**VideoCoder**

**Aria Project**

**MSc Web Technologies 2014**

Javier Rossetti – x12114952

Gerard Daly - x12114804

Paul Mannion – x12110744

Sanjeev Kumar – x00056042

Code: <https://github.com/javiros/arial_ca3>

Live: <http://videocoder.herokuapp.com>

Table of Contents

2 VideoCoder - a hosting Application for Learning Channels. 3

2.1 Motivation 3

2.2 Project Scope 3

2.3 Area of Contribution 4

3 State of the Art Review 6

3.1 Current Practice 6

3.2 RIAs and Learning 6

3.2.1 MOOC – Massive Open Online Course 7

3.2.2 Code Schools 7

3.3 YouTube Coding Playlists 9

3.4 VideoCoder 10

4 User Interface Design 12

4.1 Desktop 12

4.2 Mobiles and Devices 13

4.2.1 Navigation 14

4.2.2 Videos 14

4.3 Charts 15

4.4 Some Tools that were tried 16

5 Architecture 18

5.1 Application Architecture 18

5.2 Security 18

5.3 Toolkits and Frameworks 19

5.3.1 AngularJS 19

5.3.2 HTML5 21

5.3.3 CSS3 21

5.3.4 Bootstrap 21

5.4 Data Transfer Strategies 22

5.4.1 API – Application Programming Interface 22

5.4.2 JavaScript 24

5.4.3 JSON - JavaScript Object Notation 24

5.4.4 CanvasJS 25

6 Evaluation and Testing 26

6.1 Deployment 26

6.2 Usability 26

6.3 Testing 27

7 Summary 28

7.1 Problems & Solutions 29

7.2 Further Development 29

8 Appendices 30

8.1 Appendix I: Desktop 30

8.2 Appendix II: Mobile 36

9 References 37

# VideoCoder - a hosting Application for Learning Channels.

## Motivation

Rich Internet Applications (RIAs) (Application Programming Interface)today enhance many diverse user experiences both online and offline. They have come to dominate user interaction in the fields of education, gaming, news, entertainment and business with hundreds of millions of users throughout the world.

The RIAs in the field of education have become extremely sophisticated, allowing interactive and often personalized training through the Internet. eLearning is now integrated not only within educational institutions and businesses but is used by millions every day for self-motivated learning in an area of interest.

Open source software has become very popular in particular JavaScript, JS frameworks such as Jasmine (for testing), Backbone, Angular, and other scripting languages such as Ruby and the framework Rails, this has resulted in a myriad of sources both in the form of tutorials and videos. With vast number of online resources it could be difficult to obtain a good starting point. VideoCoder has been developed to try and help developers with a starting point in the technology of their choice; whether it is at a junior level or at a more advanced level, VideoCoder has been designed to suit these different levels of knowledge. It has been created by developers for developers.

There has been an explosion in the amount of material now available now for anybody who wants to learn more about creating with code. Whether to create applications, web sites, blogs, games or anything else, often RIAs are providing great interactive learning experiences that for many are the starting point as they explore these areas.

Watching a demonstration of coding where the learner can follow along at their chosen pace, see the expected results and any potential errors, is one of the best ways to enhance learning and for making progress. Online videos are often the first port of call for learners looking for guidance in an area whether they are at a beginners or an advanced stage. As YouTube consists of many thousands of channels catering to these needs it provides countless opportunities to find the required lessons for the area of study or interest.

Although the quantity may be very encouraging to the novice, the quality of delivery, explanation and results can vary considerably. Potentially the learner may not be finding the best instruction channels available that would be most suited to their pace of learning. The result being they may become quickly discouraged within a short time when progress made is not as expected.

VideoCoder has been designed to filter the most successful learning videos directly from YouTube to the learner. By providing channels for specific areas of coding that are linking to highly recommended playlists. This filtering of the YouTube content that has proven to be most successful and popular, for all levels, will allow a learner to quickly find the best content for their learning.

## Project Scope

VideoCoder will help to enhance the learner’s workflow by providing the most recommended and most viewed content. Although interaction with technology now is seamless for most users and the process of finding what is required is very fast, the vast volumes of material found can be time consuming for the individual to filter through, especially if they are a novice. This could be may be off putting for some as finding the appropriate quality of delivery and content takes some time.

The creators of VideoCoder will be familiar with some YouTube channels being very successful for learners and worthy of inclusion. Having focused on the learning of coding skills over the past few years, they are accustomed to searching YouTube for the most suitable content and finding the most appropriate delivery. Technology Blogs and Coding Schools also provide a lot of recommendations for learning content and these provide a valuable trusted source of quality video content.

When a learner is using VideoCoder they will find channels for each of the most popular coding languages and frameworks. Each channel will filter recommended video content from the YouTube API (Application Programming Interface).

A pie chart will display the total amount of views of the top ten videos in the channel. So the learner can quickly gauge which of these selected videos are most popular. Each video will also display the ‘date published’ as this is quite relevant in the area of coding and the total comments figure. This at a glance data will provide the learner with a very efficient learning environment.

As a portal for learning, VideoCoder is providing a place for practicing of code and getting hands dirty. This is important and understanding of concepts most often requires active exercises. There is also a resources page of links to the most useful external learning sites.

VideoCoder is designed responsively as a lot of learners will access their learning on mobiles, on the go.

VideoCoder is entirely free to use for learners, as it is basically a portal for delivery of free content from YouTube. Also as VideoCoder itself is not hosting any video content, the running costs are low.

## Area of Contribution

Users of the Internet have a growing interest in how it is working and how they may contribute. Increasingly, many areas of employment rely on the Internet for essential services, trading and maintenance of the business and there is expectation that employees have a good knowledge of not only using the Internet but also an interest in its potential for a business.

There is large demand now for skilled coders to develop and design great web applications that will keep a business competitive and profitable in today’s market. Every business today knows how vital an online presence is. This applies especially to how current and potential customers receive the news on what a business has to offer through social media and advertising through relevant online channels. Coding is gradually becoming an essential skill requirement. There is now an often heard mantra that children need to be learning not only how to read, write and do maths in school, but also to learn how to code.

VideoCoder is a starting point for learners looking for the best YouTube ‘how to code’ video content in one place. The sheer volume of video content in this field is daunting to the beginner who is searching for a solid introduction. Initially they may find the concepts and rules of a code language to be quite abstract and need reassurance by seeing exciting results fast. Maintaining motivation is difficult for learners if they are uncertain.

The quality of videos can be measured by:

* How engaging the presenter is, their personality and quality of speech
* The volume levels, as sometimes videos have a low volume level
* Quality of resolution is very important when learning to code as very small details may be of vital importance to the successful running of a piece of code
* Whether the channel is using zoom and highlighting content
* Whether the content is clear, concise and always on topic
* Whether the sequence of steps delivers the expected result in a reasonable time.

Watching inappropriate content can waste a lot of a learner’s time and also patience, resulting in progress being stifled and perhaps not as rewarding as the learner is expecting. It can be frustrating for the more advanced learner who is intending to make progress but the content they find is not quite relevant or properly explained.

Searching through Google and YouTube itself will of course enable those with some knowledge and determination to find the relevant content, but it is also word of mouth and trusted recommendations from blogs and coding schools that can guide the learner appropriately to gain the most in as little time as possible.

Watching videos on YouTube delivers a lot of related content on the right side of the screen. Depending on the level of knowledge in an area this can be helpful for some learners and for others basically a distraction. These randomly selected suggestions not being available on VideoCoder allows the learner to focus primarily on the recommended channel only.

The channels provided through VideoCoder may be seen as the beginning of a productive and managed learning path for users. They should help the learner quickly focus on the appropriate video content for their needs.

# State of the Art Review

## Current Practice

The domain of eLearning has experienced exponential growth since the arrival of Web 2.0 in 2004. While previously seen as a mere substitution for the teacher and the class room, with delivery of slides often followed by basic multiple choice options, today Rich Internet Applications (RIAs) have fundamentally changed the e-learning experience.

With the arrival of social media, highly integrated role playing and gaming, together with access to the web on not only desktops but handheld devices, learners today have become accustomed to an online experience that is not only highly interactive but intuitive, fun and fast. The fundamental technologies used over the past 20 years have developed to enable a vast amount of media information be available for the user to interact with instantly and for various states to be retained as required. There are now frameworks and libraries available to the developer that integrate (almost) seamlessly with others and that enable the fast production of reliable, secure and attractive RIAs.

Exciting interactive experiences have encouraged many to approach online learning. Users who may not take to the traditional methods of following a book, class tutorials, of even using a desktop now have the opportunity to explore various new options that can be modeled to suit the way they want to learn.

Current practice is to develop with the user in mind at all times. The objective of the user is to learn a new skill or expand on their knowledge and the objective of the RIA developer is to provide the necessary tools in the best way possible.

The benefits of eLearning are many and to ensure that the experience delivered is satisfactory, the development of RIAs must ensure:

* There is always ease of access.
* That the UX is attractive and not distracting
* That the collaborative tools function on all devices
* That there are Learning paths a user can follow as guidelines.
* That challenges are present to guide and motivate while skill building.

With the correct approach the user can benefit in ways different and unexpected in comparison to how they might learn otherwise. They can measure their progress with workflows and tools that can be used to assess their learning.

More often now, interaction is encouraged. Applications are looking for the learner to contribute through joining in with forums and sharing their knowledge with others. To engage in Questions and Answers that may even be in real time.

## RIAs and Learning

As the domain of technology in education is widening at an ever faster pace, it is important to understand the distinctions with how certain approaches to learning objectives have succeeded where others have not.

There are different groups of learners to target, which may have distinct dynamics, interests, needs and concerns. Potentially a RIA could be designed and developed to be appealing as a learning tool to as broad a group of people as possible but often it is important to initially engage with a particular demographic or type of learner. There are many ways now to do this.

### MOOC – Massive Open Online Course[[1]](#footnote-1)

The MOOC has become a popular method, especially in higher education, of allowing a student to follow a comprehensive learning path. By providing different strategies depending on the learning goals along with strong interactivity and support, MOOCs have become very popular on desktop. ‘Future Learn’ is a MOOC offshoot of the Open University, an online portal for different learning institution worldwide to provide videos, coursework and questions in a stepped learning path where students can interact with each other and with teachers.

While here in NCI, a successful hybrid model of MOOC has been in place in a computer science course. Homework entails following lectures online and the time in the classroom is enabled for hands on guidance through exercises with the teacher.

### Code Schools

Recently the audience for eLearning has grown very quickly thanks to the variety of dedicated Coding Schools available on not only desktop but also all devices. There is clearly a large audience today for learning to code, as understanding how to create and maintain a web site or application has become an integral part of any business, event or organization.

Coding schools offer an often similar experience for the learner but, as with all relatively new methods of delivery, they can differ in quality and variety. They allow the absolute beginner feel comfortable and not discouraged by complexity or the command line at an early stage.

The predominant tools used are:

* Video classes followed by multiple choice questions
* Stages of perhaps 10 steps, that enable a breakdown of fundamental concepts to be explored singularly and then in tandem with other concepts. This is then followed by multiple-choice questions of which a minimum amount of correct responses is required to proceed further.
* Drag and drop elements onto a canvas that will indicate a correct choice.
* Filling in the missing gaps on a code sample with exactly the correct answer or code characters in order for the code to function and proceed.
* Following a predefined learning path that could complement other paths as skill sets are built upon.
* Employing a style of game role-playing where points accumulate for each correct choice or move. Badges and stickers may be awarded which are visible on a learner’s avatar.
* Providing limited tips or hints if the learner requires.
* Providing a complete answer if the learner is really stuck, however this may result in a decrease of overall points.
* Forums are employed more often now as eLearning schools increase in popularity and learners who have progressed are eager to engage with others. They are also a place where beginners can seek advice and help from others if they are really stuck.
* Some eLearning schools market a fun image with quirky presenters and cute logos.

#### Team Treehouse[[2]](#footnote-2)

Team Treehouse is a very popular eLearning solution with many thousands of daily users. Once enrolled for a fixed monthly fee, the learner has a ‘mission control’ where they can view all the progress they’ve made, their total points and any areas of learning that they had started but are yet to complete.

Video plays an integral part in each learning ‘track’. These are downloadable in high quality along with the verbatim content and any necessary examples of code. The tracks are broken down into manageable parts allowing a learner to easily revisit or skip as required. Team Treehouse excels at project orientated learning.

Integration with other learners is encouraged and any learner can start a discussion or help out others who may be having problems with an area that they have already completed successfully.

Over time learners are expected to continue to build on what they have already learned and to see how all of the different skillsets play integral parts in an application’s design and development.

‘Workspaces’ are similar to native text editors used in coding, and provide online storage space of up to 50mb for projects. Learners can have up to 10 workspaces created at a time. A workspace can be forked as is common practice on a version control platform like Github, it can also be easily deployed within the Treehouse framework.

#### Code Academy[[3]](#footnote-3)

Code Academy relies primarily on the live and interactive coding console that it provides beside a left hand panel of progressively more difficult questions. The learner is encouraged straight off to dive into the console, get there hands dirty and look for hints and warnings as they go. This site has very well structured courses that motivate the learner. Similar to other coding schools it provides a sequenced progression from beginners to intermediate level of the primary coding languages: HTML, CSS, jQuery, JavaScript, PHP, Python and Ruby.

This platform is also encouraging its users to teach others and share their knowledge. “Build your reputation as an expert in your field” is good encouragement for those discovering untapped abilities and thinking of the potential that applying these new skills may have for their career or education.

#### Code School[[4]](#footnote-4)

The courses at Code School are far more in depth than those on offer at the previous examples. By using screencasts to deliver a polished breakdown of the technology or issue being discussed, with examples and graphics, the courses are set up to be completed in sequence. All content can be downloaded for reference as the learner proceeds to take on the, often quite advanced, challenges.

The site has a popular themed ‘gaming’ experience for learning Ruby on Rails called ‘Rails for Zombies’. This ‘cult status’ learning experience has five videos followed by quizzes that help the newcomer to Object Orientated Programming (OOP) visualize what is happening with the language and framework on an often quite abstract level for the beginner.

#### Rails Casts[[5]](#footnote-5)

Primarily for intermediate to advanced users of Ruby on Rails these short videos explain the new technologies and methods available to the developer. Although there is a well maintained forum and notes beneath each episode, the site is not interactive as other coding schools but demonstrates the high level that video learning can attain, while also saving the busy developer the time required to research a given issue themselves.

Railscasts keeps up to date information in relation to Ruby on Rails and the latest videos give a solid overview of the latest changes involving the release of Rails 4 and Ruby 2.

There are 2 streams of videos: free and premium. Some of the free videos are excellent as they cover the most popular Ruby Gems and topics needed to create a fully deployable application. The pro episodes offer a more in depth overview of more advanced topics such as deploying to nginx, capybara testing and more.

## YouTube Coding Playlists[[6]](#footnote-6)

YouTube allows anybody with the will and the knowhow to create their own lessons at any level of coding. The results are now thousands of channels and playlists dedicated to teaching programming at every level as well as helping learners or professionals to overcome troublesome issues. All of the videos are provided for free and as a learning resource, YouTube has a lot of potential.

A Google search for coding related material will often lead to some related YouTube material, and within the site itself there is a wealth of material to dive into. When presented with a large volume of results the best method for finding a starting point is often to find which is the most viewed video.

This is a good measure of success as it is likely to :

* Be correct
* Be presented in a clear manner with clear diction
* Be recommended on programming sites or blogs that have viewed it.
* Be worthy of the time, whether short or long, that it takes to watch the video
* Be presented in a high definition that avoids any difficulty in seeing exactly what is happening.
* Be part of a channel that has many more videos of similar standards.

Of course YouTube will recommended to the learner similar videos that may seem more relevant or at least interesting, but if the most suitable video is not found and the original purpose is side tracked, the learner may become frustrated.

Many educational institutions offer course related material on YouTube that is also helpful to those undertaking similar study or with an interest. Very well curated channels are also available from amateurs and professionals.

Following is a very small example of recommended ‘Learning to Code’ channels that are popular on YouTube.

* Learn to Program (varied content) – 140 videos
* The New Boston (varied content) – 3000+ videos
* Khan Academy – 4,190 videos
* Code.org – 76 videos
* PHP Academy – 456 videos
* Xoaxdotnet (varied content) – 250 videos
* Lynda.com (varied content) – 9,180 videos
* .net interview preparation videos – 286 videos
* teachmecomputer (varied content) – 89 videos
* Derek Banas (varied content) – 626 videos

The comment section can be useful if the user gets stuck or there is an error they see should be highlighted. By subscribing, the user can keep up to date with what is offered by the channel.

As YouTube is available on all devices it is a very convenient way of learning and revising for the novice, intermediate or experienced coder. Although the great volume of content might be off-putting for some, it is a resource that has quickly become ubiquitous with education at all levels.

## VideoCoder

VideoCoder is an application that allows the learner one point of call for fast and easy access to recommended YouTube channels and playlists at their level.

By selecting from channels for the main programming languages and frameworks, the user can access the most recommended channel of playlist for that area and quickly ascertain which has received the most views to date from the coloured pie chart. The chart provides a lot of information, quickly, within a small space. View figures are positioned beside proportionately sized pieces, tooltips provide the video name on hover and underneath the full name of each video is colour matched to the graphic.

VideoCoder is using CanvasJS to deliver the pie chart in an elegant form that integrates seamlessly with the YouTube API. The view count, and video title data provided by the API is fed into the pie chart for each channel.

Using the YouTube API allows VideoCoder to filter through a lot of data from the returned content. For video, there is the option to view as flash or HTML5 format. HTML5 allows us to view the videos embedded on all devices, this is important for VideoCoder.

A Login and Register area has been incorporated with the User App API. By having email and password details of users encrypted and stored securely by a third party, VideoCoder does not have to worry about a server side database.

AngularJS was used as it enables the extension of HTML elements. It removes the need for the JavaScript manipulation of HTML elements and CSS within the DOM. The result is an environment that is very readable, expressive and fast to develop.

# User Interface Design

Browsers today hold a substantial amount of processing power on the client side, this enables great gains in the responsiveness of applications. The result is a seamless user experience that enables total focus and concentration on the lessons.

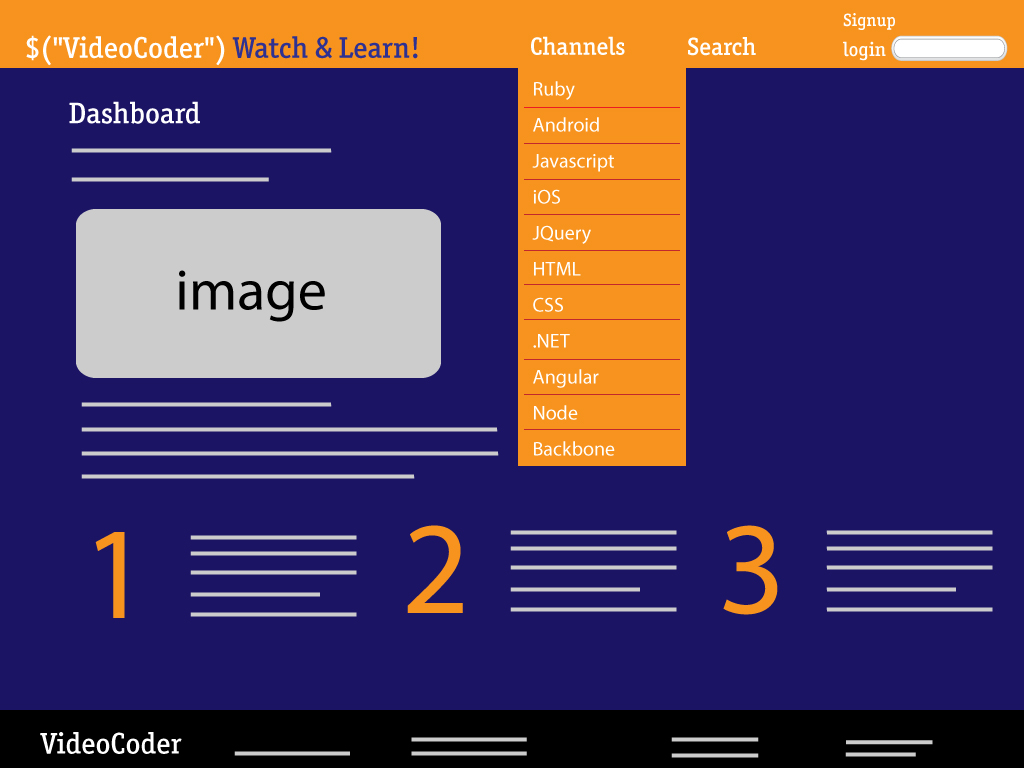
The interface is where the user interacts with the application. Getting the message and the functionality available directly across to the user as quickly and intuitively as possible is fundamental to any application and today the user’s expectations in this area are very high.

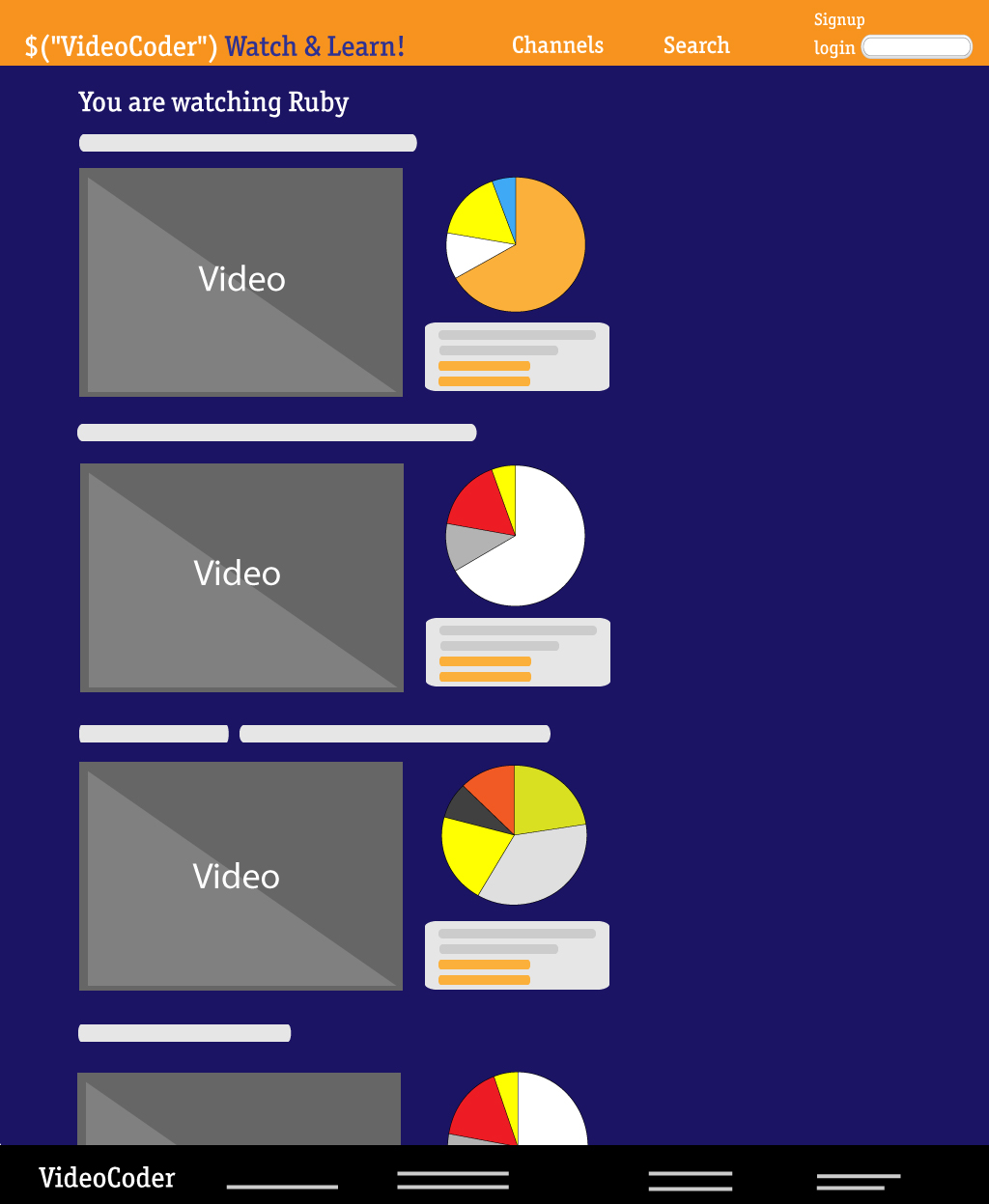
Thanks to improvements with the core front-end coding languages HTML5 and CSS3, along with the increased use of JavaScript libraries and frameworks, an online experience on any device is expected to be very fast and error free. It is becoming more common now for the business logic of applications, which used to reside primarily on the server side, to be incorporated at the front end.

In order to engage the learner and to encourage repeat visits and exploration, VideoCoder makes appropriate use of the most recent technologies in application development. The interface is designed so that the learner will not be distracted or confused. Navigation is available at all times for all of the channels and for the other core functionalities.

## Desktop

Following are the initial wireframes for the desktop design. Please see Appendix I for final screen shots.





## Mobiles and Devices

Users of VideoCoder will often want to watch the available channels and recommended videos on their mobiles and devices. Use of mobiles and devices for the Internet continues to increase exponentially, desktop browsing was overtaken by mobiles in 2012 and in 2015 80% of all internet access will be on mobiles and devices. Many popular coding schools now provide their content to growing audiences on mobiles and devices and the expectation among users is that all their online needs should be available this way as ‘m-learning’, or Mobile Learning, becomes more popular.

VideoCoder has been designed responsively to enable users have the same experience no matter how they are viewing the content.

By using the Bootstrap library’s built in responsive 12-column grid system, the elements within a webpage are responsive. The included media queries allow for custom CSS based on a number of conditions including ratios and widths but primarily focused around minimum and maximum width. Bootstrap elements are wrapped within preset rules for example a container contains rows, which may contain spans or columns. With responsive CSS a viewport below 767px will force the columns and rows to become fluid and stack vertically.

### Navigation

The ‘Channels’ drop down menu within the VideoCoder navbar needed some additional code in order to function properly on mobiles, it was becoming lost in the navbar shuffle as it fitted to a small screen. A solution was to add another CSS class to the dropdown that would override the default dropdown on mobiles.

Firstly we added ‘no-collapse’ to the dropdown. It is important not to amend the original Bootstrap code as if at any future stage, VideoCoder upgrades the Bootstrap in use, it would render customized areas redundant and they would need to then be reapplied. We added a CSS file (dropdown\_no\_collapse.css), to be loaded after the main Bootstrap CSS loads, that manually overrides all of the responsive dropdown styles.

Commenting out the following in the bootswatch ‘superhero’ CSS removed the persistent scroll bar on the responsive navbar and allowed the channels to pop out over the navigation toggle area.

/\*

.navbar-collapse.in {

overflow-y: auto;

}\*/

### Videos

One HTML element that’s difficult to contain within responsive layouts is ‘iframe’. VideoCoder is using iframe as it enables the external video be embedded within the application by including a URL that points to the content from the YouTube api. As iframe requires width and height attributes, which can’t be removed, the video was displaying at full size on mobiles and protruding outside of the containing bootstrap column elements.

The solution to have video resize with the screen’s width was to add a CSS class, video-container, to the div element that surrounds the iframe.

.video-container {

position: relative;

padding-bottom: 56.25%;

padding-top: 35px;

height: 0;

overflow: hidden;

}

By setting the position to relative we can set absolute positioning to the iframe itself. By looking at the aspect ratio of a video, 16:9, we can calculate that the height will be 56.25% of the width, this value is used for padding-bottom. We can set height at 0 as the padding-bottom value is giving the required height. Width will automatically resize with the responsive element that contains the div. Overflow at hidden will ensure that any protruding content is hidden from view.

The iframe also required some extra styling. Here we target iframes within the enclosing div with the video-container class.

.video-container iframe {

position: absolute;

top:0;

left: 0;

width: 100%;

height: 100%;

}

We are using absolute positioning as the containing div has a height of 0, by having top and left at 0 we are positioning the iframe exactly to the top left corner. Having width and height at 100% ensure the video will take up all of the space used by the containing div.

## Charts[[7]](#footnote-7)

The charts feature, enabled with CanvasJS, enables the user to quickly see which videos in their selected channel are currently the most viewed. This lightweight library of customizable charts built with HTML5 and JavaScript runs on all devices. VideoCoder is using the CanvasJS pie chart to maximize the required data in a small space. Each colour coded section of the pie chart is proportional to the quantity it represents.

ChartJS documentation allows for very easy manipulation of the default CSS by applying additional rules that will override. For example we have amended the background colour, the colours of text, legend and label.

Animation features within the pie chart add interactivity that is interesting for the user. On selecting a new channel the pie chart swings into shape like an unrolling deck of cards. On hover, tooltips appear to provide information on the video title the piece is referring to and on click, these pieces move out from the center. Numbers outside the pie chart clearly indicated the view count of each video.

One of the reasons why CanvasJS was preferred to other similar libraries is for the quite comprehensive and very thorough documentation[[8]](#footnote-8) available.

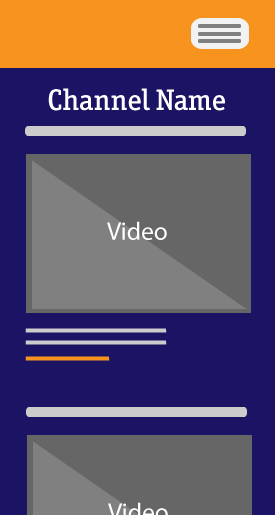
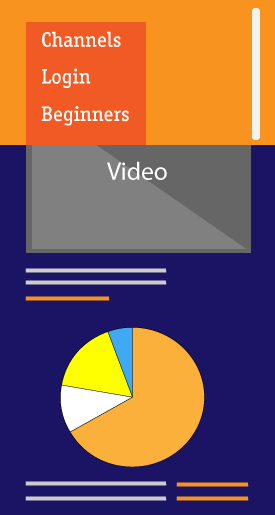
## Some Tools that were tried

It was initially intended that the navbar for mobile would be similar to that provided on Facebook or LinkedIn where the navbar is available with a left swipe. We found a JQuery plugin called ‘Sidr’[[9]](#footnote-9) that enables side menus be fully responsive. However, on desktop it required that it be toggled on or off and the solution with Bootstrap of allowing the channels dropdown to appear as a modal on small viewports was sufficient.

At the early stage, when we were exploring ways of embedding YouTube channels and playlists and displaying them to maximum effect on mobiles, we looked at the ‘Yunero’ JQuery plugin. It creates a widget within an application for displaying a video feed together with small, embedded thumbnails of further related content. However it was no longer required once we had fully embedded the YouTube streams by using iframe and when the video-container CSS class enabled the videos to display properly on all devices.

It was also intended to include a search box to try and integrate Stackoverflow[[10]](#footnote-10) within the application, this is not possible at present due to restrictions with Stackoverflow’s API[[11]](#footnote-11). Similarly it is not possible at present to utilize iframes to ‘bring’ Stackoverflow into the application.

These are the initial wireframes for the Mobile design. Please see the Appendix II for the final screen shots.

# Architecture

## Application Architecture

The application has been built utilizing the latest front end technologies. These technologies have been carefully chosen so as to provide all the necessary elements to obtain a final product that is both up to date and that adapts to the latest devices currently in use, i.e.: desktops, mobile devices and tablets.

The various JavaScript libraries are within a JS folder which contains sub folders for vendors. The AngularJS, the JQuery and the Bootstrap JavaSript files are providing a lot of functionality and means of control within the application. As they are proven methods, developed by respected third parties, they are used here to save a lot of time in the development process.

CSS in within a CSS folder which has sub folders for vendors, including the bootswatch template called ‘super hero’ used to provide colour scheme and font stylings. The vendor JavaScript and CSS files are not customizable, any of these default settings that needed to be over written were done so within the projects main ‘dashboard’ or ‘myCSS’ files. Altering these third party tools is not recommended as future updates may be invalid or require more avoidable changes else where in the application.

Within each HTML file all of the required CSS is called in before the body. After the body and once all else is loaded, the JavaScript is called. The JavaScript relies on every thing else being in place first.

The Jasmin Tests are within the test folder and the specs themselves are within a spec folder.

The HTML is within the views folder. As these files work locally but do not route correctly when deloyed to Heroku, separate PHP ‘include’ files are calling in each HTML file and linking to these appropriately within the navigation. The PHP files are outside all other folders.

A gitignore file was required to disable some Git errors regarding .idea. This too is outside all other folders at base level.

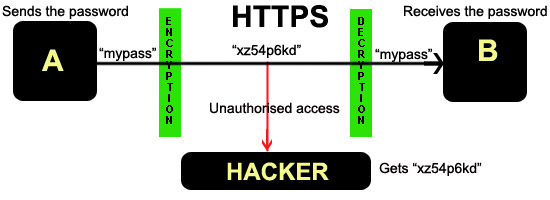
## Security

Using libraries leaves applications prone to malicious code or bugs that may include dangerous scripts that allow some party to monitor for key presses etc. There may be incompatibility issues with other libraries or frameworks in use. VideoCoder is using trusted and tried third party libraries that are well maintained and industry standard.

Due to VideoCoder being a purely client side application yet still wanting users of the application to register, a solution was proposed and implemented to integrate a third party application to accept, authenticate and store VideoCoder user data.

This service is provided through UserApp. A secure API post is made to the UserApp service when a user on VideoCoder submits their credentials. In order to integrate this service into the application a private API key was obtained. This private API key allows the data to be passed between VideoCoder and UserApp.

As stated by UserApp the transfer of the data can only be made over HTTPS, which prevents any information being stolen over the network during data transfer. As an extra layer of security all passwords are salted and hashed before being submitted to the database. A diagram below illustrates the use of https within VideoCoder.



As is shown in the above diagram through the use of HTTPS, a secure transfer of data between application and server is ensured by encrypting the transferred data over the network. Without the use of a third party this degree of security and storage would not be possible for VideoCoder.

## Toolkits and Frameworks

### AngularJS

According to the Angular website:

“HTML is great for declaring static documents, but it falters when we try to use it for declaring dynamic views in web-applications. AngularJS lets you extend HTML vocabulary for your application. The resulting environment is extraordinarily expressive, readable, and quick to develop”[[12]](#footnote-12)

Angular achieves this by making available to developers a number of tags to help make server calls and manipulate JSON objects so they are readily available in the browser environment to be manipulated in any way, shape and form that developers see fit to suit the needs of the application.

The syntax in very elegant and easy to use. This framework takes advantage of JavaScript language and makes light work of interacting with web servers in an asynchronous way.

Data binding is made simple with the use of Angular; any changes in the server are reflected in the views in an elegant way. Controls allow for data and DOM manipulation at browser level.

By taking advantage of client side processing, AngularJS allows the extension of HTML elements in an application. HTML is fine for declaring static documents but it wasn’t designed for dynamic views in web applications. AngularJS removes the need for the JavaScript manipulation of HTML elements and CSS within the DOM (Document Object Model). The result is an environment that is very readable, expressive and fast to develop. AngularJS is fully extensible and it works well with other libraries. It can be seen as a toolset for creating a framework within front end, client side control allows rendering of this framework.

Created by Google, AngularJS is known as an MVW, that is, a ‘Model View Whatever’. The framework looks after taking in the JSON from a server and rendering it on the local device, connecting the data with the JavaScript, HTML and CSS.

VideoCoder is using the AngularJS data-binding methods to automatically update the view when the model changes and to update the model when the view changes. This elimination of the requirement for DOM manipulation creates more manageable code as well as saving a lot of time for the developer.

To use, the JavaScript file ‘angular.js’ is downloaded and placed within the application.

<html lang="en" data-framework="angular.js">

It is made available for interaction with all elements by declaring it in the main div. By providing controllers (of hard coded data) Angular enables easy calling of these directly to where required.

In the ‘Channels’ dropdown of the navbar, Angular is enabling that content of the main index page, ‘dashboard.html’, be replaced with content from the selected link.

This line in the dropdown prevents each link from reloading the page

<ul class="nav" class="{{active}}" ng-click= "$event.preventDefault()">

The title of the channel appears above the embedded video streams.

<h3 ng-show="active">You are watching <b>{{active}}</b>!</h3>

On clicking a channel the text within **ng-click**=active, will replace the content currently in {{active}}.

<li><a href="#thenewboston" ng-click="active='Ruby on Rails tutorial'" class="ruby">Ruby on Rails</a></li>

The image elements and text on the dashboard.html are replaced with the new channel content by using **ng-hide**="active"> within their elements.

As the controllers are separated from the JavaScript code, Angular enables better testing.

Angular, and similar frameworks are helping to over come weak points in JavaScript regarding robustness, reliability and performance. All of the features can be modified or replaced to suit a development’s workflow and needs.

‘Deep Linking’ is an AngularJS feature that allows users to bookmark and email links to locations within apps, bypassing the issues that arise from AJAX regarding this functionality. Client Side form validation rules can be declared easily within AngularJS without having to write more JavaScript code.

In many cases JavaScript frameworks are driving development today.

### HTML5

The HTML5 canvas element allows an area be defined for drawing shapes and creating graphical elements on. HTML5 enables the display of the CanvasJS pie chart that is displaying the relevant content delivered from the YouTube API. It also animates the tool tips and each pie segment on hover or on click.

The embedded video is enabled to display on all devices by appending ?html5=1 to the end of each video url. If flash is available on a browser it will be used, and if not HTML5 will be used to display the video.

### CSS3

CSS3 makes possible new decorations and behavior that was previously restricted to JavaScript. CSS3 has a whole new set of predefined classes that makes it possible for developers to simulate animations and behavior that is expected of a Rich Internet Application today.

### Bootstrap[[13]](#footnote-13)

This library was chosen to simplify the responsiveness of the application. It allows developers to quickly generate HTML and CSS templates with its predefined classes and grid system. It is also possible to select and restrict the content to be made available to the different formats to which the application caters for. This is especially ideal bearing in mind that there is content that may not be suited for mobile devices such as iframes, embedded videos and resource rich graphics.

The Bootstrap framework of compiled and minified CSS and JavaScript resources, allow for a quick and easy method in which to apply an application’s positioned elements and components, their appearance and for some functionality. It is supported on all modern browsers and enables easy incorporation of responsive CSS through the use of media queries which allow the maximizing of the appearance of elements depending on the conditions of ratios, widths and display types.

Furthermore Bootstrap has incorporated a number of templates that can be used with minimum effort providing maximum benefits in terms of appearance and cross browser compatibility.

VideoCoder is using the customizable Bootstrap template called ‘Superhero’ from the free Bootswatch library of readymade themes. The clear typography on a dark grey background is reflective of the dark theme often selected by designers and developers of applications as they spend long periods of time in front of the screen. The simple Typography enables a clear reading of text, the orange highlight on the navbar and other features provides enough contrasting colour that is of no distraction.

## Data Transfer Strategies

### API – Application Programming Interface

An API is a set of programming instructions and standards that are used for accessing a web based application or tool. VideoCoder utilizes the YouTube API to enable use of selected streaming video channels and playlists as well as searching its content. APIs are software to software interfaces, VideoCoder and YouTube can interact seamlessly together without user knowledge or intervention.

An API can be seen as a kind of Software as a Service (SaaS). Instead of having to develop absolutely all functionality required in an application, certain functionality can be ‘contracted out’ to remote software that has allowed public access. The communication between the two is managed by Web Services which are a collection of technological standards and protocols, these include XML and JSON. Using the YouTube API means that VideoCoder need not host anything on a web server. Access to YouTube’s JSON feed is possible without the requirement of an API key.

VideoCoder is using JQuery to dynamically organize the JSON data returned from YouTube. The links in the Channels drop down are using href values containing a # symbol along with the name of selected YouTube playlists and channels.

$(function(){

$('#usersnav ul li a').on('click', function(e){

e.preventDefault();

var htmlString = '<ul id="videoslisting">';

var channelname = $(this).attr('href').substring(1);

var ytapiurl = 'http://gdata.youtube.com/feeds/api/users/'+channelname+'/uploads?alt=json&max-results=10';

The JavaScript .substring(1) method creates a new string variable starting from the second character of the href, the # symbol is the first character. Having removed the # we can place the required channel username or playlist ID into the generic YouTube API call. The result coming back has been specified to 10 and we are specifying for JSON as the response type.

Next we loop through the returned content.

$.getJSON(ytapiurl, function(data) {

$.each(data.feed.entry, function(i, item) {

var title = item['title']['$t'];

var videoid = item['id']['$t'];

//more variable names

As YouTube API returns a lot more information than we require we create variables only for the content that we are interested in. Then we create an HTML string that will be returned to the browser. This is then displayed by using an unordered list of items which is presented in the ‘videos’ div. To do this we use the jQuery .html() method. An iframe is used to embed the videos as default is to send the user to the YouTube site on click to view.

//end of above variable names

htmlString +='<li class="clearfix"><h2>' + title + '</h2>';

htmlString +='<div class="video-container col-md-9">

<iframe src="' + vlink + '" target="\_blank" width="480" height="360"></iframe></div>';

htmlString +='<div class="meta col-md-3"><p>Published on <strong>' + fulldate + '</strong></p>

<p>Total views: <strong>' + commafy(numviews) + '</strong></p>' + '<p>Total comments: <strong>'+ numcomms +'</strong></p>' + '<p><a href="'+ ytlink +'" class="external" target="\_blank">View on YouTube</a></p>' +

'<p><a href="'+ vlink +'" class="external" target="\_blank">View in Fullscreen</a></p>';

video.push(numviews);

vidTitle.push(title);

}); // end each loop

$('#videos').html(htmlString + "</ul>");

In order to ensure the comma is included in view numbers that are in the thousands, which is the case in almost all selected videos as they are popular, we are using a function called commafy() as a solution.

function commafy( arg ) {

arg += '';

var num = arg.split('.');

if (typeof num[0] !== 'undefined'){

var int = num[0];

if (int.length > 3){

int = int.split('').reverse().join('');

int = int.replace(/(\d{3})/g, "$1,");

int = int.split('').reverse().join('')

}

}

if (typeof num[1] !== 'undefined'){

var dec = num[1];

if (dec.length > 4){

dec = dec.replace(/(\d{3})/g, "$1 ");

}

}

return (typeof num[0] !== 'undefined'?int:'')

+ (typeof num[1] !== 'undefined'?'.'+dec:'');

}

});

The ‘mashup’ of YouTube within VideoCoder allows simple integration of the vast video resource to be presented to the learner in manageable chunks that are focused primarily on the most suitable material. The VideoCoder Channels can be easily amended and updated as required.

### JavaScript

JavaScript is a fully featured language that provides tremendous scope for creating functionality within interactive applications. It enables accessing of elements within the DOM through their ID or a class. JavaScript allows us to separate content and behavior. In order to ensure that all JavaScript has everything it needs in order to function, we always place the JavaScript links at the bottom of the html page to ensure that all else is loaded before it can kick in to action.

JavaScript is not a true OOP (Object Orientated Programming) Language but it can be used to implement a number of traditional OOP concepts. It is a prototype based language where objects are used as prototypes for others with the new object replicating the behavior of an existing one by cloning it.

### JSON - JavaScript Object Notation

JSON is the lightweight, native JavaScript data-interchange. As it doesn’t require to be parsed first and assigned to variables, it is extremely fast. JSON also has the advantage of being easy to read and write, as there are fewer characters to write when compared to equivalent data being represented in XML (eXtensible Markup Language), which requires types placed within the data.

When a JSON request is made a very big string of data is returned, JSON presents a collection of key or name value pairs. When accessing the data we use .parseJSON() which will recognize only JSON text and therefore avoiding any potential malicious JavaScript code (that may also be included). Most browsers now provide native JSON support

In VideoCoder, we are retrieving the JSON feed from the YouTube API. We are using the JavaScript substring(1) method to create a new string variable starting from the second character, the first being a # (hash) symbol which is being used to target each channel or playlist name. We can then place that channel or playlist name into a generic API call :

<http://gdata.youtube.com/feeds/api/users/'+channelname+'/uploads?alt=json&max-results=10>

Here the response type we need is JSON. It can be parsed easily with .parseJSON() as we loop through all of the content sent back. We don’t want to display all of the content, so we can make variables only for the content, which we will be using in VideoCoder, and organize the data on the page dynamically.

### CanvasJS

The graphic generated in the dashboard page utilizes information that is made available via JQuery from the YouTube API. It takes the titles and number of views of each returned video and populates the graph with these values. The dataPoints variable consists of an array of hashes and is utilized to pass the required parameters to the canvas function:

*dataPoints: [*

*{ y: parseInt(video[0]),legendText:vidTitle[0], label: video[0] },*

This library offers versatility, ready-made templates and is very customizable. The documentation supplies details for all of the particular parts of the charts that may be altered as required.

VideoCoder required some changes to the default white background and large fonts in the Pie Chart. By specifying particular CSS rules for the Title, Legends and Labels, we could quickly try alternatives in size, font and colour. Together with choosing one of the many themes for a colour palette we settled on a suitable look for the Pie Chart.

CanvasJS takes in any specified values in order to generate the charts. With VideoCoder it is important that the most viewed figures be the basis for Pie Chart shapes as an indication for a starting point for the learner. The label feature provides an animated Tool Tip on hover and clicking a pie shape forces it to move outwards slightly from the centre, these are nice features that also work on devices.

# Evaluation and Testing

## Deployment

VideoCoder’s local routing required some amendment in order to successfully deploy to Heroku. As Heroku will not accept standalone HTML, CSS and JavaScript files, it needs to have specific languages or frameworks in place for example Ruby on Rails or PHP. We over came this issue by simply creating an index.php file that calls in the main index page, dashboard.html.

<?php

include\_once( "views/dashboard.html");

?>

The pages linked within also required the same treatment and the navigation updated to match.

## Usability

We received invaluable feedback at the early stages of usability testing with friends and colleagues. The initial intention was to have the navigation slide from the left for mobiles and devices on a screen swipe. The JQuery plugin ‘Sidr’ which enables side menus be fully responsive was trialed but it was found to be troublesome on desktop where it required that it be toggled on or off to view the drop down menu of channels. The final solution with Bootstrap of allowing the channels to appear as a modal on small viewports was more successful regarding usability.

A Search page was in place for searching YouTube, however it was deemed inadequate from a usability point of view and we created a Beginners section, which displays an example playlist of streaming videos relevant to beginners. This functionality is in place by populating a hidden search input field with the ID number of a playlist when the user selects the Beginners channel they want to watch.

Layout of this page went through some alteration as we experimented with placement of the Beginners channels underneath and to the right of the video screen while still having the other channels from the main page still available as a dropdown. Replacing the dropdown with the Beginners section allows the user have only one place to think of for changing channels.

By clicking on a Beginner’s thumbnail the user will see the selected video display on the large main screen. This video will be HTML5 on all devices. On mobiles, the HTML player fills the screen and when finished the user is brought back to the application.

Keeping in context of VideoCoder’s main functionality as a place for learning, we included a ‘practice area’ which is utilizing the jsFiddle tool embedded in an iframe. This tool is disabled for mobiles as it does not function well on such small viewports.

Finally a ‘Stuck?’ page is simply listing various resources which may be unknown to the beginner and intermediate coder.

## Testing

The Jasmine library has been the testing framework of choice for VideoCoder from the very start. Initial plans were to use a ‘Test Driven Development’ process however time constraints dictated that the team were keen to add core functionality and add tests partly in retrospect and partly in the test first procedure.

As time became more precious and core functionality issues more abundant. The concentration of our efforts were to focus on completing core functionality.

A level of testing has been added however it has been added with little success. The recognition to define JavaScript functionality as classes did not allow tests to be completed with any success. In all the cases of running tests error messages were being returned such as “video not defined” and “user not defined”.

Upon the discovery that we had not put in place an architecture that represented our functionality as classes led to refactoring of the JavaScript to create a Video class and a User class. Tests were created on both classes, the tests created were mostly for validation purposes i.e. to check that values had been entered for a valid submission.

An example of the tests is show below:

describe("signup", function() {

it("firstname should not should not be empty", function(){

expect(function() {User.signup("tom")}).toThrow(new Error("login field empty"));

})

});

7 specs, 2 failures

[signup firstname should not should not be empty](http://localhost:63342/arial_ca3/test/SpecRunner.html?spec=signup%20firstname%20should%20not%20should%20not%20be%20empty)

The result of the example test above is shown. This was an unexpected result as the function related to this test was to take in a parameter and the test against it was to ‘Throw new Error” if the length of the expected parameter was less than one (an empty value).

It was discovered that the refactoring of the JavaScript which created a User class had negative effects on the related functionality. In the case of User signup and Login this functionality was broken due to the steps taken in the creation of the classes. As the deadline approached it was decided to remove the User class and revert back to the previous version. The tests have been left in place as there is no adverse effect to core functionality.

# Summary

The team took the approach of building the kind of resource that would have been handy to have available when we started to code ourselves. It can be slightly overwhelming for the beginner as the amount of content available grows exponentially and they are eager to get a start but don’t quite know where or how.

The goals for VideoCoder were the creation of a learning portal as a central reference point that a beginner or intermediate learner can use to access YouTube videos. The application also offers a couple of other useful features for the learner including a practice area for trying out the code as it is learned, and also a page of links to the most popular sites that provide tutorials, video casts, code samples, forums and reference material.

VideoCoder’s main feature is watching predefined channels that have been selected to cover the various learning levels, of different technologies, from basic to advanced and provide a good starting point for users without the need to search the web. After researching the many available channels and from our experience as developers and learners, we knew it was important to select the most suitable channels and playlists as starting points for learning.

When learning code, it is fundamental that the learner gets to practice and experiment with newly acquired knowledge and skills, this lead us to include a practice section where the developers can get their hands dirty, fiddle with the code and see the results in real time. Finding a sense of progress and confidence with actual results is very important when learning to code. It is important that the learner not be dissuaded when they are beginning. We have used JSFiddle placed within an iframe to facilitate this process. It enables live coding with visible results. Unfortunately it was not a successful option for small view ports and had to be disabled for mobiles.

The ‘Stuck?’ page is intended as an introduction to the very active and helpful coding communities that exist. This is only a selection that the team found useful but is by no means definitive and is only intended to be used as a guide.

While the team were creating an application to aid learning we were also learning quite a lot of new material and we researched these as best we could with the application’s requirements in mind. For example using the YouTube API was initially troublesome when we tried to pull the ‘numbers of views’ data from it to display in the charts. We explored a lot of the documentation on the Google YouTube API developer’s pages; here a lot of examples for retrieving videos were given. We found a very helpful blog on the Team Treehouse learning site which enabled us to see a fast solution for having the required playlists and channels stream on VideoCoder.

Unfortunately the results were all screen shots that linked to the YouTube site. By embedding the videos in iframes, we enabled the videos to be played within all browsers. Unfortunately on some mobiles, notably the iPhone, embedded videos are not enabled. So we were forced to search for alternatives and ways to overcome this issue. First we tried using the <video> HTML5 tag, however this did not work as expected so we requested the API to return the HTML5 version of the videos and have the flash version as a fallback in case the first option was not available.

The team took an iterative, Agile approach to development. We delegated certain areas to each team member. As the first iterations came to place, we saw that some features implemented with little difficulty while others required some more work. As the project progressed and team members were aware of problem areas we all focused on the major difficulties in the hope of sourcing solutions or finding a solution. We endeavored to have as many of the main intended features up and running, however there are certain features that will need more time and further development.

## Problems & Solutions

Some of the problems we encountered were firstly getting used to new frameworks such as Angular and Jasmine. These required time and effort from all team members, even at this point of the development process we could safely say that there is a lot more to learn and practice with the technologies we used.

The chart presented us with a number of challenges, initially we chose Chart.js to try and incorporate it to the dashboard page and feed it with information from the YouTube API. Due to limitations proper to the library this proved to be more difficult than expected, therefore we decided to try a different library and we found Canvas.js. This library is easier to adapt and customize, allowing us to change fonts, colors and most importantly to easily plug in the JSON feed from the API.

As mentioned above we had difficulties with the video tag and iframe tag, however we managed to request the API to return HTML5 based content.

## Further Development

We would like to make better use of the charts and make them more relevant to the different sections of the app.

We would like to be able to request user authentication to use the app.

We would like to have more channels and different levels to each language in the form of beginners, intermediate and advanced.

We would also like to move the app into a server environment such as Ruby on Rails and fully avail of the data that could be retrieved from a local database and further experiment with Angular.

# Appendices

## Appendix I: Desktop













## Appendix II: Mobile

# References

AngularJS. 2014. [online] Docs.angularjs.org. Available at: <http://docs.angularjs.org/api> [Accessed 17 Apr. 2014].

Berriart.com, (2014). *Sidr - A jQuery plugin for creating side menus*. [online] Available at: http://www.berriart.com/sidr/#usage [Accessed 24 Apr. 2014].

Demos.codehandling.com, (2014). *Yunero - Jquery plugin displays a video feed for YouTube channel uploads or custom playlist with subscription box*. [online] Available at: http://demos.codehandling.com/yunero/yunero.html [Accessed 24 Apr. 2014].

Elearningguild.com. 2014. *The eLearning Guild : Mobile Learning : Research Library*. [online] Available at: http://www.elearningguild.com/research/archives/index.cfm?id=117&action=viewonly [Accessed: 15 April 2014].

Falcon, A. (2014). *Top 10 Websites to Learn Coding (Interactively) Online*. [online] Hongkiat.com. Available at: http://www.hongkiat.com/blog/sites-to-learn-coding-online/ [Accessed 25 Apr. 2014].

Fausak, T. (2012). *Dropdown Menu in Twitter Bootstrap's Collapsed Navbar*. [online] taylor.fausak.me. Available at: http://taylor.fausak.me/2012/03/15/dropdown-menu-in-twitter-bootstraps-collapsed-navbar/ [Accessed 24 Apr. 2014].

Fausak, T. (2014). *tfausak/bootstrap-navbar-dropdown.html*. [online] GitHub Gists. Available at: https://gist.github.com/tfausak/2046174 [Accessed 24 Apr. 2014].

Getbootstrap.com, (2014). *Scaffolding · Bootstrap*. [online] Available at: http://getbootstrap.com/2.3.2/scaffolding.html#responsive [Accessed 24 Apr. 2014].

Handling, J., 2013. *Code Handling: Yunero - Jquery plugin displays a video feed for YouTube channel uploads or custom playlist with subscription box*. [online] Codehandling.com. Available at: <http://www.codehandling.com/2013/02/yunero-jquery-plugin-displays-video.html> [Accessed 17 Apr. 2014].

Marino, A., 2014. *Edit fiddle - JSFiddle*. [online] Jsfiddle.net. Available at: <http://jsfiddle.net/pKZPP/> [Accessed 17 Apr. 2014].

McCollin, R. (2014). *Making Embedded Content Work In Responsive Design | Smashing Magazine*. [online] Smashingmagazine.com. Available at: http://www.smashingmagazine.com/2014/02/27/making-embedded-content-work-in-responsive-design-3/ [Accessed 24 Apr. 2014].

Nair, S., 2013. *How to embed videos in your site via YouTube Data API?*. [online] Web 2.0 Start-Up Blog. Available at: <http://mewsigma.wordpress.com/2013/06/28/how-to-embed-videos-in-your-site-via-youtube-data-api/> [Accessed 17 Apr. 2014].

National College of Ireland. 2014. *Flipping the Classroom at National College of Ireland*. [online] Available at: http://www.ncirl.ie/NewsEvents/Blog/TabId/214/ArtMID/750/ArticleID/74/Flipping-the-Classroom-at-National-College-of-Ireland-.aspx [Accessed: 16 April 2014].

Rocheleau, J., 2013. *Developing over the YouTube API with JSON - Treehouse Blog*. [online] Treehouse Blog. Available at: <http://blog.teamtreehouse.com/developing-over-the-youtube-api-with-json> [Accessed 17 Apr. 2014].

Roos, D. (2014). *HowStuffWorks "What is an API?"*. [online] HowStuffWorks. Available at: http://money.howstuffworks.com/business-communications/how-to-leverage-an-api-for-conferencing1.htm [Accessed 27 Apr. 2014].

Search: list - YouTube — Google Developers. 2014. [online] Developers.google.com. Available at: <https://developers.google.com/youtube/v3/docs/search/list> [Accessed 17 Apr. 2014].

Varela, A., 2014. *Sidr - A jQuery plugin for creating side menus*. [online] Berriart.com. Available at: <http://www.berriart.com/sidr/#usage> [Accessed 17 Apr. 2014].

Wikipedia, (2014). *Application programming interface*. [online] Available at: http://en.wikipedia.org/wiki/Application\_programming\_interface [Accessed 27 Apr. 2014].

YouTube Data API: JavaScript Code Samples - YouTube — Google Developers. 2014. [online] Developers.google.com. Available at: <https://developers.google.com/youtube/v3/code\_samples/javascript#search\_by\_keyword> [Accessed 17 Apr. 2014].

YouTube for Developers - API Resources - YouTube. 2014. [online] Youtube.com. Available at: <http://www.youtube.com/yt/dev/api-resources.html> [Accessed 17 Apr. 2014].

1. MOOC List | A complete list of Massive Open Online Courses (free online courses) offered by the best universities and entities.. 2014. MOOC List | A complete list of Massive Open Online Courses (free online courses) offered by the best universities and entities.. [ONLINE] Available at: http://www.mooc-list.com/. [Accessed 25 April 2014]. [↑](#footnote-ref-1)
2. Learn Web Design, Web Development, and More | Treehouse. 2014. Learn Web Design, Web Development, and More | Treehouse. [ONLINE] Available at: http://teamtreehouse.com. [Accessed 25 April 2014]. [↑](#footnote-ref-2)
3. Learn | Codecademy. 2014. Learn | Codecademy. [ONLINE] Available at: http://www.codecademy.com/learn. [Accessed 25 April 2014]. [↑](#footnote-ref-3)
4. Learn by Doing - Code School. 2014. Learn by Doing - Code School. [ONLINE] Available at: https://www.codeschool.com/. [Accessed 25 April 2014]. [↑](#footnote-ref-4)
5. Ruby on Rails Screencasts - RailsCasts. 2014. Ruby on Rails Screencasts - RailsCasts. [ONLINE] Available at: http://railscasts.com/. [Accessed 25 April 2014]. [↑](#footnote-ref-5)
6. coding playlists - YouTube. 2014. coding playlists - YouTube. [ONLINE] Available at: http://www.youtube.com/results?search\_query=coding+playlists. [Accessed 25 April 2014]. [↑](#footnote-ref-6)
7. Beautiful HTML5 & JavaScript Charts | CanvasJS. 2014. Beautiful HTML5 & JavaScript Charts | CanvasJS. [ONLINE] Available at: http://canvasjs.com/. [Accessed 25 April 2014]. [↑](#footnote-ref-7)
8. Basics of Creating HTML5 Charts - CanvasJS. 2014. Basics of Creating HTML5 Charts - CanvasJS. [ONLINE] Available at: http://canvasjs.com/docs/charts/basics-of-creating-html5-chart/. [Accessed 25 April 2014]. [↑](#footnote-ref-8)
9. Sidr - A jQuery plugin for creating side menus. 2014. Sidr - A jQuery plugin for creating side menus. [ONLINE] Available at: http://www.berriart.com/sidr/. [Accessed 25 April 2014]. [↑](#footnote-ref-9)
10. Stack Overflow. 2014. Stack Overflow. [ONLINE] Available at: http://stackoverflow.com/. [Accessed 25 April 2014]. [↑](#footnote-ref-10)
11. Embed Stack Overflow (mechanism or itself) into my website - Meta Stack Exchange. 2014. Embed Stack Overflow (mechanism or itself) into my website - Meta Stack Exchange. [ONLINE] Available at: http://meta.stackexchange.com/questions/75138/embed-stack-overflow-mechanism-or-itself-into-my-website. [Accessed 25 April 2014]. [↑](#footnote-ref-11)
12. AngularJS — Superheroic JavaScript MVW Framework. 2014. AngularJS — Superheroic JavaScript MVW Framework. [ONLINE] Available at: https://angularjs.org/. [Accessed 27 April 2014]. [↑](#footnote-ref-12)
13. Bootstrap . 2014. Bootstrap . [ONLINE] Available at: http://getbootstrap.com/. [Accessed 27 April 2014]. [↑](#footnote-ref-13)