which is hosted on an Amazon EC2 instance.

**3.4 Tier 3**

The final tier contains the systems database. This is the tier where all the data is stored by the system and where it can access data when required. Tier 2 will communicate with tier 3 when the system needs to access or post to the database.

Amazon Web Services ecommerce platform, receives tens of millions of daily customers. This high level of demand means that Amazon require their services to be reliable, as any outages could cause a huge financial lose to both Amazon and their customers which could cause a loss of faith in Amazon’s services by customers. To combat this Amazon offers multiple database management alternatives, that are extremely reliable which makes hosting a database on Amazon Web Services advisable. ([r] George and Mathew, nd) The cost of Amazon database management services is relatively low due to users only paying for what they use. Also, Amazon Web Services offers a free tier for the first twelve months, which gives users free limited usage for new users. ([r] Amazon, nd)­

Amazon Dynamo DB, the fast, fully managed noSQL database offered by Amazon Web Services. Dynamo Db provides fast and predictable performance. Data in a DynamoDB database is store on solid state disks (SSD) and are automatically replicated across multiple Availability Zones in an AWS region. This provides the user with built-in high availability and data durability. ([r] Varia and Mathew, 2014) However, this project has the need for a relational database, DynamoDB was not a feasible option as it does not support joins which will be needed from the database chosen for the system.

Amazon Relational Database Service (RDS) was another option which is also on Amazon Web Services. RDS offers a MySQL database engine. RDS comes with Multi-Availability Zones which are automatic backups, these backups are put on a different Availability Zone than the one a user has their original copy of the database on. This is so in the unlikely event that the availability zone in which the user’s original database is on goes down, their backup will not be affected and the user will not lose any data. ([r] Laurence, 2013)

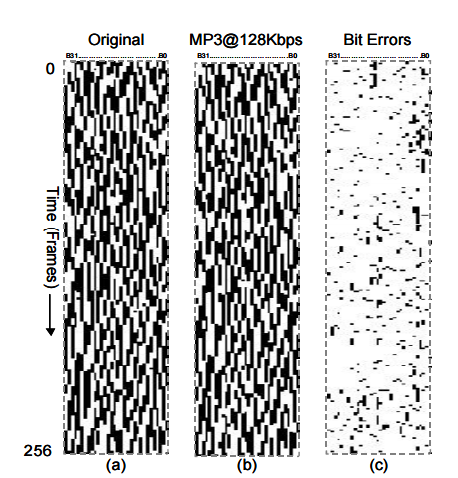
“MySQL is the most popular one of all the large-scale database servers. It is a feature rich, open-source product that powers a lot of web-sites and applications online.” ([r] SQLite vs MySQL vs PostgreSQL: A Comparison of Relational Database Management Systems) MySQL can be easily installed on a machine and can be used in conjunction with Python. MySQL is known for its scalability and speed on par with several other databases. It also offers different storage engines with different capabilities like rollback of data for example. (MySQL in a Nut Shell)

Ultimately the decision was made to use MySQL, due to having three years experience with the technology but also because MySQL is free to use. If a database option on AWS was chosen, the early stages of development and releasing of the application it shouldn’t cost too much to have the database running, but if the application were to become popular the rise in cost for the use of the services on AWS will rise also.

**3.5 Content Recognition**

**3.5.1 Audio Fingerprinting**

Audio fingerprinting is a fast and affective way of automatic content recognition. Using a spectrogram, points are plotted on a time-frequency graph. The graph represents the intensity of the frequency at certain points during the audio. These points of intensity are used to create a hash table with frequency as the key, which works as a fingerprint. As mentioned above Shazam uses audio fingerprinting to identify songs in their database from a clip sent by a user. When Shazam receive an audio clip from a user it is converted into a hash table. Each hash value is searched through all matching songs in Shazam’s database, this typically returns multiple songs, once more hash values have been searched, a pattern will begin to emerge until such a point that Shazam can return the song it has determined is the one the user is looking for. If a distinct pattern is not evident from the finger prints Shazam will return that it can’t find what the user was looking for. Shazam has partnerships with the major music labels so that they can enter the newest songs into their database as soon as they are ready for release. (Jacobs, nd).



(a) Fingerprint block of original music clip, (b) fingerprint block of a compressed version, (c) the difference between a and b showing the bit errors in black.

Due to the numerous frequency points per second in the hash tables, audio fingerprinting proves affective with background noise and distortion, to an extent. This means that apps like Shazam can still generate correct matches from poor quality recordings. (ANIL ALEXANDER)

Using audio fingerprinting for this project would mean users could record a segment of a TV show or movie and, segment would be hashed and matched against a database of hash tables relating to different movies and TV shows. The main issue with audio fingerprinting is the size of the database that would be required for this project. (Peter Jan Otto Doets) The proof of concept is on the market currently with apps like Shazam and Viggle using audio fingerprinting to match recordings with results, but neither use a database large enough to specifically cater for the audio fingerprints of every movie and TV show created.

**3.5.2 Speech to Text**

Speech-to-text has become a basic feature in mobile phones with iPhones Siri and Androids Ok Google function users can enhance their mobile experience by speaking commands to make their mobile phones execute tasks. In Siri’s case, users can use the program to set timers, alarms, check the weather forecast, locate their current position and find the nearest restaurants. All a user needs to do is press record and vocalize their question or request and Siri will use speech-to-text to analyse the request. The user’s request will appear on the screen in text form and Siri’s reply will appear below, Siri will also reply using text-to-speech. As mobile phones tend to have small processors, most of the speech-to-text process is carried out on a server, so when a user speaks a request into their phone’s microphone, the data is sent to a central server where it accesses the software and databases. (Alexandra Ossola, August 15 2014)

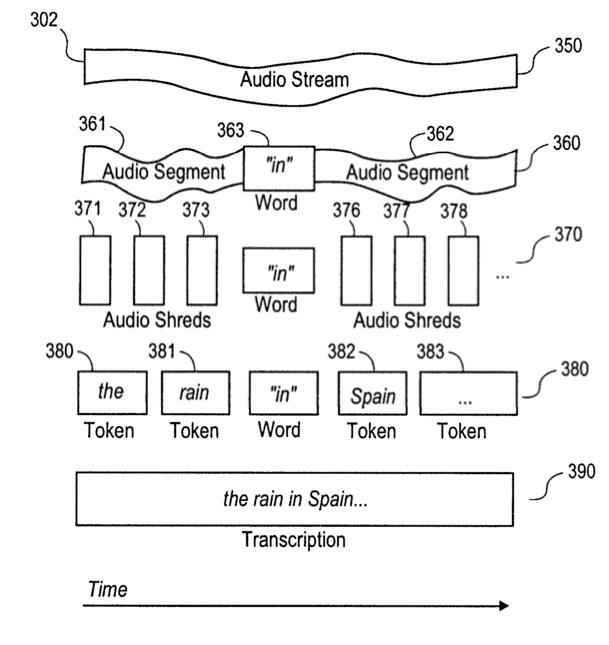


Figure 1 Exemplary Data Stream (Jon Jaroker May 27 2014)

NASA use speech to text on videos on their JPL Tube video sharing network. When a user is looking for a video they search for a sentence in the transcript and the returned results are videos with that sentence in them. When a user uploads a video to JPL Tube, a transcript is generated. This transcript is indexed so that it can be searched. When a user search returns a result midway through a video, the video will start at that point. This means a user doesn’t have to watch an hour of the video before getting to the part of the video they are interested in watching. A user can edit the transcript if the speech-to-text didn’t pick it up correctly. (Office of the Chief Knowledge Officer, NASA, 2014)

The primary issue with Speech to Text software is its robustness. Most systems require the speaker to be close to the mike so as to avoid any interference from background noise. Background noise and distortion can cause major discrepancies in the resulting text. (Noise Robust Speech Recognition) Subtitles on live TV broadcasts are typically done using speech to text, but a re-speak method is used to combat the background noise issue. In the re-speak method, a second reader is used to repeat what the first reader has spoken. The second reader will speak clearly into a microphone so that speech to text can be performed accurately as background noise has been eliminated. There is a necessity for the second reader due to likely issues from running speech to text on the first reader’s speech. It is because of this, that until speech to text technology has been greatly improved it can’t be used as a means of scripting speech straight from TV. (Real-Time Closed-Captioning Using Speech Recognition)

After weighing up the positives and negatives of both methods of recognition, Speech to text will be used for this project. The database size needed for audio fingerprinting is implausible for this project. A user will say their quote after clicking a button on the application, this speech will be picked up by their mobile’s microphone and speech to text will be performed giving a block of text. To combat any discrepancies in the speech to text system, a text box will appear which is prepopulated with the text from the user’s speech. The user can edit this text before sending it to be matched with a TV show or movie.

**3.6 API**

The OMDb API will be the main data source of the project. To retrieve data about a TV show or movie the OMDb API will be called. The API has been chosen as it returns detailed movie and TV data to display to users when a match is found. If the resulting match is a TV show, OMDb can specify the season and episode number of the show to get more precise information about the particular episode that the user is watching. The API returns a JSON object which can be used to separate different pieces of data about the show or movie. For example “data.title” will return the title of the show or movie. (Johann Schaible1 et al. July 15 2015)

**3.7 Conclusion**

After all relevant research has been completed the requirements for the project are as follows. The PhoneGap framework which I will use to create the front-end of the project. Using the hybrid technology I will create an application across multiple platforms. Amazon RDS, the relational database system to store data in a scalable relational database. Amazon EC2, an elastic compute cloud instance will be used to run a server which will handle the scripts. The OMDb API which will be used to collect TV show and movie specific data which will be used in the application.

Chapter 4. Design

The goal of my project is to display details of the television show or movie that a user is watching. To do this I plan on using speech-to-text of a piece of audio recorded by the user. The user takes out their device and presses the record button. Speech-to-text is used to get all of the quotes from the show or movie. The reason why I have chosen Speech-to-text over fingerprinting the audio is due to feasibility. To use audio fingerprinting I would need to have a database full of every television show and movie whereas with Speech-to-text I can search the text to come up with the result.

For the searching of the show or movie I will use a google search and take the first three returned results. I will do this repeatedly for each sentence, I will also add sentences together with a maximum of three sentences and use them to search also. From these multiple searches the most common result will be taken. The most common result will be used as the resulting TV show or movie.

If a definitive result can’t be found the user will be shown an error message. The error message will tell the user that their search was unsuccessful and if they wish to retry to click the record button again. This will be the default page for when a search does not turn up with a result.

When a result is returned the OMDb API will be called which will return details about the show or movie. The details will be in the form of an object, the object will be used to fill in the details into a results page which will be displayed to the user. The API contains details about the show from a plot synopsis to the cast, it also has a link to the poster for the T.V. show or movie. I will use a database to enter the details of the show into so that searching will be faster in future. The database will be searched first to try find a match and if a match can’t be found it will use the API to get the details. In the database there will be an index for recommended shows. If a user searches for two different shows they will be indexed. Each show or movie will have a top five recommended shows or movies, this will be the shows and movies that most users who have searched the original show or movie have also searched for. The user will be able to click on the titles and view details about them.

The user will also be able to view the most popular TV shows and movies of the last week, month, year and of all time to see what other users have been searching for. The application will get this data from the database as it will store how many times each show has been searched for. This will show the trends of different TV shows and movies for the user and will also allow the user to view details on these popular shows.

# Approach and Methodology

I am approaching this project using the personal scrum method. I will have weekly sprints for a small scope on putting the most important work first. With weekly deadlines I will keep myself motivated to meet my weekly goals. If one of my goals hasn’t been met at my weekly sprint it will be moved up in priority. Each task will be given a timeframe of how long I should be spending on that task during the coming week. As I am working on that task I will note the amount of time spent on it and will write down the total time taken for that task at the start of the next sprint. The reason for this is so that I will be able to make better estimations of how much time should be allotted for tasks in the future when I see the actual time taken for previous tasks.

The reason why I have chosen the personal scrum method of development is so that I can keep on track of the tasks that need to be completed using prioritization. I can use sprints to make sure I am meeting my weekly targets and progress with the project at the level I want. I will be able to see if I am falling behind in some areas. With this information I will be able to determine whether or not I will have time to add in some extra features for my project. Although there will have to be a time allowance made for a weekly sprint, it will be worthwhile as it will keep the project on track and allow for more accurate future planning.

If I have time I plan to add some features allowing users to post the show they are watching onto Facebook, Twitter and some other social media websites. This will allow user’s friends see what they are watching and allow them to click into the app to view information about the show that their friend is watching. As this is not of main priority for the apps functionality if I don’t have time to implement it I will leave it out. The app will still function fully without this feature which is why it is expendable. Using the personal scrum method I plan to be able to a lot a time for this feature but if other tasks which take priority over it take longer than planned I will know that I won’t have enough time to implement this feature and can take it out.

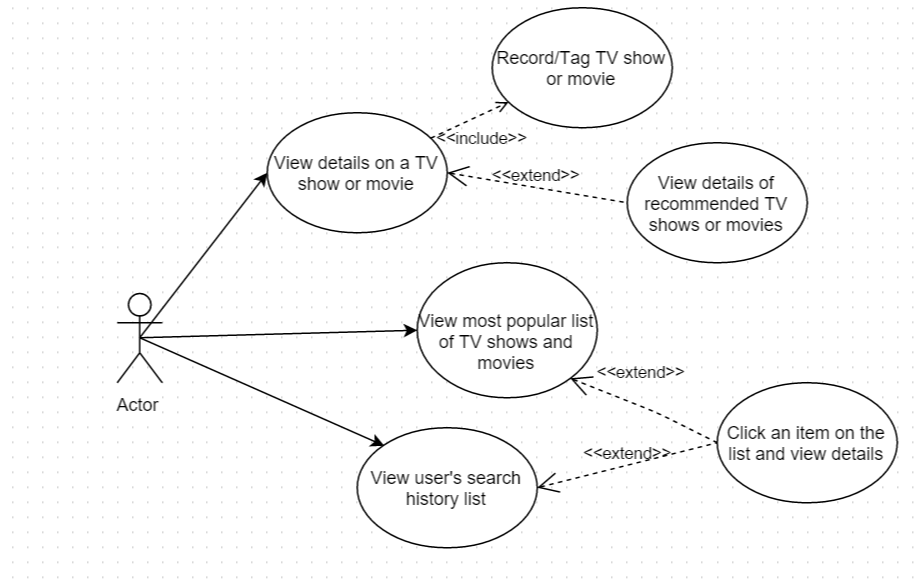
# Design

## Technical architecture diagram:

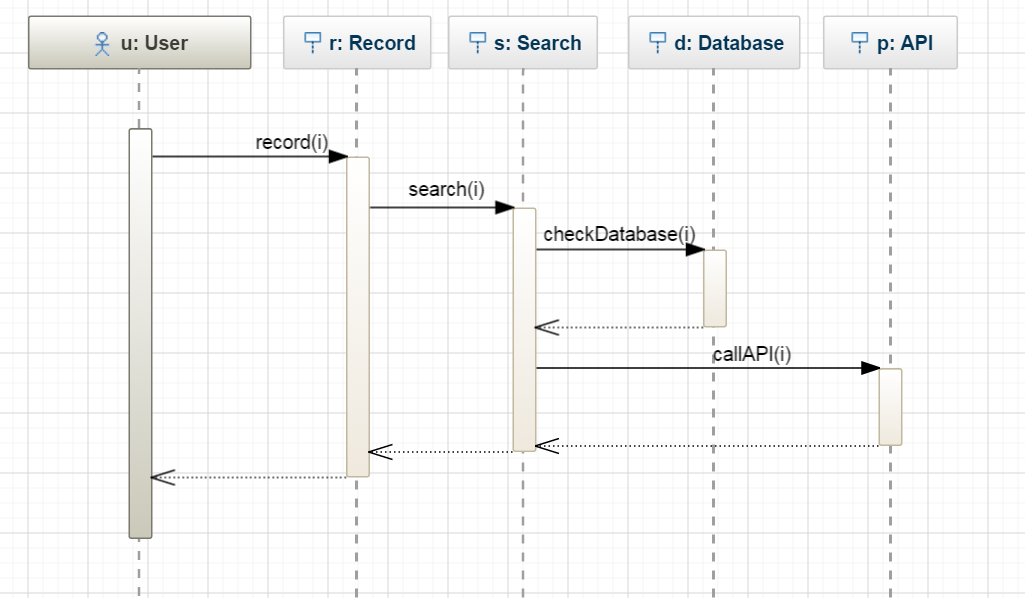
## 

## Other design documents

Use Case Diagram

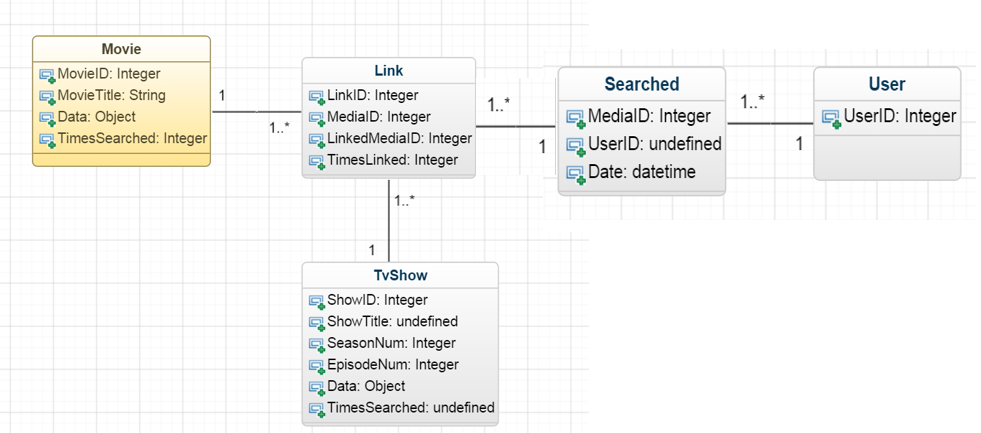


Sequence Diagram



The sequence diagram shown above shows the interaction of a user when they tag a show or movie. The user will press the record button, which will record the show or movie. After speech-to-text has been done on the recording the text will be searched to return a result. When a result is returned the database will be checked to see if data is already in the database for the result. If there is no data for the result, the OMDb API will be called to collect data on the show or movie.

Class Diagram



In the project a user can make many searches, these searches are time stamped so a user can view their most recent searches. If a user searches for the same show or movie that they have searched for before, the time stamp is updated. When a user searches for a show or movie, this show becomes linked with the other shows and movies that the user has also searched for. When a user searches for a show and receives a result, a recommended section will show up. In this recommended section will be movies and TV shows with the most common links to the show or movie that the user has searched for.

TV shows and movies will be separated in the database so that TV show can account for season and episode numbers. Both tables will have a TimesSearched variable, this variable will be used to create the list of most searched for which the user will be able to view on the application.