A study on User Interface and Experience designs in order to create a modern, interactive learning web application.

Trevor Kiggundu: 001001720
May 2020
BEng Software Engineering
Supervisor: Dr Elena Irena Popa

A dissertation submitted in fulfilment of the requirements for the module, Final Year Project, Computing and Information Systems Department, University of Greenwich.

ABSTRACT

A human software crisis; that is all it can really be described as. Technology professionals have long been aware of the accidental difficulties that affect development in the field. However, the difficulty of human error is often overlooked. The tech-based society of the 21st century has created an unusual phenomenon in which the amount of available jobs rises with the technology recruits available. Sadly, this is only happening due to the fact that companies cannot effectively fill these roles, due to the scarce number of recruits that are actually qualified enough. This points a finger toward the learning avenues and habits that are afforded and practiced by users; if all recruits are learning the same content, then why are only a small few more qualified than the vast majority? The numbers are not even close. This problem sparked the study detailed below, in which modern and primitive learning tools, as well as motivating factors, were studied in conjunction to create an interactive learning web application boasting user-friendly, modern and enjoyable user interface and experience designs.

ACKNOWLEDGEMENTS

I would like to thank everyone that was involved in the proposal, design, and eventual completion of this project. This includes, my supervisor Dr Elena Irena Popa, using her expertise to support me throughout the difficulties I faced throughout the duration of the project; it was greatly appreciated. I would also like to thank Dr Markus Wolf and Dr A.K.M Mahtab Hossain, as their ASP.NET and Web Application Development lectures respectively gave me a better insight on how to complete my project. Finally, I would like to thank all my loved ones for the continued support as I completed my final project.

Table of Contents

1 INTRODUCTION

1.1 Problem Domain

The current society is a “technology-based” (Susskind, 2015) one, and there seems to be no sign of that changing any time soon. The demand for technology professionals is at an all-time high. A career in this profession is one of the rarer ones in which the number of job opportunities continues to increase despite the parallel increase in recruits available. At least that is what it seems like from the outside looking in. Astonishingly, a large percentage of tech-based job applicants, even “those with Masters' Degrees and PhDs in computer science, manage to fail during interviews when asked to carry out basic programming tasks” (Kegel, 2014). This phenomenon begs the question; are the learning tools available well-designed, suitable, informative and modern enough to meet the demand of the changing technological field? The standard face to face teaching approach applies to most subjects, with programming being one of them, especially in universities. However, “most programming courses do not provide the kind of experience that real programming gives” (Kegel, 2014), only laying down a good base for the student to explore and learn more themselves. This emphasizes a problem regarding the external motivation of junior programmers. This also begs a final question; is this style of learning outdated? Gone are the days of picking up a book to learn and practice basic programming principles, as learning material is more accessible and abundant as ever due to the creation of the World Wide Web in 1989. It is easier than ever to learn how to program; if one is willing to practice. The programming courses available, however, only set the table for theoretical concepts to be grasped. Without the proper motivation and modernized learning tools, it is difficult for young programmers to grasp both the theoretical and practical skills needed to work in the real world, as too many graduates only possess the former as opposed to the latter.

1.2 Project Objectives

The main focus of the project is to create a web-based learning application, boasting modern and interactive user interface and experience designs in order to better motivate students to learn. Designing an “effective” and “interactive learning environment requires understanding the intricate relationships” (Repenning, 2006) between human and computer, so a large focus on user interface and experience designs (UX/UI) as well as human-computer interaction (HCI) factors is necessary. The secondary objective regarding the study of motivation factors is also extremely important, as modern learning practices, especially regarding learning to program, heavily involve the use of computers. The main investigation includes using all of the aforementioned topics in order to provide an alternative to the various available online applications that aim to teach programming as well. These applications, however, are still effective and important, as they each provide a different learning approach through games, quizzes and "visualization features to improve students' understanding of the programming concepts" (Moons et al., 2013). The bulk of these investigations will be documented in the literature review, and the web application will be created in lieu of these findings.

This will help to develop a web application that aims to provide an alternate learning experience than the traditional lecture and tutorial model used by many learning organizations. This is important, as "traditional ways of assignment grading are not scalable" and often do not "give timely or interactive feedback to students" (Tillman et al., 2013). The application itself will be created to teach HTML5 to students. A survey via questionnaire will be created for students themselves to choose which topics to learn about. The final goal indeed is to keep learners motivated, so the web application will have different interactive features such as games, practice quizzes and editors to keep the user interested in the subject, boosting user interactivity. These features will aim to keep the user more interested and engaged, as "improving learning effectiveness has always been and will be constant challenge" and the introduction of "computer games are one means to encourage learners to learn" (Long, 2017). Previous experience from the Component Programming, Systems Development Project, Web Application Development, Application Development and Database Technologies modules will be used to develop this application.

1.3 Projected Evaluation Procedures

Data collection will be an instrumental part of the process, as the proposed project attempts to solve real world programming issues. The appropriate legal, social, ethical and professional guidelines will be implemented throughout the study. After the web application is created, user feedback will be collected from a select number of participants to gather information on whether the application has had any positive/negative impact on the experience of the user. This is very important, as students can find the process of learning to program to be a "difficult and even unpleasant task" without the proper "guidance and feedback" (Watson et al., 2011) given back to them. User feedback will also be used for the black box testing that will be discussed in the later chapters of the paper. The completion of these objectives, as well as the ones documented in the 'problem domain' and 'project objectives' sections, will determine whether the project has been successful.

2 LITERATURE REVIEW

2.1 Introduction

This section aims to outline the extensive research that was conducted to make educated arguments about the topic, prior to the product being completed. Any assumptions made are documented below and are supported by numerous academic journals that have previously studied the problem or bring up new issues that the product can then aim to solve. The topics covered in this section range from subjects such as primitive learning and teaching methods, to the impact user interfaces have on consumer experiences.

2.2 Introduction to User Interfaces

User Interfaces refer to the method or space in which a user and a computer "exchange information and instructions" (BBC, 2020). These are extremely important, especially when designing a modern web application, as most, if not all, of the front-end features require a certain level of user interaction. There are 3 main types of user interfaces; command-line, menu-driven, and the most modern, graphical user interfaces (GUI).

2.2.1 Command-line Interfaces

Command-line interfaces allow users to interact with a computer via command prompts. These interfaces focus more on productivity than looks and design, as apart from the window the prompt is housed in, there is not much in terms of design. This is an advantage of this type of interface, however, as it does not have the processing power or memory requirements as other types of more graphical interfaces. It also boasts "higher levels of performance" (Durham, 1998) than some of the other interfaces, and is particularly scalable, running on multiple operating systems. Disadvantages of this type of interface include, difficulty learning the commands needed to run the prompt, though these can easily be researched and implemented.

2.2.2 Menu-driven Interfaces

Menu-driven interfaces allow users to interact with computers via a menu-based design system. In this system, the user is offered a selection of simple menus, with one menu leading to another via the use of either a menu bar or full screen menu, the latter taking up the entire screen. Menu bars are still used today, as they are very user friendly and prevalent in web application design, with navigation bars being a staple among "website usability constructs" (Lee et al., 2012). They are also much easier to use than command-line interfaces, as they do not require the user to remember commands. Menu bars do have their downsides though. If implemented incorrectly, they can become quite irritating to use, especially if "there are too many levels of menus" to browse through.

2.2.3 Graphical User Interfaces

Graphical user interfaces are the most interactive of the 3 types, as they allow for users to navigate webpages at ease due to their usually, user friendly design. Most of the more popular modern operating systems, such as Windows XP and macOS use GUIs. They also implement some of the best features from command-line and menu-driven interfaces, incorporating operating system flexibility and navigation bars respectively. As expected, there are some shortcomings involved with the usage of GUIs. They do use a significantly higher amount of memory and processing power than the other 2 interfaces, meaning that they are not suitable for all PC's, especially older ones. Just like menu-driven interfaces, they can also be irritating to use if there are too many features, so it is best for the GUI to be as simplified as possible.

2.3 User Interface features

2.3.1 Attractiveness: colour

Colour is everywhere. Gone are the days of the black and white televisions, bland projection screens and grayscale computer monitors; the successful "commercial broadcasting of colour on TV on December 17, 1953" (Bellis, 2019) changed the world forever, as it also completely revamped user expectations regarding hardware and software development. It was a very influential discovery. Colour in itself "has the potential to elicit emotions or behaviours" (Cyr el al., 2010) in human beings. Colour theory also plays a large role in the design and creation of effective web pages. The same study by Cyr et al. (2010) found that "website colour appeal is a significant determinant for website trust and satisfaction" further emphasizing the importance of colour when designing the user interface. This can be instrumental in increasing user interactivity and trust in the system, as the user is more likely to stay on the webpage if they feel like it is secure, let alone being visually pleasing. Of course, a large number of considerations must be made whilst designing the colour pattern of a web application. Colour will only help to enhance a web application that is not "poorly designed" (Lanyi, 2017) or ineffective. The colours implemented into the designed application should also be boast a high level of “accessibility for colour-deficient users” (Lanyi, 2017), as they are still relevant consumers of content displayed on these web applications. Interface designers can benefit greatly by implementing colour use into their web applications. However, they must be vigilant and take precautions to make sure that the colour scheme used complies with accessibility standards to incorporate all users.

2.3.2 Clarity and Conciseness

Clarity and conciseness are two of the most important features of user interface design. User interfaces are made to be interacted with; however, if a user cannot understand how to use the given interface, then it defeats the purpose of having one to begin with, as it makes user tasks difficult to execute as opposed to simplifying them. A study by Kusumawati et al. (2020) supported this theory, as their findings concluded that user interfaces that are “too confusing” make the specific system too “difficult to operate”. The designed web application must have clear functionality features due to this. The developer must also be careful to avoid over-clarifying the features on the GUI, as it increases the size of the interface, increasing the amount of content the user has to go through. A study by Knupfer et al. (1997) compared multiple types of web application interfaces with similar quality levels, and was able to conclude that sites that “tended to use text rather than images to present information” especially school websites, interfered with the “overall message” that those applications aimed to convey to the users. The “overuse of design backgrounds” (Knupfer et al., 1997) also interfered with the overall purpose of the application, so developers must be aware to avoid overpopulating their GUIs.

2.3.3 Familiarity and Consistency

Familiarity in design is a crucial feature for user interfaces, especially when paired with the considerations the developer has to account for regarding the clarity and conciseness of their application. The developer can bypass and/or simplify the act of making their user interface understandable by making it ‘intuitive’, which involves the “subconscious application of prior knowledge by the user” (Blessing et al., 2007) to effectively learn how to use the system. This is done by designing the user interface in a way that is familiar to other products that the user might have seen before, as they will be more likely to know how the system behaves due to their prior knowledge. Largely successful companies like Microsoft are known to do this a lot across most of their platforms. Microsoft applications feature similar menu designs regardless of application (Word vs. PowerPoint) or operating system (Windows vs. MacOS), with very minor changes made to accommodate the differences in the latter example. They also reuse designs within their own hardware/software. The popular Xbox One dashboard design features a "Windows-based operating system" (Khanji et al., 2016) first implemented on PC's and Windows phones years prior to the release of the gaming console. Developers must consider the benefits of implementing designs that are similar to other products whilst still involving extensions to their own applications, as it ensures that the user can effectively transition into using the newly designed product.

2.3.4 Responsiveness and Efficiency

Responsiveness and efficiency are arguably two of the most important user interface features concerning web application development, as they are both directly implicated regarding the usability of the system. "Enhanced user satisfaction" (See et al., 2015) plays a big role as well, as users often come up with their own "perceived" opinions, as well as pros and cons within a system. The developer must make sure that usable aspects of the application, such as button clicks, should provide feedback to the user regardless of the result of the action. The system must also boast reasonable performance variables; these being "effectiveness, efficiency, duration (time), number of errors and number of helps" (Feizi et al., 2012) the system provides the user. In terms of efficiency, the user interface must make it easy for the user to "easily accomplish what they want, instead of simply implementing access to a list" (Fadeyev, 2019) of efficiency features. The takeaway from this section is; the easier the interface developed is to use, the more efficient it will be, and developers must make sure to solidify ease of access.

2.4 User Experience features

2.4.1 Human Computer Interaction

This chapter aims to discuss the importance of human computer interaction (HCI) factors when designing a web application, especially one that focuses on user interface and experience designs like the final product will do. The purpose of human computer interaction is to ensure that a project has "safe and usable systems that function effectively" (Userhub, 2019); this is done by prioritizing the user's needs and understanding the obstacles that user's face while using technology on a day to day basis. Some of the main HCI factors include:

organizational, task, environmental, comfort, user, functionality and productive factors. For the sake of this study, only the most relevant HCI factors will be discussed in full detail.

2.4.2 Usability and task factors

Usability and user experience go hand in hand. The usability factors when discussing the topic of web development, refer to the enjoyment, satisfaction and motivation levels that the user experiences whilst using the application. This, more than definitely, defines the term 'user experience', though modern studies tend to consider UX to be an extension of usability" (Rusu et al., 2015), and not the other way around. This opinion is subjective; nonetheless, the importance of usability has never been underplayed, especially as applications become more user centric. Task factors include monitoring task allocation for the processes that take place within the system. Developers must prioritize "user-centred design" and develop "good practice in user interface design" (Bevan, 2001) to ensure that usability and task factors are accounted for.

2.4.3 Organization and Comfort factors

Organization and comfort factors refer mainly to the layout of the system that the user is interacting with. Modern web applications have adopted a mixture of the menu-driven and GUI interface designs in their structure, and this has been the norm for a number of years now. This plays favourably regarding comfort factors, as users are more likely to use applications that they are more familiar with. The example of Microsoft's scalability will be used again, due to its high success rate and usability of applications such as Microsoft Word to this day. Their ability to continually take "significant steps to improve the security and privacy of their products" (Charney, 2008) especially as these risks become more prevalent, has garnered a large amount of user trust/comfort in the company. Developers must make sure to prioritize the best interests of the user, as they will reap the rewards of consumer trust later on.

2.4.4 Environmental factors

The only environmental factors that are to be considered whilst developing a web application involve the issue of lighting; this can be directly linked to the conversation regarding colour schemes in section 2.3.1 of the paper, with lighting being interpreted as the brightness of the web application. A study by carried out Camgöz et al. (2002), aimed to investigate the effects of hue, saturation and brightness on web application design preference. Subjects of the study were presented 8 different colour squares and were to choose which they would prefer in a browser/web application. Results showed that colours having "maximum saturation and brightness were most preferred" (Camgöz et al., 2002) over their darker counterparts. Astonishingly, the study also concluded that the colour 'blue' was the "most preferred hue regardless of background", and it played a big part in the decision to choose blue as the colour variant for the final prototype web application.

2.5 Programming issues

2.5.1 Differences in teaching methods and learning styles

Various methods have been used to teach programming; some are more outdated and/or successful than others, but all were created to achieve the same goal. The conventional teacher-student model is the oldest and most common. This model is quite self-explanatory, though it does branch out into practices such as apprenticeship, more commonly known as 'placement', where the student is offered "continuous feedback" by one or more mentors to not only differ from "traditionally constructed courses", but also offer the "most efficient means of learning" (Vihavainen et al., 2011). Another form of learning is self-study, in which the student teaches themselves the required material using various means (ex: books, online websites, games, journals etc.). A study by Bergin et al. (2005) sheds a brighter view on self-regulated learning, stating a direct link between intrinsic motivation and SRL (Student Regulated Learning), with students exhibiting such behaviours being more inclined to "perform better in programming" than those with more extrusive motivational factors. Learning via web applications best fits this category, as it requires the student to explore more about the given subject in their own time.

Learning to program can be difficult, especially for beginners and novices. In addition, the notion that 'once you learn one programming language, another will be easy to learn' is subjective, as learning styles differ depending on the individual. As promising as the field of study is, programming courses often have the "highest dropout rates" (Robins et al., 2003). It is not easy at all. Multiple adaptations to the learning process have been used, such as the concept of pair programming mentioned above, as well as different types of programming completely. Robins et al. (2003) also explore the differences between object-oriented and procedural programming, the latter being taught less frequently than the former in today's day and age due to the ''naturalness, ease of use, and power'' of the OO approach.

2.5.2 Programmers are not learning

One of the key problems regarding the motivation of young programmers, is the fact that programmers are not learning in the first place. This is a problem that has no one cause and/or solution; it would be extremely lazy and simplistic to individually blame any of the key players involved in developing young programmers, whether it be the students themselves or the organizations tasked with teaching them. However, acknowledging that change is needed on both sides is crucial, as students are responsible for learning, and universities are responsible for producing material that is updated and challenging. This material should also still be engaging enough for students to stay interested as there is a "vast divide between those who can program and those who cannot program" (Atwood, 2007), meaning that there are students and universities that are thriving using the current learning model. On the other side of the spectrum, however, are the students that are far behind them, and risk falling behind even more due to the "ever-changing landscape" (Conole, 2008) of the technological field.

2.5.3 External non-learning motivation factors

The demographic of the computing field is an increasingly changing and diverse one, as the students involved in "Higher Education is increasing in numbers" and students originate from a "wide variety of backgrounds" (Jenkins, 2001). This brings to the table a large fluctuation of motivating factors regarding the decision to pursue a career in the technological field. Extrinsic factors regarding higher education, are motivational factors that involve the "desire to complete the course in order to attain some expected reward": These differ to intrinsic factors, which derive from "an interest in the subject" (Jenkins, 2001). The extremely monetizable nature of the tech field, coupled together with the promise of better jobs working for the biggest companies, unfortunately mislead some young students into a practice in which they would have had little to no interest in otherwise. This can greatly impact the learning ability and workplace preparation of a student, especially if the main goal is to benefit financially as opposed to developing the practical skills needed to succeed.

2.5.4 Programmers are giving up too quickly

This issue is a direct correspondent to the one listed above, as individuals are less likely to stick to a task if they are not fully invested into it. Students are also less likely to persevere when overwhelmed and/or frustrated by their work. These frustrations can be detrimental not only to students, but also in the workplace, as they can cause "personal dissatisfaction and loss of self-efficacy" as well as "slow learning" and "reduced participation" (Hackley et al., 2005). A programming solution is a concept such as pair programming; Paired programming involves a student working in a pair in order to meet a programming target. This is helpful as it allows partners to "encourage and motivate each other when they would otherwise give up" (Draper et al., 2004). This approach continues to be successful, even applied to the Advanced Programming module at the University of Greenwich, as it provides "collective ownership" and "better diffusion of knowledge" (Wood, 2018) while working.

2.6 Legal, Social, Professional and Ethical issues (LSPEi)

There are a variety of potential legal, social, professional and ethical issues that can arise whilst developing a mainstream web application, let alone a prototype. The numerous and rapid advances in technology since the 20th and beginning of the 21st century have brought along massive changes regarding the standards and tools available to create such applications. However, they have also made way for a new stream of legal, social, professional and ethical issues that had not existed before. This section aims to discuss some of these issues. Due to the large variety of potential (LSEPi) that can affect web application development, only the most prevalent and probable issues will be investigated.

2.6.1 Legal and Ethical Issues

A large number of legal and ethical web app development issues are actually quite intertwined. A common legality issue involves the illegal copying and usage of third-party content. The use of images, logos, products etc. are all subject to the most recent copyright infringement laws; infringement occurs when a "copyrighted work is reproduced, distributed, performed, publicly displayed, or made into a derivative work without the permission of the copyright owner" (Copyright.gov, 2020). This can only be avoided if the creator provides a written agreement in which they "assign the copyright to another person or entity". This is also an ethical issue, as the party breaching the copyright rules is often aware that the content they are using does not belong to them. Failure to properly cite and reference third-party materials can lead to lawsuits from the accusing party, so these issues must be avoided while developing the final product. All third-party content should be referenced accordingly.

2.6.2 Social Issues

The majority of social issues regarding web development involve accessibility, with the precedent being the task of affording all users the same equal access while browsing the web. The Disability Discrimination Acts of 1995 and 2010 expand more on these issues, with the latter document stating that any features on the given application must not make it "impossible or unreasonably difficult" (UK Government, 2010) to use. It also states that the application must maintain diversity and allow for a complex "range of accessibility and usability functions for all users". Features such as cookies must be implemented cautiously here, as personalized information saved from them based on user location, "browsing habits, age, marital status and political and religious affiliation" (UK Government, 2010) can contradict the Disability Discrimination Acts. Developers must make sure that any data collecting tools, such as cookies, are implemented fairly and correctly.

2.6.3 Professional Issues

The professional issues that can affect the development of a web application require special attention to be paid to the HCI factors; the developer is always going to have professional obligations to the user. A web application must be able to "maintain confidentiality, maintain anonymity and respect copyright" (Kabay et al., 2015) to the user. Failure to do this can lead in loss of consumer trust in the product, negatively affecting the credibility of the developer. The developer also has an obligation to the professional governing bodies; they must make sure to comply with the computing standards of whichever country they are based in/develop for. Developers in the UK must comply to the professional standards set by The Chartered Institute for IT, also known as the BCS. BCS members are required to hold a certain "duty to the profession" (BCS, The Chartered Institute for IT, 2020) regarding IT development. This involves "upholding the reputation and good standing of the BCS" as well respecting "professional relationships" with other development teams. Adhering to these expectations ensures that the developers do not encounter any professional issues whilst doing their work.

3 REVIEW OF SIMILAR PRODUCTS

3.1 Introduction

This chapter aims to provide some background research regarding some of the most popular similar products available to the public. This is important, as it provides a fair comparison between current/after-market products and the one that is being developed. It also allows for the strict development of an adequate quality assurance plan, as there is a 'direct competitor' to make contrasts from. It opens an avenue for criticisms about these products as well, and these can then be implemented as potential fixes in the final product. There was a wide range of existing products online. For the sake of time management and considering the short-term nature of the project, three real life products were narrowed down and chosen for review:

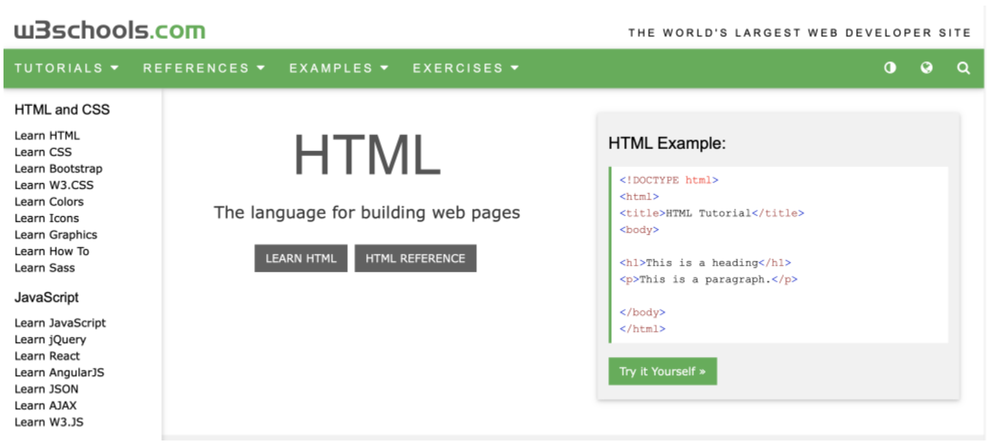
Product Name	Website URL
W3Schools	https://www.w3schools.com
CodeCombat	https://codecombat.com
CodeMoji	https://www.codemoji.com

3.2 W3Schools

W3Schools is a website that provides tools for learning web technologies online. It is the most easily accessible of the 3 examples given, as it does not require the user to create an account in order to learn and benefit from the information provided. The website offers information tutorials and references on over 20 programming languages and topics. The product is relatively good for motivating programmers, as it does not force the user to undergo any type of level progression to learn new content, allowing the user to try other solutions if any problems arise. W3Schools does offer an interactive programming experience via its 'TryIt Editor' via a button named 'Try it Yourself', by which the user is taken to a separate page where they can interact with and edit the source code provided and try the results. W3Schools differs from the proposed system a tiny bit, as its interactivity features end at the user being able to edit and view the results of the code. The topics are provided for the user and they more or less have to fend for themselves to learn the content. They do provide quizzes so that the user can test what they have learned using the content, but it is not compulsory, nor is it unavoidable.

The quizzes and exercises also track the users progress only while they are actively browsing the site; the lack of user registration/profiling means that this progress is not saved, and a new instance of the webpage will refresh the results previously achieved by the user.

Appearance



w3schools.com THE WORLD'S LARGEST WEB DEVELOPER SITE

TUTORIALS ▾ REFERENCES ▾ EXAMPLES ▾ EXERCISES ▾

HTML and CSS

Learn HTML
Learn CSS
Learn Bootstrap
Learn W3.CSS
Learn Colors
Learn Icons
Learn Graphics
Learn How To
Learn Sass

JavaScript

Learn JavaScript
Learn jQuery
Learn React
Learn AngularJS
Learn JSON
Learn AJAX
Learn W3.JS

HTML

The language for building web pages

LEARN HTML HTML REFERENCE

HTML Example:

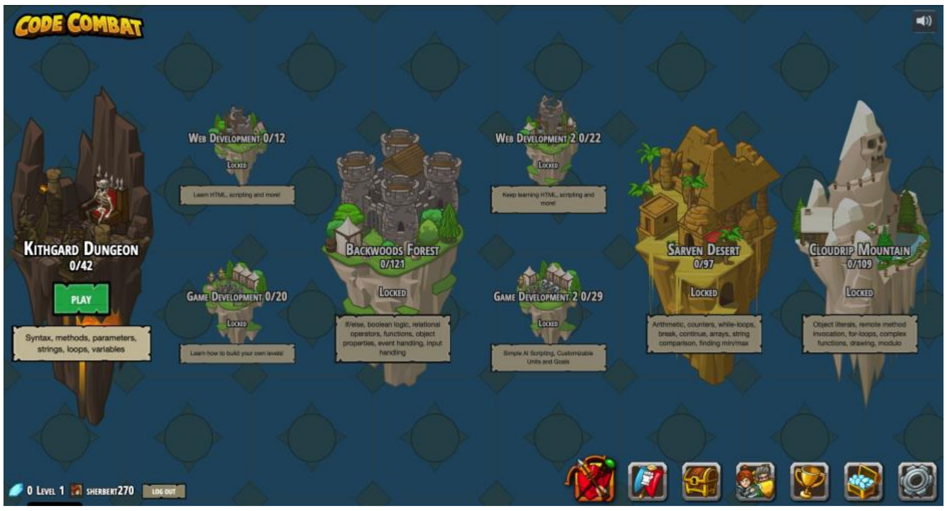
```
<!DOCTYPE html>
<html>
<title>HTML Tutorial</title>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Try it Yourself »

3.3 CodeCombat

CodeCombat is a web application that aims to help the user learn typed code via the use of programming games, quizzes and puzzles. The most recent version of the web application uses a level-based system to teach basic principles such as syntax and variables, and progresses in difficulty as the user reaches higher levels. CodeCombat is a highly interactive and innovative learning tool, as it gives the user a great deal of customizable features such the choice to create an avatar, or join a clan, providing the user with more motivation factors other than learning to program as they can compete with other clans. Users are also able to build their own levels to best suit their learning needs. The GUI is excellent, colourful and very appealing, especially for younger users.

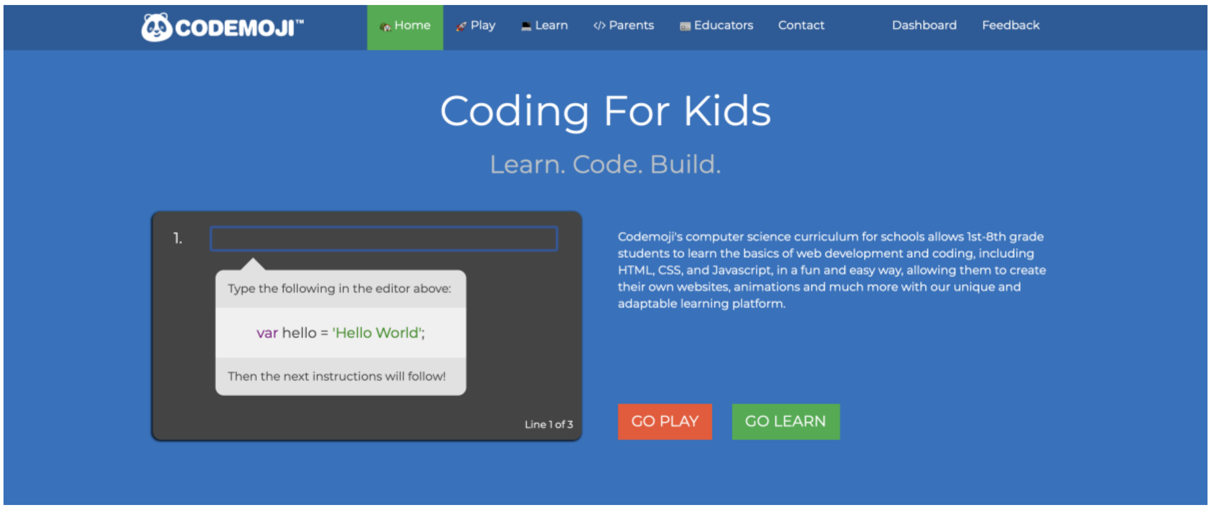
Appearance


CODE COMBAT
Kithgard Dungeon 0/42
PLAY
Syntax, methods, parameters, strings, loops, variables
Web Development 0/12
Locked
Learn HTML, scripting and more!
Game Development 0/20
Locked
Learn how to build your own levels!
Backwoods Forest 0/121
Locked
If/else, boolean logic, relational operators, functions, object properties, event handling, input handling
Web Development 2 0/22
Locked
Keep learning HTML, scripting and more!
Game Development 2 0/29
Locked
Simple AI Scripting, Customizable Units and Goals
Sarven Desert 0/97
Locked
Arithmetic, counters, while-loops, break, continue, arrays, string comparison, finding min/max
Cloudrip Mountain 0/109
Locked
Object literals, remote method invocation, for-loops, complex functions, drawing, modulo
0 Level 1 sherbert270
Log out

3.4 CodeMoji

CodeMoji is a web application that just like CodeCombat, allow users to learn code using quizzes and games; the only exception, is the fact that it implements emojis into the act of programming, making it a much more appealing to younger users. It also boasts a high level of interactivity, as the user is able to create projects and view leader boards. It, however, does not offer as many customizable features as the CodeCombat web application, as the user cannot create an avatar or compete in groups as part of a clan. The leader board function means that the user can individually compete with other though, adding to the extrinsic motivating factors that they may have regarding the task of programming.

Appearance



Coding For Kids

Learn. Code. Build.

Type the following in the editor above:

```
var hello = 'Hello World';
```

Then the next instructions will follow!

Codemoji's computer science curriculum for schools allows 1st-8th grade students to learn the basics of web development and coding, including HTML, CSS, and Javascript, in a fun and easy way, allowing them to create their own websites, animations and much more with our unique and adaptable learning platform.

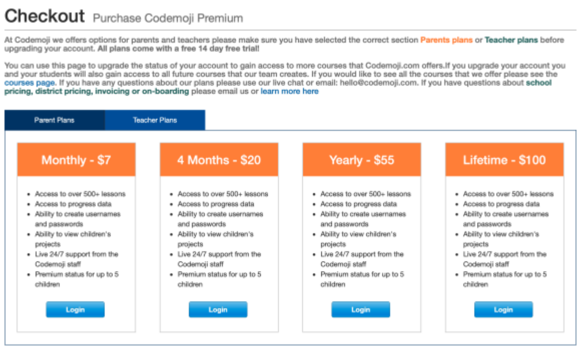
3.5 Adaptations from similar products

The review of the products mentioned above was pivotal to the project, as it helped to define the system requirements defined in chapter 4 of the paper. A result of this was the creation of a web application that aimed to boast the best features from all 3 websites. These include but are not limited to; the accessibility of W3Schools, the game-based learning approach and personalization features of both CodeCombat and CodeMoji, and finally the simple yet visually pleasing and interactive UX/UI design of the CodeMoji GUI.

3.5.1 Accessibility of W3Schools

Being a previous consumer of content from W3Schools would usually bring along a level of bias that is not good for any professional study. However, in this case, it acted as an eye opener regarding how convenient the website is to use. As mentioned before, the website does boast over 20 programming languages and topics. Features such as the ability of the user to search up the information they need, find it within seconds, and leave the website immediately without any type of restricted access is very attractive. This boosts the accessibility of the application, as the user is able to access the learning material without user registration. Also, not every user wants to create an account either. This is definitely a negative regarding applications like CodeCombat and CodeMoji, as even registered users are subject to signing up for 'premium memberships' to access specialized content that is not available to all users. W3Schools eliminates all instances of situations like this, and this feature is implemented into the final application's contingency plan. W3School's ability to enhance the knowledge of the user via an interactive practice code editor is the main adaptation implemented into the final web application.

Example of premium memberships



Checkout Purchase Codemoji Premium

At Codemoji we offers options for parents and teachers please make sure you have selected the correct section Parents plans or Teacher plans before upgrading your account. **All plans come with a free 14 day free trial!**

You can use this page to upgrade the status of your account to gain access to more courses that Codemoji.com offers.If you upgrade your account you and your students will also gain access to all future courses that our team creates. If you would like to see all the courses that we offer please see the courses page. If you have any questions about our plans please use our live chat or email: hello@codemoji.com. If you have questions about **school pricing, district pricing, invoicing or on-boarding** please email us or learn more here

Parent Plans | Teacher Plans

Monthly - $7
- Access to over 500+ lessons
- Access to progress data
- Ability to create usernames and passwords
- Ability to view children's projects
- Live 24/7 support from the Codemoji staff
- Premium status for up to 5 children

Login

4 Months - $20
- Access to over 500+ lessons
- Access to progress data
- Ability to create usernames and passwords
- Ability to view children's projects
- Live 24/7 support from the Codemoji staff
- Premium status for up to 5 children

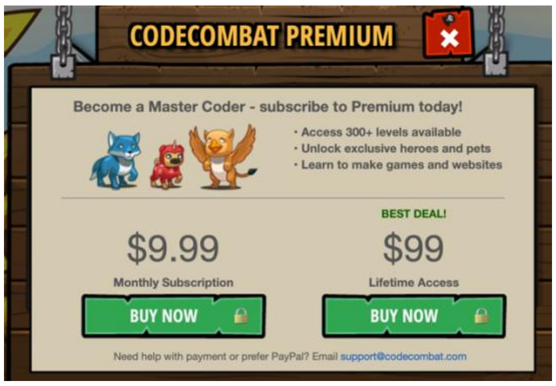
Login

Yearly - $55
- Access to over 500+ lessons
- Access to progress data
- Ability to create usernames and passwords
- Ability to view children's projects
- Live 24/7 support from the Codemoji staff
- Premium status for up to 5 children

Login

Lifetime - $100
- Access to over 500+ lessons
- Access to progress data
- Ability to create usernames and passwords
- Ability to view children's projects
- Live 24/7 support from the Codemoji staff
- Premium status for up to 5 children

Login



CODECOMBAT PREMIUM

Become a Master Coder - subscribe to Premium today!

- Access 300+ levels available
- Unlock exclusive heroes and pets
- Learn to make games and websites

$9.99
Monthly Subscription
BUY NOW

BEST DEAL!
$99
Lifetime Access
BUY NOW

Need help with payment or prefer PayPal? Email support@codecombat.com

3.5.2 Interactivity of CodeCombat and CodeMoji

CodeCombat and CodeMoji approach the task of teaching programming a bit differently than W3Schools, opting to add features that are more interactive than the quizzes the latter uses, such as user-inclusive level-based games. The addition of customizable personalization features, such as avatars, only separates them even more. A decision was made to create a game-based application similar to both CodeCombat and CodeMoji. The game created would look more like CodeCombat however, as it features full on avatars that resemble human characters as opposed to the emojis used by CodeMoji.

3.5.3 CodeMoji's visually pleasing UX/UI design

All three websites featured very visually pleasing user interface designs. However, CodeCombat and CodeMoji designs were much more enhanced to provide a better user experience while browsing the application. The addition of music, sound effects, and visible logos are much more abundant and noticeable in the latter two applications compared to W3Schools. However, CodeCombat seems to slightly overdo it with the number of website designs used. Some webpages are only distinguishable as CodeCombat pages due to the logo in the top left corner and can easily be mistaken for other gaming websites without it. CodeMoji does a much better job of having a simple consistent layout while still bestowing other interactivity features. The music that plays in the background of the CodeCombat application can also become quite irritating after extended use of the application, an issue that CodeMoji does not have to worry about, as background music is only used for gameplay. Due to these grievances, a decision was made that the best market product to base user interface and experience designs on would be CodeMoji, while also forfeiting the task of adding background music to the page when not in-game.

4 SYSTEM REQUIREMENTS

4.1 Introduction

This chapter aims to discuss the considerations regarding the requirements for the proposed web application. Detailing these requirements is important, as they are essential for all components of a system to be used efficiently. The requirements discussed below were derived from a variety of relevant sources, including but not limited to; previous coursework requirements, previous studies, the software development process and the review of similar products detailed in the previous chapter. These requirements helped to ease the process of UI and UX design, as it allowed for features that were essential to the application to take precedent over other more desirable yet less necessary features.

4.2 Analysis of Requirements

In the context of designing a software system, the system requirements have been split into two categories; functional and non-functional requirements. Functional requirements describe the different processes that a system should be able to carry out. Non-functional requirements determine how the system will carry out the processes defined by the functional requirements. These are the equivalent of ‘product plans’ and ‘process description’ respectively in the software development process. The requirements listed below will also be ranked based on importance using the ‘MoSCoW method’, which is a prioritization technique used in project management to rank the importance of the given requirements. MoSCoW is an acronym that stands for 4 different initiatives regarding requirements, entailing the ‘MUST’ haves, ‘SHOULD’ haves, ‘COULD’ haves and ‘SHOULD’ haves of a system. Shown below are the functional and non-functional requirements of the product.

4.2.1 Functional requirements

Functional requirement		MoSCoW Category
1.	Acceptable and user inclusive UX/UI	MUST
2.	User access to default homepage information without login	MUST
3.	Successful user registration and account creation	SHOULD
4.	Successful user login using email and password	SHOULD
5.	Successful user logout from application	SHOULD

6.	Successful connection between application and database	SHOULD
7.	Successful password change by user	SHOULD
8.	Successful navigation of application by user	MUST
9.	Parsing of user info from page to page during navigation	SHOULD
10.	User access to topics on home page	SHOULD
11.	User access to application contact information	COULD
12.	User access to learning information while using application	MUST
13.	User ability to play game or quiz	MUST
14.	Database ability to store application information	SHOULD
15.	Database ability to allow alteration of application information	SHOULD
16.	System ability to avoid third party invasion	SHOULD

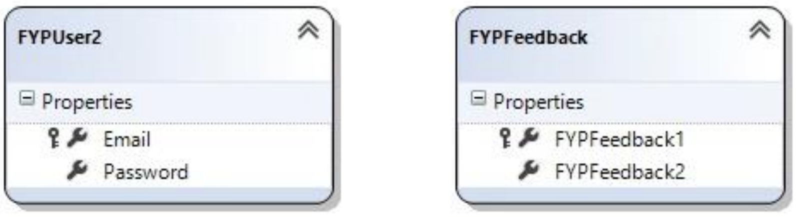
4.2.2 Non-functional requirements

Non-functional requirement		MoSCoW Category
1.	Usability: The application must have modern user interface designs consistent with modern websites, increasing overall usability.	MUST
2.	Ease of Access: The application should have a consistent layout across all webpages, being as easy to understand as possible.	SHOULD
3.	Interactivity: The application could boast multiple interactivity features to keep the user engaged, such as avatar and character creation/selection.	COULD
4.	Privacy and Security: The application must protect the personal information of the users and store it into the database securely.	MUST
5.	Accessibility: The application must be accessible by all users regardless of browser, operating system and more important circumstances such as catering for users with disabilities.	MUST
6.	Reliability: The application should be reliable and comparable to other web pages that are available on the market.	SHOULD
7.	Performance benefits: The program should perform well enough to avoid performance flaws.	SHOULD
8.	Innovativeness: The application could strive to add features that are lacking from similar real-world products.	COULD

5 SYSTEM DESIGN

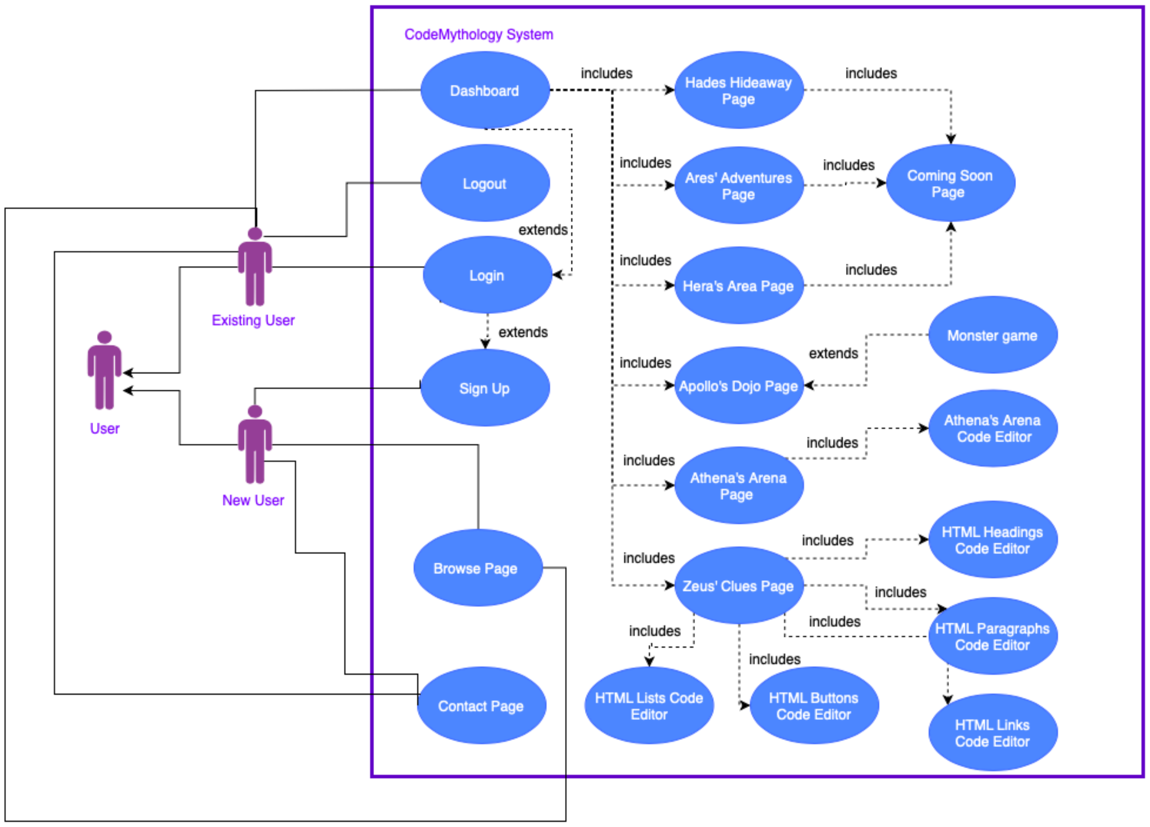
5.1 Entity Relationship Diagram (ERD)

Entity Relationship Diagrams are an important component of system design, as they aid in the "structured analysis and conceptual modelling" (Krippendorf et al., 1995) of the system. The ERD approach is also "easy to understand, powerful to model real-world problems, and readily translated into a database schema" in addition to being commonly used. The only tables that were created were to facilitate user login, registration and collect feedback via the 2 contact pages. The feedback pages were created with the intention of collecting anonymous responses, so there was no need for them to be linked to any other tables designed in the system. The table designs were also simple enough to not require any normalization, and there were no issues inserting or selecting data from the database using LINQ. Shown below is the ERD for the system:

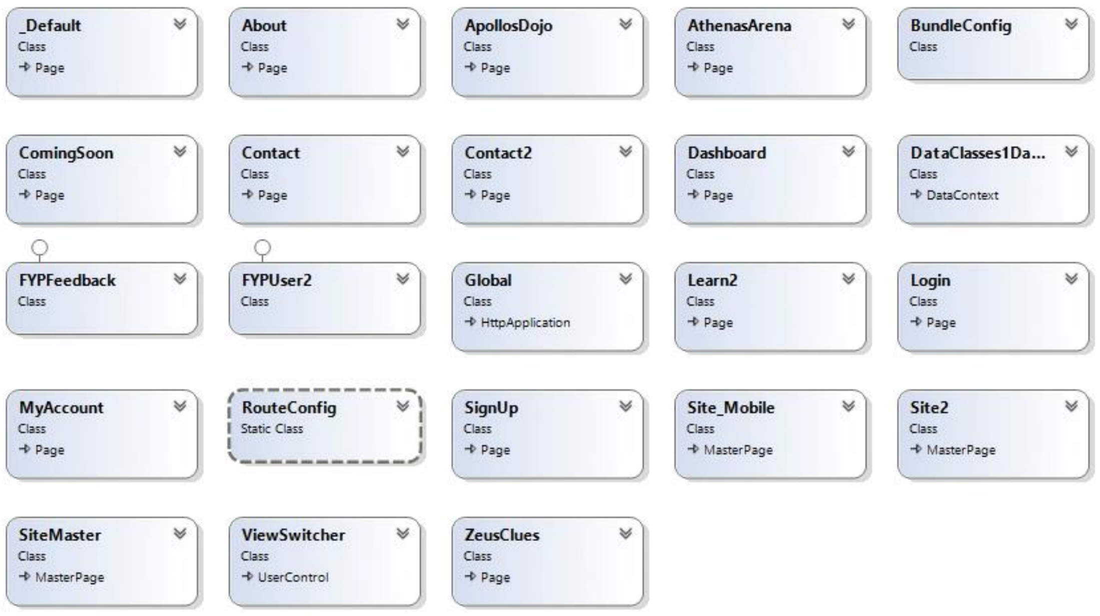


5.2 UML Use Case Diagram

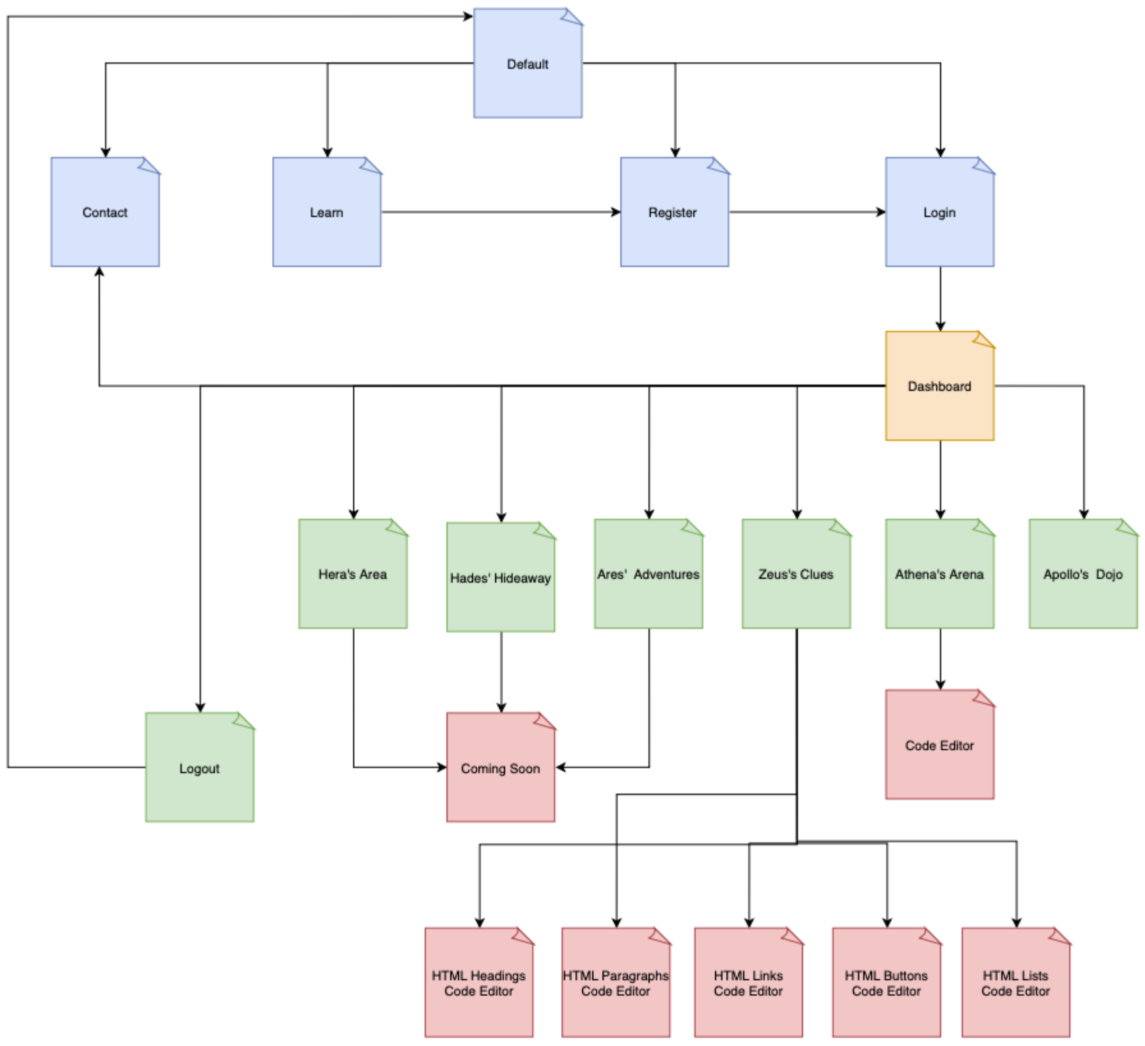
Use case diagrams are an important aspect of system design, as they define the interactions between actors (human or external system) and the system. Use cases can be used with or without supporting use case diagrams. However, a study by Gemino et al. (2009) found that participants viewing use cases with "the supporting diagram developed a significantly higher level of understanding" when evaluating the system. The diagram created shows the actions available for the user interacting with the web application. Admin functions are handled using backend functionalities, as there is no front-end admin application created for the prototype. This means that there is only one actor for the web application: the user. Shown below is the completing UML use case diagram:



5.3 UML Class Diagram



5.4 Site Map Diagram



6 SYSTEM DEVELOPMENT

6.1 Introduction

This chapter aims to discuss the system development procedures that were decided upon while creating the product, as well as the literal steps that were taken to complete the product. A decent amount of time was spent looking at existing products, and due to the fact that the proposed application was based on user interaction and design, it was imperative that the best modern examples were found for comparison. These were also used as some of the base designs of the final product. There are a number of different programming languages, IDE's and database management systems that can suffice regarding the task of web application development. However, due to prior knowledge, previous assignments and suggestions from project supervisor, the system was developed using Visual Studio 2019. The programming languages used are C#, HTML5, CSS and JavaScript. Database management is handled using Microsoft SQL Server and LINQ.

6.2 Systems Development Methodology

This section aims to detail the systems development methodologies considered during the planning of the proposed project. Regarding development models, the 'extreme programming' and 'spiral' models were eliminated immediately due to the nature of the task at hand. The 'waterfall' model was also eliminated, as it is a model that does not promote user interactivity until the end of the project cycle, a big staple in the production of this application. It is also aimed at large and expensive projects, two variables that do not describe the proposed idea. The system development methodology that would best suit this project is a close call between the 'agile development' model and the 'rapid application development' (RAD). Both models are low costing, relatively defined and informal, and have a "small-to-medium scale and of short duration" (Kyriakidou, 2018). However, the negatives of the agile development model centred around the fact that it involves customer collaboration at a much higher level than the RAD model. Given the time constraints, user interactivity and complexity of the project, the conclusion was made that the best systems development model to follow would be the rapid action development (RAD).

6.3 Web Application Development

The application was developed using a very different approach compared to systems that had been built before; this was mostly due to a previous assignment earlier in the year, in which a less than acceptable UI design was submitted. Considerations were made to ensure that the same fate did not befall this project, especially since the main research area revolved around user interface and experience design while using the application. So, in order to remedy this situation, the first task at hand was to find the similar products detailed in chapter 3 of the report, as these would become the references for the design of the prototype and final web applications. Applications such as CodeMoji were very influential to the design of the application, with features such as the colour scheme, dashboard carousel and avatar/character selection being adopted into the design of the application. These similarities have been documented in Appendix C.

Ordinarily, the first steps would have been to design an ERD diagram, normalize the diagram, and then build the database based on the final ERD. However, after viewing web pages similar to W3Schools, a decision was made that the database designed was only needed for user/admin login, as websites such as W3Schools allow users to access their content without user registration. It was also implemented as a potential contingency, provided there were database integration problems at any time during the development of the product, allowing for total functionality of the webpage without database connection. This would have still been a valid implementation of the product, as it would have adopted features from the real life W3Schools website, and it would have still satisfied the main primary and secondary objects; providing the user with the necessary content to practice programming on a modern interactive application. Due to this, a prototype management system was created, with the most essential features of the application being focused on in the first prototypes, and the more trivial features being added to the latter and final versions.

6.3.1 Prototype 1

Prototype 1 was the most time consuming as it was created from scratch, using only the basic web forms template and the designs adapted from the researched similar products. As mentioned before, the colour scheme of the webpage closely resembles that of CodeMoji. The analogous colour scheme was also checked for compliance with the WCAG 2.0(AA) guidelines, using an accessibility colour wheel created by ‘Altervista’, found at https://gmazzocato.altervista.org/en/colorwheel/wheel.php. The previously mentioned basic web forms template also handled issues regarding responsiveness; issues like the resizing of the application window were handled without too much hassle. Default documents like bootstrap and master pages were also already created and linked and were easily editable with the proper knowledge. It was imperative that the web forms template was used rather than an empty website format, as programming the website completely from scratch would have likely been too time consuming and caused delays in the completion of the project. Prototype 1 was created mainly to test navigation features regarding usability, productivity, organization, comfort and environment factors, 5 of the 6 previously mentioned HCI factors in chapter 2 of the report.

6.3.2 Prototype 2

Prototype 2 was created after the user interface of the application had been finalized, focusing less on frontend features like the GUI, and more on the backend development of the application. This prototype integrated the interactivity features of the application, such as character selection and page navigation with the more functional elements of the application, such as validation and database connectivity/functionality. Luckily, the validation for most features on the application (textbox filling, email validation, password confirmation) was handled by the borrowed code from the Microsoft ASP.NET practice application, ‘WingTip Toys’. These were already available in prototype 1 and were used to create the “Sign Up’, ‘Login’ and ‘Settings’ pages. However, there was no validation to check whether or not the user signing in had an account, and access was granted to any user with a valid email input in the registration textbox. Validated user registration and login via SQL Server using LINQ, as well as parsing of user information from page to page was the main development of this prototype. The pre-generated CSS files that are created as part of the default web form design were also quite helpful, as they helped in the completion of prototype 2. Minor changes were made to the actual design of the application during prototype 2.

6.3.3 Prototype 3

This was the final prototype, and it involved developing the final interactivity features that were promised to the user, such as quizzes, games and a code editor to practice programming effectively. A few more changes were also made regarding the GUI design and user interface. Black box testing was also carried out, and the results were compared to the ones garnered from the previously carried out white box tests. A few different versions of the quizzes and games were also tested for functionality, and the best fitting ones were added to the final application. This also required a change in some of the visual designs of a few pages, but nothing was drastic enough to report. An overall evaluation and reflection of the final system was carried out, as well as an analysis of system bugs, failures and recommendations for future studies.

7 TESTING

7.1 Introduction

This chapter aims to discuss the considerations regarding testing requirements for the completed web application. Testing is important, as it determines the quality of software after a programmer develops it. A study by Janzen et al. (2006) found that software projects that inherit a test-driven development (TDD) approach tend to demonstrate benefits over traditional approaches during testing, boasting benefits such as better “external quality” and more “developer-centric” aspects; thus improving both “productivity and confidence” on the developer side. The negatives of this type of testing, however, include developer “motivation and discipline” (Janzen et al., 2006) as it is a very rigorous and frequent form of testing. Due to this, a decision was made that the best type of testing environment would require a hybrid of both TDD and the traditional testing approach; with iterative undocumented tests throughout development, leading up to a documented linear last test upon completion of the web application.

7.2 Analysis of Testing Procedures

In the context of designing a software system, the testing requirements have been split into two categories; white box and black box testing. White box testing is a software testing method in which the internal design is known to the tester. This approach can also be used to “generate test cases” (Ostrand, 2002) for the black box testing; a method in which the tester is unaware of the specifics of the system. Detailed below are the test cases created for the white box testing of the application.

7.2.1 White box testing

Test Case		Expected Result	Actual Result	Success?
1.	Run the application	Direction to default/home page	Direction to default/home page	Yes
2.	Access ‘Contact’ page without user registration	Redirection to contact page	Redirection to contact page	Yes
3.	Access ‘Learn’ page without user registration	Redirection to learn page	Redirection to learn page	Yes

4.	Access ‘Login’ page without user registration	Redirection to login page	Redirection to login page	Yes
5.	Access ‘Sign Up’ page without user registration	Redirection to registration page	Redirection to registration page	Yes
6.	Access ‘Dashboard’ page	Redirection to dashboard page only via successful login	Redirection to dashboard page only via successful login	Yes
7.	Incorrect email format during registration	Error message requesting proper format	Error message requesting proper format	Yes
8.	Non-matching passwords during user registration	Error message requesting proper format	Error message requesting proper format	Yes
9.	User login with non-existent user login details	Error message requesting valid details	Error message requesting valid details	Partial
10.	User login with validated user login details	Redirection to dashboard page	Redirection to dashboard page	Yes
11.	View lessons available on ‘Dashboard’ page.	Redirection to dashboard page only via successful login	Redirection to dashboard page only via successful login	Yes
12.	Choose a character via selection menu on ‘Dashboard’ page	Successful character selection via arrow keys provided	Successful character selection via arrow keys provided	Yes
13.	Access to ‘Zeus’ Clues’ learning page on dashboard	Redirection to progress page whilst logged in	Redirection to settings page whilst logged in	Yes
14.	Access to code editor via learning pages	Redirection to code editor page	Redirection to code editor page	Yes
15.	Successfully use code editor to practice HTML	Working code editor on page	Working code editor on page	Yes
16.	Access to ‘Apollo’s Dojo’ game page	Redirection to game page	Redirection to game page	Yes
17.	Successfully play ‘Apollo’s Dojo’ game	Working game on page	Working game on page	Yes

18.	Access to remaining learning pages	Redirection to 'Coming Soon' page	Redirection to 'Coming Soon' page	Yes
19.	Use 'Log Out' button to exit application	Redirection to default/home page after successful log out	Redirection to default/home page after successful log out	Yes

7.2.1 Black box testing

Test Case		Expected Result	Actual Result	Success?
1.	Run the application	Direction to default/home page	Direction to default/home page	Yes
2.	Access 'Contact' page without user registration	Redirection to contact page	Redirection to contact page	Yes
3.	Access 'Learn' page without user registration	Redirection to learn page	Redirection to learn page	Yes
4.	Access 'Login' page without user registration	Redirection to login page	Redirection to login page	Yes
5.	Access 'Sign Up' page without user registration	Redirection to registration page	Redirection to registration page	Yes
6.	Incorrect email format during registration	Error message requesting proper format	Error message requesting proper format	Yes
7.	Non-matching passwords during user registration	Error message requesting proper format	Error message requesting proper format	Yes
8.	User login with non-existent user login details	Error message requesting valid details	Error message requesting valid details	No
9.	User login with validated user login details	Redirection to dashboard page	Redirection to dashboard page	Yes
10.	Access 'Dashboard' page	Redirection to dashboard page only via successful login	Redirection to dashboard page only via successful login	Yes

11.	Choose a character via selection menu on ‘Dashboard’ page	Successful character selection via arrow keys provided	Successful character selection via arrow keys provided	Yes
12.	Access to ‘Zeus’ Clues’ learning page on dashboard	Redirection to progress page whilst logged in	Redirection to settings page whilst logged in	Yes
13.	Access to code editor via learning pages	Redirection to code editor page	Redirection to code editor page	Yes
14.	Successfully use code editor to practice HTML	Working code editor on page	Working code editor on page	Yes
15.	Access to ‘Apollo’s Dojo’ game page	Redirection to game page	Redirection to game page	Yes
16.	Successfully play ‘Apollo’s Dojo’ game	Working game on page	Working game on page	Yes
17.	Access to remaining learning pages	Redirection to ‘Coming Soon’ page	Redirection to ‘Coming Soon’ page	Yes
18.	Use ‘Log Out’ button to exit application	Redirection to default/home page after successful log out	Redirection to default/home page after successful log out	Yes

8 EVALUATION

8.1 Introduction

It is important for a system to be evaluated, as it provides feedback regarding whether or not the goals established by the developer prior to the development have been achieved. Evaluation also acts as a contingency for future versions of the product being developed, as there will always be "accidental difficulties" (Brooks, 1987) that are unavoidable whilst in the development process. However, companies can limit the damage and "successfully improve the accuracy of their new product evaluation decisions" (Ozer, 2005) by evaluating their prototypes efficiently before final release.

8.1 Process Evaluation

This section aims to critically evaluate the processes that were implemented in order to complete the product. The most influential processes that were implemented to the study have been highlighted, as well as a list of a few procedures that would have resulted in a better product overall. Discussed below are the pros and cons of the final prototype web application.

8.1.1 Evaluation of the process

Process Positives	Process Negatives
Clear and concise project aim, with defined objectives.	Lack of normalized ERD designs due to simple entities created.
Extensive literature review, covering a large array of relevant topics.	Lack of prototypes with variable designs: all the prototypes developed were built on previous designs of each other.
Critical review of similar/competing products, as well relevant adaptations for the final prototype.	Lack of extensive black box testing by the relevant users due to the COVID-19 guidelines: Additional testing was originally to be done in person via computer labs.
Considerately defined functional and non-functional requirements based on prior knowledge and existing standards	Poor time management leading to loss of quality in some sections.
Creation of entity relationship, UML and site map diagrams as expected in the professional field. These were also	

compared to existing diagrams for proposed prototypes.	
Analysis and implementation of system development and testing guidelines based on existing standards.	

8.2 Product Evaluation

This section aims to critically evaluate the final prototype web application that was produced as a result of the procedures detailed in chapters 1-6 of the paper. Comparing results from the black and white box testing to each other is an effective evaluation procedure, as it allows for critical assessment of the functionality of the system from an unbiased third-party actor. However, a product like this can and should also be evaluated by simply comparing the positives and negatives of the overall system. Discussed below are the pros and cons of the final prototype web application.

8.2.1 Evaluation of the product

Product Positives	Product Negatives
Validated user input features such as user registration, login and password textboxes.	Third-party code editor: Code editor was borrowed and modified.
Responsive web application, able to change orientation regardless of window size.	Third-party game developer: game was implemented using an already existed design.
Accessibility features: Application can be accessed using multiple browsers (Google Chrome, Microsoft Edge etc.)	Inability to change password without changing user login as well
Interactivity features such as character selection and a code editor, to practice programming and playable games.	Lack of external version control system, such as Git.
Modern and visually pleasing GUI and interface design based on real world example (CodeMoji).	Lack of forms validation for certain webpages.
Reliable database connectivity features regarding user registration and login to ensure user can access other webpages.	
Parsing of user information across webpages to ensure user is logged in whilst accessing	

the features behind the registration/login wall.	
Learning pages offering reliable information and the ability to practice via code editor.	
Clear and concise labelled buttons and instructions to aid in user understandability of the system	

8.3 Further Research Ideas

Future research regarding the motivation of students to learn programming could venture into exploring the reasons that students choose to pursue this job path today. As much as it was the secondary objective, it is still an important aspect of research that should not be ignored, as it still affects the field of computing. Also, technological advancements over the last few decades mean that the range of topics that fall underneath the tech field is more extensive than ever, ranging from game design and creation to industry software development. The gender gap of professionals involved in the field has also decreased greatly, with women "building careers as professional engineers in a high technology industrial organization" (Evetts, 1998) more frequently than ever. Similar trends have also been noted regarding age gap, with individuals picking up the practice of programming at younger ages than ever before, mostly through online games and reasoning programs such as the one created in this project; these seem to be promising tools for "assessing computational thinking in young programmers" (Touretzky et al., 2017) of the future.

9 CONCLUSION

The process of proposing, researching, designing and developing a system was one that I had never experienced before on my own. Various previous endeavours such as 1st/2nd year modules had offered certain aspects of this process, but they were always either not as extensive, or assigned as group work; alleviating the workload required per person. Nonetheless, I was very grateful for my experience during those modules, as they prepared me for some of the tasks I was to carry out. Time management has always been an issue I struggled with, especially when given multiple goals to accomplish. However, due to the guidance given to me, I was able to strive and reach my goals. Regarding the product, the goal in the problem domain was to create an interactive and modern web application to teach programming. I am happy to say that albeit a few bugs here and there that are to be expected whilst developing a prototype, that the goal was achieved successfully. A web application was created with interactivity and customizability features, as well as implementing modern user interface and experience designs that are comparable and identical in some ways to the industry standard for teaching programming. It was a very challenging environment; however, I feel like it has better prepared me for a career in the field, as many of the procedures carried out there were replicated throughout the project. I will forever be grateful for the valuable experience this final project has given me while I continue to strive and achieve my dreams.

10 REFERENCES

- 50 free vector icons of Greek Mythology designed by Freepik. (n.d.). Retrieved May 19, 2020, from https://www.flaticon.com/packs/greek-mythology-10

- Altervista. (n.d.). Accessibility Color Wheel. Retrieved May 19, 2020, from https://gmazzocato.altervista.org/en/colorwheel/wheel.php

- Ancient Greek Gods & Goddesses Facts For Kids. (n.d.). Retrieved May 19, 2020, from https://www.historyforkids.net/ancient-greek-gods.html

- Anon (1995) Disability Discrimination Act 1995, Legislation.gov.uk, Statute Law Database, [online] Available at:http://www.legislation.gov.uk/ukpga/1995/50/contents (Accessed December 9, 2019).

- BCS Code of Conduct. (2020). Retrieved May 19, 2020, from https://www.bcs.org/membership/become-a-member/bcs-code-of-conduct/

- Bellis, M. (2019, November 24). How Did Color TV Come to Be? Retrieved May 19, 2020, from https://www.thoughtco.com/color-television-history-4070934

- Bergin, S., Reilly, R. and Traynor, D. (2005) Examining the role of self-regulated learning on introductory programming performance, Proceedings of the 2005 international workshop on Computing education research - ICER 05.

- Bevan, N. (2006). International Standards for HCI. Encyclopedia of Human Computer Interaction, 362-372. doi:10.4018/978-1-59140-562-7.ch056

- Brockmann, R. J., & Bradford, A. N. (1986). Writing better computer user documentation. IEEE Transactions on Professional Communication, PC-29(3), 37-37. doi:10.1109/tpc.1986.6448254

- Brooks. (1987). No Silver Bullet Essence and Accidents of Software Engineering. Computer, 20(4), 10-19. doi:10.1109/mc.1987.1663532

- Camgöz, N., Yener, C., & Güvenç, D. (2002). Effects of hue, saturation, and brightness on preference. Color Research & Application, 27(3), 199-207. doi:10.1002/col.10051

- Charney. (2008). Formalizing End-To-End Context-Aware Trust Relationships In Collaborative Activities. Proceedings of the International Conference on Security and Cryptography. doi:10.5220/0001926905460553

- Cyr, D., Head, M., & Larios, H. (2010). Colour appeal in website design within and across cultures: A multi-method evaluation. International Journal of Human-Computer Studies, 68(1-2), 1-21. doi:10.1016/j.ijhcs.2009.08.005

- Definitions. (n.d.). Retrieved May 19, 2020, from https://www.copyright.gov/help/faq/faq-definitions.html

- Erikre. (n.d.). Getting Started with ASP.NET 4.7 Web Forms and Visual Studio 2017. Retrieved May 19, 2020, from https://docs.microsoft.com/en-us/aspnet/web-forms/overview/getting-started/getting-started-with-aspnet-45-web-forms/introduction-and-overview

- Evetts, J. (1998) Managing the technology but not the organization: women and career in engineering, Women in Management Review, 13(8), pp. 283–290.

- Feizi, A., & Wong, C. Y. (2012). Usability of user interface styles for learning a graphical software application. 2012 International Conference on Computer & Information Science (ICCIS). doi:10.1109/iccisci.2012.6297188

- Gemino, A., & Parker, D. (2009). Use Case Diagrams in Support of Use Case Modeling. Journal of Database Management, 20(1), 1-24. doi:10.4018/jdm.2009010101

- Gemoets, L. A., & Mahmood, M. A. (1990). Effect of the quality of user documentation on user satisfaction with information systems. Information & Management, 18(1), 47-54. doi:10.1016/0378-7206(90)90063-n

- Hui, S. L., & See, S. L. (2015). Enhancing User Experience Through Customisation of UI Design. Procedia Manufacturing, 3, 1932-1937. doi:10.1016/j.promfg.2015.07.237

- Janzen, D., & Saiedian, H. (2006). On the Influence of Test-Driven Development on Software Design. 19th Conference on Software Engineering Education & Training (CSEET'06). doi:10.1109/cseet.2006.25

- JavaScript and HTML5 Canvas game tutorial code. (n.d.). Retrieved May 19, 2020, from https://www.codemahal.com/javascript-and-html5-canvas-game-tutorial-code/

- Kabay, M. E., Salveggio, E., Guess, R., & Rosco, R. D. (2015). Anonymity and Identity in Cyberspace. Computer Security Handbook. doi:10.1002/9781118820650.ch70

- Kegel, D. (2014) How To Get Hired -- What CS Students Need to Know, How To Get Hired -- What CS Students Need to Know, Kegel, [online] Available at: http://www.kegel.com/academy/getting-hired.html (Accessed October 27, 2019).

- Khanji, S., Jabir, R., Iqbal, F., & Marrington, A. (2016). Forensic analysis of xbox one and playstation 4 gaming consoles. 2016 IEEE International Workshop on Information Forensics and Security (WIFS). doi:10.1109/wifs.2016.7823917

- Knupfer, & Nelson, N. (1996, November 30). Visual Aesthetics and Functionality of Web Pages: Where is the Design?. Retrieved May 19, 2020, from https://eric.ed.gov/?id=ED409846

- Krippendorf, M., & Song, I. (1995). The translation of star schema into entity-relationship diagrams. Database and Expert Systems Applications. 8th International Conference, DEXA '97. Proceedings. doi:10.1109/dexa.1997.617320

- Kusumawati, R. E., Muslim, E., & Nugroho, D. (2020). Usability testing on touchscreen based electronic kiosk machine in convenience store. RECENT PROGRESS ON: MECHANICAL, INFRASTRUCTURE AND INDUSTRIAL ENGINEERING: Proceedings of International Symposium on Advances in Mechanical Engineering (ISAME): Quality in Research 2019. doi:10.1063/5.0000982

- Kyriakidou, A. (2018) COMP1632: Systems Development Project, COMP1632: Systems Development Project, London.

- Lanyi, C. S. (2017). Choosing effective colours for websites. Colour Design, 619-640. doi:10.1016/b978-0-08-101270-3.00026-6

- LDG. (n.d.). Retrieved May 19, 2020, from http://www.lostdecadegames.com/how-to-make-a-simple-html5-canvas-game/

- Lee, Y., & Kozar, K. A. (2012). Understanding of website usability: Specifying and measuring constructs and their relationships. Decision Support Systems, 52(2), 450-463. doi:10.1016/j.dss.2011.10.004

- Long, J. (2007) Just For Fun: Using Programming Games in Software Programming Training and Education, Journal of Information Technology Education: Research, 6, pp. 279–290.

- Moons, J. and Backer, C. D. (2013) The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism, Computers & Education, 60(1), pp. 368–384.

- Ostrand, T. (2002). Black-Box Testing. Encyclopedia of Software Engineering. doi:10.1002/0471028959.sof022

- Ostrand, T. (2002). White-Box Testing. Encyclopedia of Software Engineering. doi:10.1002/0471028959.sof378

- Ozer, M. (2005). Factors which influence decision making in new product evaluation. European Journal of Operational Research, 163(3), 784-801. doi:10.1016/j.ejor.2003.11.002

- Repenning, A. (2006) Programming Substrates to Create Interactive Learning Environments, Interactive Learning Environments, 4(1), pp. 045–074.

- Rusu, C., Rusu, V., Roncagliolo, S., & González, C. (2015). Usability and User Experience. International Journal of Information Technologies and Systems Approach, 8(2), 1-12. doi:10.4018/ijitsa.2015070101

- Susskind, R. E. and Susskind, D. (2017) The future of the professions: how technology will transform the work of human experts, Oxford, United Kingdom, Oxford University Press.

- Tillmann, N., Halleux, J. D., Xie, T., Gulwani, S. and Bishop, J. (2013) Teaching and learning programming and software engineering via interactive gaming, 2013 35th International Conference on Software Engineering (ICSE).

- ToonJam. (2020, May 01). Hire Leading Freelance Cartoon Artist Jamie Sale. Retrieved May 19, 2020, from https://www.jamiesale-cartoonist.com/

- Touretzky, D. S., Gardner-Mccune, C. and Aggarwal, A. (2017) Semantic Reasoning in Young Programmers, Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education - SIGCSE 17.

- UK Government(n.d.) Equality Act 2010, Legislation.gov.uk, Statute Law Database, [online] Available at: http://www.legislation.gov.uk/ukpga/2010/15/contents (Accessed December 9, 2019).

- Userhub. (2017, June 04). Human–Computer Interaction (HCI): Articles, HCI, Literature: Leading UX in Bangladesh. Retrieved May 19, 2020, from https://www.theuserhub.com/literature/human-computer-interaction-hci/

- Vihavainen, A., Paksula, M. and Luukkainen, M. (2011) Extreme apprenticeship method in teaching programming for beginners, Proceedings of the 42nd ACM technical symposium on Computer science education - SIGCSE 11.

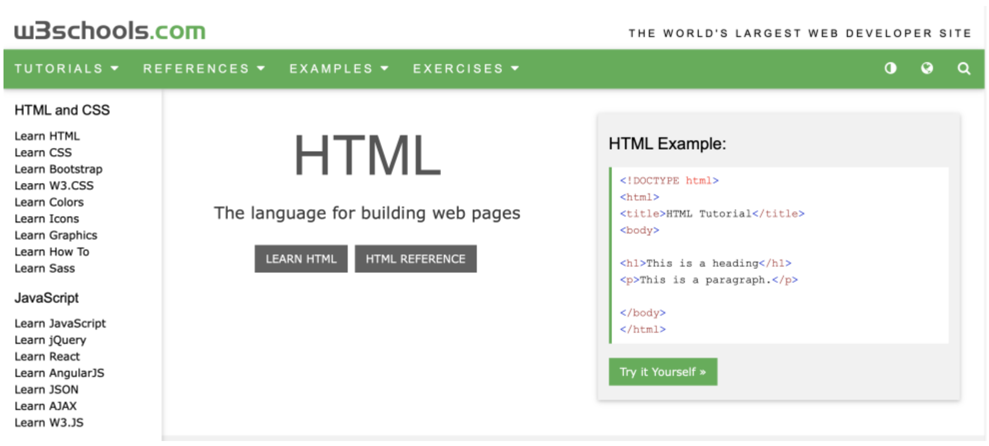
- Watson, C., Li, F. W. B. and Lau, R. W. H. (2011) Learning Programming Languages through Corrective Feedback and Concept Visualisation, Advances in Web-Based Learning - ICWL 2011 Lecture Notes in Computer Science, pp. 11–20.

- WUHCAG. (n.d.). WCAG 2.0 checklist - a free and simple guide to WCAG 2.0. Retrieved May 19, 2020, from https://www.wuhcag.com/wcag-checklist/

APPENDIX A: Similar products

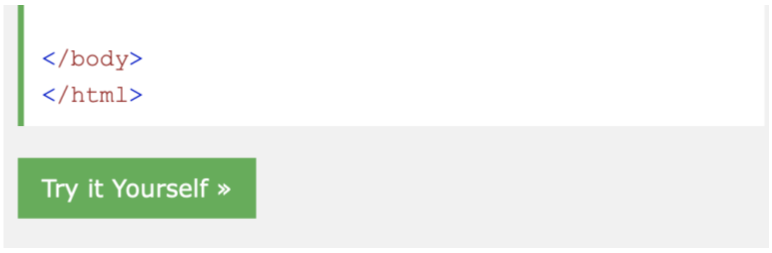
1 W3Schools

Home Page

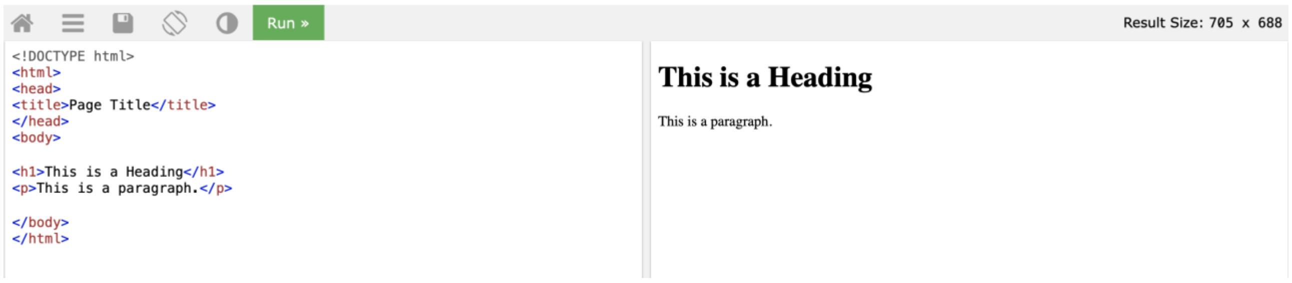


```
<!DOCTYPE html>
<html>
<title>HTML Tutorial</title>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Try it Yourself editor



```
</body>
</html>
```

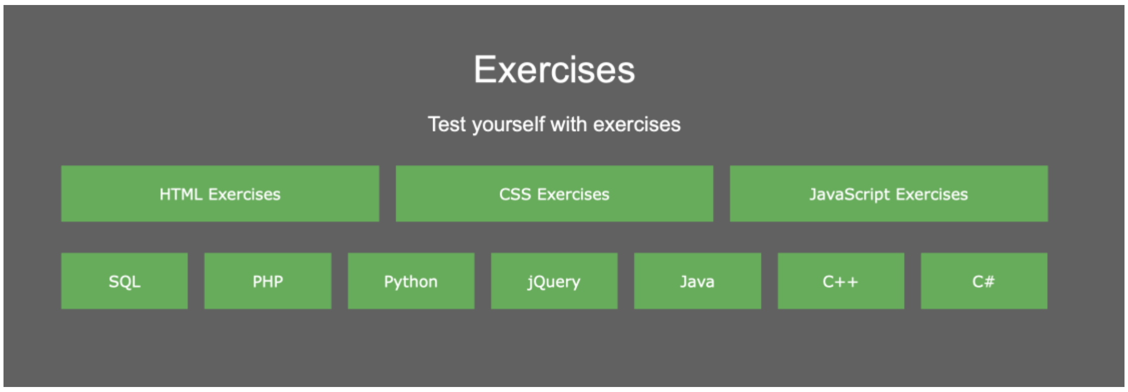


```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

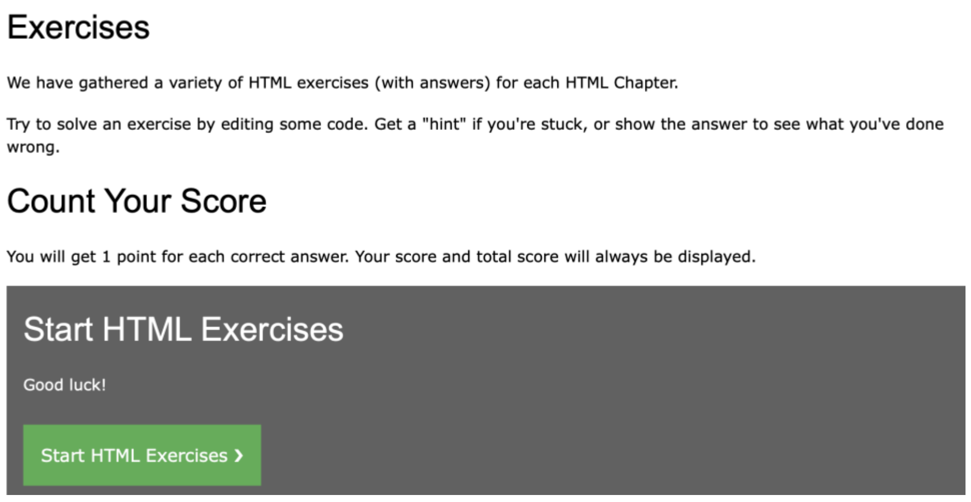
This is a Heading

This is a paragraph.

Quizzes



Exercises



Exercises

We have gathered a variety of HTML exercises (with answers) for each HTML Chapter.

Try to solve an exercise by editing some code. Get a "hint" if you're stuck, or show the answer to see what you've done wrong.

Count Your Score

You will get 1 point for each correct answer. Your score and total score will always be displayed.

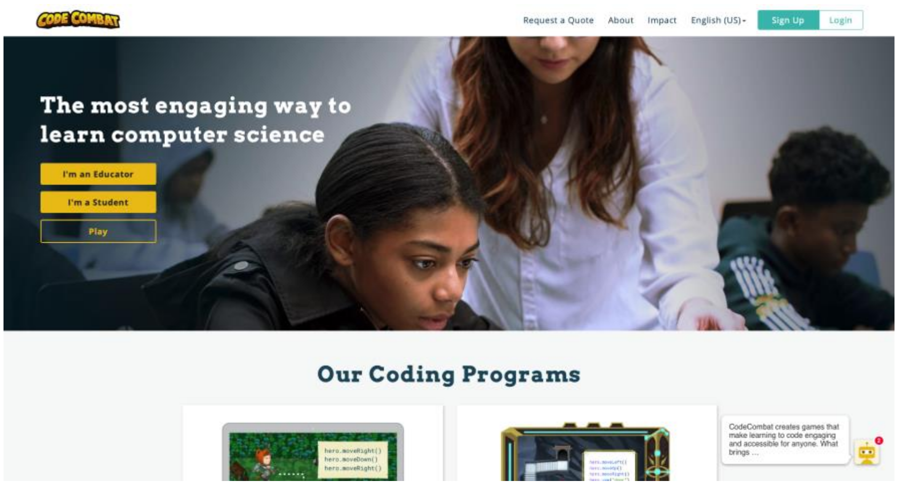
Start HTML Exercises

Good luck!

Start HTML Exercises ❯

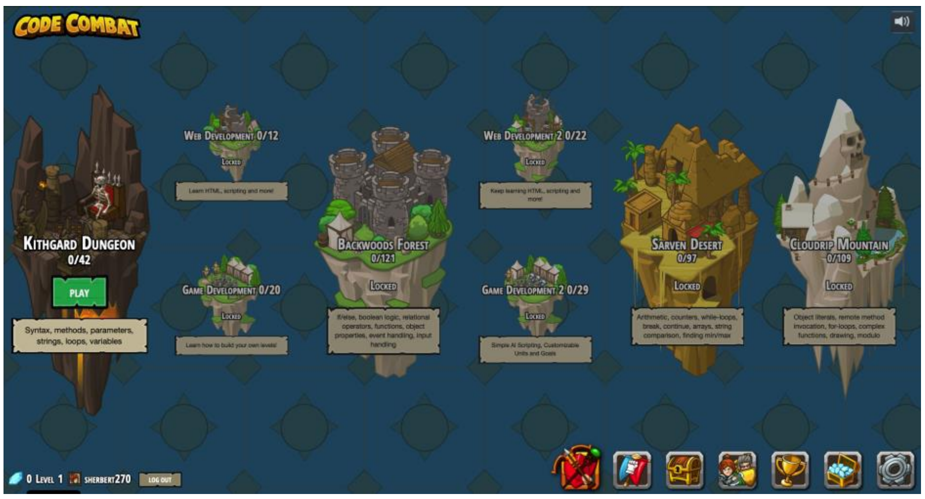
2 CodeCombat

Home Page

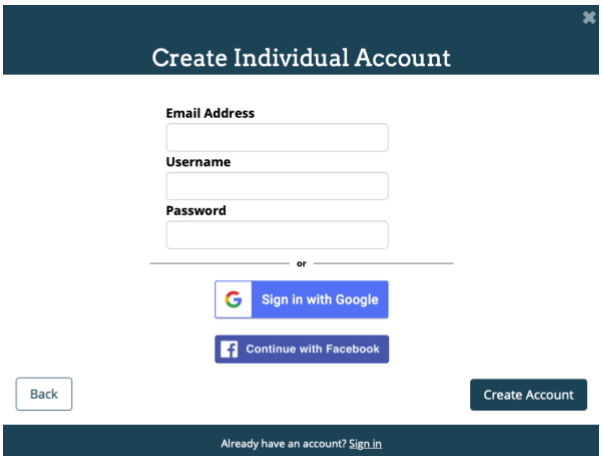


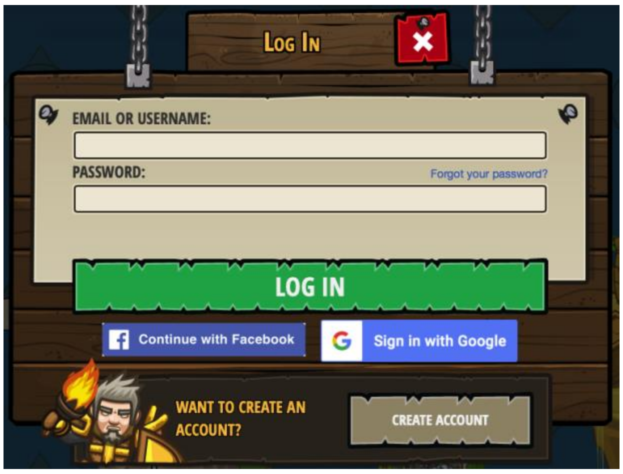
CodeCombat creates games that make learning to code engaging and accessible for anyone. What brings ...

Dashboard

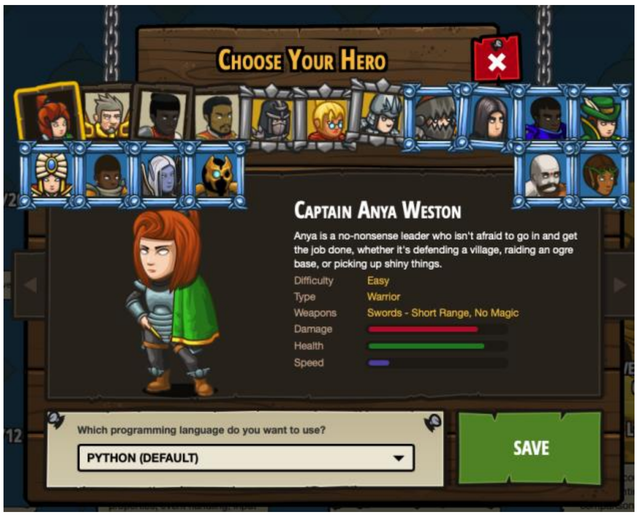

CODE COMBAT
Kithgard Dungeon 0/42
Play
Syntax, methods, parameters, strings, loops, variables
Web Development 0/12
Locked
Learn HTML, scripting and more!
Game Development 0/20
Locked
Learn how to build your own levels!
Backwoods Forest 0/121
Locked
If/else, boolean logic, relational operators, functions, object properties, event handling, input handling
Web Development 2 0/22
Locked
Keep learning HTML, scripting and more!
Game Development 2 0/29
Locked
Simple AI Scripting, Customizable Units and Goals
Sarven Desert 0/97
Locked
Arithmetic, counters, while-loops, break, continue, arrays, string comparison, finding min/max
Cloudrip Mountain 0/109
Locked
Object literals, remote method invocation, for-loops, complex functions, drawing, modulo
0 Level 1
sherbert270
Log out

User Registration





Avatar Creation/Selection



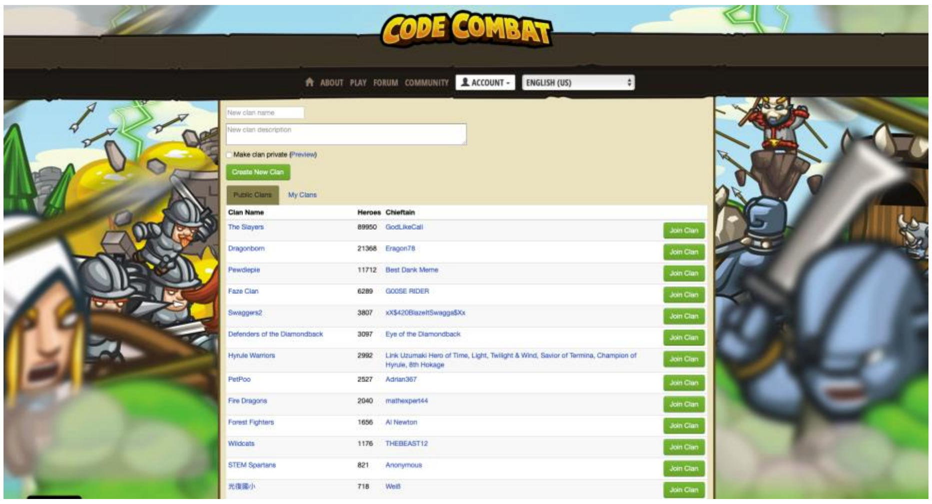
Captain Anya Weston

Anya is a no-nonsense leader who isn't afraid to go in and get the job done, whether it's defending a village, raiding an ogre base, or picking up shiny things.

Difficulty	Easy
Type	Warrior
Weapons	Swords - Short Range, No Magic
Damage	
Health	
Speed	

Which programming language do you want to use? PYTHON (DEFAULT)

Clan Selection

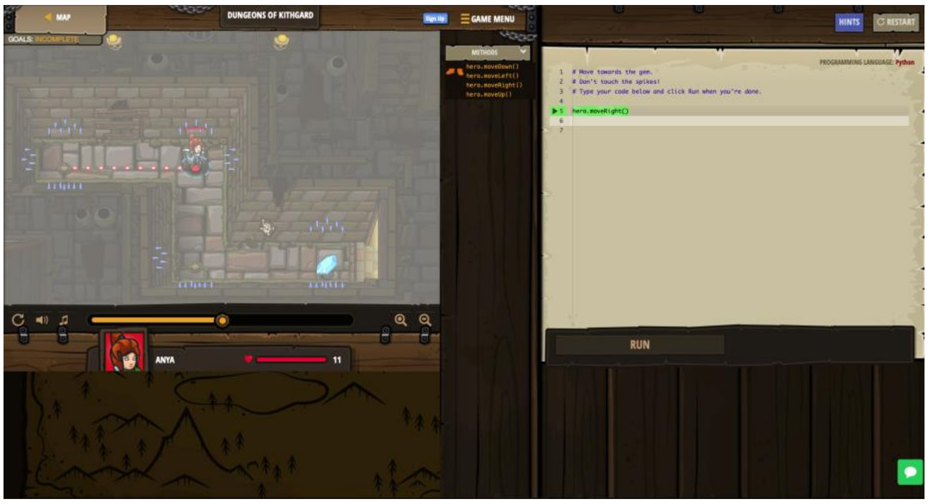


Clan Name	Heroes	Chieftain	
The Slayers	89950	GodLikeCall	Join Clan
Dragonborn	21368	Eragon78	Join Clan
Pewdiepie	11712	Best Dank Meme	Join Clan
Faze Clan	6289	G00SE RIDER	Join Clan
Swaggers2	3807	xX$420BlazeItSwaggaSXx	Join Clan
Defenders of the Diamondback	3097	Eye of the Diamondback	Join Clan
Hyrule Warriors	2992	Link Uzumaki Hero of Time, Light, Twilight & Wind, Savior of Termina, Champion of Hyrule, 8th Hokage	Join Clan
PetPoo	2527	Adrian367	Join Clan
Fire Dragons	2040	mathexpert44	Join Clan
Forest Fighters	1656	Al Newton	Join Clan
Wildcats	1176	THEBEAST12	Join Clan
STEM Spartans	821	Anonymous	Join Clan
光復國小	718	Wei8	Join Clan

Game Example: ‘Dungeons of Kithgard’

The user has to type the correct code in the right-sided window in order to play the game in the left-handed window. As this is one of the easier levels, the methods are provided for the user to help them solve the level. The GUI also includes a ‘Hint’ button in order to help the user even more if they are still in need of assistance.

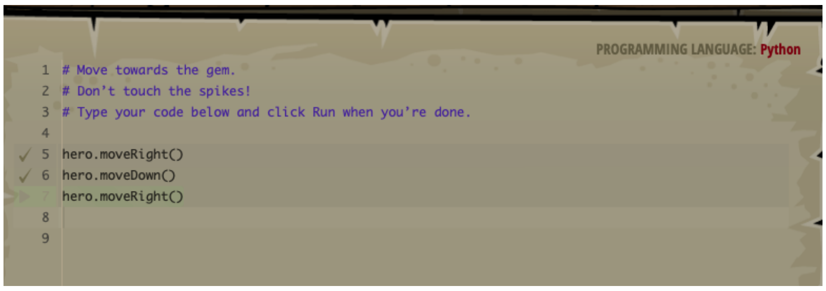
Gameplay



```
hero.moveDown()
hero.moveLeft()
hero.moveRight()
hero.moveUp()
```

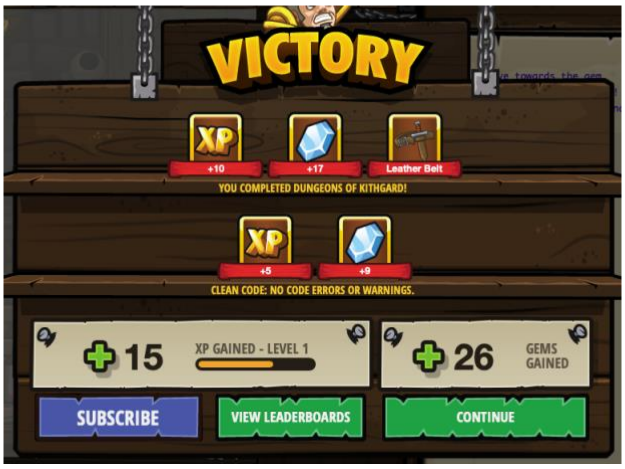
```
# Move towards the gem.
# Don't touch the spikes!
# Type your code below and click Run when you're done.

hero.moveRight()
```

Correct Input



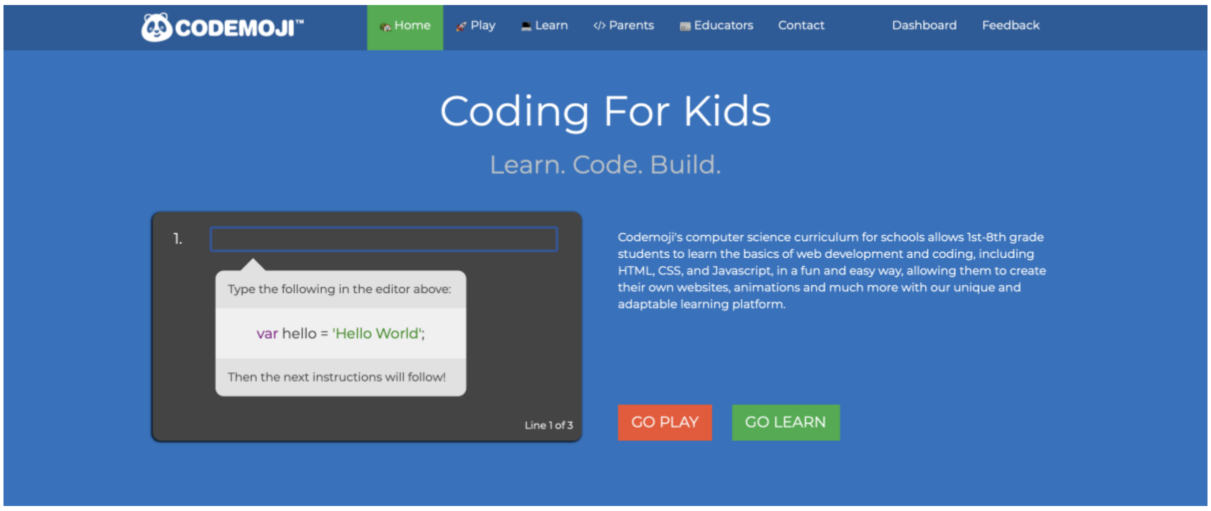
```
# Move towards the gem.
# Don't touch the spikes!
# Type your code below and click Run when you're done.

hero.moveRight()
hero.moveDown()
hero.moveRight()
```

Victory/Level Progression



3 CodeMoji

Appearance



Coding For Kids

Learn. Code. Build.

1.

Type the following in the editor above:

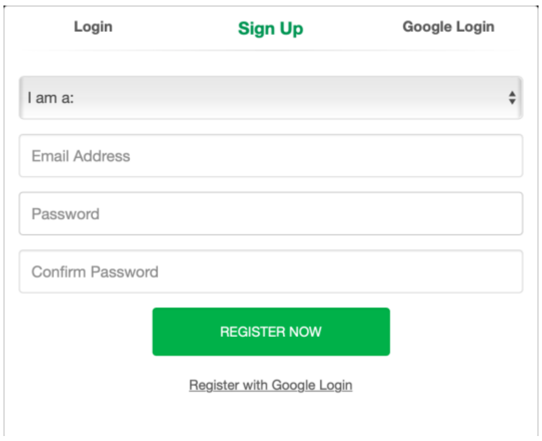
var hello = 'Hello World';

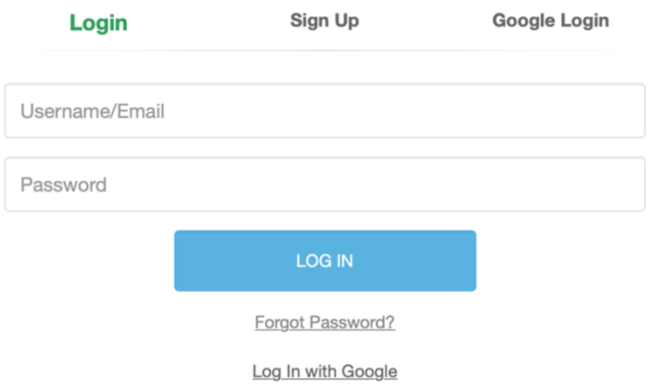
Then the next instructions will follow!

Line 1 of 3

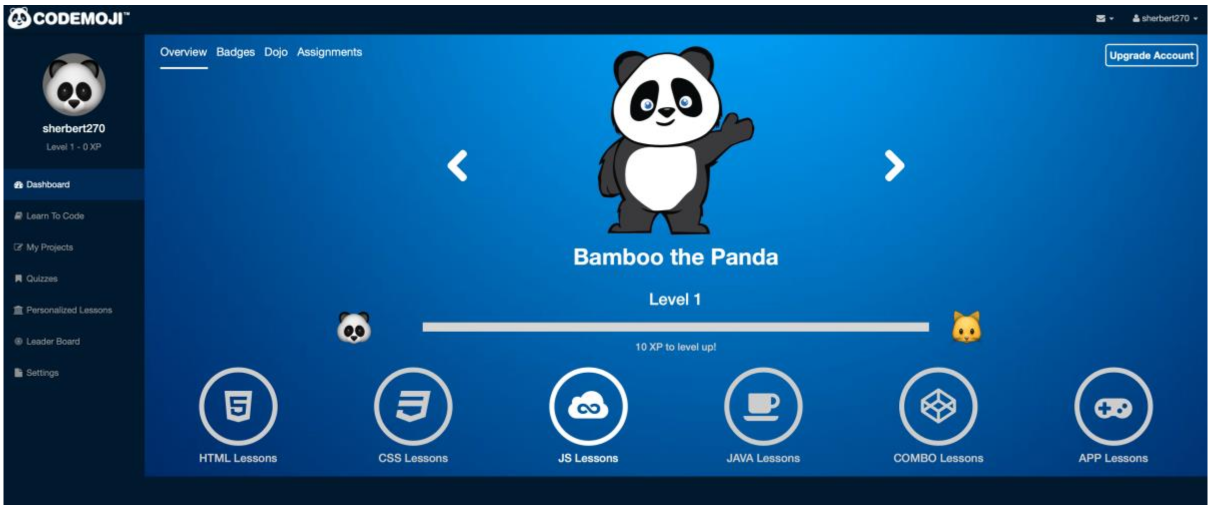
Codemoji's computer science curriculum for schools allows 1st-8th grade students to learn the basics of web development and coding, including HTML, CSS, and Javascript, in a fun and easy way, allowing them to create their own websites, animations and much more with our unique and adaptable learning platform.

User Registration and Login



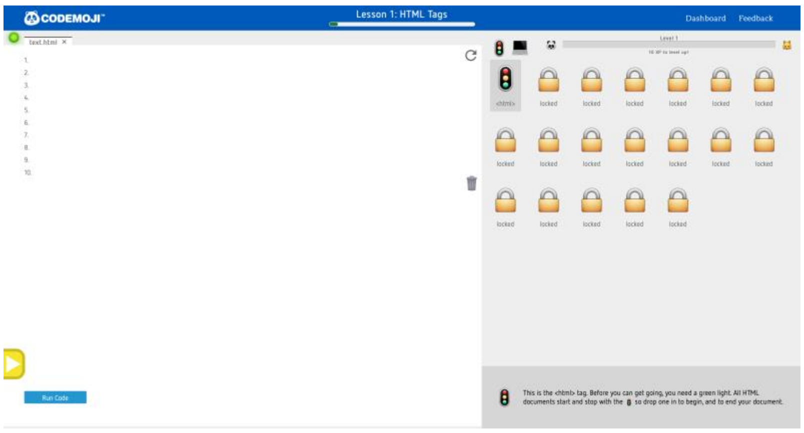


Dashboard



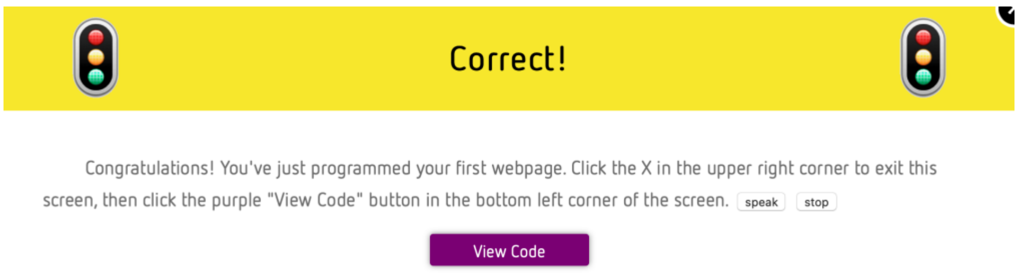
Game Example: ‘HTML5 Intro’

Just like the in CodeCombat application, the user has 2 windows in which to input the code. This is where the similarities end however, as CodeMoji requires the user to program using emojis, a task that definitely appeals more to younger audiences. In the HTML example below, the stop light symbolizes the HTML tag, which should be present at the beginning and end of any HTML document. Once the user has inputted the correct answer, they can also view the actual code behind the emojis they have just used. The game also has quizzes between levels, testing the user’s knowledge before moving on to the next level.



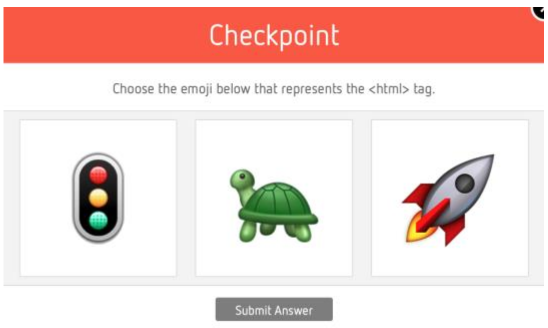
This is the <html> tag. Before you can get going, you need a green light. All HTML documents start and stop with the 🚦 so drop one in to begin, and to end your document.

Victory/Level Progression



Congratulations! You've just programmed your first webpage. Click the X in the upper right corner to exit this screen, then click the purple "View Code" button in the bottom left corner of the screen.

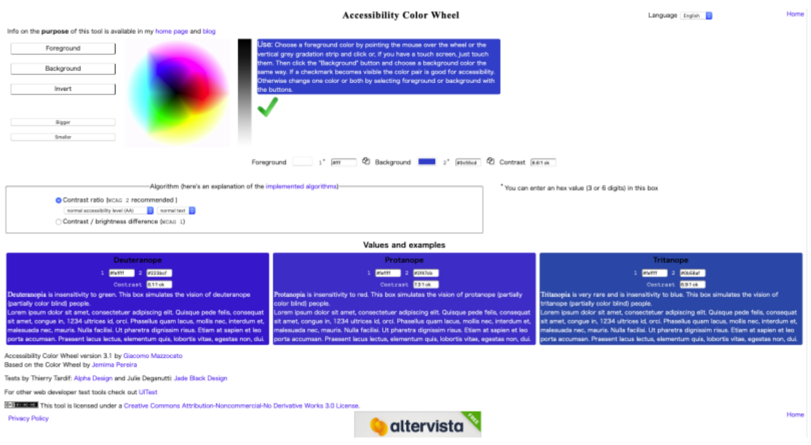
Checkpoint Quiz



Choose the emoji below that represents the <html> tag.

APPENDIX B: Prototypes and Design

Validated colourways:


Accessibility Color Wheel

Language English

Home

Info on the **purpose** of this tool is available in my home page and blog

Foreground

Background

Invert

Bigger

Smaller

Use: Choose a foreground color by pointing the mouse over the wheel or the vertical grey gradation strip and click or, if you have a touch screen, just touch them. Then click the "Background" button and choose a background color the same way. If a checkmark becomes visible the color pair is good for accessibility. Otherwise change one color or both by selecting foreground or background with the buttons.

Foreground 1* #fff Background 2* #0c56cd Contrast 6.6:1 ok

Algorithm (here's an explanation of the implemented algorithms)

[✓ Contrast ratio (WCAG 2 recommended)] normal accessibility level (AA) normal text

Contrast / brightness difference (WCAG 1)

* You can enter an hex value (3 or 6 digits) in this box

Values and examples

Deuteranope	Protanope	Tritanope
1 #fefff 2 #223bcf	1 #fefff 2 #2f47cb	1 #fefff 2 #0b58af
Contrast 8.1:1 ok	Contrast 7.3:1 ok	Contrast 6.9:1 ok
Deuteranopia is insensitivity to green. This box simulates the vision of deuteranope (partially color blind) people. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Quisque pede felis, consequat sit amet, congue in, 1234 ultrices id, orci. Phasellus quam lacus, mollis nec, interdum et, malesuada nec, mauris. Nulla facilisi. Ut pharetra dignissim risus. Etiam at sapien et leo porta accumsan. Praesent lacus lectus, elementum quis, lobortis vitae, egestas non, dui.	**Protanopia** is insensitivity to red. This box simulates the vision of protanope (partially color blind) people. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Quisque pede felis, consequat sit amet, congue in, 1234 ultrices id, orci. Phasellus quam lacus, mollis nec, interdum et, malesuada nec, mauris. Nulla facilisi. Ut pharetra dignissim risus. Etiam at sapien et leo porta accumsan. Praesent lacus lectus, elementum quis, lobortis vitae, egestas non, dui.	**Tritanopia** is very rare and is insensitivity to blue. This box simulates the vision of tritanope (partially color blind) people. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Quisque pede felis, consequat sit amet, congue in, 1234 ultrices id, orci. Phasellus quam lacus, mollis nec, interdum et, malesuada nec, mauris. Nulla facilisi. Ut pharetra dignissim risus. Etiam at sapien et leo porta accumsan. Praesent lacus lectus, elementum quis, lobortis vitae, egestas non, dui.

Accessibility Color Wheel version 3.1 by Giacomo Mazzocato
Based on the Color Wheel by Jemima Pereira

Tests by Thierry Tardif: Alpha Design and Julie Deganutti: Jade Black Design

For other web developer test tools check out UITest

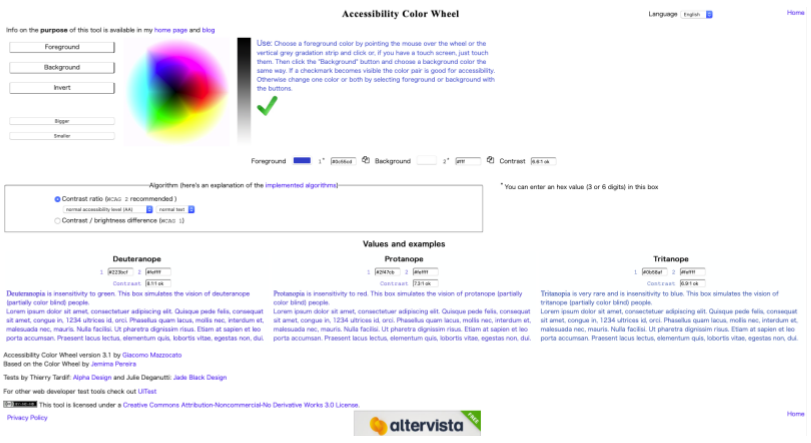
This tool is licensed under a Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 License.

Privacy Policy

Home

altervista

Inverse:


Accessibility Color Wheel

Language English

Home

Info on the **purpose** of this tool is available in my home page and blog

Foreground

Background

Invert

Bigger

Smaller

Use: Choose a foreground color by pointing the mouse over the wheel or the vertical grey gradation strip and click or, if you have a touch screen, just touch them. Then click the "Background" button and choose a background color the same way. If a checkmark becomes visible the color pair is good for accessibility. Otherwise change one color or both by selecting foreground or background with the buttons.

Foreground 1* #0c56cd Background 2* #fff Contrast 6.6:1 ok

Algorithm (here's an explanation of the implemented algorithms)

[✓ Contrast ratio (WCAG 2 recommended)] normal accessibility level (AA) normal text

Contrast / brightness difference (WCAG 1)

* You can enter an hex value (3 or 6 digits) in this box

Values and examples

Deuteranope	Protanope	Tritanope
1 #223bcf 2 #fefff	1 #2f47cb 2 #fefff	1 #0b58af 2 #fefff
Contrast 8.1:1 ok	Contrast 7.3:1 ok	Contrast 6.9:1 ok
Deuteranopia is insensitivity to green. This box simulates the vision of deuteranope (partially color blind) people. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Quisque pede felis, consequat sit amet, congue in, 1234 ultrices id, orci. Phasellus quam lacus, mollis nec, interdum et, malesuada nec, mauris. Nulla facilisi. Ut pharetra dignissim risus. Etiam at sapien et leo porta accumsan. Praesent lacus lectus, elementum quis, lobortis vitae, egestas non, dui.	**Protanopia** is insensitivity to red. This box simulates the vision of protanope (partially color blind) people. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Quisque pede felis, consequat sit amet, congue in, 1234 ultrices id, orci. Phasellus quam lacus, mollis nec, interdum et, malesuada nec, mauris. Nulla facilisi. Ut pharetra dignissim risus. Etiam at sapien et leo porta accumsan. Praesent lacus lectus, elementum quis, lobortis vitae, egestas non, dui.	**Tritanopia** is very rare and is insensitivity to blue. This box simulates the vision of tritanope (partially color blind) people. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Quisque pede felis, consequat sit amet, congue in, 1234 ultrices id, orci. Phasellus quam lacus, mollis nec, interdum et, malesuada nec, mauris. Nulla facilisi. Ut pharetra dignissim risus. Etiam at sapien et leo porta accumsan. Praesent lacus lectus, elementum quis, lobortis vitae, egestas non, dui.

Accessibility Color Wheel version 3.1 by Giacomo Mazzocato
Based on the Color Wheel by Jemima Pereira

Tests by Thierry Tardif: Alpha Design and Julie Deganutti: Jade Black Design

For other web developer test tools check out UITest

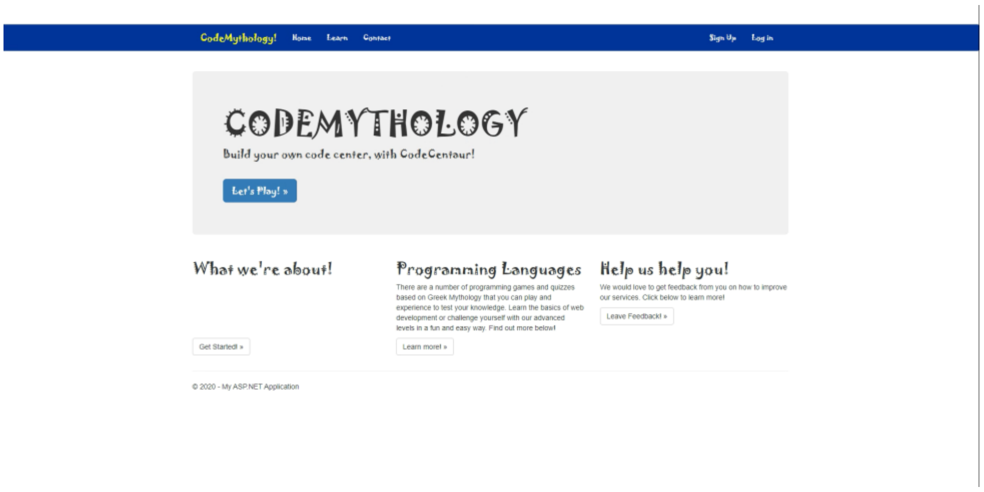
This tool is licensed under a Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 License.

Privacy Policy

Home

altervista

Original Design:


CodeMythology! Home Learn Contact Sign Up Log in

CODEMYTHOLOGY

Build your own code center, with CodeCentaur!

Let's Play! »

What we're about!

Get Started! »

Programming Languages

There are a number of programming games and quizzes based on Greek Mythology that you can play and experience to test your knowledge. Learn the basics of web development or challenge yourself with our advanced levels in a fun and easy way. Find out more below!

Learn more! »

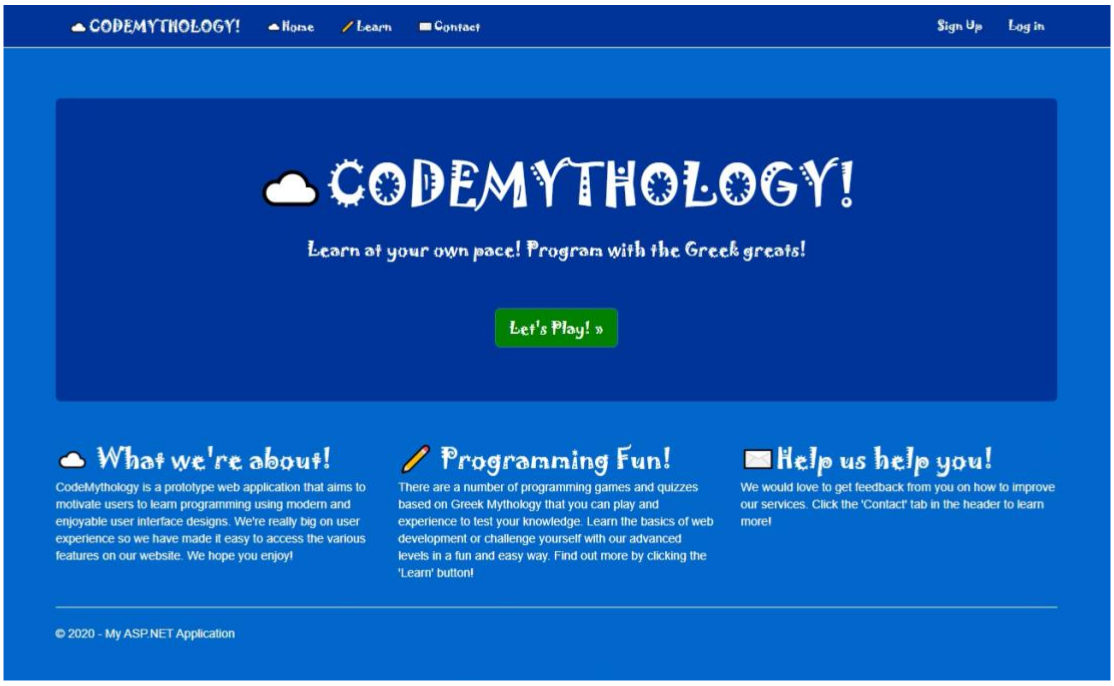
Help us help you!

We would love to get feedback from you on how to improve our services. Click below to learn more!

Leave Feedback! »

© 2020 - My ASP.NET Application

Final Prototype:


CODEMYTHOLOGY! Home Learn Contact Sign Up Log in

CODEMYTHOLOGY!

Learn at your own pace! Program with the Greek greats!

Let's Play! »

What we're about!

CodeMythology is a prototype web application that aims to motivate users to learn programming using modern and enjoyable user interface designs. We're really big on user experience so we have made it easy to access the various features on our website. We hope you enjoy!

Programming Fun!

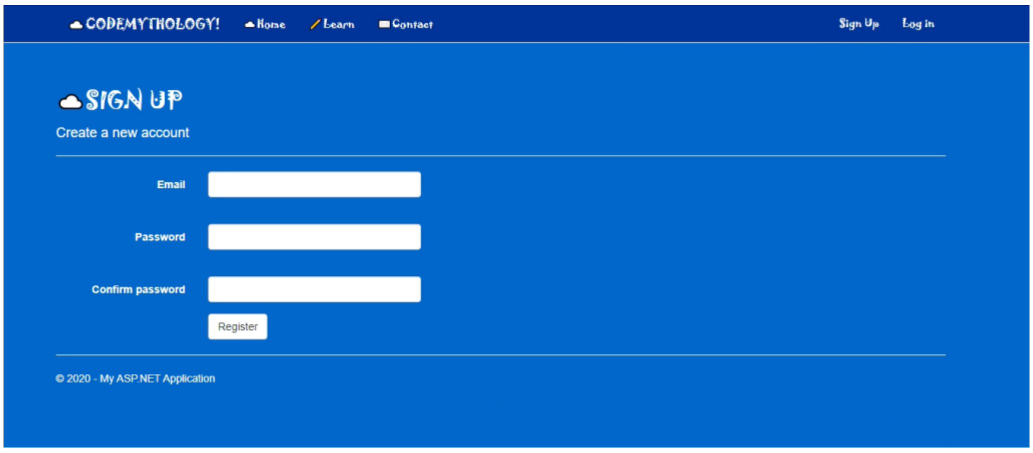
There are a number of programming games and quizzes based on Greek Mythology that you can play and experience to test your knowledge. Learn the basics of web development or challenge yourself with our advanced levels in a fun and easy way. Find out more by clicking the 'Learn' button!

Help us help you!

We would love to get feedback from you on how to improve our services. Click the 'Contact' tab in the header to learn more!

© 2020 - My ASP.NET Application

Sign Up Page:


CODEMYTHOLOGY! Home Learn Contact Sign Up Log in

SIGN UP

Create a new account

Email

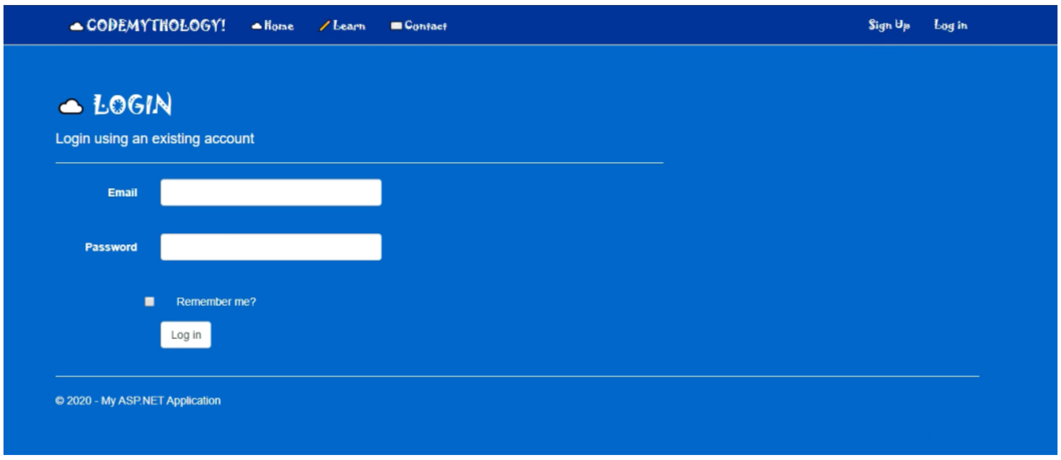
Password

Confirm password

Register

© 2020 - My ASP.NET Application

Login Page:


CODEMYTHOLOGY! Home Learn Contact Sign Up Log in

LOGIN

Login using an existing account

Email

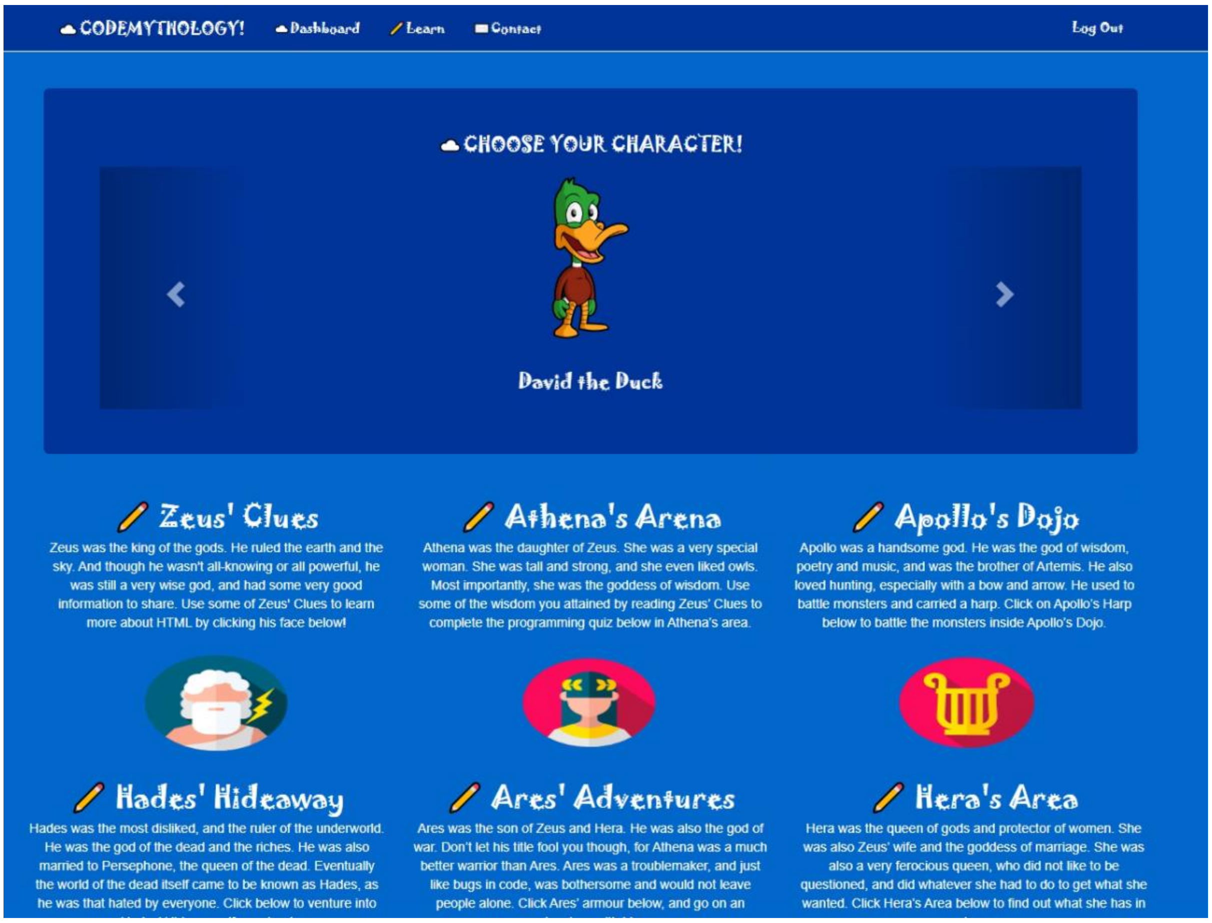
Password

Remember me?

Log in

© 2020 - My ASP.NET Application

Dashboard:



Zeus' Clues

Zeus was the king of the gods. He ruled the earth and the sky. And though he wasn't all-knowing or all powerful, he was still a very wise god, and had some very good information to share. Use some of Zeus' Clues to learn more about HTML by clicking his face below!

Athena's Arena

Athena was the daughter of Zeus. She was a very special woman. She was tall and strong, and she even liked owls. Most importantly, she was the goddess of wisdom. Use some of the wisdom you attained by reading Zeus' Clues to complete the programming quiz below in Athena's area.

Apollo's Dojo

Apollo was a handsome god. He was the god of wisdom, poetry and music, and was the brother of Artemis. He also loved hunting, especially with a bow and arrow. He used to battle monsters and carried a harp. Click on Apollo's Harp below to battle the monsters inside Apollo's Dojo.

Hades' Hideaway

Hades was the most disliked, and the ruler of the underworld. He was the god of the dead and the riches. He was also married to Persephone, the queen of the dead. Eventually the world of the dead itself came to be known as Hades, as he was that hated by everyone. Click below to venture into

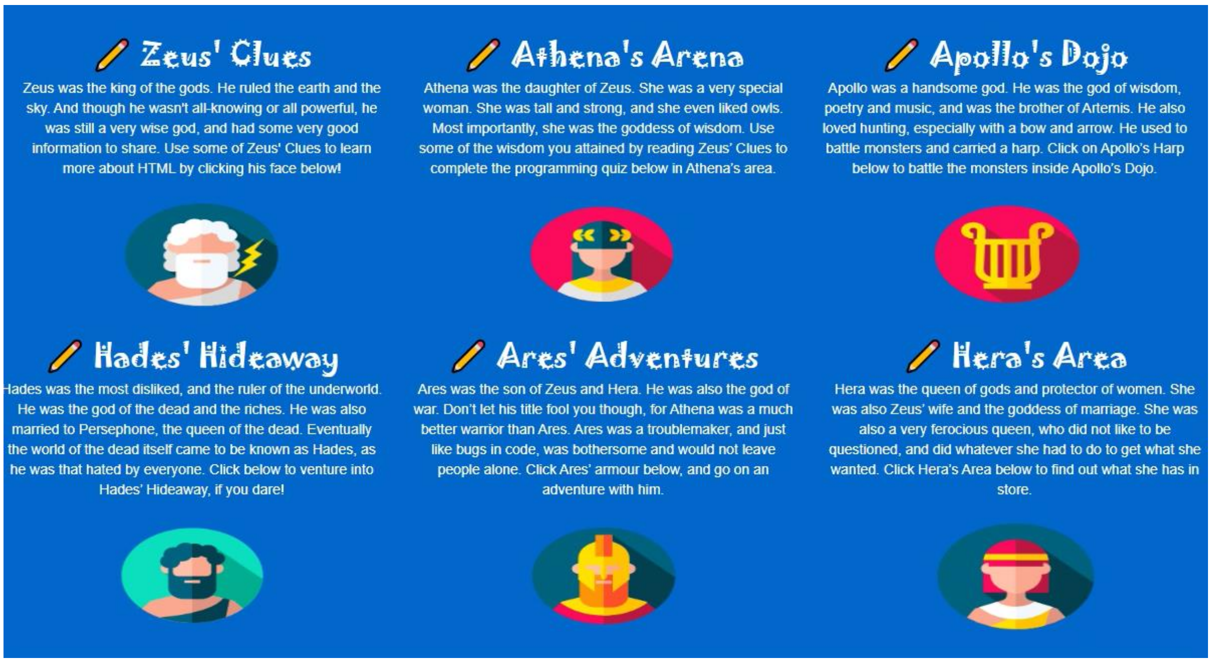
Ares' Adventures

Ares was the son of Zeus and Hera. He was also the god of war. Don't let his title fool you though, for Athena was a much better warrior than Ares. Ares was a troublemaker, and just like bugs in code, was bothersome and would not leave people alone. Click Ares' armour below, and go on an

Hera's Area

Hera was the queen of gods and protector of women. She was also Zeus' wife and the goddess of marriage. She was also a very ferocious queen, who did not like to be questioned, and did whatever she had to do to get what she wanted. Click Hera's Area below to find out what she has in

Dashboard Lessons:



Zeus' Clues

Zeus was the king of the gods. He ruled the earth and the sky. And though he wasn't all-knowing or all powerful, he was still a very wise god, and had some very good information to share. Use some of Zeus' Clues to learn more about HTML by clicking his face below!

Athena's Arena

Athena was the daughter of Zeus. She was a very special woman. She was tall and strong, and she even liked owls. Most importantly, she was the goddess of wisdom. Use some of the wisdom you attained by reading Zeus' Clues to complete the programming quiz below in Athena's area.

Apollo's Dojo

Apollo was a handsome god. He was the god of wisdom, poetry and music, and was the brother of Artemis. He also loved hunting, especially with a bow and arrow. He used to battle monsters and carried a harp. Click on Apollo's Harp below to battle the monsters inside Apollo's Dojo.

Hades' Hideaway

Hades was the most disliked, and the ruler of the underworld. He was the god of the dead and the riches. He was also married to Persephone, the queen of the dead. Eventually the world of the dead itself came to be known as Hades, as he was that hated by everyone. Click below to venture into Hades' Hideaway, if you dare!

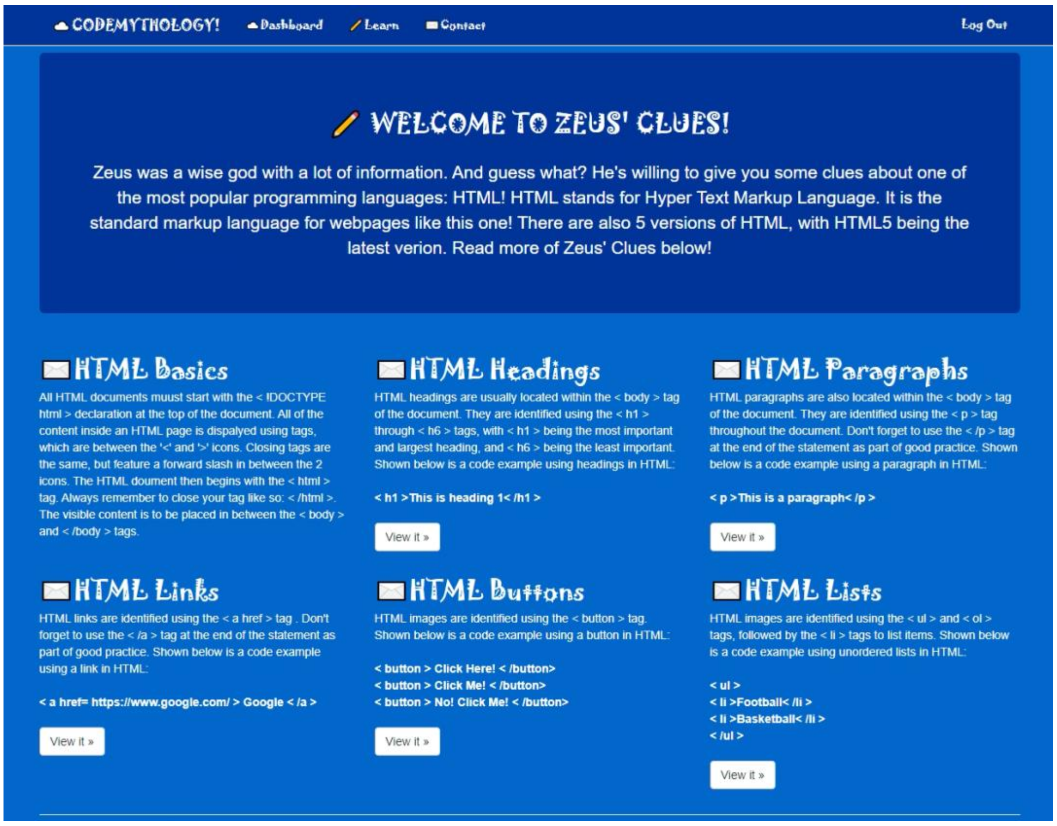
Ares' Adventures

Ares was the son of Zeus and Hera. He was also the god of war. Don't let his title fool you though, for Athena was a much better warrior than Ares. Ares was a troublemaker, and just like bugs in code, was bothersome and would not leave people alone. Click Ares' armour below, and go on an adventure with him.

Hera's Area

Hera was the queen of gods and protector of women. She was also Zeus' wife and the goddess of marriage. She was also a very ferocious queen, who did not like to be questioned, and did whatever she had to do to get what she wanted. Click Hera's Area below to find out what she has in store.

Zeus’ Clues Learning Page:



WELCOME TO ZEUS' CLUES!

Zeus was a wise god with a lot of information. And guess what? He's willing to give you some clues about one of the most popular programming languages: HTML! HTML stands for Hyper Text Markup Language. It is the standard markup language for webpages like this one! There are also 5 versions of HTML, with HTML5 being the latest verion. Read more of Zeus' Clues below!

HTML Basics

All HTML documents muust start with the < !DOCTYPE html > declaration at the top of the document. All of the content inside an HTML page is dispalyed using tags, which are between the '<' and '>' icons. Closing tags are the same, but feature a forward slash in between the 2 icons. The HTML doument then begins with the < html > tag. Always remember to close your tag like so: < /html >. The visible content is to be placed in between the < body > and < /body > tags.

HTML Headings

HTML headings are usually located within the < body > tag of the document. They are identified using the < h1 > through < h6 > tags, with < h1 > being the most important and largest heading, and < h6 > being the least important. Shown below is a code example using headings in HTML:

< h1 >This is heading 1< /h1 >

HTML Paragraphs

HTML paragraphs are also located within the < body > tag of the document. They are identified using the < p > tag throughout the document. Don't forget to use the < /p > tag at the end of the statement as part of good practice. Shown below is a code example using a paragraph in HTML:

< p >This is a paragraph< /p >

HTML Links

HTML links are identified using the < a href > tag . Don't forget to use the < /a > tag at the end of the statement as part of good practice. Shown below is a code example using a link in HTML:

< a href= https://www.google.com/ > Google < /a >

HTML Buttons

HTML images are identified using the < button > tag. Shown below is a code example using a button in HTML:

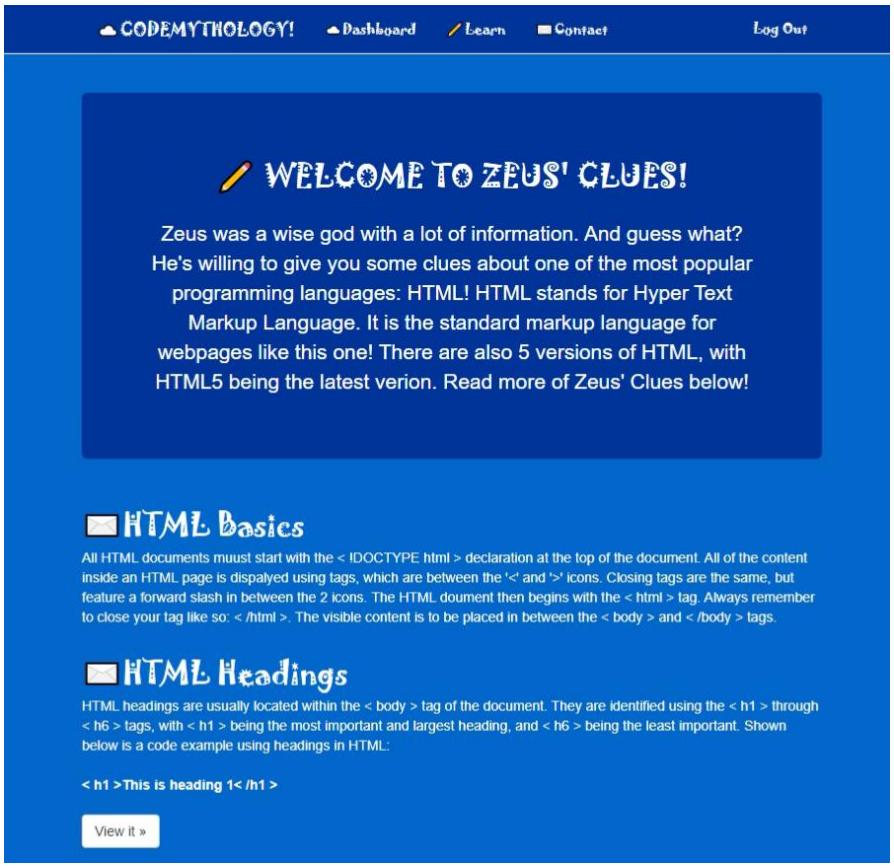
< button > Click Here! < /button>
< button > Click Me! < /button>
< button > No! Click Me! < /button>

HTML Lists

HTML images are identified using the < ul > and < ol > tags, followed by the < li > tags to list items. Shown below is a code example using unordered lists in HTML:

< ul >
< li >Football< /li >
< li >Basketball< /li >
< /ul >

Responsive Design: Zeus’ Clues Learning Page



WELCOME TO ZEUS' CLUES!

Zeus was a wise god with a lot of information. And guess what? He's willing to give you some clues about one of the most popular programming languages: HTML! HTML stands for Hyper Text Markup Language. It is the standard markup language for webpages like this one! There are also 5 versions of HTML, with HTML5 being the latest verion. Read more of Zeus' Clues below!

HTML Basics

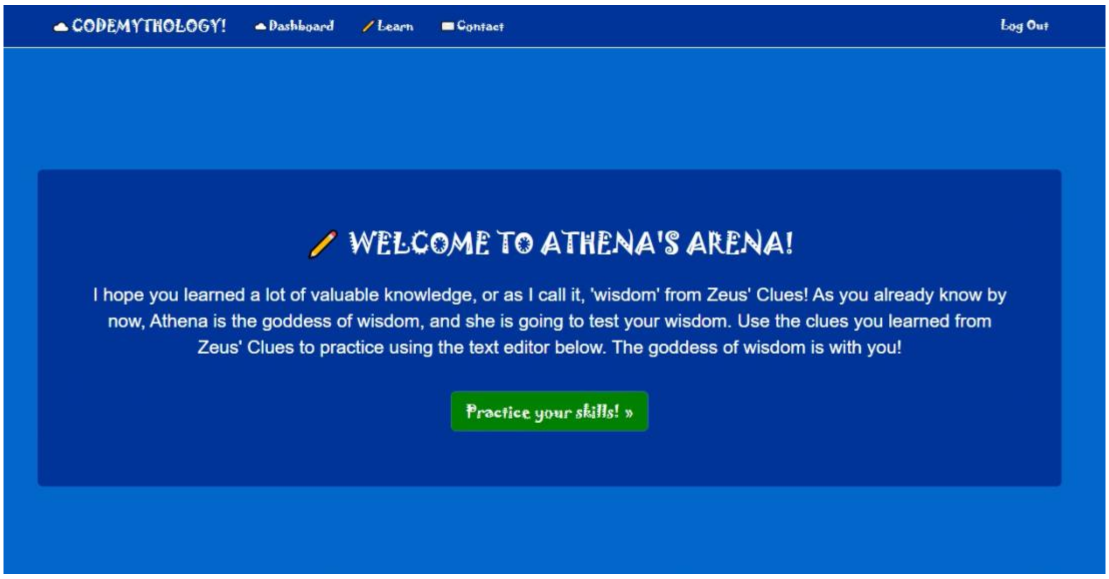
All HTML documents muust start with the < !DOCTYPE html > declaration at the top of the document. All of the content inside an HTML page is dispalyed using tags, which are between the '<' and '>' icons. Closing tags are the same, but feature a forward slash in between the 2 icons. The HTML doument then begins with the < html > tag. Always remember to close your tag like so: < /html >. The visible content is to be placed in between the < body > and < /body > tags.

HTML Headings

HTML headings are usually located within the < body > tag of the document. They are identified using the < h1 > through < h6 > tags, with < h1 > being the most important and largest heading, and < h6 > being the least important. Shown below is a code example using headings in HTML:

< h1 >This is heading 1< /h1 >

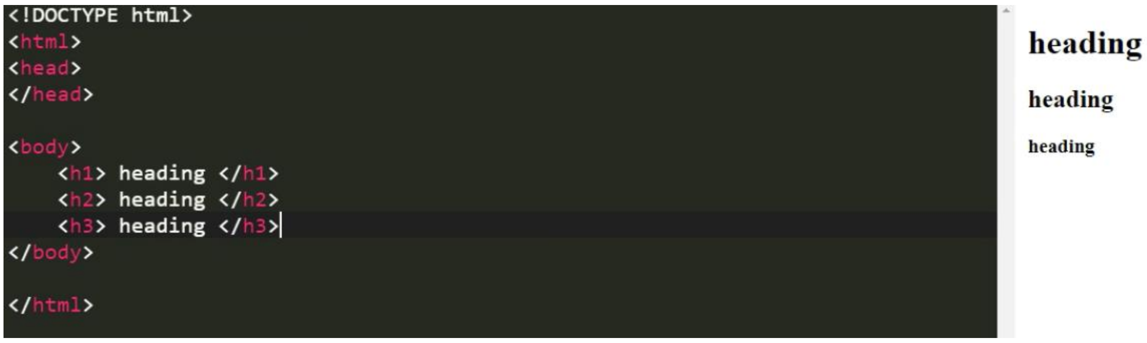
Athena's Arena Learning Page:



WELCOME TO ATHENA'S ARENA!

I hope you learned a lot of valuable knowledge, or as I call it, 'wisdom' from Zeus' Clues! As you already know by now, Athena is the goddess of wisdom, and she is going to test your wisdom. Use the clues you learned from Zeus' Clues to practice using the text editor below. The goddess of wisdom is with you!

Athena's Arena Learning Page: Code Editor



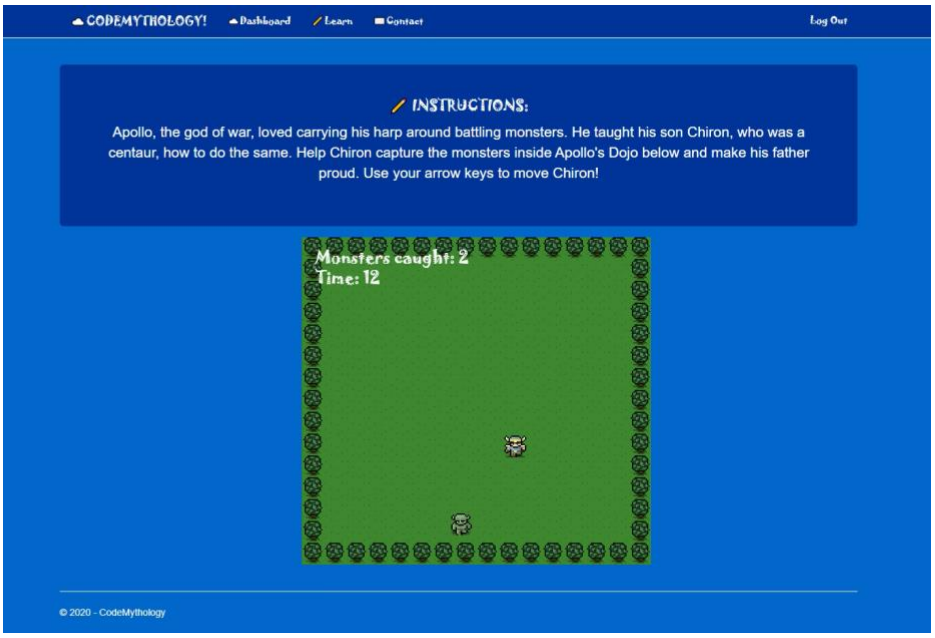
```
<!DOCTYPE html>
<html>
<head>
</head>

<body>
    <h1> heading </h1>
    <h2> heading </h2>
    <h3> heading </h3>
</body>

</html>
```

heading

heading

heading

Apollo's Dojo Game Page:

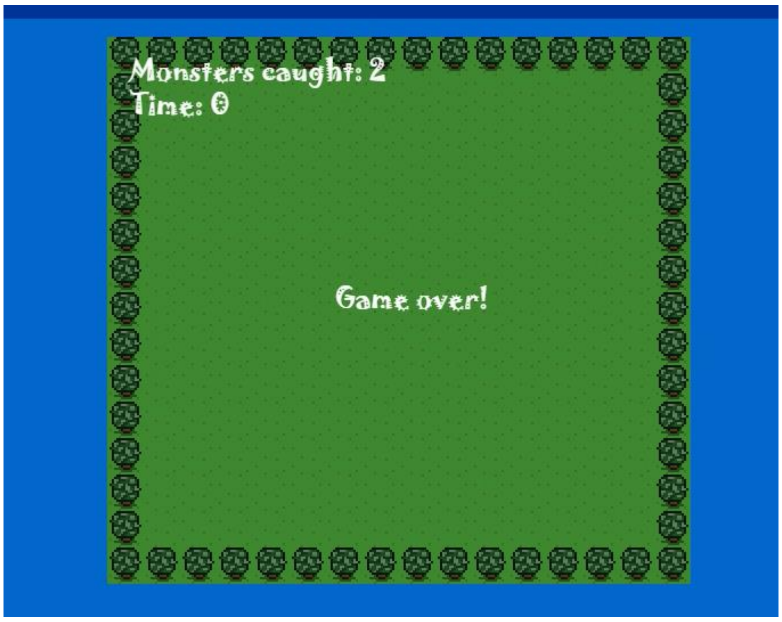


INSTRUCTIONS:

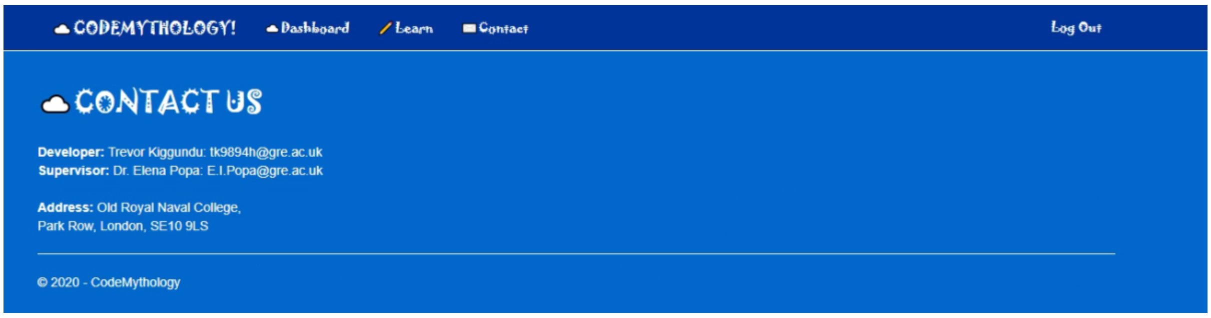
Apollo, the god of war, loved carrying his harp around battling monsters. He taught his son Chiron, who was a centaur, how to do the same. Help Chiron capture the monsters inside Apollo's Dojo below and make his father proud. Use your arrow keys to move Chiron!

© 2020 - CodeMythology

Apollo’s Dojo Game Over:



Contact Page:



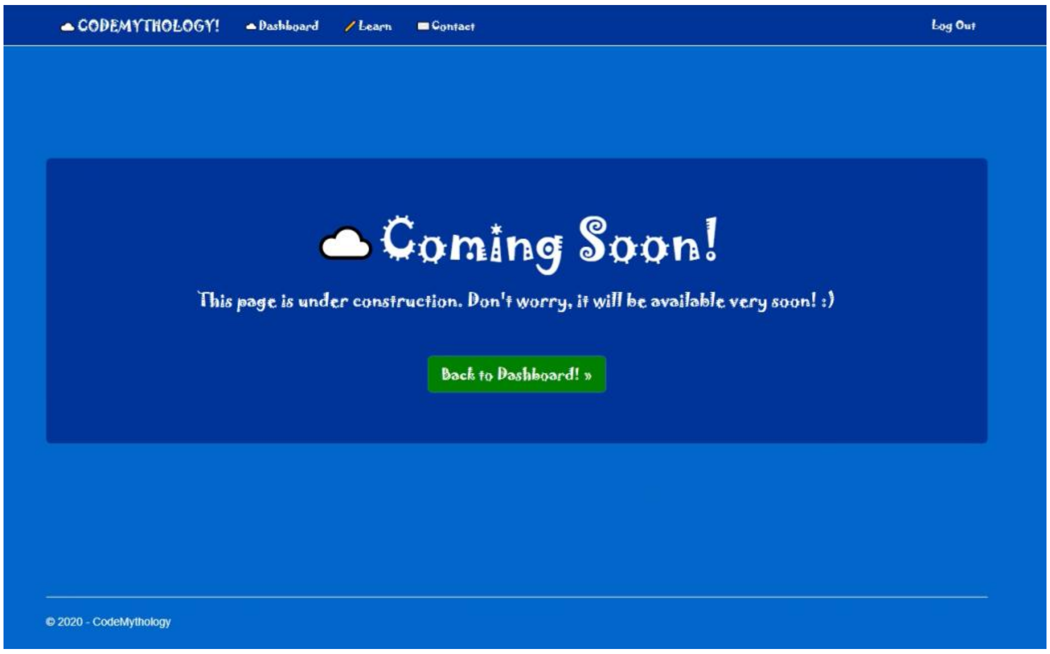
CONTACT US

Developer: Trevor Kiggundu: tk9894h@gre.ac.uk
Supervisor: Dr. Elena Popa: E.I.Popa@gre.ac.uk

Address: Old Royal Naval College,
Park Row, London, SE10 9LS

© 2020 - CodeMythology

Coming Soon Page:



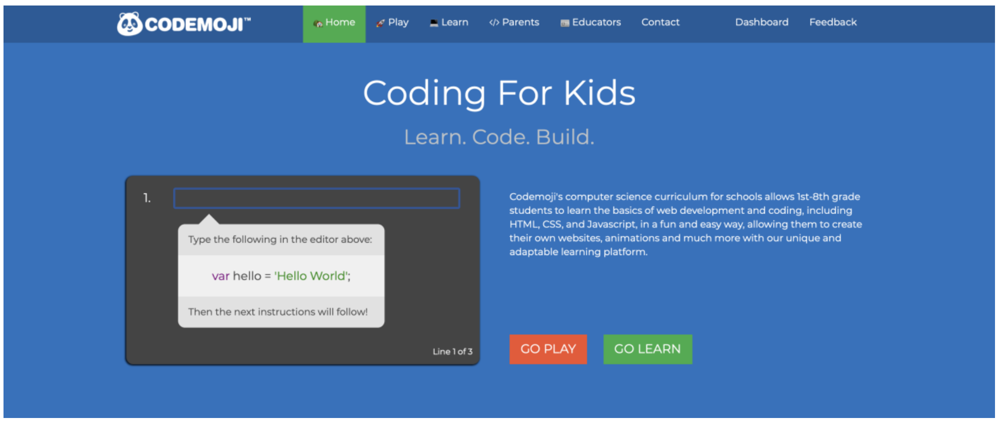
Coming Soon!

This page is under construction. Don't worry, it will be available very soon! :)

© 2020 - CodeMythology

APPENDIX C: Similarities in design

CodeMoji home page:



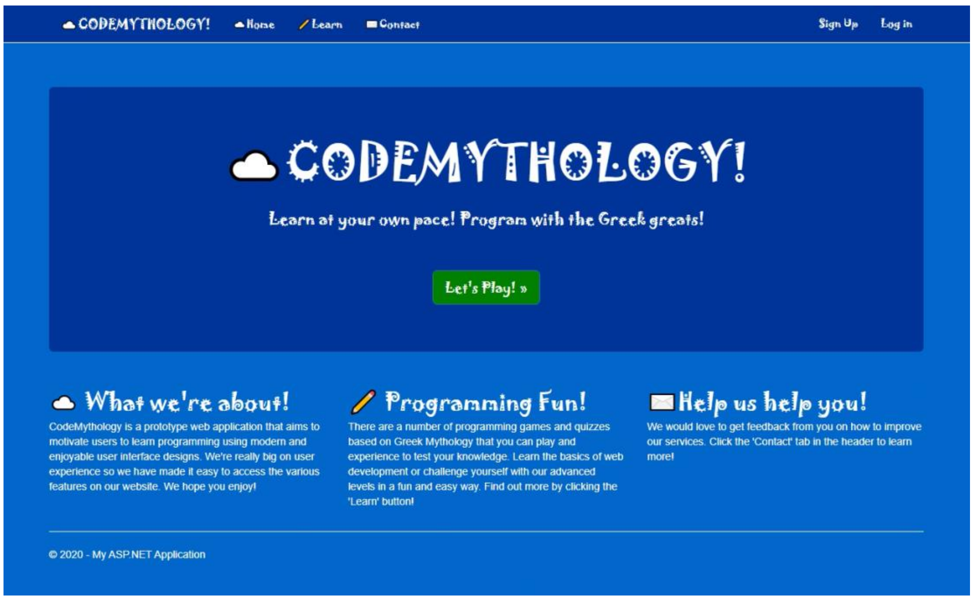
Type the following in the editor above:

var hello = 'Hello World';

Then the next instructions will follow!

Codemoji's computer science curriculum for schools allows 1st-8th grade students to learn the basics of web development and coding, including HTML, CSS, and Javascript, in a fun and easy way, allowing them to create their own websites, animations and much more with our unique and adaptable learning platform.

CodeMythology home page:



Learn at your own pace! Program with the Greek greats!

What we're about!

CodeMythology is a prototype web application that aims to motivate users to learn programming using modern and enjoyable user interface designs. We're really big on user experience so we have made it easy to access the various features on our website. We hope you enjoy!

Programming Fun!

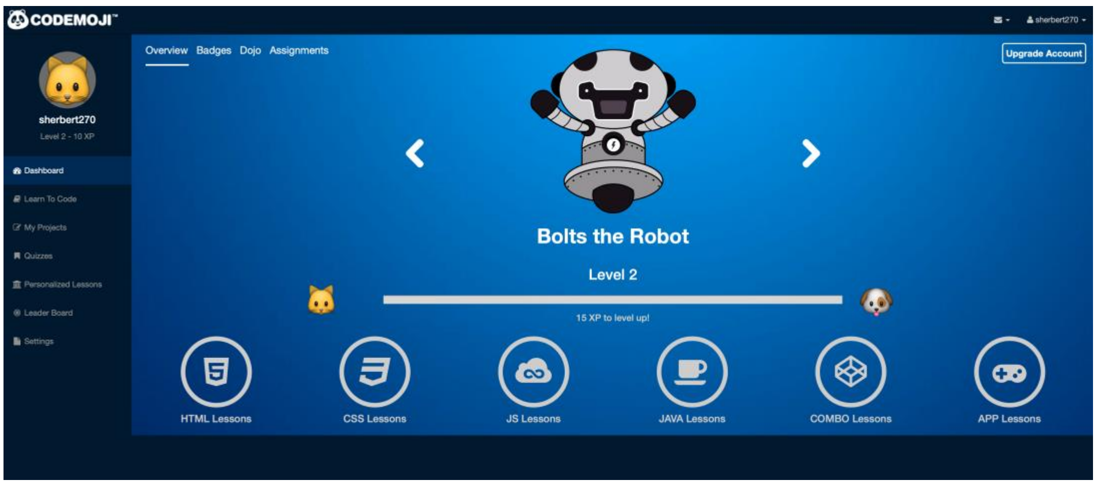
There are a number of programming games and quizzes based on Greek Mythology that you can play and experience to test your knowledge. Learn the basics of web development or challenge yourself with our advanced levels in a fun and easy way. Find out more by clicking the 'Learn' button!

Help us help you!

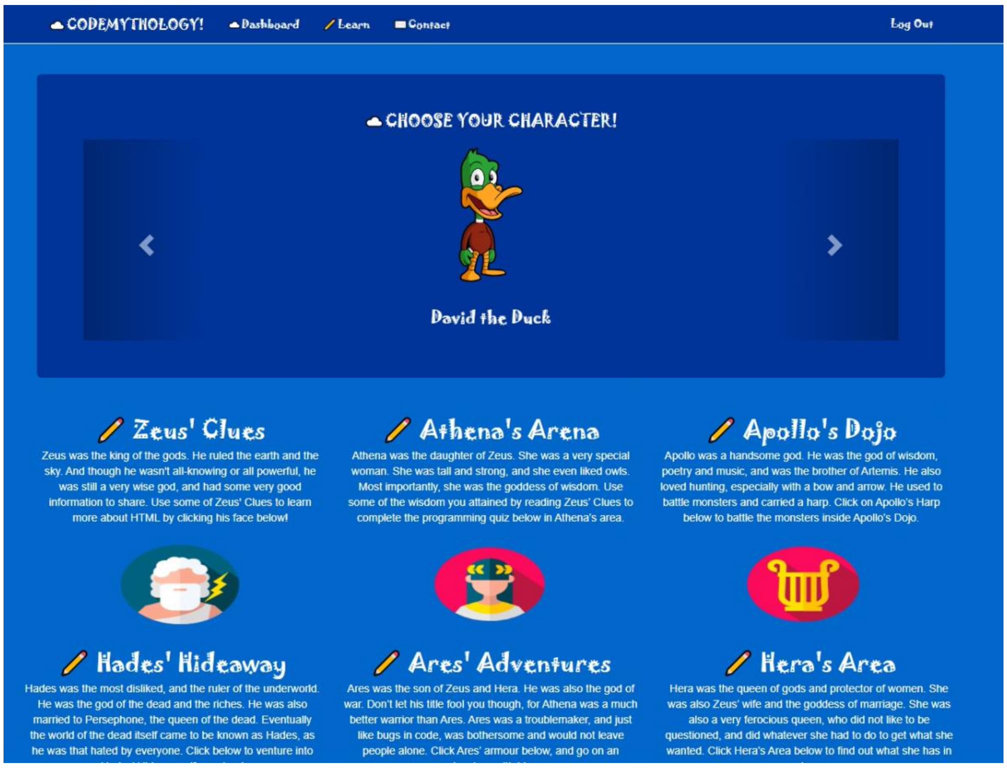
We would love to get feedback from you on how to improve our services. Click the 'Contact' tab in the header to learn more!

© 2020 - My ASP.NET Application

CodeMoji dashboard:



CodeMythology dashboard:



Zeus' Clues

Zeus was the king of the gods. He ruled the earth and the sky. And though he wasn't all-knowing or all powerful, he was still a very wise god, and had some very good information to share. Use some of Zeus' Clues to learn more about HTML by clicking his face below!

Athena's Arena

Athena was the daughter of Zeus. She was a very special woman. She was tall and strong, and she even liked owls. Most importantly, she was the goddess of wisdom. Use some of the wisdom you attained by reading Zeus' Clues to complete the programming quiz below in Athena's area.

Apollo's Dojo

Apollo was a handsome god. He was the god of wisdom, poetry and music, and was the brother of Artemis. He also loved hunting, especially with a bow and arrow. He used to battle monsters and carried a harp. Click on Apollo's Harp below to battle the monsters inside Apollo's Dojo.

Hades' Hideaway

Hades was the most disliked, and the ruler of the underworld. He was the god of the dead and the riches. He was also married to Persephone, the queen of the dead. Eventually the world of the dead itself came to be known as Hades, as he was that hated by everyone. Click below to venture into

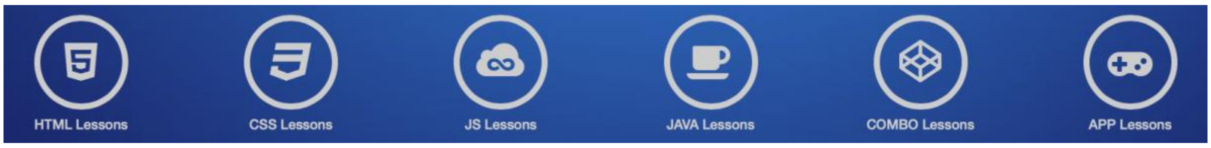
Ares' Adventures

Ares was the son of Zeus and Hera. He was also the god of war. Don't let his title fool you though, for Athena was a much better warrior than Ares. Ares was a troublemaker, and just like bugs in code, was bothersome and would not leave people alone. Click Ares' armour below, and go on an

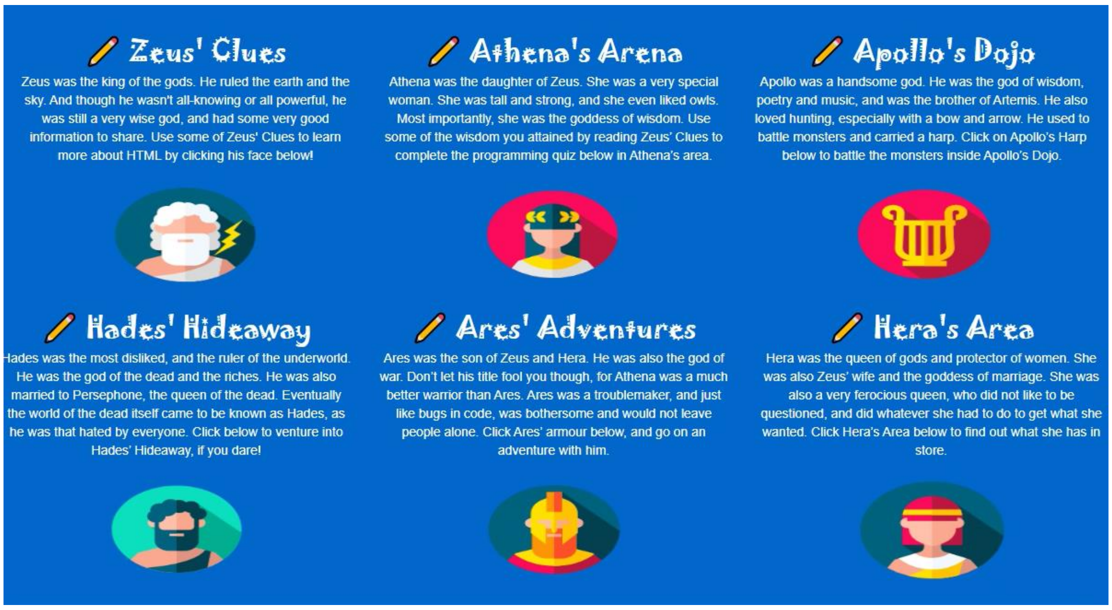
Hera's Area

Hera was the queen of gods and protector of women. She was also Zeus' wife and the goddess of marriage. She was also a very ferocious queen, who did not like to be questioned, and did whatever she had to do to get what she wanted. Click Hera's Area below to find out what she has in

CodeMoji lessons:



CodeMythology lessons:



Zeus' Clues

Zeus was the king of the gods. He ruled the earth and the sky. And though he wasn't all-knowing or all powerful, he was still a very wise god, and had some very good information to share. Use some of Zeus' Clues to learn more about HTML by clicking his face below!

Athena's Arena

Athena was the daughter of Zeus. She was a very special woman. She was tall and strong, and she even liked owls. Most importantly, she was the goddess of wisdom. Use some of the wisdom you attained by reading Zeus' Clues to complete the programming quiz below in Athena's area.

Apollo's Dojo

Apollo was a handsome god. He was the god of wisdom, poetry and music, and was the brother of Artemis. He also loved hunting, especially with a bow and arrow. He used to battle monsters and carried a harp. Click on Apollo's Harp below to battle the monsters inside Apollo's Dojo.

Hades' Hideaway

Hades was the most disliked, and the ruler of the underworld. He was the god of the dead and the riches. He was also married to Persephone, the queen of the dead. Eventually the world of the dead itself came to be known as Hades, as he was that hated by everyone. Click below to venture into Hades' Hideaway, if you dare!

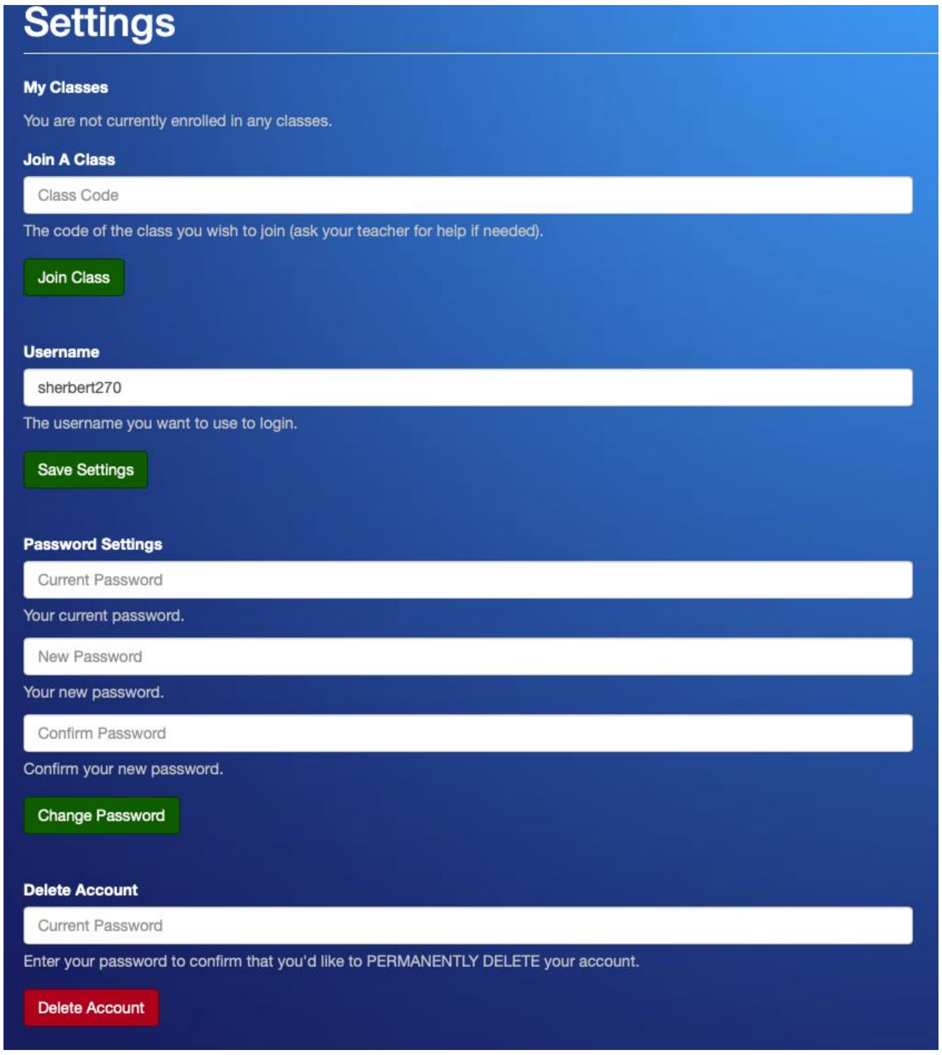
Ares' Adventures

Ares was the son of Zeus and Hera. He was also the god of war. Don't let his title fool you though, for Athena was a much better warrior than Ares. Ares was a troublemaker, and just like bugs in code, was bothersome and would not leave people alone. Click Ares' armour below, and go on an adventure with him.

Hera's Area

Hera was the queen of gods and protector of women. She was also Zeus' wife and the goddess of marriage. She was also a very ferocious queen, who did not like to be questioned, and did whatever she had to do to get what she wanted. Click Hera's Area below to find out what she has in store.

CodeMoji settings page:



Settings

My Classes

You are not currently enrolled in any classes.

Join A Class

Class Code

The code of the class you wish to join (ask your teacher for help if needed).

Join Class

Username

sherbert270

The username you want to use to login.

Save Settings

Password Settings

Current Password

Your current password.

New Password

Your new password.

Confirm Password

Confirm your new password.

Change Password

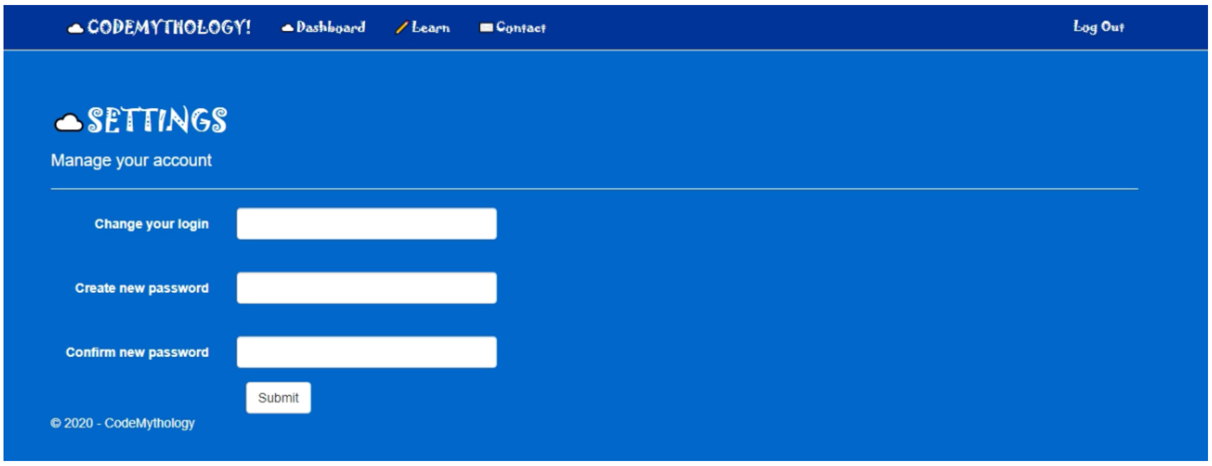
Delete Account

Current Password

Enter your password to confirm that you'd like to PERMANENTLY DELETE your account.

Delete Account

CodeMythology settings page:



CODEMYTHOLOGY! Dashboard Learn Contact Log Out

SETTINGS

Manage your account

Change your login

Create new password

Confirm new password

Submit

© 2020 - CodeMythology

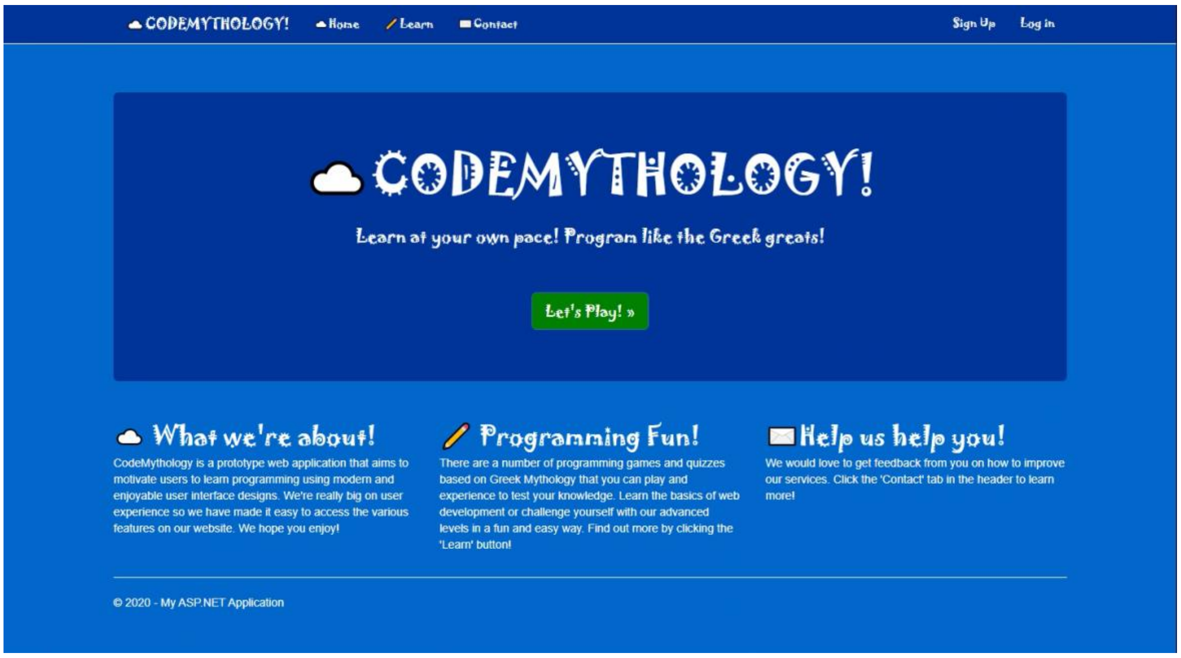
APPENDIX D: Extended User Documentation

Introduction

This section aims to provide an extended user documentation regarding the functionality of the application to potential end users. This is important, as it might not be completely obvious to the user how the software that has been developed fully works. A study carried out by Gemoets et al. (2002) found that the "user satisfaction of a system is strongly influenced by its documentation", as it provides the user with a better understanding of the system they are using. Another study also recommended that it was good practice to write computer user documentation, as it reflects "industry procedures" (Brockmann et al., 1986). Of course, with this said, the user documentation must be well designed to satisfy these requirements. Shown below is the user documentation for the web application.

Documentation

Upon first viewing the application, the user will find a well-designed CodeMythology home page. The name 'CodeMythology' integrates coding with Greek Mythology, so a lot of the features in the application will have to do with that. There are three paragraphs at the bottom of the page describing what the application does, and the buttons on the GUI are labelled and easily visible. There are 6 menu-bar icons running across the top of the screen labelled; 'CodeMythology', 'Home', 'Learn', 'Contact', Sign Up' and 'Login':



What we're about!

CodeMythology is a prototype web application that aims to motivate users to learn programming using modern and enjoyable user interface designs. We're really big on user experience so we have made it easy to access the various features on our website. We hope you enjoy!

Programming Fun!

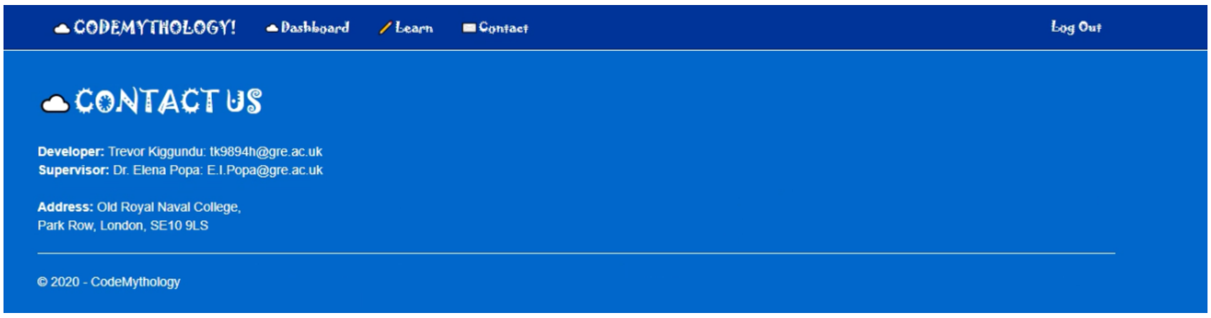
There are a number of programming games and quizzes based on Greek Mythology that you can play and experience to test your knowledge. Learn the basics of web development or challenge yourself with our advanced levels in a fun and easy way. Find out more by clicking the 'Learn' button!

Help us help you!

We would love to get feedback from you on how to improve our services. Click the 'Contact' tab in the header to learn more!

© 2020 - My ASP.NET Application

Clicking ‘Contact’ on the navigation bar will redirect the user to the contact page, where they can view the contact details for the developer:



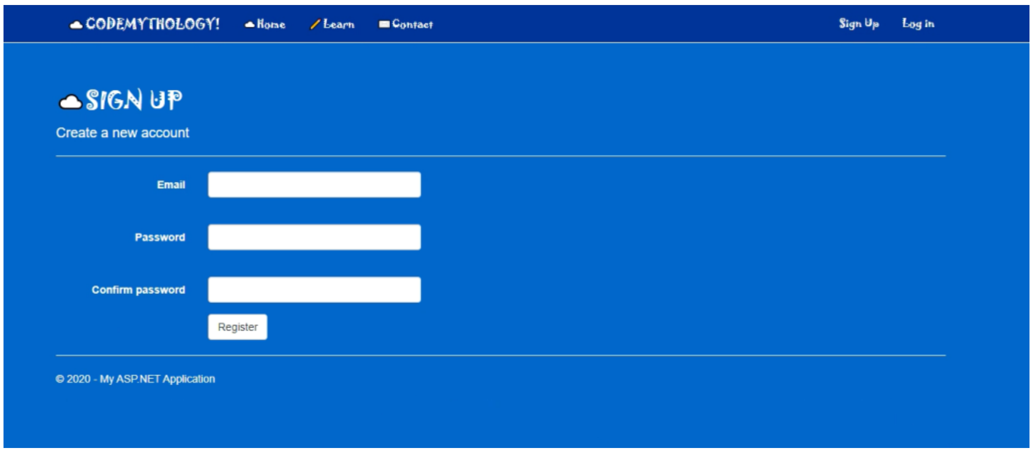
CONTACT US

Developer: Trevor Kiggundu: tk9894h@gre.ac.uk
Supervisor: Dr. Elena Popa: E.I.Popa@gre.ac.uk

Address: Old Royal Naval College,
Park Row, London, SE10 9LS

© 2020 - CodeMythology

Clicking the ‘Learn’ or ‘Sign Up’ icons will redirect the user to the registration page, where they can create an account to access the other features of the application:



SIGN UP

Create a new account

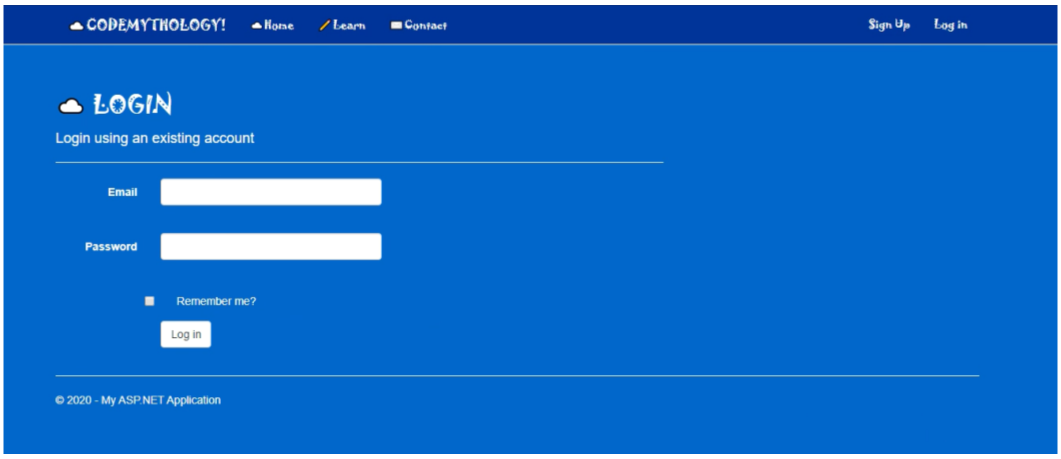
Email

Password

Confirm password

© 2020 - My ASP.NET Application

Successfully creating an account will redirect the user to the ‘Login’ page, where they can log in and access the dashboard page. Users can also access the login page by clicking the green ‘Let’s Play’ button, or the ‘Login’ button on the navigation bar.



LOGIN

Login using an existing account

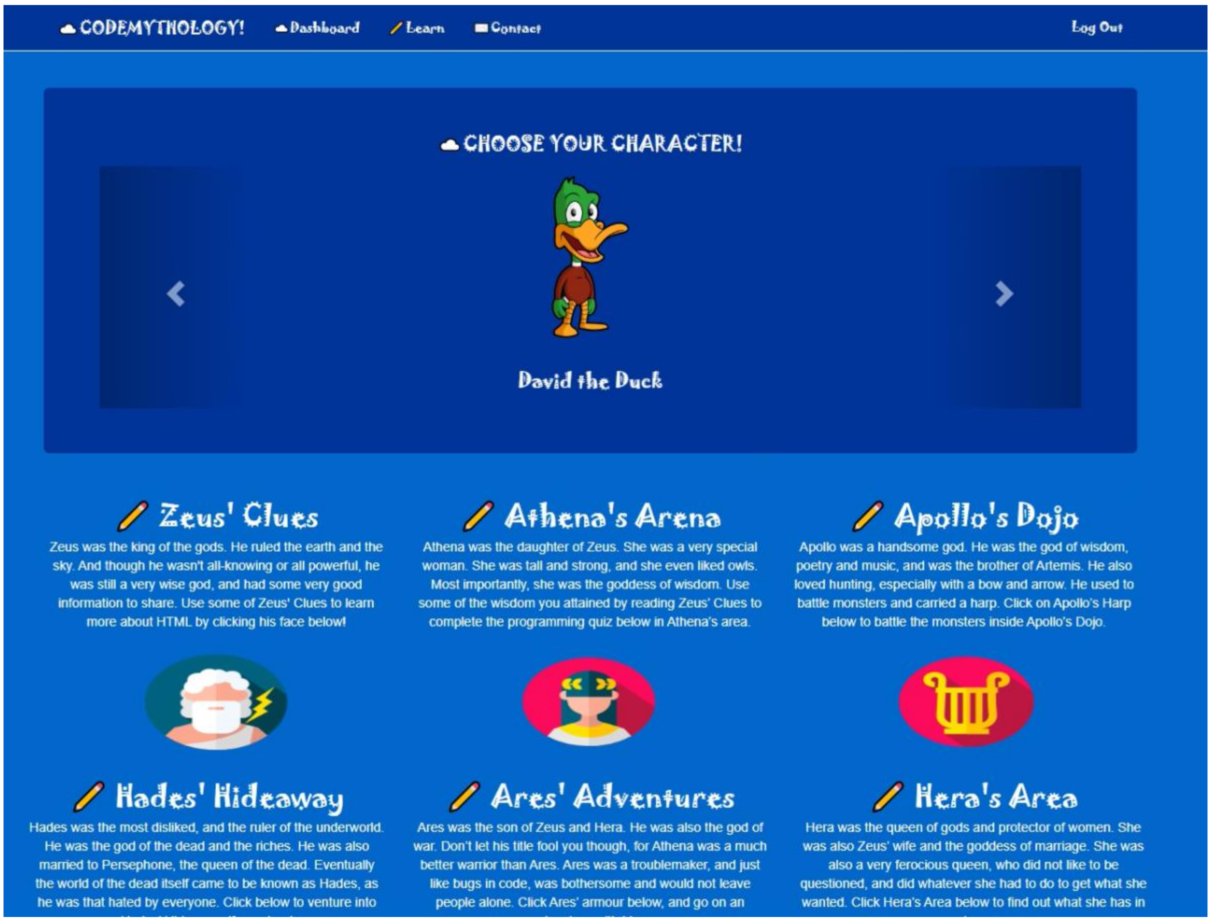
Email

Password

Remember me?

© 2020 - My ASP.NET Application

Successful user login will lead the user to the ‘Dashboard’ page, where the user will find the most interactivity features in the application. Notice how the icons in the navigation bar have changed and now read: ‘Dashboard’, ‘Learn’ ‘Contact’ and ‘Logout’. Clicking the ‘Logout button will log the user out of the application, and the ‘Contact’ page remains the same:



Zeus' Clues

Zeus was the king of the gods. He ruled the earth and the sky. And though he wasn't all-knowing or all powerful, he was still a very wise god, and had some very good information to share. Use some of Zeus' Clues to learn more about HTML by clicking his face below!

Athena's Arena

Athena was the daughter of Zeus. She was a very special woman. She was tall and strong, and she even liked owls. Most importantly, she was the goddess of wisdom. Use some of the wisdom you attained by reading Zeus' Clues to complete the programming quiz below in Athena's area.

Apollo's Dojo

Apollo was a handsome god. He was the god of wisdom, poetry and music, and was the brother of Artemis. He also loved hunting, especially with a bow and arrow. He used to battle monsters and carried a harp. Click on Apollo's Harp below to battle the monsters inside Apollo's Dojo.

Hades' Hideaway

Hades was the most disliked, and the ruler of the underworld. He was the god of the dead and the riches. He was also married to Persephone, the queen of the dead. Eventually the world of the dead itself came to be known as Hades, as he was that hated by everyone. Click below to venture into

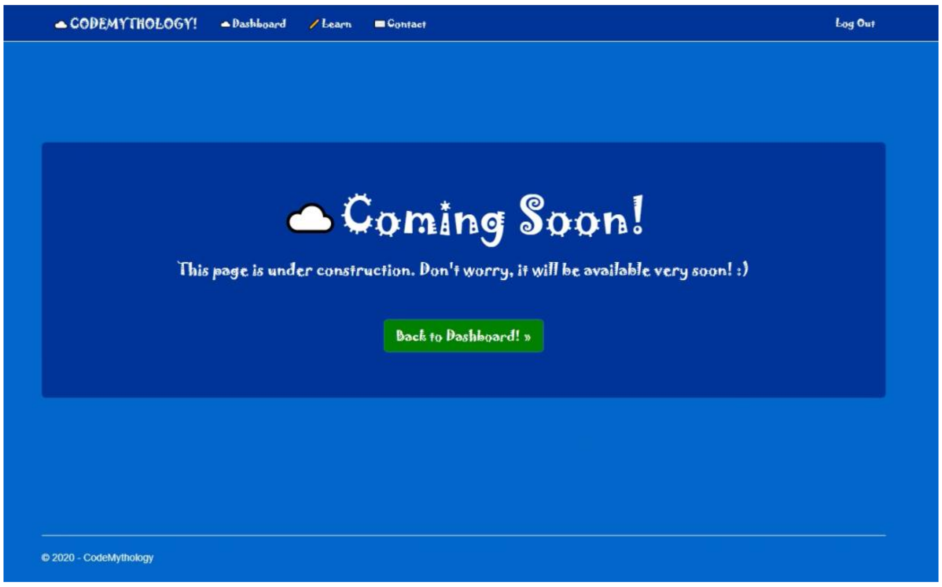
Ares' Adventures

Ares was the son of Zeus and Hera. He was also the god of war. Don't let his title fool you though, for Athena was a much better warrior than Ares. Ares was a troublemaker, and just like bugs in code, was bothersome and would not leave people alone. Click Ares' armour below, and go on an

Hera's Area

Hera was the queen of gods and protector of women. She was also Zeus' wife and the goddess of marriage. She was also a very ferocious queen, who did not like to be questioned, and did whatever she had to do to get what she wanted. Click Hera's Area below to find out what she has in

The user can choose which character they would like to accompany them on their journey by clicking the ‘left’ and ‘right’ arrows on the carousel. The user will also find a list of 6 lessons named after Greek gods to fit the Greek Mythology. Unfortunately, the final three lessons are still under construction, so clicking them will redirect the user to the ‘Coming Soon’ page:

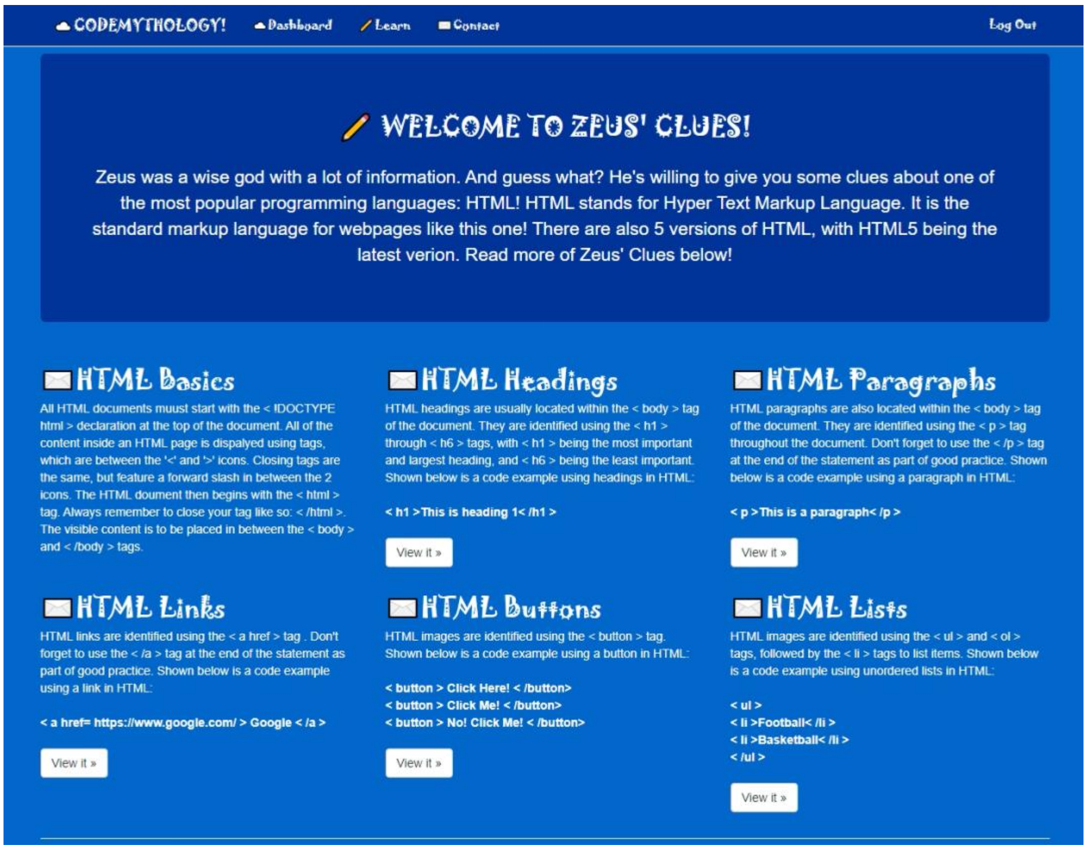


Coming Soon!

This page is under construction. Don't worry, it will be available very soon! :)

© 2020 - CodeMythology

Clicking the ‘Zeus’ Clues’ icon on the dashboard will redirect the user to a page where they can learn the basics of HTML:



WELCOME TO ZEUS' CLUES!

Zeus was a wise god with a lot of information. And guess what? He's willing to give you some clues about one of the most popular programming languages: HTML! HTML stands for Hyper Text Markup Language. It is the standard markup language for webpages like this one! There are also 5 versions of HTML, with HTML5 being the latest verion. Read more of Zeus' Clues below!

HTML Basics

All HTML documents muust start with the < !DOCTYPE html > declaration at the top of the document. All of the content inside an HTML page is dispalyed using tags, which are between the '<' and '>' icons. Closing tags are the same, but feature a forward slash in between the 2 icons. The HTML doument then begins with the < html > tag. Always remember to close your tag like so: < /html >. The visible content is to be placed in between the < body > and < /body > tags.

HTML Headings

HTML headings are usually located within the < body > tag of the document. They are identified using the < h1 > through < h6 > tags, with < h1 > being the most important and largest heading, and < h6 > being the least important. Shown below is a code example using headings in HTML:

< h1 >This is heading 1< /h1 >

View it »

HTML Paragraphs

HTML paragraphs are also located within the < body > tag of the document. They are identified using the < p > tag throughout the document. Don't forget to use the < /p > tag at the end of the statement as part of good practice. Shown below is a code example using a paragraph in HTML:

< p >This is a paragraph< /p >

View it »

HTML Links

HTML links are identified using the < a href > tag . Don't forget to use the < /a > tag at the end of the statement as part of good practice. Shown below is a code example using a link in HTML:

< a href= https://www.google.com/ > Google < /a >

View it »

HTML Buttons

HTML images are identified using the < button > tag. Shown below is a code example using a button in HTML:

< button > Click Here! < /button>
< button > Click Me! < /button>
< button > No! Click Me! < /button>

View it »

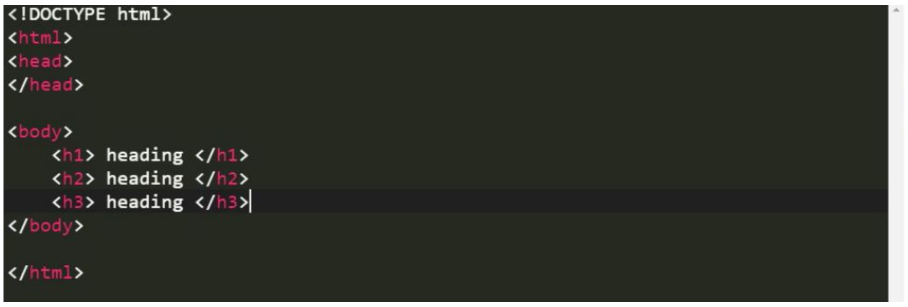
HTML Lists

HTML images are identified using the < ul > and < ol > tags, followed by the < li > tags to list items. Shown below is a code example using unordered lists in HTML:

< ul >
< li >Football< /li >
< li >Basketball< /li >
< /ul >

View it »

Each code example features CodeMythology’s version of W3Schools’s ‘Try it Editor’, where they can practice the skills they have just learned. Shown below is an example:



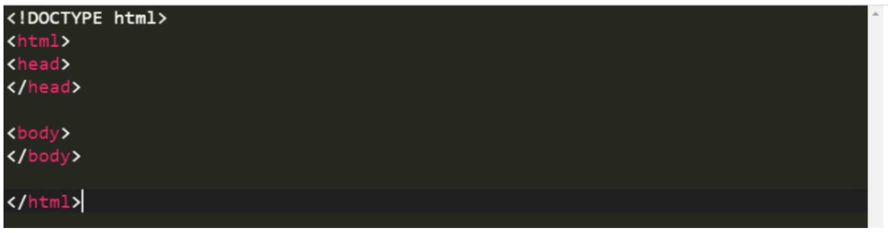
```
<!DOCTYPE html>
<html>
<head>
</head>

<body>
    <h1> heading </h1>
    <h2> heading </h2>
    <h3> heading </h3>
</body>

</html>
```

heading

heading

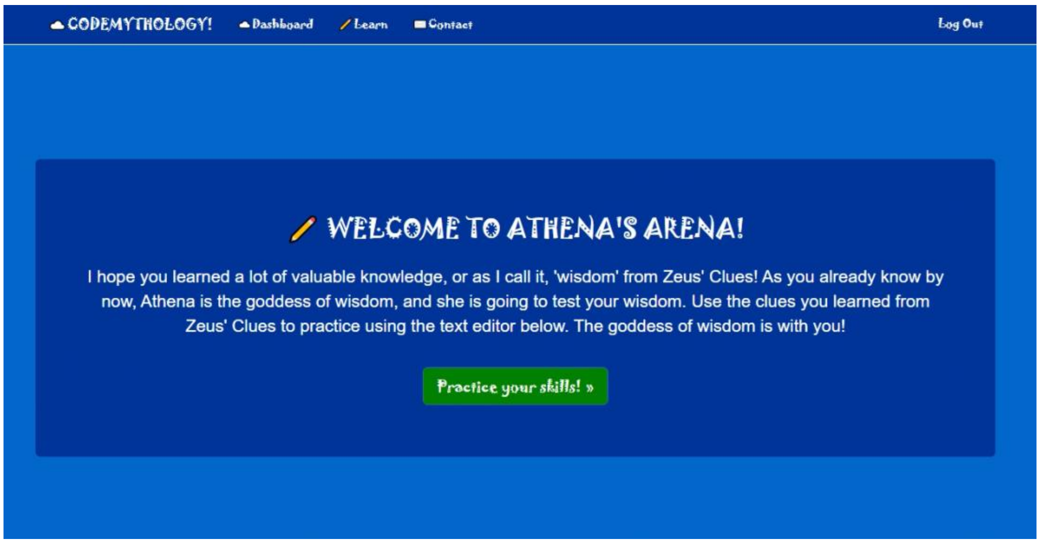
heading

Clicking the Athena’s Arena icon on the dashboard will redirect the user to a page where they can practice the basics of HTML on their own without pre-loaded HTML:



```
<!DOCTYPE html>
<html>
<head>
</head>

<body>
</body>

</html>
```

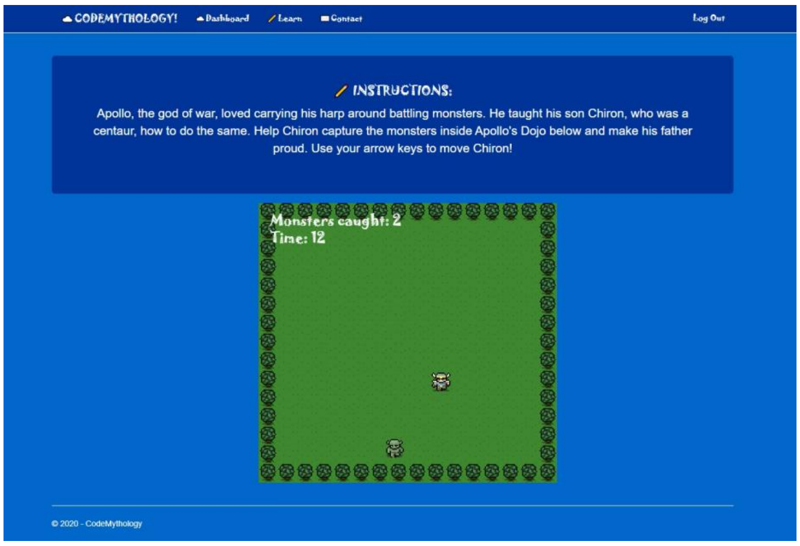
The ‘Athena’s Arena’ page leading to the code editor is just a regular webpage with instructions. Clicking the green ‘Practice your skills!’ button will lead the user to the empty code editor shown above:



WELCOME TO ATHENA'S ARENA!

I hope you learned a lot of valuable knowledge, or as I call it, 'wisdom' from Zeus' Clues! As you already know by now, Athena is the goddess of wisdom, and she is going to test your wisdom. Use the clues you learned from Zeus' Clues to practice using the text editor below. The goddess of wisdom is with you!

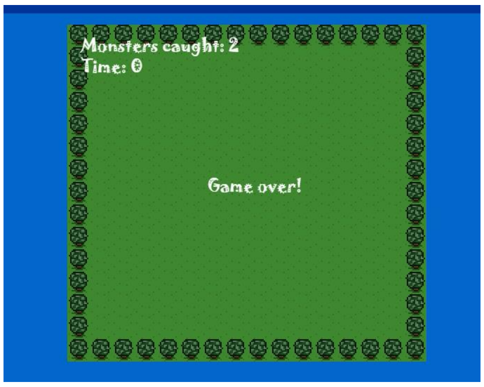
Clicking the ‘Apollo’s Dojo’ icon on the dashboard will redirect the user to a page where they can play a Greek Mythology game, based on actual stories:



INSTRUCTIONS:

Apollo, the god of war, loved carrying his harp around battling monsters. He taught his son Chiron, who was a centaur, how to do the same. Help Chiron capture the monsters inside Apollo's Dojo below and make his father proud. Use your arrow keys to move Chiron!

The game will display a ‘Game over!’ message, and it is up to the user to try to beat their best score in under 30 seconds.



APPENDIX E: Project Proposal

1 Introduction

The current society is a “technology-based” (Susskind, 2015) one, and there seems to be no sign of that stopping any time soon. The demand for these types of technology professionals is at an all-time high. A career in this profession is also one of the rarer ones in which the number of job opportunities continues to increase despite the parallel increase in recruits available. Or at least that is what it seems like on the outside. Astonishingly, a large percentage of job applicants, even “those with masters' degrees and PhDs in computer science, fail during interviews when asked to carry out basic programming tasks” (Kegel, 2014). This points to a problem regarding the external motivation of junior programmers, as “most programming courses don't provide the kind of experience that real programming gives” (Kegel, 2014), only laying down a good base for the student to explore and learn more themselves. Without the proper motivation, and the correct learning tools, it is difficult for young programmers to possess both the theoretical and practical skills needed to work in the real world, as too many graduates only possess the former as opposed to the latter.

2 Problem Domain

The main focus of this project is to create an innovative and interactive web-based learning application to motivate students to learn programming. Designing an “effective” and “interactive learning environment requires understanding the intricate relationships among people” (Repenning, 2006), so there will need to be more background research on learning practices. In order to do this, a secondary objective will be introduced, which is to research and investigate the motivating factors that students have regarding the task of learning to program. The main investigation will include conducting deeper research into more innovative UX and UI designs, and HCI factors in order to provide an alternative to the various already available online websites that aim to teach programming through web applications. These applications, however, are still important, as they provide a different learning approach through games, quizzes and “visualization features to improve students’ understanding of the programming concepts” (Moons et al., 2013). The bulk of these investigations will be documented in the literature review, and the web application will be created in lieu of these findings.

2.1 Further Research Ideas

Future research regarding the motivation of students to learn programming could venture into exploring the reasons that students choose to pursue this job path today. Technological advancements over the years mean that the range of topics that fall underneath the tech field are vast, from games design to industry software development. The diversity in the age and gender of those involved in the field has also been noted, with women "building careers as professional engineers in a high technology industrial organization" (Evetts, 1998) more frequently than ever. Individuals are also picking up the practice of programming at an even younger age than before, mostly through online games and reasoning programs, as these seem to be promising tools for "assessing computational thinking in young programmers" (Touretzky et al., 2017). A study could be done on whether these young learners are being guided/persuaded into joining the field or are just inquisitive enough and have completely intrinsic intentions towards learning to program.

3 Methodology

3.1 Background Research

Various methods have been used to teach programming; some are more outdated and/or successful than others, but all were created to achieve the same goal. The conventional teacher-student model is the oldest and most common. This model is quite self-explanatory, though it does branch out into practices such as apprenticeship, more commonly known as 'placement', where the student is offered "continuous feedback" by one or more mentors to not only differ from "traditionally constructed courses", but also offer the "most efficient means of learning" (Vihavainen et al., 2011). Another form of learning is self-study, in which the student teaches themselves the required material using various means (ex: books, online websites, games, journals etc.). A study by Bergin et al. (2005) sheds a brighter view on self-regulated learning, stating a direct link between intrinsic motivation and SRL (Student Regulated Learning), with students exhibiting such behaviors being more inclined to "perform better in programming" than those with more extrusive motivational factors. Learning via web applications best fits this category, as it requires the student to explore more about the given subject in their own time. Detailed below are the various objectives and methodologies that will be implemented in the proposed project.

3.2 SMART Objectives

Smart Objective 1:

- Within the allocated time-period (1.5 months), aim to collect user feedback regarding the usefulness of the learning application created at the half-way level. User feedback will also be used to determine which topic will be taught on the application. The feedback will be anonymous and will be directed accordingly to create the least amount of bias possible for more reliable data collection.

Smart Objective 2:

- Within the allocated time-period (1.5 months), plan to create a web-based application prototype to be both client and developer tested. This will allow for early feedback so that the appropriate changes can be made before the final product is presented.

Smart Objective 3:

- Within the allocated time-period (6 months), aim to create a fully functional learning-based web application with more interactive features than the prototype. The final product will be white and black box tested, as well as client and developer tested again. Feedback will be taken from this final product and compared with the prototype.

Smart Objective 4:

- Within the allocated time-period (6 months), aim to compile all the research, data and results collected in order to write a completed research paper, poster and presentation in fulfilment of the requirements for the module and final year project. This will be subject to the success of the web application created, as well as the user feedback.

3.3 Deliverables

The proposed project will deliver two specific products; an innovative and interactive web application in conjunction with a project report written in LaTex. The latter will be a direct product of technical research, literary analysis, and the findings documented from the study conducted.

3.4 Project Methodology

As mentioned before, the proposed project will not only include investigating the motivating factors regarding learning programming, but also the user interface designs of the already available learning avenues. This will help to develop a web application that aims to provide an alternate learning experience than the traditional lecture and tutorial model used by many learning organizations. This is important, as "traditional ways of assignment grading are not scalable" and often do not "give timely or interactive feedback to students" (Tillman et al., 2013). The application itself will be created to teach a programming language/topic to first and/or second year students that has not been chosen yet; this will be handled in either one or two ways. A survey via questionnaire will be uploaded online for students themselves to choose which topic to learn about. Alternatively, the web application will be catered to teach a subject that is already being taught by lecturers at the university, as this will provide a sample size, at most, as large as 250-300 students, provided they all respond to the survey. The final goal indeed is to keep learners motivated, so the web application will have different interactive features such as quizzes, checkpoints and level progression to keep the user interested in the subject. These will aim to keep the user more interested, as "improving learning effectiveness has always been a constant challenge" and "computer games are one means to encourage learners to learn" (Long, 2017). Previous experience from the Web Application Development, Database Technologies and Application Development modules will be used to develop this application.

3.5 Systems Development Methodology

This section aims to detail the systems development methodologies considered during the planning of the proposed project. The Extreme Programming and Spiral models were eliminated immediately due to the nature of the task at hand. The Waterfall model was also eliminated as it is a model that does not promote user interactivity until the end of the project cycle, a big staple in the production of this application. It is also aimed at large and expensive projects, two variables that do not describe the proposed idea. The system development methodology that would best suit this project is a close call between the Agile Development model and the Rapid Application Development (RAD). Both models are low costing, relatively defined and informal, and have a "small-to-medium scale and of short duration" (Kyriakidou, 2018). However, the negatives of the agile development model centered around the fact that it involves customer collaboration at a much higher level than the RAD model.

Given the time constraints, user interactivity and complexity of the project, the conclusion was made that the best systems development model to follow would be the Rapid Action Development (RAD).

3.6 Test Bed

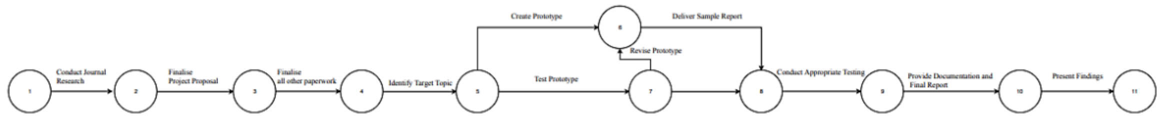
A series of white and black box tests, as well as unit tests, will be run to test the functionality of the web application, database integration, HCI factors and the user documented features of the project. The testing will be run on both developer and user side to eliminate any possible bias and receive as much feedback as possible.

3.7 Models

Models to be implemented will include UML diagrams, class diagrams and updated Pert/Gantt charts. Functional and non-functional requirements as well as ERD diagrams will also be included.

3.7.1 Pert Chart

Shown below is a pert chart detailing some of the objectives that are to be achieved during the duration of the proposed project.



4 Evaluation

Data collection will be an instrumental part of the process, as the proposed project attempts to solve real world programming issues. Previous project proposals were halted due to the difficulties that accompanied the task of finding reliable and readily available test data. This proposal will be different; as year 1 and 2 students will provide data about various topics such as interest in programming, desired topics, and general user interface feedback. The appropriate legal, social, ethical and professional guidelines will be implemented throughout the study. After the web application is created, user feedback will be collected from a select number of participants to gather information on whether the application has had any positive/negative impact on the concept of the selected topic, and if it has increased confidence/motivation in the individual. This is very important, as students can find the process of learning to program to be a “difficult and even unpleasant task” without the proper “guidance and feedback” (Watson et al., 2011) given back to them. The completion of these objectives, as well as the ones documented in the ‘problem domain’ and ‘project methodology’ sections, will determine whether the proposed project will be successful.

APPENDIX F: Supporting Documents

Questionnaire:

A study based on the downfalls of modern programming practices and user interface designs and their effects on students in order to aid in the creation of an innovative and interactive web-based learning application to motivate learning.

Trevor Kiggundu

The purpose of the questionnaire will be to decide the programming language that will be focused on in the web application. All collected responses and results will be destroyed apart from the use of questionnaire results for the final paper. There will be no personal data collected during the duration of this study. This survey will take you between 5-10 minutes to complete.

DISCLAIMER

This study is being conducted by Trevor Kiggundu, a BEng Software Engineering student at the University of Greenwich. The study is part of the COMP1682 Final Year Project module, and will be supervised by Dr. Elena Popa. Thank you for agreeing to participate in this study.

QUESTIONNAIRE

How much programming knowledge do you currently possess?

- Low
- Intermediate
- High

Which programming languages are you familiar with?

- JavaScript
- C#
- Java
- Python
- Other (Specify Below):
  ____

Would you consider your theoretical programming skills to be stronger than your practical programming skills? (**theoretical**: knowing about programming; **practical:** physically practicing programming)

- Yes
- No

How confident are you in your practical programming skills? Rate yourself from **1-10** (1 being the lowest, and 10 being the highest).

Not Confident **0 1 2 3 4 5 6 7 8 9 10** Highly Confident

Which programming language would you like to learn more about?

- JavaScript

- C#
- Java
- Python
- Other (Specify Below):

  ____

What do you think about the teaching methods the university offers? Are they helpful enough?

____

Rate your experience learning programming in a classroom setting from **1-10** (1 being the lowest, and 10 being the highest).

Dissatisfied **0 1 2 3 4 5 6 7 8 9 10** Very Satisfied

Learning to program can often be a difficult and daunting task. What difficulties do you face while learning to program?

- It is difficult
- I don’t enjoy it
- Not interesting enough
- Lack of time/Other hobbies
- Other (Specify Below)

  ____

Do you think a web application would make it easier for you to learn to program outside of the classroom?

- Yes
- No

Do you think a web application with games/quizzes would help you stay more interested while learning to program?

- Yes
- No

Are there any types of games/quizzes that you like playing in your spare time?

- Yes (Please Specify)
- No

____

Which of the following online learning tools have you used before?

- CodeMonkey
- W3Schools
- CodeCombat
- CodeMoji
- Other (Specify Below)

____

Rate your experience using online learning tools to learn programming on a scale of **1-10** (1 being the lowest, and 10 being the highest)

Dissatisfied **0 1 2 3 4 5 6 7 8 9 10** Very Satisfied

Do you think that online learning tools are better than conventional learning environments?

____

Do you have any suggestions for this study?

____

APPENDIX G: Borrowed Code

Bootstrap Carousel

```
<div id="myCarousel" class="carousel slide" data-ride="carousel">
  <!-- Indicators -->
  <ol class="carousel-indicators">
    <li data-target="#myCarousel" data-slide-to="0" class="active"></li>
    <li data-target="#myCarousel" data-slide-to="1"></li>
    <li data-target="#myCarousel" data-slide-to="2"></li>
  </ol>

  <!-- Wrapper for slides -->
  <div class="carousel-inner">
    <div class="item active">
      <img src="la.jpg" alt="Los Angeles">
    </div>

    <div class="item">
      <img src="chicago.jpg" alt="Chicago">
    </div>

    <div class="item">
      <img src="ny.jpg" alt="New York">
    </div>
  </div>

  <!-- Left and right controls -->
  <a class="left carousel-control" href="#myCarousel" data-slide="prev">
    <span class="glyphicon glyphicon-chevron-left"></span>
    <span class="sr-only">Previous</span>
  </a>
  <a class="right carousel-control" href="#myCarousel" data-slide="next">
    <span class="glyphicon glyphicon-chevron-right"></span>
    <span class="sr-only">Next</span>
  </a>
</div>
```

HTML Code Editor

```
<!DOCTYPE html>
<html>
<head>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/ace/1.4.11/ace.js"></script>

    <style>
        html, body {
            margin: 0;
            padding: 0;
            height: 100%;
            width: 100%;
            overflow: hidden;
        }

        #editor {
            height: 100%;
            width: 50%;
            display: inline-block;
        }

        #iframe {
            height: 100%;
            width: 50%;
            display: inline-block;
        }

        #container {
            height: 100%;
            width: auto;
            white-space: nowrap;
            overflow: hidden;
        }

        /* disable tag matching */
        .ace_editor .ace_marker-layer .ace_bracket {
            display: none
        }
    </style>

</head>

<body onload="ready()">
    <div id="container">

        <div id="editor">
        </div>

        <iframe id="iframe" frameborder="0"></iframe>
    </div>

    <script>

        function update() {
            var idoc = document.getElementById('iframe').contentWindow.document;

            idoc.open();
            idoc.write(editor.getValue());
            idoc.close();
        }

        function setupEditor() {
            window.editor = ace.edit("editor");
            editor.setTheme("ace/theme/monokai");
            editor.getSession().setMode("ace/mode/html");
            editor.setValue(`<!DOCTYPE html>
<html>
<head>
</head>

<body>
</body>

</html>`, 1); //1 = moves cursor to end

            editor.getSession().on('change', function () {
                update();
            });

            editor.focus();


            editor.setOptions({
                fontSize: "16pt",
                showLineNumbers: false,
                showGutter: false,
                vScrollBarAlwaysVisible: true,
                enableBasicAutocompletion: false, enableLiveAutocompletion: false
            });

            editor.setShowPrintMargin(false);
            editor.setBehavioursEnabled(false);
        }

        setupEditor();
        update();

    </script>

</body>
</html>
```

CSS for Login Page

```
<%@ Page Title="Login" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="Login.aspx.cs" Inherits="FYPTrevor1.Login" %>
<asp:Content ID="Content1" ContentPlaceHolderID="MainContent" runat="server">
    <br />
    <br />
     <h2 style="font-family:Jokerman" <span class="glyphicon glyphicon-cloud">
LOGIN</h2>
```

```
  <div class="row">
     <div class="col-md-8">
        <section id="loginForm">
           <div class="form-horizontal">
              <h4>Login using an existing account</h4>
              <hr />
              <asp:PlaceHolder runat="server" ID="ErrorMessage" Visible="false">
                 <p class="text-danger">
                    <asp:Literal runat="server" ID="FailureText" />
                 </p>
              </asp:PlaceHolder>
              <div class="form-group">
                 <asp:Label runat="server" AssociatedControlID="Email" CssClass="col-md-2
control-label">Email</asp:Label>
                 <div class="col-md-10">
                    <asp:TextBox runat="server" ID="Email" CssClass="form-control"
TextMode="Email" />
                    <asp:RequiredFieldValidator runat="server" ControlToValidate="Email"
                       CssClass="text-danger" ErrorMessage="The email field is required." />
                 </div>
              </div>

              <div class="form-group">
                 <asp:Label runat="server" AssociatedControlID="Password" CssClass="col-
md-2 control-label">Password</asp:Label>
                 <div class="col-md-10">
                    <asp:TextBox runat="server" ID="Password" TextMode="Password"
CssClass="form-control" />
                    <asp:RequiredFieldValidator runat="server"
ControlToValidate="Password" CssClass="text-danger" ErrorMessage="The password field
is required." />
                 </div>
              </div>

              <div class="form-group">
                 <div class="col-md-offset-2 col-md-10">
                    <div class="checkbox">
                       <asp:CheckBox runat="server" ID="RememberMe" />
                       <asp:Label runat="server"
AssociatedControlID="RememberMe">Remember me?</asp:Label>
                    </div>
                 </div>
              </div>

               <div class="form-group">
                 <div class="col-md-offset-2 col-md-10">
                    <asp:Button runat="server" Text="Log in" CssClass="btn btn-default"
OnClick="Unnamed6_Click" />
                 </div>
              </div>
```

```
                </div>



            </section>
        </div>



        </div>
</asp:Content>
```

CSS for Registration Page

```
<%@ Page Title="Sign Up" Language="C#" MasterPageFile="~/Site.Master"
AutoEventWireup="true" CodeBehind="SignUp.aspx.cs" Inherits="FYPTrevor1.SignUp"
%>
<asp:Content ID="Content1" ContentPlaceHolderID="MainContent" runat="server">
     <br />
    <br />
<h2 style="font-family:Jokerman" <span class="glyphicon glyphicon-cloud">SIGN
UP</h2>
    <p class="text-danger">
        <asp:Literal runat="server" ID="ErrorMessage" />
    </p>

    <div class="form-horizontal">
        <h4>Create a new account</h4>
        <hr />
        <asp:ValidationSummary runat="server" CssClass="text-danger" />
        <div class="form-group">
            <asp:Label runat="server" AssociatedControlID="Email" CssClass="col-md-2
control-label">Email</asp:Label>
            <div class="col-md-10">
                <asp:TextBox runat="server" ID="Email" CssClass="form-control"
TextMode="Email" />
                <asp:RequiredFieldValidator runat="server" ControlToValidate="Email"
                    CssClass="text-danger" ErrorMessage="The email field is required." />
            </div>
        </div>
        <div class="form-group">
            <asp:Label runat="server" AssociatedControlID="Password" CssClass="col-md-2
control-label">Password</asp:Label>
            <div class="col-md-10">
                <asp:TextBox runat="server" ID="Password" TextMode="Password"
CssClass="form-control" />
                <asp:RequiredFieldValidator runat="server" ControlToValidate="Password"
                    CssClass="text-danger" ErrorMessage="The password field is required." />
            </div>
        </div>
        <div class="form-group">
```

```
            <asp:Label runat="server" AssociatedControlID="ConfirmPassword" CssClass="col-
md-2 control-label">Confirm password</asp:Label>
            <div class="col-md-10">
                <asp:TextBox runat="server" ID="ConfirmPassword" TextMode="Password"
CssClass="form-control" />
                <asp:RequiredFieldValidator runat="server"
ControlToValidate="ConfirmPassword"
                    CssClass="text-danger" Display="Dynamic" ErrorMessage="The confirm
password field is required." />
                <asp:CompareValidator runat="server" ControlToCompare="Password"
ControlToValidate="ConfirmPassword"
                    CssClass="text-danger" Display="Dynamic" ErrorMessage="The password and
confirmation password do not match." />
            </div>
        </div>
        <div class="form-group">
            <div class="col-md-offset-2 col-md-10">
                <asp:Button runat="server" Text="Register" CssClass="btn btn-default"
OnClick="Unnamed9_Click" />
            </div>
        </div>
    </div>
</asp:Content>
```

Monster Game

```
/*
  Code modified from:
  http://www.lostdecadegames.com/how-to-make-a-simple-html5-canvas-game/
  using graphics purchased from vectorstock.com
*/
// Create the canvas for the game to display in
var canvas = document.createElement("canvas");
var ctx = canvas.getContext("2d");
canvas.width = 512;
canvas.height = 480;
canvas.style.left = "38%";
canvas.style.top = "35%";
canvas.style.position = "absolute";
document.body.appendChild(canvas);

// Load the background image
var bgReady = false;
var bgImage = new Image();
bgImage.onload = function () {
    // show the background image
    bgReady = true;
};
bgImage.src = "Images/world.png";
```

```
// Load the centaur image
var centaurReady = false;
var centaurImage = new Image();
centaurImage.onload = function () {

   // show the here image
   centaurReady = true;
};
centaurImage.src = "Images/centaur.png";

// Load the monster image
var monsterReady = false;
var monsterImage = new Image();
monsterImage.onload = function () {

   // show the monster image
   monsterReady = true;
};
monsterImage.src = "images/monster.png";

// Create the game objects
var centaur = {
   speed: 256 // movement speed of centaur in pixels per second
};
var monster = {};
var monstersCaught = 0;

// Handle keyboard controls
var keysDown = {};

// Check for keys pressed where key represents the keycode captured
addEventListener("keydown", function (key) {
   keysDown[key.keyCode] = true;
}, false);
addEventListener("keyup", function (key) {
   delete keysDown[key.keyCode];
}, false);

// Reset the player and monster positions when player catches a monster
var reset = function () {

   // Reset player's position to centre of canvas
   centaur.x = canvas.width / 2;
   centaur.y = canvas.height / 2;
   // Place the monster somewhere on the canvas randomly
   monster.x = 32 + (Math.random() * (canvas.width - 64));
   monster.y = 32 + (Math.random() * (canvas.height - 64));
};
// Update game objects - change player position based on key pressed
```

```
var update = function (modifier) {
   if (38 in keysDown) { // Player is holding up key
      centaur.y -= centaur.speed * modifier;
   }
   if (40 in keysDown) { // Player is holding down key
      centaur.y += centaur.speed * modifier;
   }
   if (37 in keysDown) { // Player is holding left key
      centaur.x -= centaur.speed * modifier;
   }
   if (39 in keysDown) { // Player is holding right key
      centaur.x += centaur.speed * modifier;
   }
   // Check if player and monster collider
   if (
      centaur.x <= (monster.x + 32)
      && monster.x <= (centaur.x + 32)
      && centaur.y <= (monster.y + 32)
      && monster.y <= (centaur.y + 32)
   ) {
      ++monstersCaught;
      reset();
   }
};
// Draw everything on the canvas
var render = function () {
   if (bgReady) {
      ctx.drawImage(bgImage, 0, 0);
   }
   if (centaurReady) {
      ctx.drawImage(centaurImage, centaur.x, centaur.y);
   }
   if (monsterReady) {
      ctx.drawImage(monsterImage, monster.x, monster.y);
   }
   // Display score and time
   ctx.fillStyle = "rgb(250, 250, 250)";
   ctx.font = "24px Jokerman";
   ctx.textAlign = "left";
   ctx.textBaseline = "top";
   ctx.fillText("Monsters caught: " + monstersCaught, 20, 20);
   ctx.fillText("Time: " + count, 20, 50);
   // Display game over message when timer finished
   if (finished == true) {
      ctx.fillText("Game over!", 200, 220);
   }

};
var count = 30; // how many seconds the game lasts for - default 30
var finished = false;
var counter = function () {
   count = count - 1; // countown by 1 every second
   // when count reaches 0 clear the timer, hide monster and
   // centaur and finish the game
   if (count <= 0) {
      // stop the timer
      clearInterval(counter);
      // set game to finished
      finished = true;
      count = 0;
      // hider monster and centaur
      monsterReady = false;
      centaurReady = false;
   }
}
// timer interval is every second (1000ms)
setInterval(counter, 1000);
// The main game loop
var main = function () {
   // run the update function
   update(0.02); // do not change
   // run the render function
   render();
   // Request to do this again ASAP
   requestAnimationFrame(main);
};
// Cross-browser support for requestAnimationFrame
var w = window;
requestAnimationFrame = w.requestAnimationFrame || w.webkitRequestAnimationFrame ||
w.msRequestAnimationFrame || w.mozRequestAnimationFrame;
// Let's play this game!
reset();
main();
```