



UNIVERSITY OF NATIONAL AND WORLD ECONOMY
The Spirit Makes The Power

Faculty: Applied Informatics and Statistics

Department: Information Technologies and Communications

Diploma Project

Design and development of a specialized Web site

Student: Head of the Project:_____

Petar Marinov Petrov

/ Prof. D. Velev /

Faculty Nr.: 12114099; Group 1320

Sofia 2017

Declaration of Authorship

I hereby certify that the graduation project is developed by me and it is based on my own work on the topic. All references, used literature and all sources of information, including charts and data, are quoted (including with URL addresses).

Date

Signature

Table of Contents

Introduction	4
I. Analysis of the Subject Field	5
1. Static Web pages.....	5
Step 1: Plan Your Web Presence	5
Step 2: Select the Tools for Making Your Home on the Web.....	6
Step 3: Make Key Design Decisions	8
Step 4: Learn the Code.....	10
Step 5: Optimize Your Site for Search Engines	13
Step 6: Testing Your Site	16
Step 7: Go Online.....	16
2. Dynamic Web Pages.....	17
Conclusion	19
II. Used Technologies, Methodologies and Platforms	19
1.Types of programming languages, platforms and comparison analysis	19
Scripting Languages.....	19
Databases	24
Web Servers	26
2.Choice of Languages, Server and DB Server	28
Conclusion:	30
III. Website projection.....	31
Conceptual Analysis	31
Functional Requirements.....	31
Conclusion	31
IV. Website Creation – Program Code, Techniques and Methodologies used	32
Site Map	32
Website Design – CSS code	34
Adding Content and Functionality – HTML code	36
PHP + MySQL.....	44
Database Schema	45
Userss Table	46
Emails Table.....	47
Roles Table	48

User_roles Table	48
PHP	49
Forum – PHPBB	55
Conclusion	56
V. Used Literature – References	57

Introduction

Nowadays the sport events are becoming more and more popular and the reason behind that is that they can be watched simultaneously all over the world. The global communications such as TV and Internet are making the biggest influence and creating the possibility that updates and news can be spread faster and more efficiently than ever. It is just a matter of minutes before a certain news becomes known worldwide. The information is delivered mostly via Internet. It is first published on the official website of the club or country, if it is official. If it isn't then there are sites which use their sources close to the club and leak some information which can prove to be untruthful after some time.

There are plenty of websites online, which can serve their visitors in many different ways, depending on what they want. They can be more general, or more specific, about plenty of sports or about one, about analyses or news, about a specific league, competition or about a certain club. This is the case that I have decided to make a website about. I am a FC Liverpool fan since I was a kid and it is a pleasure for me to read and watch different news about the team, that is what helped me decide about the topic of my diploma work.

The main purpose of the website is to inform and update its users about the recent news and also show them content that can be interesting to them such as videos, pictures, downloads, analyses of the game and etc. I have tried to create the website using the latest trends and tools and also to give the website a modern and pleasing to the end-user design.

I. Analysis of the Subject Field

The main purpose of that stage of the development is to gather enough information about the website creation and to assure the successful execution of the stages of its design and development. Here I'm going to explain in general what is required to understand in order to create your own website and then will go into details about my project and the methods and tools I used to create it. First of all and most important is to know that there are two types of web sites – static and dynamic. The static websites represent html pages that in order to change anything, you have to change the code. Nowadays though, the dynamic pages become more and more used, because they are way easier to create and to maintain. They use CMS – Control Management System, thanks to which, they can be made by people, that don't have an idea of programming. I am going to explain more about the static type of webpages first and will provide more information about it, because I have used that method to create my project and then will explain briefly about the dynamic sites and the CMS.

1. Static Web pages

Step 1: Plan Your Web Presence

Defining your Customers and Mission

You may think this goes against common sense, but the essence of your Web site isn't really about you. Your Web site is a specialized tool, one that enables

you to reach countless new customers and, if it's a retail site, sell to them and process their purchases. Here, your primary purpose is to know your customers so well that you answer any questions they might have before they ask, then make it easy for them to buy what you're selling. This bedrock principle applies whether you're creating a one- or two-page site that simply tells who you are and where you can be reached by e-mail, snail mail and phone; or a fully functioning retail site with hundreds, even thousands, of pages and a "shopping cart" that let's your buyers collect products and pay for them, comfortable that their financial and other personal data are secure. Exactly who are they and what do you know about them, what they want, what they need, what they don't know they need, what gives them the willies on the Web? • How old are they? Are they men, women, kids?

- What do they expect when they come to a company like yours?
- How smart are they and what specific talents or skills do they have?
- Where do they live? What are those places like?
- Are they Web savvy or are they just beginning to use it? In either case, what are their concerns about doing business on the Web – what scares them off?

Step 2: Select the Tools for Making Your Home on the Web

Web Hosting

Unless you own or plan to invest in a server – a powerful computer that's always online, and "big" enough to store all your Web site files, as well as the content and operations of your company's network – you need to find and hire a reliable Web host. Just like someone who accepts you into their home and tends to your needs, a Web host accepts your site into its computers, securely stores all of your files and data, and ensures that it will be available every day, around the clock, to you and your customers.

The host also handles most of your other technical needs, including up-to-date backups of your entire site; properly tuning the software; and giving you enough bandwidth to keep from slowing down your site's functions, and how fast the pages load. Because there are a whole lot of hosts, all trying to get your business, most keep their prices low (some are even free), for any size business and budget.

Hosts commonly offer other necessary Web site services, either with all-in-one discount packages, or individual low-cost add-ons. Just remember, you will be placing your entire Web site and all its functions in the host's safekeeping, so don't be tempted to use anything but a well-established outfit with a proven track record. There's plenty of comparative information, user reviews and other critical material online to provide this confidence.

FTP: File Transfer Protocol

So you don't get confused, understand that "FTP" is both a noun – referring to the software that transfers or "uploads" Web site files from your computer to your host's server – and a verb – referring to the actual transfer: "I'm going to FTP these files." That also pretty well takes care of explaining what it does. While hosts commonly include an FTP tool as part of their service, there's often a limit on the size of the uploads it can handle. No matter. Plenty of free downloadable software on the Web can easily transfer your files to your host. A few that we like:

- FTP Navigator
- FileZilla
- CoffeeCup Free FTP
- Core FTP Lite

With some hosting packages (such as Office Live Small Business), no FTP software is required. You can just find your document, image or file on your desktop and easily upload it.

Managing your Web Images

Unless you plan to hire a designer to take care of all the photos and other graphics on your Web site, you'll need a tool to do it yourself. Basic digital photo and graphics editors are available for free whereas sophisticated top end programs like Adobe Photoshop or Microsoft Digital Image Suite, pretty well recognized as the professional gold standard, can cost into the hundreds of dollars. What you're looking for is editing software that can resize and crop images; repair problems with color and contrast; set their resolution, which controls how sharp your graphics are on the Web page; and save them using color modes and formats specifically for the Web. The photo organizer built into your operating system, like Windows Photo Gallery packed with Microsoft's new

Vista, might even take care of your needs. Unless you really want to get into graphics editing and creating your own unique images and photo illustrations, you don't need to understand the technical ins and outs. But you should be sure that your choice of software supports all standard graphics formats for the Web, mainly JPEG (jpg), GIF, Bitmap (bmp and others) and Ping (png). Here's a few good choices at a range of price points:

- Microsoft Digital Image Suite
- CoffeeCup Flash Photo Gallery
- Adobe Photoshop Elements
- Corel Paint Shop Pro
- Quick Web Photo Resizer
- PhotoPlus

Step 3: Make Key Design Decisions

This is where the hard work you did in Step 1 comes into play. Having a clear definition of your target customer will help guide many of your decisions when the specific work of designing your new business Web site begins. Doesn't everybody want basically the same things from a Web site? Well, yes and no. Any visitor wants to know quickly what your site is about, what you have to offer that's of value to them, a well-designed system to move them through its pages and freedom from sensory assault by unexpected, unwelcome noisy and flashy graphics which can slow page load times. Remember always: Your Web site is there to serve your customers and their needs. If you're turned off by endless popups, grating audio and graphics that look like they've been lifted from the Vegas strip, then you shouldn't expect your Web site visitors to react any differently.

General Design Principles

Don't be a showoff. That's another way of saying what we've stressed before, and will again: When it comes to Web design, as in so many other things, simple is better. Of course you want photos and other graphic images to tell your company's story in the best way. And without some eye candy, any Web page

is blah. But use only what's needed to enhance your central message and tell it quickly and clearly in an attractive setting. Never make your customers work to get the information they need. As you move ahead in building your site, stick to these basic design rules:

- Keep it clean. Empty white space on your Web pages is itself a design element. Use enough to keep each page uncluttered and uncramped. Do the same if you decide to use a dark background.
- In the dark. Never use dark text on dark backgrounds, or for that matter, light colored text on a white background. Black-on-white is a safe bet.
- Gray blocks. Because you're already keeping it simple, make your text as concise and straightforward as possible. Don't waste words – they waste your customers' time. And break up long paragraphs. What the eyes see in a split second – about all it takes for a Web user to split from your site – is a big, challenging block of gray text. Give it some air.
- Choose colors carefully. You wouldn't wear red plaid pants with an orange striped shirt (we hope!), and you should use the same design sense in picking the color palette for your Web site. There are even free tools to help. (See page 25.)
- Use successful models. The things you like or hate about other Web sites are probably the same for most other users. Take notes on what works and what you'd like to imitate. Better yet, save a screenshot in your design file. It's easy:
 - With your cursor anywhere on the Web page you've chosen, hold down the Alt key and press the Print Screen key.
 - Nothing happened? Don't worry, you just couldn't see it.
 - Now open a blank document page in your word processor or Microsoft Paint, right click anywhere on it and choose Paste. An exact duplicate of the Web page you selected will appear!

Getting Around on Your Web Site

Easy navigation through your site is absolutely essential to a successful design. If the path you lay out for your customers to follow is long, twisted and forks off without reason, they'll get lost – and you'll lose the sale. As part of planning in Step 1, we asked you to draw a simple diagram of all the pages on your future Web site, beginning with the home page, then connect them in the order you expect customers to follow. Did it get messy? Too complicated? That's your draft. Now you'll refine it. Try the same exercise by starting with the last page on your site diagram and working back to the home page. A lot of designers find

that much easier. Now, is every page linked directly to the home page like spokes on a wheel? That can work, but it requires your customers to go back to the home page every time they want find more information, more page links. Do you have patience with that kind of back and-forth?

Step 4: Learn the Code

What is Hypertext Markup Language?

Yes, it's a new language to learn. But HTML has been the basic framework of all Web design for as long as it's existed, largely because it's easy to understand. It's just words. Plain text, common words mixed with some special but simple "punctuation" marks. You may be surprised to learn that every Web page, no matter how many slick tricks and graphics it has, is built on nothing but text. It's like that old wizard behind the curtain: You don't see him – unless you know where to look. Go to a Web page you like and right-click your mouse on an empty space. When a menu appears, look for "View Source" or "View Page Source" and left-click it. A new screen appears, filled with plain English text and familiar punctuation marks – but arranged in a different way. (If it's one long unbroken block of gobbledygook, pick another page. Whoever wrote the code didn't bother to break the text into lines and sections for easy reading.) This is HTML and it controls everything on that page – every sentence, every graphic, every link and form, every sound, all of it. Your Web browser reads this text and translates it into the visual, functional Web page. It's as user-friendly as code gets, and you don't need anything more than a word processor or simple text editor – like Notepad – to write or manipulate it. And it works on any kind of computer with any operating system.

Understanding HTML Tools

As mentioned earlier, you really don't need any special software or programs to work with HTML. Plenty of Web designers use nothing more than Microsoft Word to create HTML content.

Let's decipher one more techie acronym here in case you run across it: ASCII – say "ask-ee" – stands for American Standard Code for Information Interchange, the most common standard for handling text on computers. ASCII documents are basically text files, easily viewed and managed. Because HTML works with any operating system – Windows, Mac, Linux – saving your HTML files in ASCII text

format is the easiest and most effective way to go. In Microsoft Word, just choose "Simple Text," "Text" or "Text Only" when it's time to close and save your file. Text editors are simpler than word processing programs, but cover your same needs for writing HTML. On PCs running the Windows operating system (or OS), you'll find Notepad or WordPad built into all but the oldest versions; on Macs, it's SimpleText. There's a big advantage, however, to getting an inexpensive program like the CoffeeCup HTML Editor, because it lets you easily switch between a text screen and a visual editor so you can see how your HTML looks on a Web page.

WYSIWYG vs. HTML Software

The two most common types of design software are WYSIWYG and HTML, which is used to build a Web site with Hypertext Markup Language. Better software combines both, automatically converting your visual design to HTML. WYSIWYG (say "wiziwig") makes Webbuilding a lot easier for the ones new to the whole thing. It's an acronym for What You See is What You Get –you watch your site come together on the screen while dragging and dropping its pieces into place. But if you're building anything more than a basic Web site with limited functions, HTML is the way to go. The code isn't hard to learn, if you have the time, and gives you endless flexibility and options, and better control over every element of your new site and how it looks online. A blend of both is best and usually offered in higher end – more expensive – design software.

Five Best WYSIWYG HTML Editors

You can make a strong argument for hand-coding HTML, but the appeal of a What You See Is What You Get editor for beginners is undeniable. Here's a look at five of the most popular WYSIWYG HTML editing tools.

[Kompozer](#) (Windows/Mac/Linux, Free)

Kompozer has a lot going for it, foremost of which is the free-as-in-beer price tag. Kompozer sports tabbed editing—WYSIWYG in one tab, raw HTML in the other—on-the-fly editing via the built-in FTP site manager, and a highly customizable interface with easily modified toolbars. Kompozer has a markup cleaner and a W3C call function to validate your HTML against current

standards. It's free, available on Windows, Mac, and Linux machines, and it has a strong focus on standards compliance and clean code.

[iWeb](#) (Mac, \$99 for [iLife bundle](#))

The "It just works!" design philosophy that permeates Apple offerings is strong with iWeb—the WYSIWYG HTML editor bundled with iLife—and interacting with it is so drag-and-drop and user friendly that even your friends least likely to learn HTML could whip together a functioning web site. Apple provides a number of polished templates and dozens of web site widgets that are all a mouse click away. iWeb's built-in site manager makes it easy to publish to multiple sites or just keep a close eye on your ever-expanding digital manifesto.

[Adobe Dreamweaver](#) (Windows/Mac, \$300)

Dreamweaver is a titan in the WYSIWYG world. Now part of the Adobe portfolio but originally launched by Macromedia, Dreamweaver has offered WYSIWYG editing since 1997 when the web was a maze of tiled backgrounds, electric blue links and blinking GIFs. Dreamweaver offers hybrid editing, you can work completely in WYSIWYG mode without ever seeing a bit of code, you can work directly in the code only switching over to preview your work, or you can work in a dual-pane environment to take advantage of WYSIWYG and hand-coding simultaneously. Dreamweaver is extensible with dozens of free and commercial plugs-ins available for everything from web effects and widgets to shopping carts and image galleries.

[Microsoft Expression Web](#) (Windows, \$125)

Expression Web is Microsoft's current offering in the WYSIWYG arena (the popular but much maligned FrontPage was retired in 2003). For those of you who associate Microsoft with poor web standards compliance, take comfort knowing that Expression Web has a totally separate engine from Internet Explorer and is compliant with a wide range of current web standards. It shares a lot of features with the other WYSIWYG editors featured here, like highlighting

code errors and non-compliant code, a built-in CSS editor, and more, it also stands out for features like search engine optimization—offering you tips and ideas to optimize your sites for better crawling and search engine ranking.

[Flux](#) (Mac, \$75)

Flux is a Mac-based WYSIWYG editor that has received high praise for being a powerful editor with a reasonable price tag. Flux's interface offers a fine degree of control over editing everything from the margins and padding to over all size of your elements including altering CSS code with simple mouse movements. Flux offers dual-pane editing so you can switch between hand-editing and drag-and-drop editing instantly or just watch the HTML code unfold as you WYSIWYG edit to study what's going on under the hood. Like Dreamweaver, Flux supports third-party plug-ins which are available for download through the Flux application.

Step 5: Optimize Your Site for Search Engines

What is SEO?

Search Engine Optimization is the process of making your Web site as easy to find as possible for search engines and, through them, your clients and customers. For that to happen, your Web pages have to contain the keywords and phrases most likely to be used when a customer enters search requests in an engine, and your pages must be organized in way that's most "friendly" to those high-tech seek-and-find services.

There are two dominant types of search engines:

- Crawler-Based - Google, Yahoo, MSN, Live.com and other top search engines operate automatically, coming up with their rankings by sending "spiders" out to "crawl" Web sites, analyze their contents and rank them according to how likely they are to have what users want.
- Human-Powered Directories - These depend on Web site owners or someone working on their behalf to manually enter their listings, or enough information for directory editors to look over the site and write their own reviews. If you don't submit your site to these directories, it won't show up when they're searched.

How SEO Works

Most crawler-based search engines have three key ingredients:

- The spider or crawler that seeks out a Web page, reads it and follows links to other pages on the site. These powerful little tech-mites return every few weeks to look for changes and adjust results.
- An indexed archive where all content uncovered by the spiders is stored. Also known as the catalog, it contains a copy of every Web page found, and is updated as a Web site changes or grows – for example, when you add new products.
- Software that zips through the index database to “match” search requests and rank them by relevance. Because most crawler-based search engines use their own technology, search results vary between them.

How Search Engines Rank Web Sites

Unlike a human archivist or librarian, Internet search engines don’t interact with users and ask for more details, or use judgment and past experience to rank Web pages. Instead, they rely on mathematic formulas called “algorithms.” Despite what you may hear, nobody but the search engine owner knows exactly how their algorithm works.

But they do follow a universal practice known as the keyword location/frequency method. Search engines look over your Web site to see if the search keywords show up at the top your pages, in the headlines or the first few lines of text content. They assume that any page relevant to a given search topic will mention those magic words right from the start. “Frequency” is how often keywords appear in relation to other words on a Web page. Those with higher frequency are given more relevance, and higher rankings.

SEO Best Practices Boosting your Web site’s visibility is a very competitive game, and you should assume that rival sites are playing it. No worries. Here are some well-proven ways to optimize your Web site’s visibility:

Get the most from your URL: Be specific and creative with your domain name. Use one that uniquely identifies your company and your brand.

Create search-friendly page titles: Be sure to use relevant keywords first in your page titles, and keep them under 60 characters.

Don't use "home page" in your title: Studies show it decreases your Google ranking.

Highlight your keywords: Be sure the individual words and phrases are in the meta tag description of your site, which you build into the code with your design or site-builder software. The description should be no more than 200 characters long.

Focus on density: Use multiple key words in a coherent, creative and compelling way on your pages. Be sure your keyword "density" is never more than 5 percent for pages with a lot of text, or 10 percent for pages with little copy, or your rankings could nosedive. SEOChat.com has a free keyword density tool.

Emphasize your text links: The wording of the links on each of your Web pages is one of the most important requirements of SEO, and will significantly affect your search engine ranking. They should always include relevant keywords.

Keyword stemming: Including all the possible variations of your keywords is called "stemming." For example, variations of the keyword "optimization" include "optimal," "optimize" and "optimum." UsingEnglish.com has a free tool to find similar or stem words.

Page linking: Be sure every page on your site is linked to the other pages. Search spiders follow these trails to rank your Web site.

Avoid "spamalot" syndrome: Search engines will drop your site if they think you're "spamming" – and using any of these things:

- Meta refresh tags
- Invisible text
- Irrelevant keywords in the title and meta tags
- Excessive repetition of keywords
- Identical or nearly identical pages
- Submitting to an inappropriate directory category
- A dizzying slew of links that are of low value

Step 6: Testing Your Site

Before you start telegraphing to the world that you're online, be sure to test for the following:

- Have you checked for spelling and grammar mistakes? Even the slickest, most professional site will look like amateur hour if you've ignored basic content editing. And don't trust your spellchecker. It doesn't know the difference between "their" and "there" and a lot more.
- Are your pages and navigation consistent across the site?
- Do all of the links work, taking users where they expect to go?

Testing on Different Platforms and Browser

Just because your Web site looks great and runs smoothly on your Windows Vista machine in Internet Explorer doesn't mean it will look the same on Firefox, a Mac or other browsers or operating systems. So:

- Be sure to test your site across all browsers, and keep up on new releases or upgrades after you go live.
- Most of your customers will be using Windows XP, Vista or Macintosh OS X, so it's essential that your site works on all platforms, or operating systems.
- Take a look at your Web site on a variety of monitors. Color settings and screen resolutions vary, and an LCD monitor's quality is much higher than a standard CRT monitor's. You can't do much about it, but if you see drastic differences between monitors, fine tune a little to cut the gap.

Step 7: Go Online

This is of course the final step from the cycle. After you go online comes the even harder part - maintenance and constant updates to keep it organized and working.

2. Dynamic Web Pages

Definition - What does *Content Management System (CMS)* mean?

A content management system provides a simple, accessible website interface that can be used to add content to a page in a highly structured manner. The overall approach of a CMS is to allow for the generation of standards-compliant content.

Generally, access to a CMS is defined for a particular set of users who can view, add, edit and publish the content within the CMS. This reduces duplicate work by allowing privileged users to view the work status of content that has already been worked on by other users in the group.

The CMS allows users to create, edit and publish content from anywhere and at any time. Because content is added to the CMS server, the operational aspects of the CMS are not installed on users' personal computers.

Content management systems are categorized into four different types: enterprise content management systems, Web content management systems, Web group content management systems and component content management systems.

The 21st century brought with itself a revolution in the field of website designing and development in the form of Content Management Systems (CMS) — the software packages which allow you to organize, publish, modify and maintain content on a website, without requiring much of technical expertise. All you need to do is install a CMS, choose a domain name, give your inputs in the template design and hop onto the web server with your new online identity.

15 years into the new millennium, we are surrounded with hundreds of CMSs to choose from. And the cherry on the top is that most of them are open source platforms, implying that they can be used for free. Although owing to the pros and cons of all, no CMS can be termed as the best but some are more popular than the rest. Here is a list of the 5 most widely used ones:

1. WordPress

Ask any CMS Website Development Company for an open source platform, and the first suggestion would be WordPress. Built on PHP and MySQL, this CMS powers around 15% of the world's top one million websites and 24% of the total

websites. Be it hosting, installation or administration, WordPress makes everything hassle-free. Sites developed on this open source platform are W3C compliant, search engine optimized and have excellent Web 2.0 functionality. Its huge universe of themes and plugins allows you to create virtually any type of website.

2. Joomla

This open source CMS enjoys high levels of popularity, particularly amongst the startups and SMEs owing to its flexibility and easy-to-use admin controls. However, many bigshot names like MTV and Harvard University are also running their websites on Joomla. This polyglot platform has created millions of websites in 64 different languages. Also, if your website is on Joomla, its promotional capabilities can enhance your viewership to a great extent.

3. Drupal

Another free, PHP-based Content Management System, Drupal is being used by prominent names like New York Observer, Popular Science and MIT amongst several others. Similar to WordPress and Joomla, it also possesses a treasure of thousands of extensions to increase your site's functional range. If your website has a lot of complex content to be organized, Drupal is best suited for you with its unique features for taxonomy and ability to tag.

4. ModX

Although the name of ModX is not much heard of in common parlance but there are over 100,000 websites running on this CMS. Its core strength lies in its usability allowing even non-technical handlers to conveniently work with it. A logically laid out interface makes it easy to customize the designs. To top it all, no additional plugins are required for SEO in ModX sites.

5. ExpressionEngine

Developed by EllisLab, this CMS is based on PHP and offers both open source and premium software for creating websites. Although, its number of plugins is nowhere even close to what popular CMSs are offering, this platform does provide complete liberty to design the sites and search engine friendly URLs. Official support is available on purchase of license.

Conclusion

In this chapter, I tried the basics of web site development and the different types of techniques used to create webpages. Now when we know the basics, I'll move on with explaining of how I created my website and what techniques and programming languages I have used.

II. Used Technologies, Methodologies and Platforms

1.Types of programming languages, platforms and comparison analysis

Scripting Languages

A scripting or script language is a programming language that supports scripts, programs written for a special run-time environment that automate the execution of tasks that could alternatively be executed one-by-one by a human operator. Scripting languages are often interpreted (rather than compiled). Primitives are usually the elementary tasks or API calls, and the language allows them to be combined into more complex programs. Environments that can be automated through scripting include software applications, web pages within a web browser, the shells of operating systems (OS), embedded systems, as well as numerous games.

PHP

Stands for "Hypertext Preprocessor." (It is a recursive acronym, if you can understand what that means.) PHP is an HTML-embedded Web scripting language. This means PHP code can be inserted into the HTML of a Web page. When a PHP page is accessed, the PHP code is read or "parsed" by the server the page resides on. The output from the PHP functions on the page are typically returned as HTML code, which can be read by the browser. Because the PHP code is transformed into HTML before the page is loaded, users cannot

view the PHP code on a page. This make PHP pages secure enough to access databases and other secure information.

A lot of the syntax of PHP is borrowed from other languages such as C, Java and Perl. However, PHP has a number of unique features and specific functions as well. The goal of the language is to allow Web developers to write dynamically generated pages quickly and easily. PHP is also great for creating database-driven Web sites. If you would like to learn more about PHP, the official site is PHP.net.

HTML

First developed by Tim Berners-Lee in 1990, HTML is short for HyperText Markup Language. HTML is used to create electronic documents (called pages) that are displayed on the World Wide Web. Each page contains a series of connections to other pages called hyperlinks. Every web page you see on the Internet is written using one version of HTML code or another.

HTML code ensures the proper formatting of text and images so that your Internet browser may display them as they are intended to look. Without HTML, a browser would not know how to display text as elements or load images or other elements. HTML also provides a basic structure of the page, upon which Cascading Style Sheets are overlaid to change its appearance. One could think of HTML as the bones (structure) of a web page, and CSS as its skin (appearance).

HTML5 is the update made to HTML from HTML4 (XHTML follows a different version numbering scheme). It uses the same basic rules as HTML4, but adds some new tags and attributes which allow for better semantics and for dynamic elements that are activated using JavaScript. New elements include section, <article>, <aside>, <audio>, <bdi>, <canvas>, <datalist>, <details>, <embed>, <figure>, <figcaption>, <footer>, <header>, <keygen>, <mark>, <meter>, <nav>, <output>, <progress>, <rp>, <rt>, <ruby>, <time>, <track>, <video>, and <wbr>. There are also new input types for forms, which include tel, search, url, email, datetime, date, month, week, time, datetime-local, number, range, and color.

With the increasing movement to keep structure and style separate, a number of styling elements have been removed along with those that had accessibility issues or saw very little use. These following elements should no longer be used in

HTML code: <acronym>, <applet>, <basefont>, <big>, <center>, <dir>, , <frame>, <frameset>, <noframes>, <strike>, and <tt>.

CSS

Most web pages are made from **HTML**, or hypertext markup language. This is the standard way to decorate plain web text with fonts, colors, graphic doodads, and hyperlinks (clickable text that magically transports the user somewhere else). But websites can get really big. When that happens, HTML is a very hard way to do a very easy thing. **CSS** (cascading style sheets) can make decorating web sites easy again!

Think of CSS as a kind of computer dress code. **CSS** mainly does just one thing: it describes how web pages should look. Even better, CSS can be easily separated from HTML, so that the dress code is easy to find, easy to modify, and can rapidly change the entire look of your web site. Like a dress code at school, you can change your CSS and the look of your students will change with it. Style sheets allow you to rapidly alter entire websites as you please, just like a fashion craze allows people to change with the times yet remain the same people.

A really neat thing about CSS, is that it **cascades**. Each style you define adds to the overall theme, yet you can make the most recent style override earlier styles. For example, with CSS we can start by saying we want all of our text 12px (12 units) high. Later we can say we want it to be red, too. Still later, we can tell it we want one phrase to be in bold or italics, or blue rather than red.

Three Types of CSS

CSS comes in three types:

- In a separate file (**external**)
- At the top of a web page document (**internal**)
- Right next to the text it decorates (**inline**)

External style sheets are separate files full of CSS instructions (with the file extension .css). When any web page includes an external stylesheet, its look and feel will be controlled by this CSS file (unless you decide to override a style using one of these next two types). This is how you change a whole website at once. And that's perfect if you want to keep up with the latest fashion in web pages without rewriting every page!

Internal styles are placed at the top of each web page document, before any of the content is listed. This is the next best thing to external, because they're easy to find, yet allow you to 'override' an external style sheet -- for that special page that wants to be a nonconformist!

Inline styles are placed right where you need them, next to the text or graphic you wish to decorate. You can insert inline styles anywhere in the middle of your HTML code, giving you real freedom to specify each web page element. On the other hand, this can make maintaining web pages a real chore!

CSS Instructions

Before we introduce CSS, let's briefly review HTML. A simple web page is made of **tags**. Everything must go between the opening and closing `<html>` tags. The `<head>` section contains invisible directions called **meta information**. The `<body>` section is where we put all the visible stuff. Here's a super simple HTML example:

External CSS

Now here is a short, simple example of CSS using an external file (we'll call it 'stylish.css'). We're going to tell our web page to be white and to make our h1 heading, noted in our simple HTML example as 'John Adams', appear in red at 24 units high:

In the sample file, the top line is a comment and doesn't do anything. The next part (called body) tells the web page what background color to use for the **body** section. Right after that, the h1 part says we want our largest heading (h1) to be the color red and its font size to be 24 units high.

```
/* CSS Document */

body {
  background-color: white;
}

h1 {
  color: red;
  font-size: 24px;
}
```

```
<html>
<head>
  <!-- An example of including an external CSS stylesheet -->
  <link rel="stylesheet" type="text/css" href="stylish.css">
</head>
<body>
  <!-- Large headings are labeled h1 -->
  <h1>John Adams</h1>
</body>
</html>
```

How to include an external CSS file

```
<html> <!-- Opening tag -->

<head>
  <!-- The head section contains hidden meta information -->
</head>

<body>
  <!-- The body section contains our web page content -->

  <!-- Large headings are labeled h1 -->
  <h1>John Adams</h1>
</body>

</html> <!-- Closing tag -->
```

HTML code for a typical HTML page

John Adams

What the page would look like

What the page would look like

Internal CSS

We don't need to include an external CSS file just to decorate one web page. We can just define our styles at the top of the page, in the <head> section. Here's a quick example in which we're making our heading (h1) blue at a font size of 28 px:



JavaScript

JavaScript is a programming language commonly used in web development. It was originally developed by Netscape as a means to add dynamic and interactive elements to websites. While JavaScript is influenced by Java, the syntax is more similar to C and is based on ECMAScript, a scripting language developed by Sun Microsystems.

JavaScript is a client-side scripting language, which means the source code is processed by the client's web browser rather than on the web server. This means JavaScript functions can run after a webpage has loaded without communicating with the server. For example, a JavaScript function may check a web form before it is submitted to make sure all the required fields have been filled out. The JavaScript code can produce an error message before any information is actually transmitted to the server.

Like server-side scripting languages, such as PHP and ASP, JavaScript code can be inserted anywhere within the HTML of a webpage. However, only the output of server-side code is displayed in the HTML, while JavaScript code remains fully visible in the source of the webpage. It can also be referenced in a separate .JS file, which may also be viewed in a browser.

Below is an example of a basic JavaScript function that adds two numbers. The function is called with the parameters 7 and 11. If the code below were included in the HTML of a webpage, it would display the text "18" in an alert box.

```
<script>
function sum(a,b)
{
    return a + b;
}
var total = sum(7,11);
alert(total);
</script>
```

JavaScript functions can be called within <script> tags or when specific events take place. Examples include onClick, onMouseDown, onMouseUp, onKeyDown, onKeyUp, onFocus, onBlur, onSubmit, and many others. While standard JavaScript is still used for performing basic client-side functions, many web developers now prefer to use JavaScript libraries like jQuery to add more advanced dynamic elements to websites.

Ajax

Ajax is not a programming language or a tool, but a concept. Ajax is a client-side script that communicates to and from a server/database without the need for a postback or a complete page refresh. The best definition I've read for Ajax is "the method of exchanging data with a server, and updating parts of a web page – without reloading the entire page." Ajax itself is mostly a generic term for various JavaScript techniques used to connect to a web server dynamically without necessarily loading multiple pages. In a more narrowly-defined sense, it refers to the use of XMLHttpRequest objects to interact with a web server dynamically via JavaScript.

Databases

MySQL

MySQL was a free-software database engine originally developed and first released in 1995. MySQL is named after My, the daughter Michael Widenius, of one of the product's originators. It was originally produced under the GNU General Public License, in which source code is made freely available.

MySQL, pronounced either "My S-Q-L" or "My Sequel," is an open source relational database management system. It is based on the structure query language (SQL), which is used for adding, removing, and modifying information in the database. Standard SQL commands, such as ADD, DROP, INSERT, and UPDATE can be used with MySQL.

MySQL can be used for a variety of applications, but is most commonly found on Web servers. A website that uses MySQL may include Web pages that access information from a database. These pages are often referred to as "dynamic," meaning the content of each page is generated from a database as the page loads. Websites that use dynamic Web pages are often referred to as database-driven websites.

Many database-driven websites that use MySQL also use a Web scripting language like PHP to access information from the database. MySQL commands can be incorporated into the PHP code, allowing part or all of a Web page to be generated from database information. Because both MySQL and PHP are both open source (meaning they are free to download and use), the PHP/MySQL combination has become a popular choice for database-driven websites.

Microsoft SQL Server

SQL Server is Microsoft's relational database management system (RDBMS). It is a full-featured database primarily designed to compete against competitors Oracle Database (DB) and MySQL.

Like all major RDBMS, SQL Server supports ANSI SQL, the standard SQL language. However, SQL Server also contains T-SQL, its own SQL implementation. SQL Server Management Studio (SSMS) (previously known as Enterprise Manager) is SQL Server's main interface tool, and it supports 32-bit and 64-bit environments.

SQL Server is sometimes referred to as MSSQL and Microsoft SQL Server.

Oracle Database

Oracle DB rivals Microsoft's SQL Server in the enterprise database market. There are other database offerings, but most of these command a tiny market share compared to Oracle DB and SQL Server. Fortunately, the structures of Oracle DB and SQL Server are quite similar, which is a benefit when learning database administration.

Oracle DB runs on most major platforms, including Windows, UNIX, Linux and Mac OS. Different software versions are available, based on requirements and budget. Oracle DB editions are hierarchically broken down as follows:

- Enterprise Edition: Offers all features, including superior performance and security, and is the most robust
- Standard Edition: Contains base functionality for users that do not require Enterprise Edition's robust package
- Express Edition (XE): The lightweight, free and limited Windows and Linux edition
- Oracle Lite: For mobile devices

A key feature of Oracle is that its architecture is split between the logical and the physical. This structure means that for large-scale distributed computing, also known as grid computing, the data location is irrelevant and transparent to the user, allowing for a more modular physical structure that can be added to and altered without affecting the activity of the database, its data or users. The sharing of resources in this way allows for very flexible data networks whose capacity can be adjusted up or down to suit demand, without degradation of service. It also allows for a robust system to be devised as there is no single point at which a failure can bring down the database, as the networked schema of the storage resources means that any failure would be local only.

Web Servers

Apache

Apache is the most popular Web server software. It enables a computer to host one or more websites that can be accessed over the Internet using a Web

browser. The first version of Apache was released in 1995 by the Apache Group. In 1999, the Apache Group became the Apache Software Foundation, a non-profit organization that currently maintains the development of the Apache Web server software.

Apache's popularity in the Web hosting market is largely because it is open source and free to use. Therefore, Web hosting companies can offer Apache-based Web hosting solutions at minimal costs. Other server software, such as Windows Server, requires a commercial license. Apache also supports multiple platforms, including Linux, Windows, and Macintosh operating systems. Since many Linux distributions are also open-source, the Linux/Apache combination has become the most popular Web hosting configuration.

Apache can host static websites, as well as dynamic websites that use server-side scripting languages, such as PHP, Python, or Perl. Support for these and other languages is implemented through modules, or installation packages that are added to the standard Apache installation. Apache also supports other modules, which offer advanced security options, file management tools, and other features. Most Apache installations include a URL rewriting module called "mod_rewrite," which has become a common way for webmasters to create custom URLs.

While the Apache Web server software is commonly referred to as just "Apache," it is technically called "Apache HTTP Server," since the software serves webpages over the HTTP protocol. When Apache is running, its process name is "httpd," which is short for "HTTP daemon."

lighttpd (pronounced "lighty") is an open-source web server optimized for speed-critical environments while remaining standards-compliant, secure and flexible. It was originally written by Jan Kneschke as a proof-of-concept of the c10k problem – how to handle 10,000 connections in parallel on one server, but has gained worldwide popularity. Its name is a portmanteau of "light" and "httpd."

Nginx

Nginx is an open source HTTP Web server and reverse proxy server. Pronounced as "Engine-Ex," Nginx has emerged as the third most popular Web server behind the Apache Web server and Microsoft's IIS, and it currently powers popular websites like Pinterest, WordPress.com, Netflix, Hulu, CloudFlare, Zappos and Zynga.

In addition to offering HTTP server capabilities, Nginx can also operate as an IMAP/POP3 mail proxy server as well as function as a load balancer and HTTP cache server. Nginx can run on Linux, Mac OS X, Solaris, AIX, HP-UX and BSD variants.

2.Choice of Languages, Server and DB Server

The choice for Database is:

MySQL with phpmyadmin management tool. I have selected MySQL because it is:

- Relational DB with open source
- Supports API for large amount of programming languages (incl. PHP)
- Offers fast and reliable working environment
- Easy administration

And phpmyadmin is the tool that is used to implement the database into the website via PHP. PHPMyAdmin is open source free software, designed to handle the administration and management of MySQL databases through a graphic user interface. Written in PHP, PHPMyAdmin has become one of the most popular web-based MySQL management tools. PHPMyAdmin comes with detailed documentation and is being supported by a large multi-language community. PHPMyAdmin's ever growing list of features supports all commonly used operations such as browsing, dropping, creating, altering MySQL databases, tables, fields and indexes. Also, PHPMyAdmin enables you to manage MySQL users and user privileges. Another commonly used PHPMyAdmin feature is its import function. With PHPMyAdmin, MySQL database import from backup is made easy and you can import an SQL or CSV dump with a few mouse clicks. Also, you can easily export your database in CSV, SQL, XML, Excel and other popular formats.

The choice for scripting languages is:

for formatting languages: HTML and CSS (to design the webpage) and for scripting(programming) language I've chosen PHP. Here is why:

❖ HTML and CSS

- Free
- Most widely used
- Supported by all browsers
- Don't require additional software
- Easy syntax

For HTML, PHP and CSS editor I've chosen Microsoft Expression Web 4, because it is easy to work with and I've had some experience with it earlier during my study.

❖ PHP

- Free with open source
- Offers wide range of possibilities – work with sessions, variables, objects, files, databases, libraries and etc.
- Highly productive and flexible
- Platform independent
- Big and active society, which can help anytime

For the Forum Page of my website I've chosen to use phpBB3. phpBB is one of the most popular and widely used open source bulletin board systems currently available. Forums are powered by [phpBB](#) that cater to virtually any sized audience from a handful of users per month to thousands per day.

Sites powered by phpBB are frequently used to provide support, facilitate community interaction, and are even useful for small, collaborative teams searching for a platform where multiple discussions can be easily searched and read.

phpBB is released under the [GNU Public License](#) and can be modified and expanded upon. An abundance of add-ons and other third-party modifications are available for free through multiple phpBB user groups and resources. phpBB has a large and active user community, with many contributors producing third-party tools to extend phpBB's functionality well beyond what's available with the core product.

Features of phpBB

- Written in PHP
- Advanced Administrative Control Panel
- Extensive Moderation Tools
- Search Engine Spider Handling
- Unread Message Tracking
- Private Messaging System
- Fully Customizable User Registration System
- Attachment Support
- Support for Forum Categories, Rules, Password Protection, URL Redirection, Forum-Specific Styling, etc.
- Advanced Spam Filtering and Blocking Built in
- Supports Multiple Database Storage Systems (MySQL, Microsoft SQL, PostgreSQL, Oracle Database, Firebird, SQLite, etc.)
- Advanced Caching Built in
- Powerful Integrated Search
- Plugins and Custom Styles Supported
- Comprehensive Notifications System

The choice for a server is Apache because it is:

- Free to use
- Platform independent
- Fast, reliable and secure
- Easy to configure
- The most popular web server

Conclusion:

In this chapter I've explained in detail about some options that can be used in order to create a working website with a database connected to it, I've stated my choices and gave arguments why I've chosen exactly them, and also given some information about the software I have used in the process.

III. Website design

Conceptual Analysis

The purpose of the website that I'm creating is to provide information about the football club in the easiest way possible. The system is user-friendly, which means it is easy to be used by any website visitor, without needing to have any special computer skills. The main purpose is providing news and information relevant to Liverpool FC via the appropriate presentation and this information to be sufficient for the user without having to visit other websites or having to access other sources of information. The target-group is made out of football fans, who show interest in any way about Liverpool Football Club, who can visit only this website on the web if they need information and updates about the club. The website has simplified and logical structure which also supports its easy usage.

Functional Requirements

In order to be user-friendly as mentioned above the first and most important step is to make the design simple and clear. The user has to know where everything is located right from the home page and the navigation has to be formatted well. This helps for the longer stay of the person using the website and is the main factor for additional and regular website usage. The next step is keeping the website up-to date – no one wants to read the news from the past week, everybody wants to be updated ASAP and in order to realize that, there has to be someone who is constantly updating the information and a template of the page has to be kept ready to use, so it can be filled with information right away when available. Another very important aspect is the communication between users and to give them the opportunity to express their feelings and thoughts about the club (and not only). In order to do that, a forum has to be created and it has to be maintained as well.

Conclusion

In this chapter I've explained briefly about my vision of the website that, I'm going to create and the basic principles that have to be met in order the site to be successful.

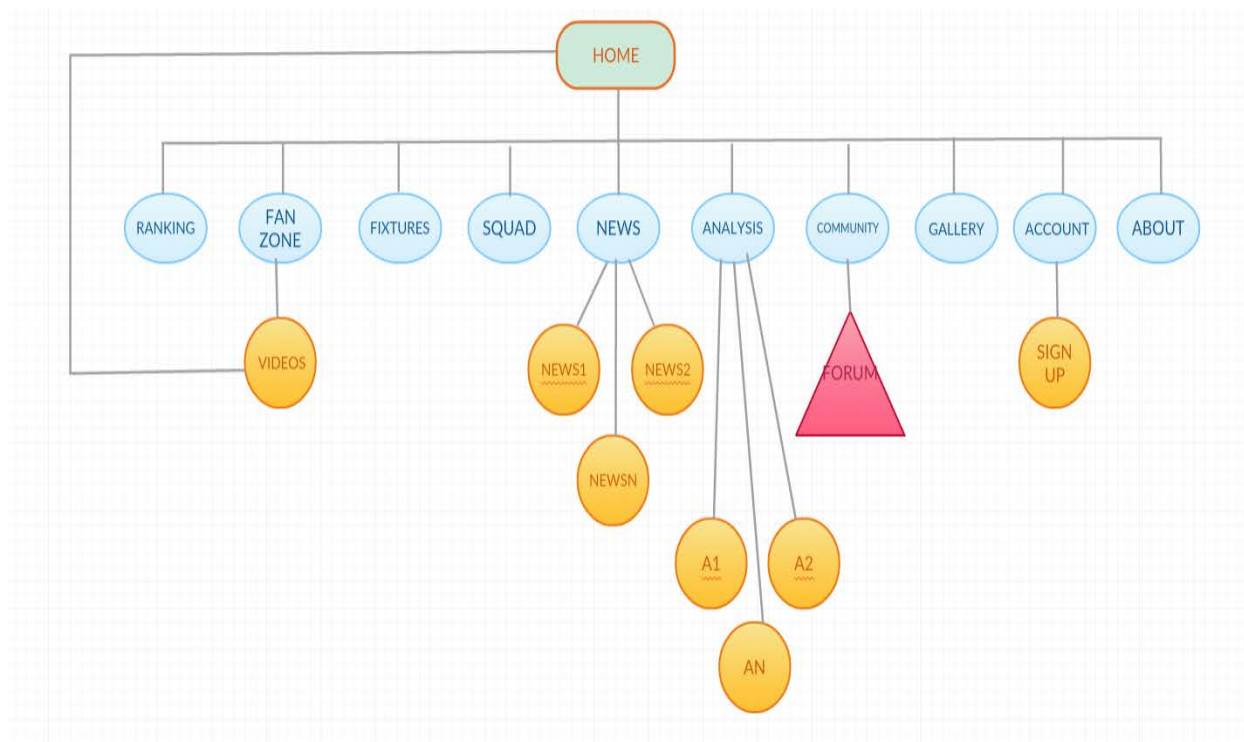
IV. Website Implementation – Program Code, Techniques and Methodologies used

In this chapter, I'm going to explain in detail about the creation and the further development of the website. I'm going to start off explaining step by step about the creation of every page, beginning with the site map, then moving to the design of the website, then creation of the database, then the forum and for each step I'll provide the used techniques, the code that I've written and pictures of the results as well.

Site Map

First of all, and most importantly, you have to have an idea of the content of the site that you want to be displayed. There are some things that are essential for the type of site that I'm going to create like news, fixtures, results, current rankings and etc. But this is never enough for the die-hard fan of a club, he always wants more information, more content, more discussions and media about his favorite club. So I've came up with few more suggestions and implemented them in my website. Here they are:

- Analysis page – everyone thinks they understand a lot about football, but this is not always the case. So I've implemented a special webpage to keep the visitors updated about the situation in the club from the viewpoint of the specialists
- Fan Zone page which is made up of downloads and video links to youtube.com
- Account page which logs the users in
- Community page which leads to a forum, so discussions can be held



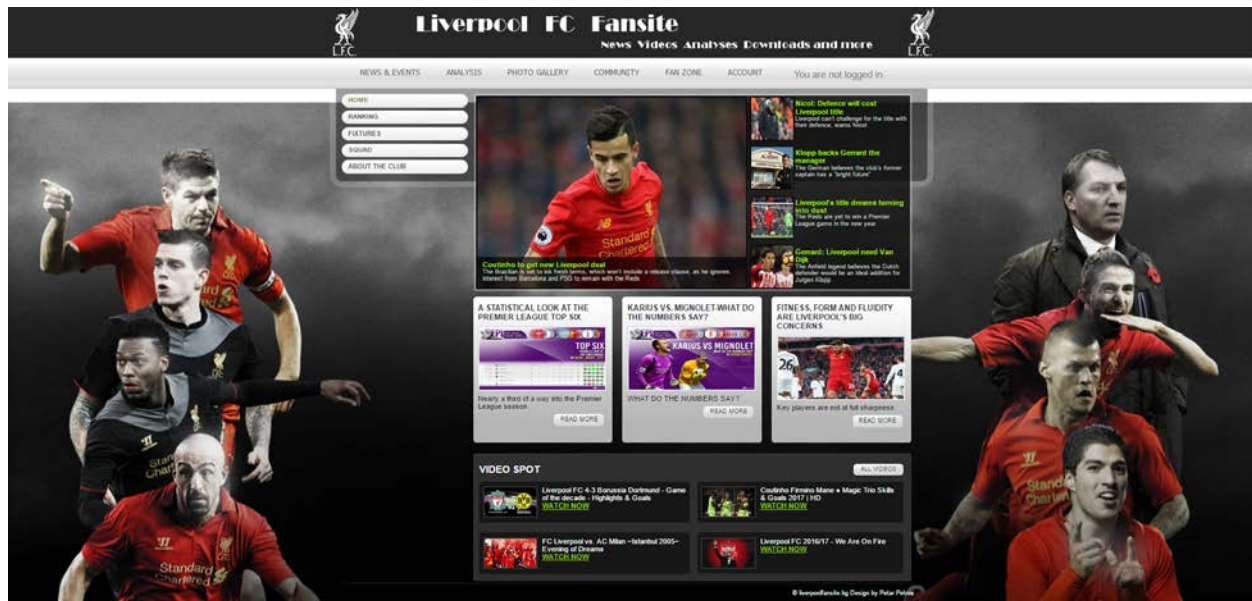
In the picture above I've created a model of my website (a map). The main idea is to have everything interconnected and easy to access from every page on the website. The home page (green button) + the blue colored ellipses are part of the so-called *navigation* part of the site. They can be reached from any page. The orange circles (news pages, analysis pages) are going to grow in time and it will become impossible to place them on the website so they can be reached from anywhere, that is why there is a page, which is designed to organize them and to give access to them easily. They can be accessed only from that page (except for the newest news and analysis which appear on the home page, and the videos page which has a special button implemented on the home page) but they can access every page from the navigation menu. The forum is the only webpage which is designed to lead to a destination outside of this website, but this is the better thing to do, because a forum is a separate unit in the whole concept and project, with its own database and design, which are made again in order to be easier and user-friendly. Basically this was the first step, which is more of a preparation than a development part. Here is the next thing to do:

Website Design – CSS code

The next step that I think has to be made is the creation of the design of the webpage. As mentioned earlier the design is made mostly using CSS, which serves as a skin to the bones (HTML). There are plenty of pre-made templates online, most of them are paid for, there are some free. I decided not to use a paid one, although they are really good, they don't need much improvement. So I chose to download a free one to have it as a foundation and to improve and redesign it with my pictures and styles. I have changed most of the stuff, actually I've changed so much that at one point I was angry and felt sorry I didn't chose to start from the beginning and make my own, but whatever. So here I will post the CSS code used and will include a sample screenshot of the homepage and a random page. The reason they are different is that the home page is made of many fields and various type of information has to be displayed and the other pages are specified and simpler. In order to edit the CSS code I used Microsoft Expression Web and in order to create, edit and adjust the images, I used ACDSee Photo Editor 6. I've designed the header image, the background image and the buttons by editing, cropping and combining different images, but this is not exactly relevant to the website, so I won't go any further in explanations.

CSS is using classes (.) and ID's (#), which can later be used by HTML and thus are making it easier to operate with the language.

Here is a sample of the front page with the current news and the images according to them scaled to 75% in order to show all the content. In order to change the view, when a new news or a video, or analysis is submitted, the css has to be accessed and new image according to the new content has to be uploaded. I have scaled the background image to 100%, so it is not affected by such changes or changes in resolution.



Here is a sample of a random page. There the content is added via html code and isn't dependent on the CSS, so for example when a new news is added with images, they can be inserted only via command `...` And this is basically how the website will look on a resolution 1920x1080.



So this is how I designed my webpage, we can now move on to the next part: adding content to the site.

Adding Content and Functionality – HTML code

In this section I'm going to explain more detailed about every page and try to give an explanation why the user might be interested in the given page. I will provide HTML syntax only for some parts, because it will be too long and pointless to copy/paste the whole code inside this paperwork. So again I'm going to start with the start(home) page then move on to the navigation pages until I reach the specific news and analysis pages and at last I'm going to give a brief explanation about the forum page.

Regarding the home page:

I've tried to show the most important and recent information there in a modern and eye-pleasing view. There is a section for news, analysis underneath it and videos at the bottom. I have updated the title on every page to correspond to the given content. It is shown at the top of the browser tab section. For all the news and analyses I've given the option to connect to the selected news/analysis by clicking on either the title or the image.

```
<a href="coutinho.php"></a>
```

```
<a href="coutinho.php">Coutinho to get new Liverpool deal</a>
```

This is a sample for the main news which goes for all the other news and analysis. I have also added an alternative if the image isn't available at the moment, which is writing a text instead using the command "alt".

Regarding the videos they can also be accessed using the text and the img, but there is also a command added to open the link in a new tab, because they lead to an external site (youtube.com). This is done using the command :

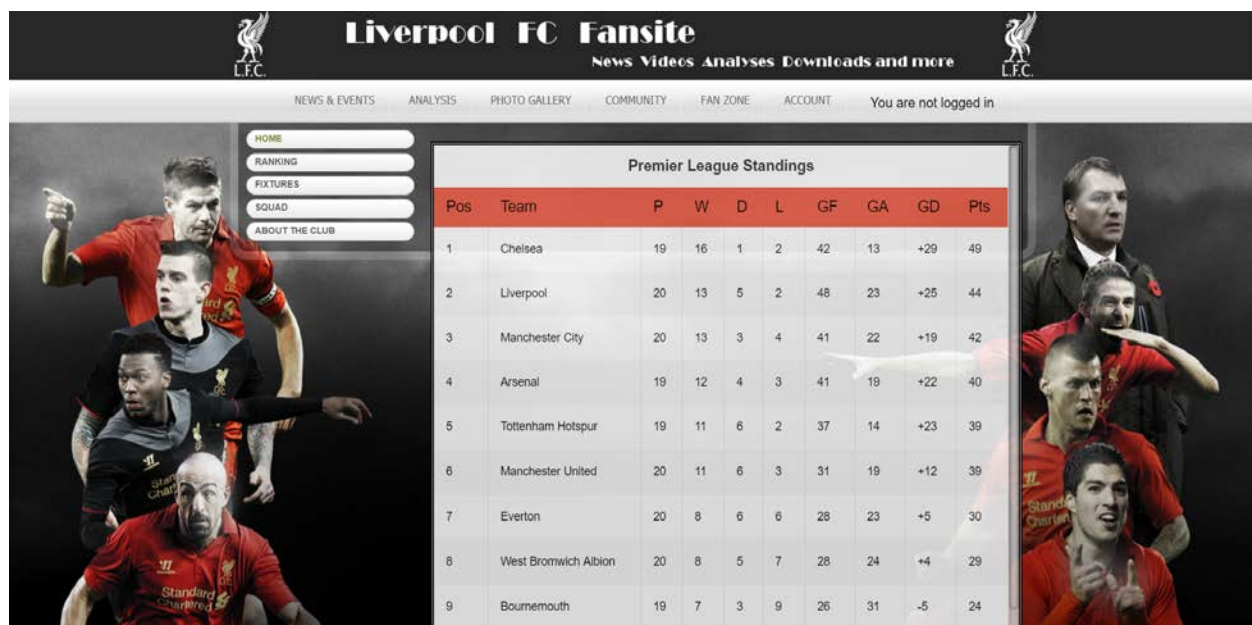
```
target="_blank"
```

First thing to mention before starting to explain about the next pages is that I've created the content field (the field where the information will be presented) adding a whole grey block and making it a <div> field. I have made it with 85% opacity and added the option "overflow: auto" which means that if the information (content) written in the block exceeds its boundaries it creates

automatically scroll bars at the bottom (if it is exceeded horizontally) or on the right side (if exceeded vertically).

Ranking page:

The ranking page represents basically a table of the premier league. It has rows for every team in the table and columns that contain the most important factors that influence the place which a team holds. It takes time to create it initially, but after it is done, it can be updated easily by only writing the new values in the already designed table. I have made the table fit to the window horizontally for easier usage. Here is a screenshot of the page:



Pos	Team	P	W	D	L	GF	GA	GD	Pts
1	Chelsea	19	16	1	2	42	13	+29	49
2	Liverpool	20	13	5	2	48	23	+25	44
3	Manchester City	20	13	3	4	41	22	+19	42
4	Arsenal	19	12	4	3	41	19	+22	40
5	Tottenham Hotspur	19	11	6	2	37	14	+23	39
6	Manchester United	20	11	6	3	31	19	+12	39
7	Everton	20	8	6	6	28	23	+5	30
8	West Bromwich Albion	20	8	5	7	28	24	+4	29
9	Bournemouth	19	7	3	9	26	31	-5	24

Fixtures Page:

Regarding Fixtures page, it is again made of tables, but this time it is not one whole table, but a table for each different match. This is also made for easier later updates, because after a match is played it is then moved to the bottom and the result is updated. I've shown the next matches first with the corresponding time and the already played games at the bottom with the date and result. On the side of each, there is information given about for which competition the given game is played. Again here it only expands(overflows) on the side and is fitted to the box on the sides.



Liverpool FC Fansite

News Videos Analyses Downloads and more



[NEWS & EVENTS](#)
[ANALYSIS](#)
[PHOTO GALLERY](#)
[COMMUNITY](#)
[FAN ZONE](#)
[ACCOUNT](#)
You are not logged in

[HOME](#)
[RANKING](#)
[FIXTURES](#)
[SQUAD](#)
[ABOUT THE CLUB](#)

Liverpool FC Fixtures and Results 2016/17

All fixtures are subject to changes of date and kick-off time due to television, policing and scheduling matters.

Tuesday, 31 January 2017	Liverpool	20:00	Chelsea	EPL
Saturday, 4 February 2017	Hull	15:00	Liverpool	EPL
Saturday, 11 February 2017	Liverpool	17:30	Tottenham	EPL
Monday, 27 February 2017	Leicester	20:00	Liverpool	EPL
Saturday, 4 March 2017	Liverpool	17:30	Arsenal	EPL
Sunday, 12 March 2017				

Squad Page:

The squad page is once again one big table representing the squad of LFC including the coaches and managers for season 2016/17. It is separated and shows each player on a different row, giving information about the position he plays at, small picture, the number he has, the nationality (including the flag) and date of birth.





Liverpool FC Fansite

News Videos Analyses Downloads and more



[NEWS & EVENTS](#)
[ANALYSIS](#)
[PHOTO GALLERY](#)
[COMMUNITY](#)
[FAN ZONE](#)
[ACCOUNT](#)
You are not logged in

[HOME](#)
[RANKING](#)
[FIXTURES](#)
[SQUAD](#)
[ABOUT THE CLUB](#)

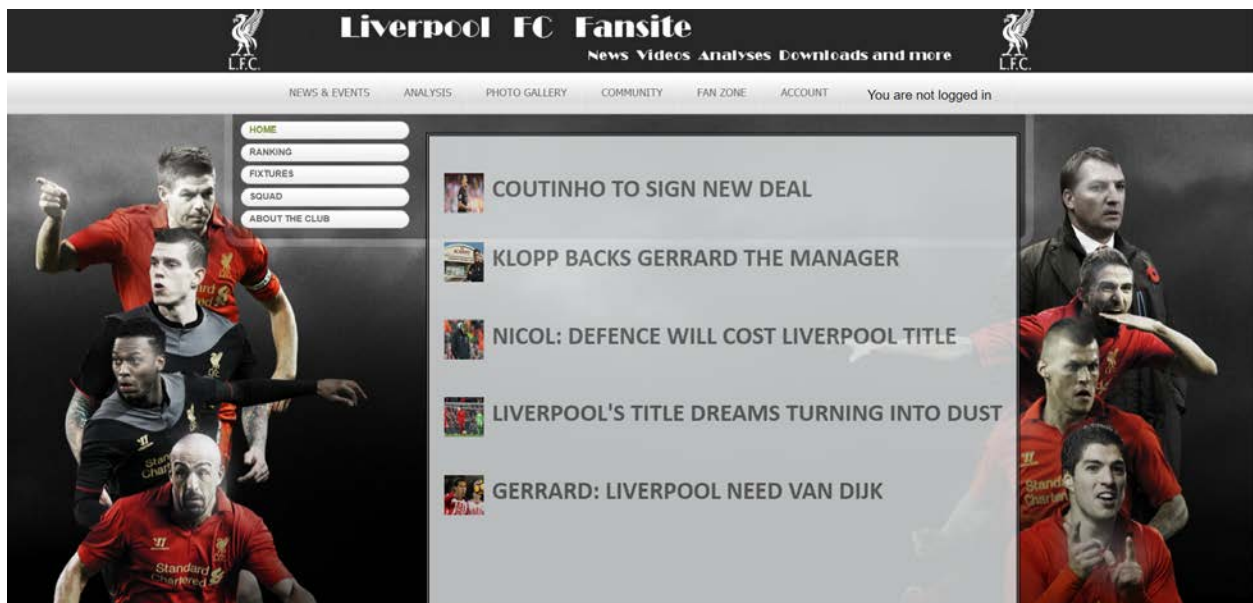
Goalkeeper				
1	Loris Karius		Germany	22/06/1993
13	Alexander Manninger		Austria	04/06/1977
22	Simon Mignolet		Belgium	06/03/1988
Defender				
18	Alberto Moreno		Spain	05/07/1992
66	Trent Alexander-Arnold		England	07/10/1998
2	Nathaniel Clyne		England	05/04/1991
12	Joe Gomez		England	23/05/1997
17	Ragnar Klavan		Estonia	30/10/1985
6	Dejan Lovren		Croatia	05/07/1989
32	Joel Matip		Cameroon	08/08/1991
56	Connor Randall		England	21/10/1995
3	Mamadou Sakho		France	13/02/1990
Midfielder				
23	Emre Can		Germany	12/01/1994

About the club page:

This is the page which gives a brief information about the club starting with trophies won through the years and moving on to a really brief history. It is just raw text and there isn't anything special to mention about that page.

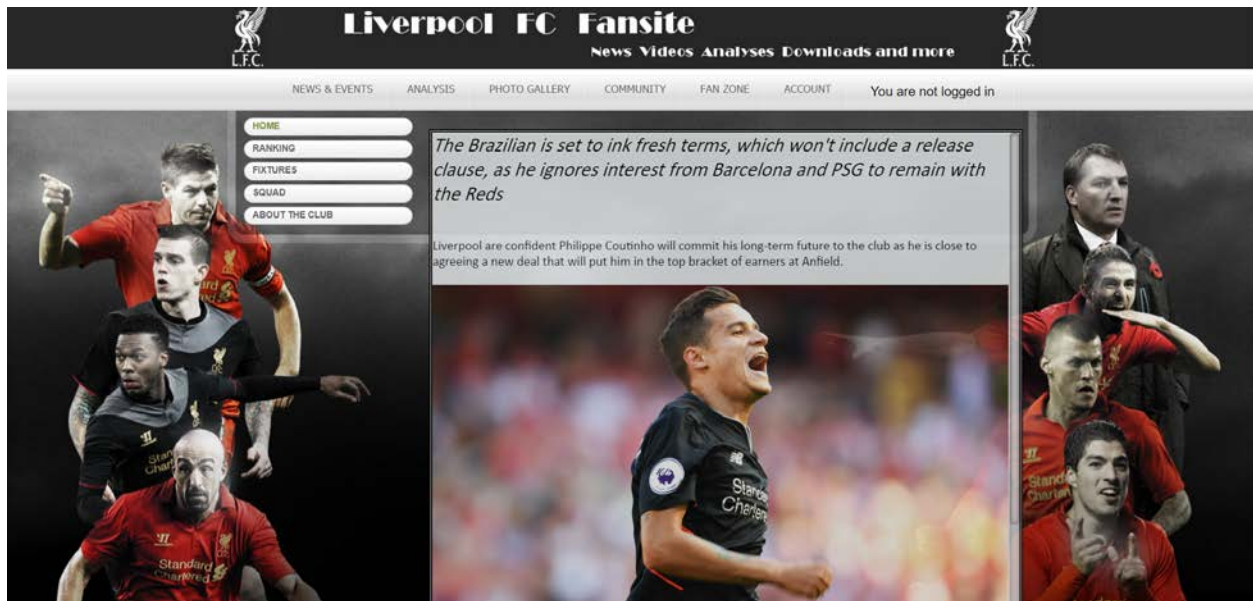
News & Events:

This is the page that organizes and stores all the news in one place. It consists of a list of all the news that have already been uploaded to the page, but because the site is new, there is only 5 news given and they can be updated easily over time. There is a small image in front of every news title. Here is a screenshot:



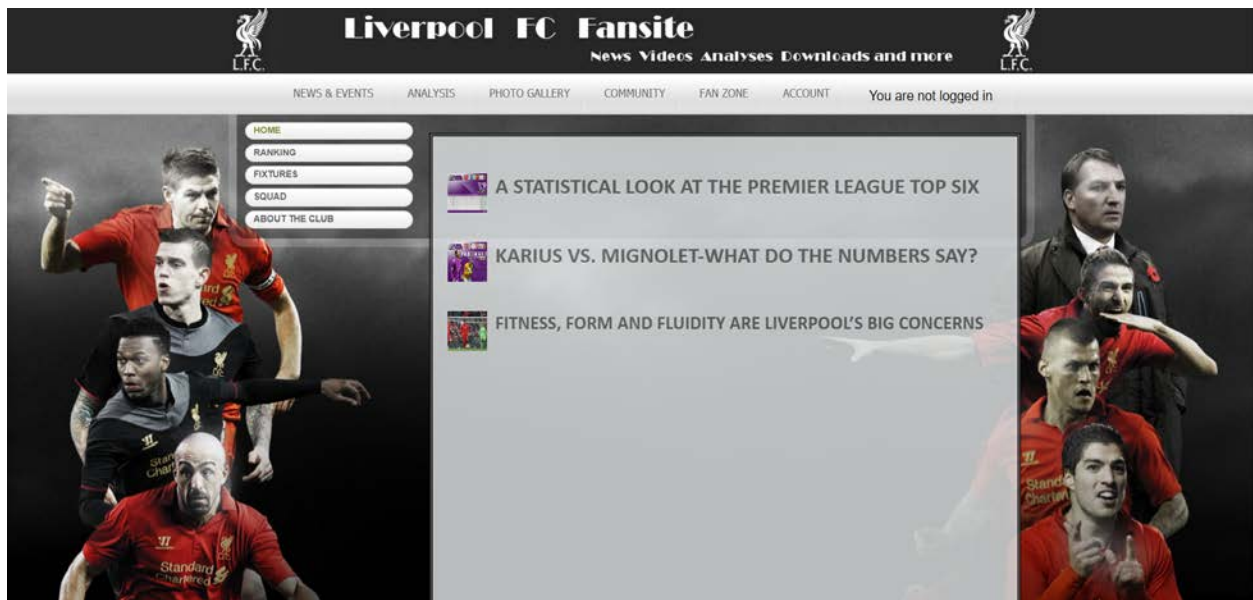
Specific (random) News page:

Every news starts with a summary with a bigger font size which shows what the news is about. The title is given at the top of the browser (the current tab) as a page title. All the news are fitted to the grey box and include text and images regarding the specific news.



Analysis Page

This is the page which organizes all the analyses of the so called specialists and is designed basically on the same principle as the news page. The same goes for the specific analysis page, so I won't go in any further explanations about it.



Gallery Page:

The gallery page is nothing special, just a bunch of photos from games and whatever the admin decides what to add. It is a table and represents rescaled images which contain a reference to the real image and it can be accessed by clicking the image. New images can be added by adding new content to the table using the command <td>



Fan Zone page:

This page is about feeding the website visitors with content different than the raw information. Here the downloads will be placed and also there is a reference to the video page section of the website. The design is made simple and I've included buttons, with text in them describing what they stand for. Most of the buttons are download links and upon click, the download starts right away. This is done using the html command <a href... download> for example:

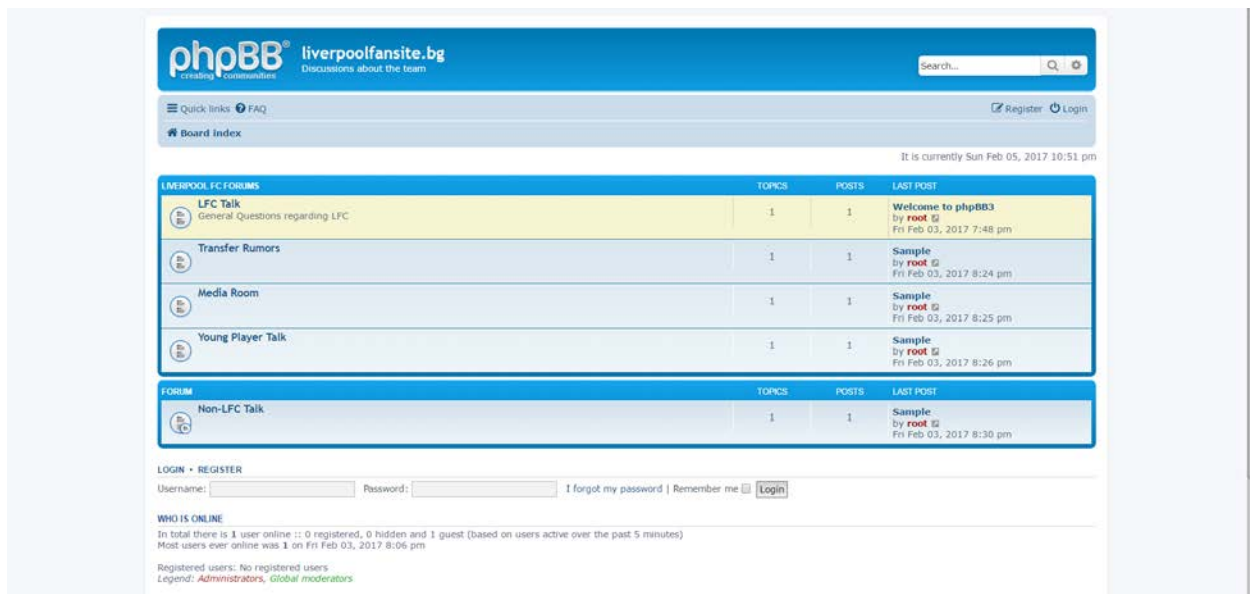
```
<a href="anfield_screensaver.exe" download style="left: 70px; top: 582px"></a></div>
```

I have included downloads for the club most well-known song which stands as a hymn – You'll never walk alone, Windows 7 theme and a screensaver with images of Liverpool FC stadium – Anfield. Here is the image:



Community page:

This is the link which leads to the forums of the website. It is made as mentioned earlier on php and I will explain further about it in the next section. Here I will provide just a photo:



It is just a basic forum, without further development and with the initial design. Basically I've just added the Categories and different forums, but this is the most important think for start, after a while it can be developed and maintained easily.

Account page:

This is the page where the users are going to log in to their accounts, sign up if they don't have one, log out if they wish. It will also contain some information about the user, that he has provided upon registration. There are fields for Username and Password entry underneath which there is a button for login.


```
<input type="text" name="username" placeholder="Username" style="margin:10px;height:26px"><br>
```

```
<input type="password" name="password" placeholder="Password" style="margin:10px;height:26px"><br>
```

The input type tag is placed to recognize that if there is some entry in the password field, it will remain hidden for privacy. The placeholder tag is because it stays over the entry field before some input there is made and to give the user information what to enter there.

On the right side, there are 2 buttons – 1 for sign up and one for log out. The log out button obviously logs the user out of the system, and the sign up button leads to a sign up page, where a new registration can be made.

Beneath all the buttons there is a box where the information about the logged in user is given – names, age, username, nationality.



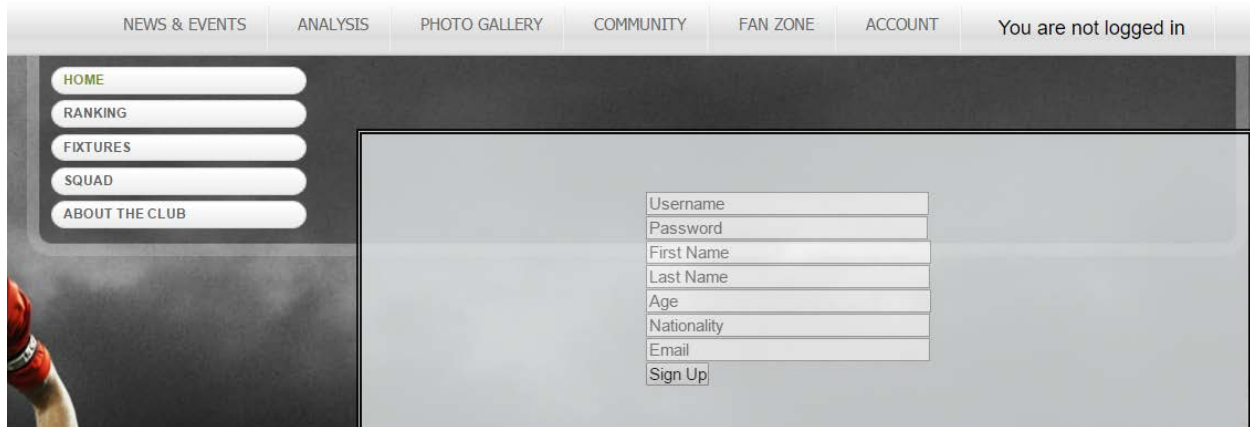
The screenshot shows the Liverpool FC Fansite account page. At the top is the Liverpool FC logo and the text "Liverpool FC Fansite" with links for "News Videos Analyses Downloads and more". Below this is a navigation bar with links: "NEWS & EVENTS", "ANALYSIS", "PHOTO GALLERY", "COMMUNITY", "FAN ZONE", "ACCOUNT", and "You are not logged in". The main content area is divided into two columns. The left column contains a sidebar with links: "HOME", "RANKING", "FIXTURES", "SQUAD", and "ABOUT THE CLUB". The right column contains a login form with "Username" and "Password" input fields, a "Login" button, a "SIGN UP" button, and a "LOGOUT" button. Below the login form is a "USER INFO" section with a box containing the text "You are not logged in".

USER INFO

First Name: Petar
Family Name: Petrov
Username: petarp
Age: 23
Nationality: Bulgarian
Email Address: makaronski12@gmail.com

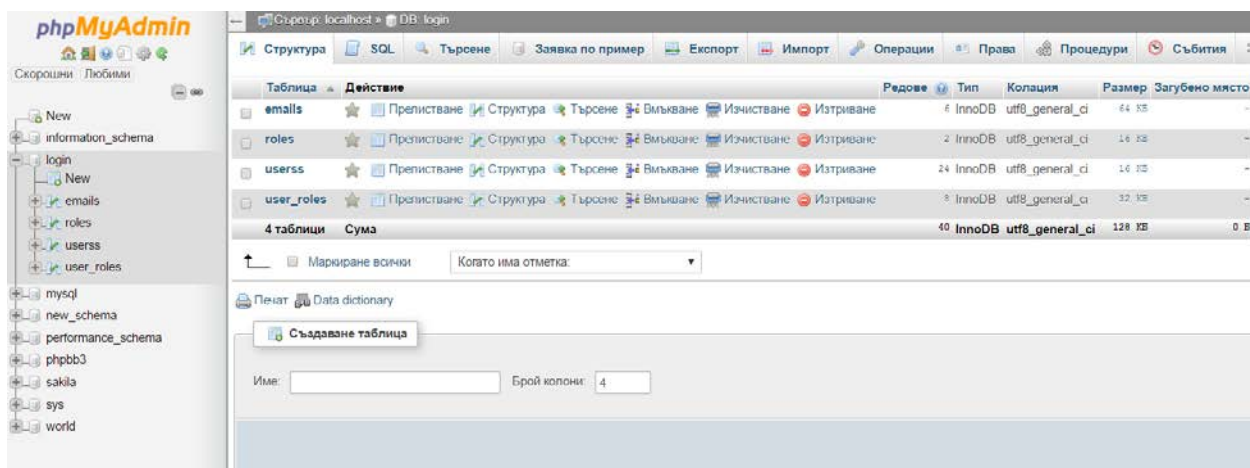
Sign Up page:

Finally we've come to the description of the last page. It contains just a sign up form containing several fields where has to be entered specific information and there is a button on the bottom. After sign up, if successful it leads to the login page, if not you stay on the same page and error message appears to show you what is wrong, but more about this in the next section.



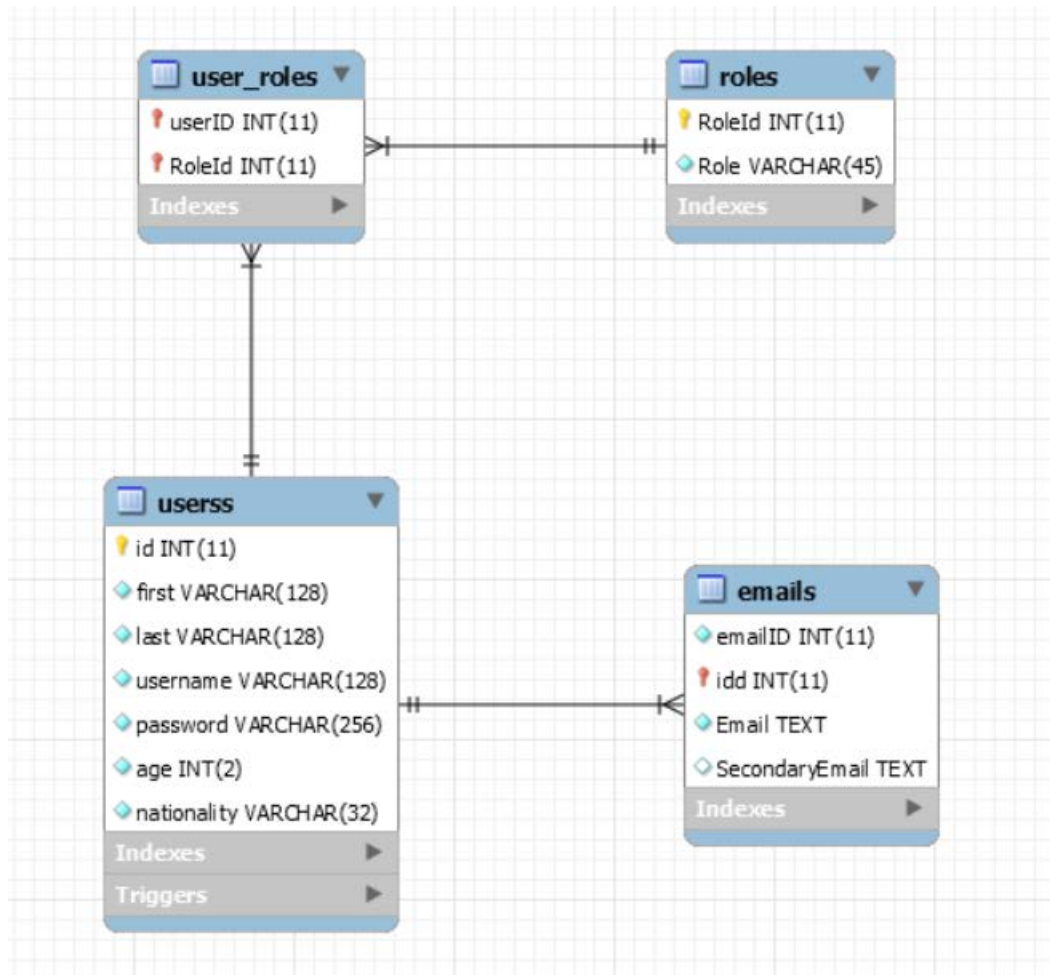
PHP + MySQL

Here comes the other part of the project which involves connection of a database to the website in order to be implemented a login system. This is the more technical part and is where the things get interesting. To be able to achieve this as I mentioned earlier I had to install MySQL and PHPmyAdmin. Here is a screenshot of phpmyadmin page with all the tables.



Database Schema

I decided to add four tables into my DB design to achieve a good level of normalization. Here is a screenshot of the database schema, created with the Reverse Engineering feature of MySql, with all the relationships between the tables:



The primary table is **userss**, and it has many to many connections between it and the **emails** table and **user_roles** table. **Userss** table stores the primary user information as you can see – the names the passwords, username and etc. It's connected to the **emails** table via the primary key **userss.id** and the primary key of **emails** table which also acts as a foreign key for the relationship between the two tables – **emails.idd**. The connection between **userss** and **user_roles** is again

between the userss.id and user_roles.userID. The roles table is already populated with the 2 available roles for the website – Admin and User, and they are given RoleId – 1 or 2. Admins are 1 and users – 2. It is connected to the user_roles using its primary key RolesId and user_roles.RoleId. This is done so, so that a level of normalization can be reached and the data to be scattered around more tables, because it is easier and faster for the DB engine to store and process the information in a normalized way. Now I am going to explain a little bit more about each table.

Userss Table

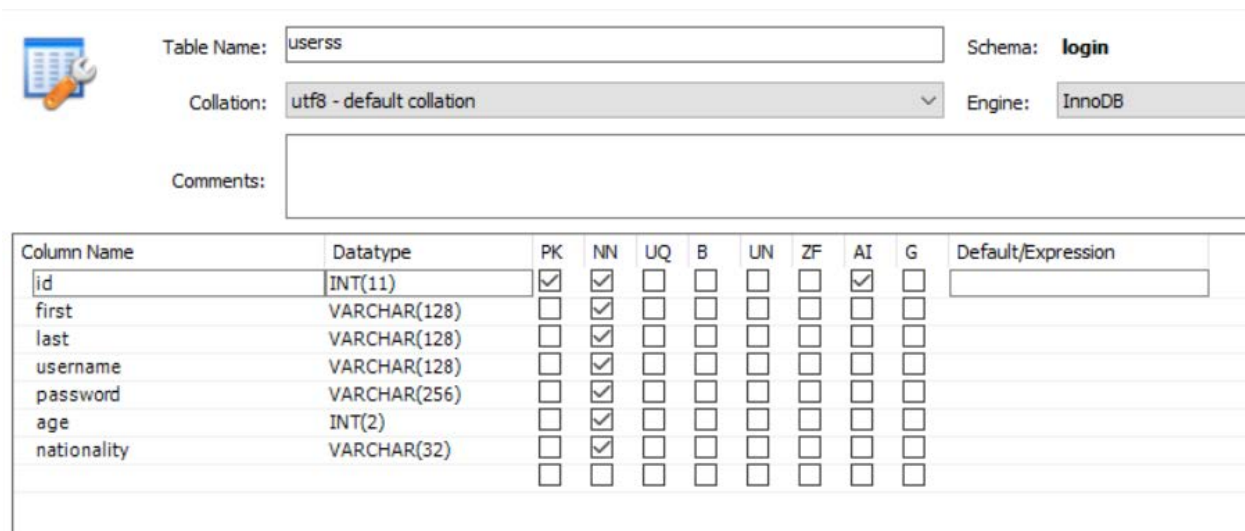


Table Name: Schema: **login**

Collation: Engine: **InnoDB**

Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
first	VARCHAR(128)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
last	VARCHAR(128)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
username	VARCHAR(128)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
password	VARCHAR(256)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
age	INT(2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
nationality	VARCHAR(32)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

It shows that the primary key is the id, which is auto generated integer and is unique for each new entry (new user), also first and last names of the user which are of the type varchar with max length of 128 symbols. Then comes the username which is pretty much with the same characteristics and the password which only difference is that the max length is 256 symbols. After that comes the age which I've created as integer with max length of 2 numbers (assuming there isn't a user over the age of 99). And last comes the Nationality field which is again varchar with max length of 32 symbols. This table doesn't have any foreign keys as it is the primary table and is primary being referenced instead of it referencing any other table. Instead it has a Trigger built in it, which updates the user_roles table every time a new user is created and a new row is inserted in userss table with the same user id as the one in the userss. Here is the syntax:

```

delimiter |

CREATE TRIGGER UpdateRoles After INSERT ON login.userss

FOR EACH ROW

BEGIN

INSERT INTO login.user_roles SET userid = NEW.id;


END;

|

delimiter ;

```


Emails Table

		Table Name: <input type="text" value="emails"/>	Schema: login							
		Collation: <input type="text" value="utf8 - default collation"/>	Engine: <input type="text" value="InnoDB"/>							
Comments: <input type="text"/>										
Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
emailID	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
idd	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Email	TEXT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
SecondaryEmail	TEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

This table is created and used for storage of the emails of the users. The structure is simple: It has column with name idd which is the primary key and also acts as a foreign key for the relationship between emails and userss tables. Email column cannot be null, which means a user cannot be created without specifying an email, but a secondary email is not required, but if the person wants, it can be entered.

Foreign Key Name	Referenced Table	Column	Referenced Column
emails_ibfk_1	`login`.`userss`	<input type="checkbox"/> emailID	
		<input checked="" type="checkbox"/> idd	id
		<input type="checkbox"/> Email	
		<input type="checkbox"/> SecondaryEmail	

Roles Table



 Table Name: Schema: **login**
 Collation: Engine:
 Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
RoleId	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Role	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Here is the next table. It only serves as for reference for the user_roles table and is not designed to be updated. Column roleid is the PK and column Role is the role name. The options are already entered inside the table – 1 is for Admin and 2 is for User.

<div><div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div><div></div></div>						RoleId	Role	
<div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div>	<div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div>	Редакция	<div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div>	Копиране	<div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div>	Изтриване	1	Admin
<div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div>	<div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div>	Редакция	<div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div>	Копиране	<div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div><div><div></div><div></div><div></div></div></div>	Изтриване	2	User

User_roles Table


 Table Name: Schema: **login**
 Collation: Engine:
 Comments:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
userID	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
RoleId	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'2'

This is the table which doesn't have an option to manually enter information (new rows) in it, but rather references other tables using foreign keys. This is a typical example of Normalization. The userID references the id column of userss table and the RoleId references the RoleId column of Roles table. It has a default value of 2, because when a new user is registered, that are the privileges

he gets by default. Together both columns create the primary key of this table (Combined primary key) which means, that there cannot be a combination of the same user with the same role twice.

Foreign Key Name	Referenced Table	Column	Referenced Column
ibfk_1	`login`.`userss`	<input type="checkbox"/> userID	
user_roles_ibfk_2	`login`.`roles`	<input checked="" type="checkbox"/> RoleId	RoleId

Foreign Key Name	Referenced Table	Column	Referenced Column
ibfk_1	`login`.`userss`	<input checked="" type="checkbox"/> userID	id
user_roles_ibfk_2	`login`.`roles`	<input type="checkbox"/> RoleId	

This was the last table of my database design. After that comes the task of integrating this database in the site and making the login system work. That is where PHP comes to help. I have already explained about how the system looks now here I will give an explanation how it works.

PHP

First of all a connection has to be made to the database. It is initiated by creating a separate php class which is then used every time when accessing the database. Here is the syntax.

```
<?php
$conn = mysqli_connect("localhost", "root", "anfield", "login");
if (!$conn) {
    die ("connection failed: ".mysqli_connect_error());
}
```

Where: localhost stays for the server name, root is the username, anfield is the password and login is the database to which I'm trying to connect. Also I have added basic error handling showing a message why the connection isn't working if it isn't.

After that naturally comes the Sign Up page and as explained earlier I've added fields for the required information and a sign up button in the bottom after all fields are filled in. Here I've added some error handling as well, explaining the user, that if he hasn't filled all the fields he can't register and if there is an already a registered user with the same username, he has to choose a different one. First as mentioned above comes the include command to connect to the database. After that variables are created and next comes the error handling. If everything is OK, the information is entered in the table and the user is sent to the home page.

```
<?php
session_start();
include 'dbh.php';
$first = $_POST['first'];
$last = $_POST['last'];
$username = $_POST['username'];
$password = $_POST['password'];
$age = $_POST['age'];
$nationality = $_POST['nationality'];
if (empty($first)){
    header ("Location: signuppg.php?error=empty");
    exit();
}
if (empty($nationality)){
    header ("Location: signuppg.php?error=empty");
    exit();
}
if (empty($username)){
    header ("Location: signuppg.php?error=empty");
    exit();
}
```

```

    }
    if (empty($password)){
        header ("Location: signuappg.php?error=empty");
        exit();
    }
    if (empty($age)){
        header ("Location: signuappg.php?error=empty");
        exit();
    }
    if (empty($last)){
        header ("Location: signuappg.php?error=empty");
        exit();
    }
    else{
        $sql = "SELECT username FROM userss WHERE username='$username'";
        $result = mysqli_query($conn,$sql);
        $uidcheck = mysqli_num_rows($result);
        if($uidcheck > 0){
            header ("Location: signuappg.php?error=username");
            exit();
        } else {
            $sql = "INSERT INTO userss (first, last, username, password, age, nationality)
VALUES ('$first', '$last', '$username', '$password','$age', '$nationality')";
            $result = mysqli_query($conn,$sql);
            header("Location: acc.php");
        }
    }
}

```

If a problem (error) exists, then a message appears to inform the user about the problem.

```
<div style="height: 37px; font-size: x-large; text-align: center;">
    <?php
        $url = "http://".$_SERVER['HTTP_HOST'].$_SERVER['REQUEST_URI'];
        if (strpos($url, 'error=empty')!==false){
            echo "Fill all fields!";
        }
        elseif (strpos($url, 'error=username')!==false){
            echo "Username already exists!";
        }
    ?>
</div>
```

Fill all fields!

Username already exists!

Regarding the login page: When a user enters correct information, he logs in. After that he is sent to the front page and **a session is made** which is included on the start of every single page of the site. When the session is made, the most right field on the navigation bar is stating Welcome (FIRST NAME), instead of You are not logged in. Also on the account page, information about the user is shown.

You are not logged in

Welcome Petar

First Name: Petar Family Name: Petrov Username: makaronski Age: 23 Nationality: Bulgarian

If the information entered is not right, then a short message appears that the info given isn't right. Here is also the code used:

Username or password is incorrect!

```
<?php
session_start();
include 'dbh.php';
$username = $_POST['username'];
$password = $_POST['password'];
$sql = "SELECT * FROM userss WHERE username='$username' AND
password='$password'";
$result = mysqli_query($conn,$sql);
if (!$row = mysqli_fetch_assoc($result)) {
    header ("Location: acc.php?error=empty");
    exit();
} else {
    $_SESSION['first'] = $row['first'] ;
    $_SESSION['username'] = $row['username'];
    $_SESSION['last'] = $row['last'];
    $_SESSION['age'] = $row['age'];
    $_SESSION['nationality'] = $row['nationality'];
}
header("Location: index.php");
```

Then the class is called with the html:

```
<form action="login.php" method="POST".....>
```

And the same code for the error handling is made as above. For the field that shows the information the code is:

```
<?php
    if (isset($_SESSION['first'])){

        echo "First Name:\n {$_SESSION['first']}". "\r\n";
        echo nl2br("\r\nFamily Name: {$_SESSION['last']}");
        echo nl2br("\nUsername: {$_SESSION['username']}");
        echo nl2br("\nAge: {$_SESSION['age']}");
        echo nl2br("\r\n Nationality: {$_SESSION['nationality']}")
    } else {
        echo "You are not logged in";
    }
?>
```

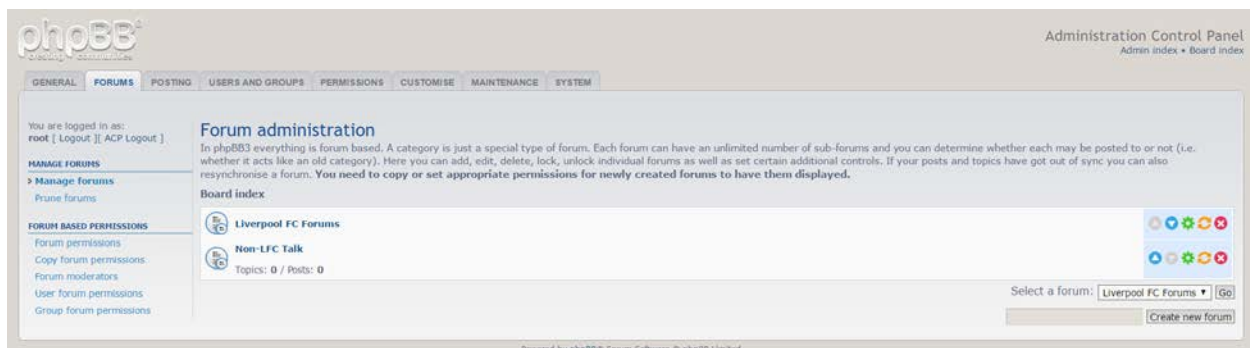
And for the field in the navigation bar it is basically the same, only it refers just to the first name of the user.

And last, but not least, comes the **Log out button**. It basically just ends the session and logs out the current user. After pressing it the information about the user is no longer shown and the user is sent to the front page.

```
<?php  
session_start();  
session_destroy();  
header ("Location: index.php");
```

Forum – PHPBB

I am going to explain briefly about the forum page using PHPBB. It is basically an independent page, with independent database and is the best and easiest way to add a forum to your website. It has way more tables in the database and is way more complicated than the website, but it has a different purpose of course – it has to store user's information and also a search engine and etc. The things I modified are done from the administrator control panel (ACP). It looks differently than the rest of the forum page and is basically the place where the most important stuff is done. There is the place to change permissions, add or remove forums and categories, change the style + many more options to make it work better and to adjust it for the required target group of people using it. Here is an image of the ACP, with which I am going to end the explanation part of the project and will move on to the final part – the final conclusion.



Conclusion

Here comes the point, where I have to end this project and stop explaining how it is working and how I built it. I have offered my vision on the subject and started to understand better how a website has to be built and managed (which I think is the main point) and also about the tools and programs that help us to do it. It is really essential to understand how something works from the inside, in order to make it look good from the outside and to impress the end user. The one thing I know though is that pretty much everything in the internet is a work in progress, as is the website I created – it is still in initial phase but has the potential and the functionality to be put online and to be visited by real users. But before that happens, the most important thing is to be liked by you and I really hope this is the case!

V. Used Literature – References

1. www.techopedia.com
2. www.w3schools.com
3. www.youtube.com
4. stackoverflow.com
5. 11 steps to create a successful website – Startup Nation