This document will very crudely describe the idea of the project and its implementation.

IDEA

First and foremost – what is this site about? The very idea behind it is to take its users on a sort of code-cracking adventure with a number of moments, which can be traced back to very small disadvantages in the Software University (those disadvantages are mainly regarding the building, the university resides in, nothing too offensive… hopefully).

But why would we have to make the students ‘work for’, of sort, for a few little inside jokes, when many other sites will just present them that as-is? Well, I believe that true satisfaction (be it from a game or anything else) comes from **working** for it. A number of writers, philosophers, etc. have written and spoken about how the road to success is far more rewarding than the success itself. And that’s what this site is striving to achieve (well, besides uniqueness) – to further away the price from the user, making him/her travel the road and experience what is regarded as **true** satisfaction.

HOW

Oh, well here we go then. HOW do we do this? Quite easy actually. We have the user do all the work, of course. What we need to do is give enough clues, so the user can add 2 and 2 and continue on the path. I know, this is too cryptic. Ok, let’s start at… the start!

First screen – besides the popup, mentioning who we are, what the project is about etc., the first screen, the very first page is quite basic – simply it’s a decision between left and right. The middle is a white vertical line, on the left we’ll have the “red” part – representing the right part of the brain (painting, singing, etc.) and it will lead to tasks, catering to that part of the brain; on the right we’ll have the “blue” part – representing the left part of the brain (mathematics, logic, etc.) and it will lead to different tasks, catering to, you guessed it, that part of the brain. If someone decides he doesn’t want to pick side and goes for the middle, he will be greeted with the very hardest tasks, because that’s what you get for being a smartass.

So, the “red” and “blue” parts will consist of three pages each, both leading to the same result – the parodied location of SoftUni (clearly picking at the fact that the university is quite far for most of us). The middle one will consist of **I don’t know yet how many and will lead to I don’t know where yet. It’s up for discussion if you want, but soon enough I will figure it out. I do know now however that the middle would be called “WHITE” and it will be the hardest one, maybe even needing a bit of code-making.**

“RED” PART, PAGE 1

The following text will be present at the top of the page:

Welcome,

,Be ever vigilant

;perceive that,

,which cannot be seen.

Good luck

Nothing about this message is random. I will explain later. Below this message, there will be a large image, which absolutely has to be some obscure painting. Why? Because such paintings have very few sources and chances are small that (if a person decides to do reference check between our image and the original one) someone will find the clue within the image before solving the task above.

Okay, so – here we go. There are three total clues in the above message. The first one is semantic – the message itself is a clue to finding the clue within the message. The second one is the clue, which the first one talks about – the message has a number of &nbsp; and space (‘ ‘) elements scattered about. They’re not random. One should be able to see that each set of space and &nbsp; is of 16 elements. &nbsp is 1, space is 0. These are 16-bit representations of numbers. The third clue are the commas and semicolons. If someone finds the 16-bit numbers, he (or she) will come up with 4 numbers total and if he puts those numbers along with the commas and semicolons, which don’t belong, he’ll have a,b;c,d (a,b,c,d are actual numbers). So what are those? Those are positions on an x-y plane. The first pair will point to the top-left corner of a section in the painting, the second pair will point to the bottom-right corner of that section. So what would a person find in that section if he looks more closely? Well, with a number of filters, which can be found at Paint.NET, a message will appear - /nlsQp.html. GUESS what that is. Yeah.

By the way – it may be a bit too hard to hide /nlsqup.html in its current form. It needs to probably be represented by something small – dots, dashes! We need morse code!

-..-. -. .-.. ... --.- .--. .-.-.- .... - -- .-.. This is the representation of /nlsqp.html

The painting is “The blue moon” by Pablo Picasso. It symbolizes the hidden message within the top message (just like the painting itself hides another painting). The above message will be hidden in a block, starting with 123,76 (01111011, 01001100). This means, right after Welcome, the message should have the following: “ &nbsp;&nbsp;&nbsp;&nbsp; &nbsp;&nbsp;”

“RED” PART, PAGE 2

The second page will start with the following message:

Good work,

But are the efforts you expend enough to dig out the diamonds you need to build your own dynasty? Are you prepared to be the **master of the game**?

Only time will tell.

Just to mention, the bolded text is not to be bolded in the site – those words should be in different color, but should very well match the others. What I mean – if all other letters are of color #000000, these letters should be about #000007. But why should we make those like that? Well, Master of the game is a Sydney Sheldon book, which speaks of a diamond dynasty (The diamond dynasty is actually the Bulgarian equivalent). Next part of the puzzle is this:

# # # # #

# # # # #

….

Each # here represents a block of certain color. Upon finding the hash code of each of those colors, the person should notice something weird – all of them consist of letters and only 1 or 2 numbers. For example EF13AA or FFFFF1. This should ring a bell that only those numbers are important. But what are they? #:#:#:#:# - chapter:paragraph:sentence:word:letter. It’s not an uncommon encrypting mechanism. So what will the find after the decrypt all of those numbers – rpQLstPP. I think you can guess what that is.

However, on this page there’s one more thing – a recording. This recording should include 3 main things – a timestamp, which is of another lecture, where XOR is discussed, a problem in the sound and the phrase “next time” (or similar). XOR will be needed for the next page, the problem in the sound is poking fun at the stupid microphone problems, “next time” is to refer to Page 3.

“RED” PART, PAGE 3

Again, we start with a message:

good work,

you are truly a SoftUni student.

a little more and salvation is at hand.

you shall soon be Home.

Again, nothing random about this message either. Below this message there should be a painting of a person (real, of legends or mythology), which has been known to always speak opposite of what he means. This image will have nothing else hidden in it – it is there to say that what the person needs to do is reverse anything the next string has to say. Will go in more detail in a bit.

The next thing, which should be shown is a string of letters, which, when converted to numbers and XOR-ed, represent the latitude and longitude of SoftUni’s parodied location. This should include any special symbols.

So how’s that the end? Well, they’ll be left with two things – SoftUni and Home jutting out of the above text, as well as the awesome town of Madurai, India from the coordinates from the string. Eh? Eh? Yeah, the humor is mild with this one, but this is in fact the end of the road. There might be another page, which tells the one that didn’t get the joke that the pages are over, but I haven’t thought of a way to hide that yet.

To be continued with the “BLUE” part.

BLUE Page 1:

What need to be shown on this page are a few things: a number (representing the sum of the ascii numbers), a poem (which from the first letters you get ‘ascii sum’), a set of numbers, which represent the percentages of the ascii in that ascii sum.

So what will we get after finding the ascii? Why, /stussw.html of course. Okay, let’s start.