Cambridge Academy for Science and Technology

AQA Computer Science

Practical Computing Project

CRYPTOGRAPHY ONLINE

Author
J.P. Jacob Powell
Candidate Number: 8032
Center Number: 22231

Supervisor
B.C. Barry Cooper

April 29, 2018

Contents

Chapter 1

Analysis

1.1 What is Cryptography

Cryptography is idea of allowing 2 parties to communicate securely and not allowing an adversary to listen in on the communications. In a general sense it is the art of building secure protocols that allow us to communicate securely. This idea has been around for centuries and is constantly evolving with the increase in demand for technology in our everyday lives and the need make certain that everything that is in the public domain is secure.

1.1.1 Quick History

1.1.1.1 Ancient History

One of the first recorded instances of cryptography is from the Roman Empire when *Julius Caesar* created what is now known as the Caesar Cipher. This cipher works by shifting the letters of the alphabet a given number of times. According to *Gaius Suetonius Tranquillus*, Julius Caesar used a shift of +3 places to protect messages of military importance. Even though it is unknown how effective the cipher was at the time it can be assumed that it was reasonably secure. Not least because most of Caesars enemies would have been illiterate and others would have just presumed that the messages were written in an unknown foreign language.

At the time there was no recorded way to break these ciphers. The earliest records of attacks that could be used to break the cipher are from the 9^{th} century, with the work of *Al-Kindi* in the Arab world with the discovery of frequency analysis.

1.1.1.2 Recent History

WW1: One of the most notable events during WW1 would be the British interception and decryption of the Zimmermann Telegram. Room 40, the section in British admiralty most identified with cryptanalysis effort during WW1, heavily contributed to the decryption of the Zimmermann Telegram.

This decryption is often referred to as the most significant triumph in signals intelligence for Britain in the First World War, and one of the first times in which a single piece of signals intelligence directly influenced world events. In this case the United States of America joined the War on 6^{th} April 1917.

WW2: During World War 2, cryptography was used extensively with a plethora of ciphers systems fielded by the nations involved during the war. In addition to the advancements of cipher systems, the theoretical and practical aspects of cryptanalysis was much more advanced.

Probably the most important code breaking event during WW2 was the breaking and decryption of the *Enigma* Cipher. The first 'full' break of the enigma cipher was in 1932 by Poland, with the techniques and insights were passed on to the British and French Allies just before the start of the War in 1939. These were improved significantly by the British efforts at the Bletchly Park research station during the War. The decryption of the Enigma Cipher allowed the allies to read important parts of the German Radio networks and it provided an invaluable source of military intelligence throughout the war. Intelligence from this source (including other high level sources, including the Fish Ciphers), was eventually called *Ultra*.

A similar break into an important Japanese cipher (PURPLE) by the US Army Signals Intelligence started before the US entered the War. The product of this effort was given the codename *MAGIC*, it was the highest security Japanese diplomatic cipher.

1.1.2 Modern Day Cryptography

1.1.2.1 Symmetric

Symmetric-Key Algorithms are algorithms for cryptography that use the same Key for encryption and decryption of the plaintext and ciphertext respectively. These *Keys* are a representation of a shared secret between 2 or more parties, that can be used to maintain a confidential link of communication. The main drawback of this system is that all the parties communicating need to know the secret key in order to communicate with each other.

There are 2 main types of Symmetric Cryptography Algorithms:

- Stream Ciphers

  Stream ciphers will go through each bit of the plaintext and encrypt it one at a time

- Block Ciphers

  Block Ciphers will divide the plaintext into groups of bits and will encrypt each of those *Blocks* at once, padding the plaintext if the length of the plaintext does not equally divide the block size of the cipher. Blocks of 64-bits were commonly used but now a days a block size of 128-bits is more common. The Advanced Encryption Standard (approved by NIST in 2001) and the GCM Block cipher both use block sizes of 128-bits.

1.1.2.2 Asymmetric

Public Key Cryptography or Asymmetric Cryptography is any cryptographic system that instead of using a single secret key, uses pairs of keys:

- Public Keys

  These could be known widely to the public domain

- Private Keys

  These are only known to the Owner

This accomplishes 2 things: Authentication, where the public key verifies that the holder of the paired private keys sent the message, and encryption, where only the paired private key holder can decrypt the message encrypted with the public key.

Public Key systems are often based on complex mathematical systems that currently have no reasonable solution to them. Some of these include certain integer factorization, discrete logarithm and elliptic curve problems. Unlike symmetric key algorithms, asymmetric algorithms do not require a secure channel of communication for the initial exchange of one or more of the secret keys between the parties.

Because of the computational complexity of asymmetric cryptography, it is usually only used for small blocks of data. This is typically the transfer of a symmetric encryption key or *Session Key*. This session key is the used to encrypt the rest of the potentially longer message. This symmetric encryption/decryption is based on simpler algorithms and is much faster.

In a public key signature system a person can combine their message with a private key to create a short digital signature on the message. Anyone with the corresponding public key can combine a message, a digital signature and the known public key to verify if the signature is valid or not. Therefore this provides authentication of a message provided that the owner of the private key keeps the private key secret.

These public key algorithms play fundamental roles in cryptosystems, applications and protocols. They are heavily involved in internet standards such as TLS (Transport Layer Security), S/MIME, PGP and GPG. Some public key algorithms provide key distribution and secrecy. Examples include the Diffie-Hellman Key Exchange. Some algorithms provide digital signatures. Examples include the Digital Signature Algorithm and some provide both, e.g. RSA.

Public Key Cryptography finds applications in the Information Security Discipline. Information Security (IS) is concerned with all aspects of protecting electronic information assets against security threats. Public Key cryptography is used as a method of assuring the confidentiality, authenticity and non-reputability of electronic communications.

1.1.2.3 Cryptanalysis

Cryptanalysis is the study of information systems working towards the goal of unveiling the hidden aspects of the system. This knowledge is then used

to break the information system and gain access to the contents of encrypted information.

In addition to mathematical analysis of cryptographic algorithms, cryptanalysis also involves looking at side-channel attacks. These attacks don't specifically look for weaknesses in the algorithm itself but for a weakness in the implementation of algorithm.

Considering the end goal has been the same, the specific ways in which we get there have evolved and developed drastically of the past few decades. We have gone from the pen-and-papers methods of the past, to the machines like the British bombes and colossus supercomputer at Bletchly part to the mathematically advanced computerized schemes of the preset. Most cryptanalysis of modern day algorithms involves solving complex pure mathematical problems, including the integer-factorization problem and the discrete logarithm problem.

1.1.2.4 Protocols

A security protocol is an abstract protocol that performs a security function and applies cryptographic methods, often grouped together as sequences of cryptographic primitives. The protocol defines how the algorithms should be used.

Cryptographic protocols are widely used for application-level data transport. A cryptographic protocol usually incorporates at least a few of the following features

- Key Agreement / Establishment
- Entity Authentication
- Symmetric Encryption and message material construction
- Secure Application-level data transport
- Non-repudiation methods
- Secret Sharing Methods
- Secure Multi-party computation

Example: Transport Layer Security An example of one of these algorithms could be the Transport Layer Security, or TLS, protocol. TLS is used to secure web (HTTP/HTTPS) connections. It has an entity authentication mechanism based on the X.509 system; a key setup phase, where a symmetric key is formed by employing public-key cryptography; and a secure application-level data transport system. These 3 functions have very important interconnections. Standard TLS does not however implement Non-repudiation methods.

There are other types of cryptographic protocols as well, and even the spelling has multiple meanings. Cryptographic Application Protocols, often

use one or more underlying key agreement methods, which are sometimes themselves referred to as cryptographic protocols. For example, TLS employs the Diffie-Hellman Key Exchange, which although is only a part of the TLS protocol, could be seen as a completely independent cryptographic protocol in itself for other applications.

1.1.3 Applications of Modern Cryptography in Everyday Life

Cryptography is present in nearly every single electronic device that has been produced in the past decade. This ranges from your new ePassport to your new mobile phone to your smart fridge. It is everywhere and because of this it has become even more important.

For the most part Cryptography provides security to a service that we use in out daily lives. Examples of these services could be:

- ATM Cash Withdraw
- File Storage
- Emails
- TV
- Text Messaging
- Web Browsing
- Payment Systems like PayPal

1.2 Important Cryptographic Algorithms

1.2.1 DES

1.2.1.1 What is DES?

DES, or the Data Encryption Standard, is a symmetric block cipher used for the encryption and decryption of electronic data. Even though it has now been proven that DES is insecure it was a major keystone in the development of modern cryptography.

Developed in the early 1970's by IBM, based of an earlier design by *Horst Feistal*, the algorithm was submitted to the National Bureau of Standards (NBS) following the agency's invitation to propose a candidate for the protection of unclassified, sensitive electronic government data. In 1976, after consultation with the National Security Agency (NSA), the NBS eventually selected a modified version of DES which was published as a Federal Information Processing Standard (FIPS) for the United States in 1977. It should be noted that this modified version helped protect against differential cryptanalysis but weakened it significantly against brute-force attacks, the attack that

would eventually make NIST (National Institute for Standards and Technology, the new NBS) ask for the newer Advanced Encryption Standard (AES).

1.2.1.2 Overview of DES

DES is a Block Cipher. It takes a fixed-length amount of plaintext bits and transforms it through a series of complicated operations which results in ciphertext bits. In the case of DES, its block-size is 64-bits, this means that you would split up your plaintext into 64-bit blocks and pass them through the algorithm one at a time. In the case that your plaintext does not fully divide into 64-bits you would use padding to increase the size of the plaintext until it fully divided into 64-bits. DES also uses a Key that heavily effects the transformation, so decryption can only be done if the user has the Key. DES uses a Key size of 64-bits. However it should be noted that 8 of these bits are used solely for parity checks so the effective Key size of DES is 56-bits. The Key is stored or transferred as 8 bytes with odd parity. According to *ANSI INCITS 92-1981*, section 3.5:

> *One bit in every byte of the KEY may be utilized for error detection in key generation, distribution and storage. Bits 8,16,..,64 are for use in ensuring that each byte is of odd parity.*

Like all Block Ciphers, DES on its own in not inherently secure, it must be used in a mode of operation to achieve this. FIPS-81 specifies several modes to be used with DES.

Decryption works the same way as encryption, but the keys are used in reverse order. This gives the advantage that you can use the same hardware/software for both encryption and decryption.

Overview of Internal Structure: DES uses 16 rounds, this means that the core blocks of the cipher are repeated 16 times on each block of plaintext. There is also an Initial Permutation and a Final Permutation, name IP and FP respectively. These have no significant cryptographic impact on the cipher but were added to allow the loading of blocks in and out of mid-1970s 8-bit based hardware.

Before the main rounds the block is divided into two 32-bit halves, being processes alternatively. This structure is known as a Feistel Network. This way of constructing a block cipher ensures that decryption and encryption are very similar processes, the only difference being that the subkeys are applied in reverse for decryption. Figure 1.1 shows this.

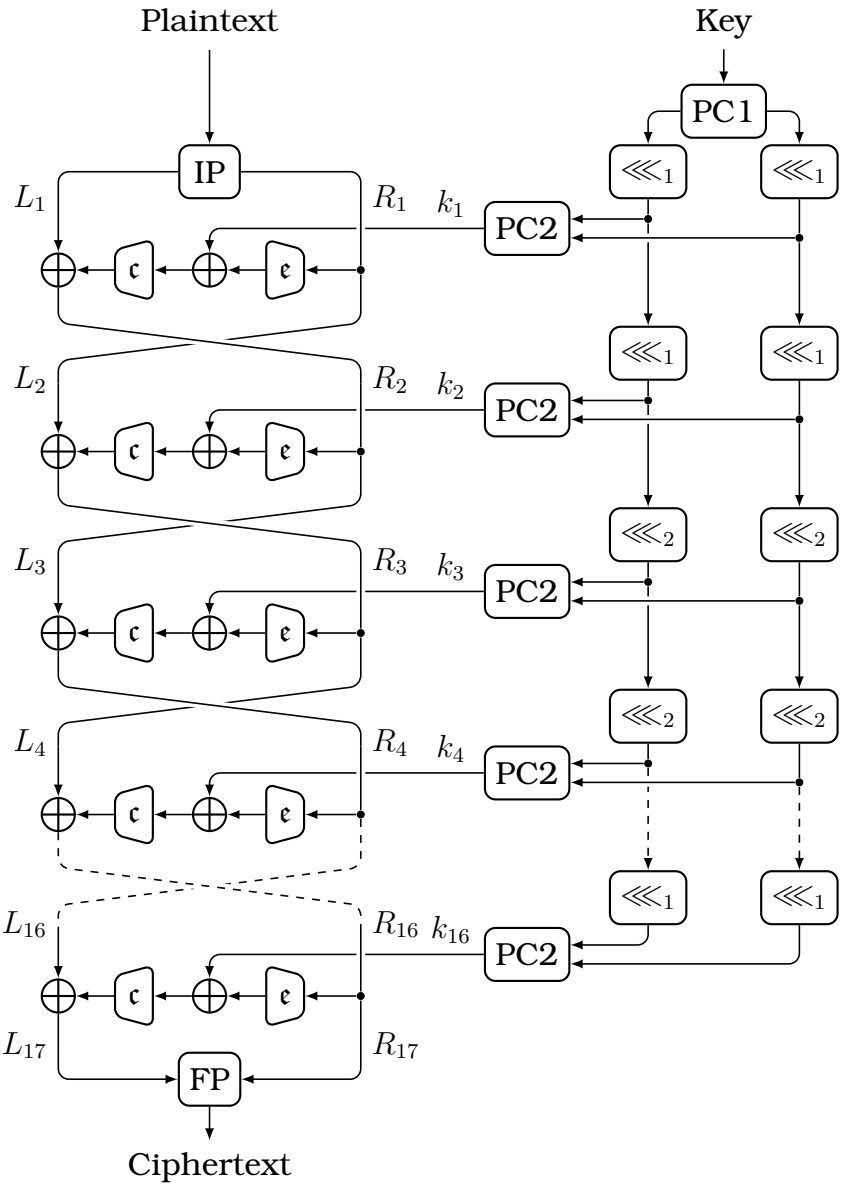


Figure 1.1: High Level Overview of the DES Block Cipher

1.2.2 AES

1.2.2.1 What is AES?

AES, or the Advanced Encryption Standard, is a specification for the encryption of electronic data established by the National Institute for Standards and Technology (NIST) in 2001. AES is a subset of the Rijndael Cipher, developed by 2 Belgium cryptographers. *Vincent Rijmen* and *Joan Daemen*, who submitted the cipher during the AES selection phase. Rijndael is a family of Block Ciphers which support multiple block sizes and key lengths.

For AES, NIST selected only 3 members of the Rijndael cipher. Each using

a block size of 128-bits and supports using key lengths of 128, 192, 256 bits.

AES was first adopted by the US Government and nowadays is used world-wide. It has become the defacto standard for securing electronic data. It has replaced the Data Encryption Standard which was first published in 1977. AES was first announced by NIST on *November 26, 2001* as a FIPS PUB 197 (FIPS 197). This announcement followed a five year standardization process on which fifteen competing designs were presented and evaluated, before the Rijndael Cipher was accepted as the most suitable.

AES became effective as a federal government standard on *May 26, 2002*, after its approval by the state of commerce. AES is included in ISO/IEC 18033-3 Standard. AES is available in multiple encryption packages and is the first(and only) cipher approved by the National Security Agency (NSA) for top secret information when used in an NSA cryptographic module.

1.2.2.2 Overview of AES

AES is based on what is known as a substitution-permutation network, and is very fast in both hardware and software. Unlike its predecessor DES, AES is not build on a Feistal Network. Since AES is a variant of the Rijndael Cipher using a Block size of 128-bits and supporting key sizes of 128, 192, 256 bits.

AES operates on a 4 x 4 column major order matrix of bytes, known as the state of the cipher. The calculations of this Cipher are performed in the Galios Field of 2^8, $GF(2^8)$, this is covered in detail in *Section 2.4.1*. The byte ordering in the matrix is shown in Figure 1.2.

b_0	b_4	b_8	b_{12}
b_1	b_5	b_9	b_{13}
b_2	b_6	b_{10}	b_{14}
b_3	b_7	b_{11}	b_{15}

Figure 1.2: AES Byte Ordering

The Key size used for the AES cipher specifies the number of rounds the used to convert the plaintext block into the ciphertext block.

- 128-bits : 10 Rounds
- 192-bits : 12 Rounds
- 256-bits : 14 Rounds

Each round consists of several processing steps, each containing 4 steps including one that depends on the key supplied to the algorithm. A set of

reverse rounds are applied to transform the ciphertext back into the plaintext using the same key. A High Level description of the algorithm is given below in Figure 1.3:

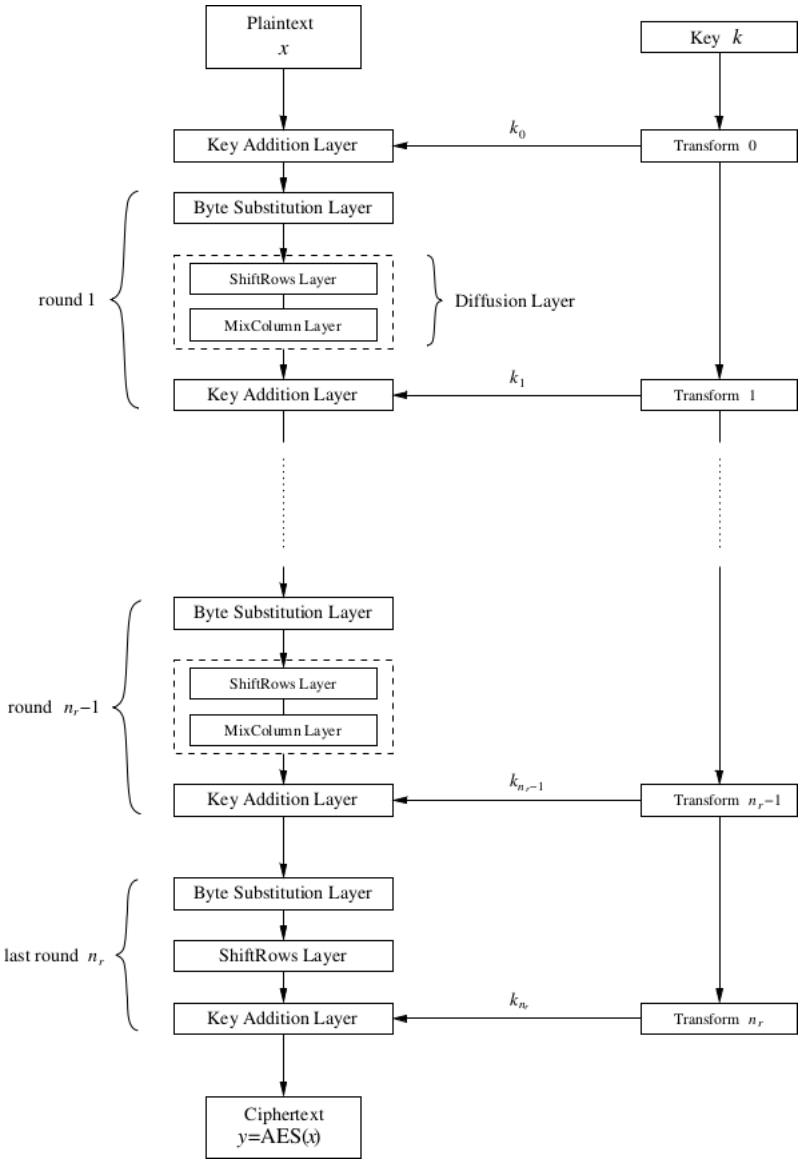


Figure 1.3: AES High Level Overview

1.3 Why is this important?

1.3.1 The Importance of Cryptography in the Modern day

In the modern era, the amount of reliance we have on technology has exponentially increased over the past few decades. With this dramatic increase, we

have started using this new technology to complete tasks that use very important and critical information about us, like online banking. This has opened a new opportunity for criminals to gain access to your critical accounts and now, rather than having to rob a bank, steal your credit cards, mug you, they only need to be in the same coffee shop connected to the same pubic Wifi as you. This is a very bad situation.

Cryptography plays a key role in this as it is the only thing that stops attackers from being able to access all of our critical information. As long our data is encrypted securely, it doesn't matter whether or not they can see what we are doing by being connected to the same public Wifi, all they will see is jiberish that makes no sense. And as long as the algorithms we use are secure and we make sure they don't get access to our private cryptographic keys they will never be able to access our critical data.

This is why Cryptography is very very important in the Modern Day. Cryptography is the only defense we have against attackers who have access to our data. The counter argument could be made that as long as they never get access to our data we will be fine. But, recent cyber attacks against corporate companies have proven that even the most secure facilities are not 100% secure. So preventing the attackers from getting the data in the first place is important, but it is equally important to make sure that when they get the data, that data is useless to them.

1.3.2 Analysis of Current Solutions

Now we know that cryptography is possible one of the most important concepts in our digital age we need to make sure that it stays that way. This means we need to educate and train the next generation of cryptographers so that in 100 years time, we don't run out of people that can develop, test, maintain the next wave of cryptographic algorithms. So, what are the current ways that an individual can learn the ways of the cryptographers.

1.3.2.1 1. Academia

The most traditional way to break into a given industry, you enroll at a institution that allows you to study cryptography. You will learn everything, from the basics of cryptography all the way up to why specifically the designers of certain algorithms did it that way rather than another way.

- Advantages

  With this method being the most traditional way of learning a given subject, it has been tried and tested for many many years and works pretty much every time. With the additional bonus of having staff who will provide you any assistance you require over your few years of study, whether that would be on understanding a given topic or just to answer some of your questions. You also have ability to use any on-site facilities that the institute provide for you and this can be a massive help as it means you

don't have to go out and spend money on specialist equipment to test something. You can just use the institutes.

- Disadvantages

  Even though this is the most traditional way to learn something in depth, like everything it does have flaws. For some people, learning this way just doesn't work well. You will spend most of your time learning about the theoretical side of the subject and will only look at the practical side on a few occasions. Couple this along with the fact that this route is very very expensive it makes it pointless for somebody that learns best by doing practical work.

1.3.2.2 2. Self Taught

Becoming a more and more popular way to learn about cryptography is to just teach yourself, its the route it took. This way allows you to quickly focus in on exactly the aspect of cryptography that you find interesting and learn about it in depth.

- Advantages

  With the internet becoming more and more accessible to the average person it has made an abundance of learning resources available to anyone with a browser from the last decade. This given us an opportunity to learn more things than ever before and it is all completely free. There is plenty of information on cryptography that you can learn from, this includes YouTube series on every aspect of modern cryptography to research papers about an up and coming area of cryptography, like Quantum Cryptography. Combine this with the fact that you can learn at exactly the right pace for you and it is a very good option.

- Disadvantages

  The main drawback of this option is that you will need a device that allows you to connect to the internet, and you will need internet which does cost a fair amount. You also need the commitment to stick to it. Because you can go at your own pace, it does mean there is no one that will make you stick to learning about it. The entire responsibility of learning about the subject is in your own hands and for some this could be to much and they will just give up after about a week.

1.3.2.3 3. Learning on the Job

This option mixes the aspects of both the previously mentioned options. Some people will have already gotten a job in the technology industry before realizing how important cryptography is and will want to learn it so they know that they are doing there job right.

- Advantages

  With already working, you will be moved away from the theoretical and more the practical. This is great for some people. You will also be able to see exactly how these highly complicated concepts directly affect the way we use the internet. You may also have the opportunity through your employer to attend various courses and conferences that allow you to gain various certifications in cryptography and maybe even the broader InfoSec industry.

- Disadvantages

  The main disadvantage here is that you will need a job in a somewhat relevant area in the industry and in order to get the job in the first place you will probably have gone through options 1 and 2 anyway.

1.4 My Proposed Solution

1.4.1 End Goal

The end goal of my project is to provide a website that combines the best parts of Options 1 and 2 from the above list. It will teach the users everything from the core concepts of cryptography all the way to the very complicated systems that implement modern cryptography. This way people have the ability to learn as much as they would while in academia while getting the same freedom as if they were teaching themselves.

1.4.2 Proposed End Users

In the short term there my End Users will be a few students from the Computer Science class. Over the past few weeks while I have been learning Cryptography they have noticed and thought it was pretty cool and wanted to learn as well. But most of them just don't have the commitment to sit through hours of lectures like I did to learn the content so I though that I should build something that they can use.

Due to the content in this project, I will also be making it available to the public so anyone that really wants to learn about Cryptography will have the option to. This allows for a much larger range of people who will have the option to learn about Cryptography which everyone can see a being a good thing.

1.4.3 How the User will interact with the system

The main way that uses will interact with my system is quite simple. First they will need to create an account so they can record and keep track of there progress. They will then select a specific area to study. This is completely up the the user, they can work there way through chronologically going through

each section or just jump to the middle and crack on with the hard stuff. Once in the selected section, they will read through the information content displayed on the screen. Once the user feels confident they understand that content, they can attempt to answer the sample questions at the bottom of the page. These questions will cover all content shown on that given page and will vary in difficulty. Once they have answered the questions they can check if they are right, this information will be shown in the report section, any questions they get wrong will be shown and the provided solutions as well so the user can see where they went wrong and learn from the mistakes made. Once they are happy with how they have performed in that given section they can move on to the next section repeating this process.

1.4.4 Technical Analysis of my System

1.4.4.1 Option 1: Classical Approach

My first idea about how I was going to attempt to build this system was arguably the 'classical' approach. By using the 3 core web technologies, (HTML CSS JS), I would have all the features that would be needed to fully achieve my goal.

- Advantages

  The biggest advantage of choosing this option would be that it has been around for decades. It is truly a tried and tested method. This means that not only does it have the flexibility that I would require, but if I were to run across any problems then I would just have to use the internet and chances are someone else has had that problem before and they would know how I could fix it and make it work. This is a massive perk over the other 2 choices as it means that even if I come across an abundance of problems, I will definitely be able to find a solution in a timely manner.

  An another advantage is that Javascript, that main language I would be using to 'code' the website, has many frameworks and extensions that I would be able to use freely. This allows me the ability to be a lot more ambitious about what I want to do as I wouldn't have to develop the majority of the core complex functions. I would just have to know how I need to implement them.

- Disadvantages

  The main disadvantage of this option for me is that personally I tend to stay away from this kind of work. I personally am much more of a back-end type developer. So I would definitely go for one of the other options.

  I also have much more experience with the back-end type languages and feel I would be spending the majority of my time learning the concepts and syntax of this option, whereas if I were to choose one of the other options I already know the syntax so could start working straight away.

1.4.4.2 Option 2: Modern Approach

This Option is the most 'modern' approach I could think of. This option considers all options with similarities to ASP.NET. I consider this the 'modern' approach as we are not using specific web technologies to create our website. We are just using a standard High Level language to do it, the main 2 contenders here are clearly C-Sharp and Java.

- Advantages

  The main advantage here is that by using a normal programming language, you have lots of flexibility. Because this is a normal programming language you are not bound by the paradigms of normal web development. You can effectively create a Desktop application and with very little modifications, it will be possible to turn it into a Web application. Another advantage is that it is very easy to incorporate previously coded modules into your web application.

  This also has the advantage that it suits my needs better as well. Since I have lots more experience with these types of languages, I would have much more time to spend on the actual web application rather than learning the language. This means I can get more done and have a better, more complete end result.

  This option is also new technology, not so new that there is no documentation and help but not so old there is the same amount of help that Option 1 provides. Since it is in this mid-stage area, it will only be used more and more so looking at it in terms of what industry might need in a few years, this would give me a good opportunity to learn a technology that will make me more employable than the average person.

- Disadvantages

  The main disadvantage of this option is that since it revolves around fairly new technology, it will not be streamlined to the point of Option 1. What I mean by this is that it may take more effort to create the same thing from Option 1 using this method. This method may provide the best solution, but it may take a far longer time to get to the 'best' solution when compared to a 'client ready' solution from Option 1. The solution from Option 1 may not be as good as this one, but on a scale of 'first conception' to 'ready to ship', this Option is not the one to choose.

1.4.4.3 Option 3: The Hybrid

The final option is a sort of combination of the other Options but with a sort of twist. It will be using the C++ language, which is by no means the go to for creating web applications. It also includes using a 3rd party library which will provide the necessary core web technology functionality.

- Advantages

The clear advantage of this option is that because it uses a 'mid' level language, that allows us to write very fast code. Rather than having to go through a framework and converting from source to bytecode to asm, we can go directly from the source to the asm. This is a very important factor as the the whole purpose of this project is to teach Cryptography and in a lot of situations, the key algorithms need to be fast. So, in that aspect there is no other option.

This option is also my preferred option as C++ is a pure back-end language. I also have the most experience with this language therefore I would be most comfortable working with it. This also removes the need to learn a completely new language to continue with the project. Thus eliminating a massive bottleneck in the process of development.

I would also consider this a very good option to make myself more employable in the future as lots of the older systems are written in C++ and in order to keep them running you need someone who is familiar with the language.

- Disadvantages

  The key disadvantage with this option is that it uses a not-so friendly language. Meaning that it takes much more effort to do the same task when compared to the other options. It also has the disadvantage that you need to learn a completely new library to do anything which can be very daunting to consider.

1.5 All Different Areas of Cryptography

General Overview

- Cryptography
  - Basic information about Cryptography
    - Private-Key Cryptography
    - Introduction to Public-Key Cryptography
      - Security Mechanisms
      - Authenticity of Public Keys
      - Important Public-Key Algorithms
      - Key Lengths and Security Levels
    - Explain the simple protocols
  - Overview of the types of Ciphers
    - Block Cipher
    - Stream Cipher
    - Encryption and Decryption using both types of ciphers
  - Overview of Hashing Functions

  - Brief History on Cryptography
    - Caesar Cipher
    - Vernam Machine

- Cryptanalysis
  - Very Basic overview of what Cryptanalysis involves
  - Brief History on Cryptanalysis

More Detailed look

- Cryptography
  - Public-Key Cryptography
    - Cryptosystems based of the Discrete Logarithm Problem
      - Diffie-Hellman Key Exchange
      - The Discrete Logarithm Problem
      - The Elgamel Encryption Scheme
    - RSA Cryptosystem
    - Eliptic Curve
    - Digital Signatures
      - RSA Signature Scheme
      - Elgamel Digital Signature Scheme
      - Digital Signature Scheme (DSA)
      - Elliptic Curve Digital Signature Scheme (ELDSC)
    - Key Establishment
    - Message Authentication Codes (MACs)
      - MACs from Hash Functions
      - MACS from Block Ciphers CBC-MAC
      - Galois Counter Message Authentication Code (GMAC)
  - More detailed look into the common algorithms
    - Block Ciphers
      - Data Encryption Standard (DES)
      - Advanced Encryption Standard (AES)
    - Talk about the modes of operation for Block Ciphers
      - Electronic Codebook Mode (ECB)
      - Cipher Block Chaining (CBC)
      - Output Feedback Mode (OFB)
      - Cipher Feedback Mode (CFB)
      - Counter Mode (CTR)
      - Galois Counter Mode (GCM)
    - Stream Ciphers

        - Stream ciphers based off Linear Feedback Shift Register (LFSR)
    * Hashing Functions
        - MD5
        - SHA-1
        - SHA-2
        - SHA-3
- Implementations of the AES Cipher
- Create theory questions that require the user to demonstrate their understanding of the various Cryptography sections

1.6 Requirements / Objectives / Limitations

This whole section is going to serve a check list of the requirements, objectives and limitations of my project. The requirements serve as a detailed check list for me to go over and use to see if my project has all the content I want it to have. The objectives serve as a more 'is my project complete' check list and that is what will be used when checking as to whether or not my project is complete per say. And finally, limitations are similar to objectives but are things that I would like to achieve but couldn't due a variety of reasons including but not limited to time constraints and complexity of the problem.

There are 2 main sections here relating to the User and System. Anything that relates the the User is about how the platform is designed and orientated around the user. When it comes to the System it is mainly about my implementation for the solution.

1.6.1 User Requirements

- Easy to read font
- Intuitive user interface
- Good flow through out the entirety of the website
- Content explained makes sense
- Content should be easy to follow

1.6.2 System Objectives

1.6.2.1 Platform Objectives

I have split up this section into 3 different areas, Website Management, Website Content, User Interaction to make it easier to understand. It is also organized from most important to least important for each section. E.g. the higher up in the list the more important the item is.

1. Website Management
   (a) Website that I can launch and connect to from my Computer
   (b) Buying a Domain for the Website
   (c) Renting a VM for my Website to live in so it can be accessible to anyone
   (d) Renting a SSL Certificate so I have HTTPS enabled for secure communication between the client and the server
2. Website Content
   (a) Create Navigation Grid that can link to all the different sections of the website
   (b) Create Content Template that allows me to easily add new content
   (c) Create Question Template that allows me to easily add new Questions
   (d) Create online code-editor with standard code-editor features. E.g. Auto-complete, re factoring, a home made 'intellisense'
   (e) Create questions that require the need to write code in order to solve them
   (f) Review the Coded solutions and suggest improvements
3. User Interaction
   (a) Login system is implemented
   (b) Authentication for users is implemented
   (c) Relevant user information is stored in local database
   (d) Email verification for accounts is implemented
   (e) Store answers to questions into local database
   (f) Analyze the users answers to questions and produce report on what they need to work on

1.6.2.2 AES Implementation Objectives

1. (a) Key Schedule Implemented
   (b) Sub Bytes Implemented
   (c) Shift Rows Implemented
   (d) Mix Columns Implemented
   (e) Add Round Key Implemented
2. Key sizes supported
   (a) 128-bit

(b) 192-bit
(c) 256-bit

3. Different Modes of Operation
   (a) ECB (Electronic Code Book
   (b) CBC (Cipher Block Chaining)
   (c) OFB (Output Feedback Mode)
   (d) CFB (Cipher Feedback Mode)
   (e) CTR (Counter Mode)
   (f) GCM (Galois Counter Mode)

1.6.3 Acceptable Limitations

1.6.3.1 Platform Limitations

1. Website Management
   (a) Limit to concurrent connections to the website
   (b) Making the website live to the public
   (c) Getting SSL/HTTPS setup for the website
2. Website Content
   (a) Only a few sections have Content and Questions
   (b) Online Code-editor does not support all the modern features
3. User Interaction
   (a) Analysis of user answers not fully finished

1.6.3.2 AES Implementation Limitations

1. Only 128-bit Key size is supported
2. Only configure ECB Mode of operation
3. Padding is not implemented

1.7 Data Usage in the System

1.7.1 Data Security Overview

This will be covered more in depth in the Documented Design section but I feel it is important to briefly go over it here.

So for all important user critical information stored in the local database, the appropriate security measures will be implemented. Since the only critical user information is the passwords to the users account, before storing the password it will be hashed using a secure hashing algorithm, such as bcrypt, SHA2, SHA3, etc. The reason we hash it rather than encrypt it is that a hash is known as a 1-way function. You can never get the plaintext back from the ciphertext, in this case our plaintext is the password and the ciphertext is the password hash or digest.

1.7.2 Data Sources

The main data sources for my project will be for the users answers to the questions on the various sections provided. I will also receive data from the user as an input for the AES algorithm. For the answers to the questions, the data will be processed and stored in a local database. The input into the AES algorithm will not be stored and will only be processed by the algorithm itself.

1.7.3 Data Destinations

The main data destination for my project will be the back end database in use. This is where the majority of the key information like the authentication information and user data will be stored. All appropriate security measures will also be taken to ensure confidentiality, integrity and availability of the data.

1.7.4 Data Analysis

1.7.4.1 User Data

All user data will be stored in the database table db_user. This data will only be able to be accessed through my systems database api.

1.7.4.2 System Data

All system data will stored in custom built data structures. Important information like authentication information will be stored in the following database tables:

- auth_information
- auth_identity
- auth_token

Information regarding question data will be stored in a table called questions.

1.7.4.3 Temporary Data

Temporary data will be stored in a data structure from the web application and then when it is no longer needed the data will be disposed of.

Chapter 2

Documented Design

2.1 High-Level Overview of System

2.2 Project Versioning

2.2.1 File Structure

This section will talk about the general file structure I will be using and the style that I will be coding against. We will be using an example file, file˙structure.cc, which is shown below.

```
1  /**
2   * @file file_structure.cc
3   * @date 21/02/2018
4   * @version 0.01
5   *
6   * @brief A brief explanation of what is contained in this file and what ↩
        the purpose of the file is
7   *
8   * @author Jacob Powell
9   */
10
11
12 /**
13  * @class A
14  * @version 0.01
15  *
16  * @brief A brief explanation of what the purpose of the class is and ↩
        what it does
17  */
18 class A
19 {
20
21 };
22
23 /**
24  * @class TestClass
25  * @version 0.01
```

```
26  * @implements TestClass
27  *
28  * @brief A brief explanation of what the purpose of the class is and ↩
        what it does
29  */
30 class B : A
31 {
32 public:
33     /**
34      * @brief A brief explanation of what the function does
35      *
36      * More detailed explanation below
37      */
38     TestClass() = default;
39
40     /**
41      * @brief A brief explanation of what the function does
42      *
43      * @param f A description of the parameters is given here
44      * @param e
45      * @return Describes what the function returns
46      *
47      * More detailed explanation below
48      */
49     int C(int f, bool e);
50
51 private:
52     int f; /**< Detailed information on a member of the class */
53
54 };
```

2.2.1.1 File Header

Firstly we will be looking at the generic file header for the *.h* and *.cc* files used in my project. The file header is given below:

```
1 /**
2  * @file file_structure.cc
3  * @date 21/02/2018
4  * @version 0.01
5  *
6  * @brief A brief explanation of what is contained in this file and what ↩
       the purpose of the file is
7  *
8  * @author Jacob Powell
9  */
```

There are 5 key sections included in the file header,

- @file - this declared the name of the file, including its extension
- @date - this is the date of creation of the file

- @version - this documents what the version of the file is. It will change when there are small changes and when there are large changes. The significance of the change is shown by which number is changed, e.g. changing from version 0.01 to 0.1 is a more significant change than going from version 0.01 to 0.02
- @brief - this is where a brief description of what the files purpose is and what it contains
- @author - this is where the author of the files name is given

The reason I am including a header block is because when browsing through my project most people will not be able to meticulously go through all the code working out what it does exactly. The header block serves for the purpose of explaining what the code in the file does, giving the reader any key information like version, date of creation and author as soon as they look at the file. This way the reader doesn't have to read all the code, they can just read the header block and they will understand, on a high level, what that section of the code does.

2.2.1.2 File Commenting

Throughout the project I will be sticking to a particular commenting style, which applies rules to how document my code. This section explains how I comment and what style I use for each scenario.

The first area we will look at is how we document classes, their members and their methods. We will look at the example class B and in our file˙structure.cc example file. The class extract is shown below.

Classes An extract of our example classes is given below:

```
1  /**
2   * @class A
3   * @version 0.01
4   *
5   * @brief A brief explanation of what the purpose of the class is and ↩
        what it does
6   */
7  class A
8  {
9
10 };
11
12 /**
13  * @class TestClass
14  * @version 0.01
15  * @implements TestClass
16  *
17  * @brief A brief explanation of what the purpose of the class is and ↩
        what it does
```

```
18  */
19 class B : A
20 {
21
22 };
```

A class will contain 4 sections:

- @class - The name of the class, the name should also always use Camel-Casing
- @version - The current version of the class
- @implements - If the class inherits from another class then the name of the inherited class is given here
- @brief - This provides a short explanation of what the class does and what its purpose is

Methods An extract of class method documentation is given below:

```
1  class B : A
2  {
3  public:
4      /**
5       * @brief A brief explanation of what the function does
6       *
7       * More detailed explanation below
8       */
9      TestClass() = default;
10
11     /**
12      * @brief A brief explanation of what the function does
13      *
14      * @param f A description of the parameters is given here
15      * @param e
16      * @return Describes what the function returns
17      *
18      * More detailed explanation below
19      */
20     int C(int f, bool e);
```

- @brief - Explains what the method does
- @param - If the method takes parameters then these will be declared and explained here
- @return - If the method returns something this explains what that is
- Then after these tags a more detailed explanation of what happens in the method will be given below

Members An extract of class member documentation is given below

```
class B : A
{
private:
    int f; /**< Detailed information on a member of the class */

};
```

- Next to the member use the style /**< */ to describe what its purpose in the class is

2.2.2 Version Changelog

Lets look at the file header again, it is given below:

```
/**
 * @file file_structure.cc
 * @date 21/02/2018
 * @version 0.01
 *
 * @brief A brief explanation of what is contained in this file and what ↩
    the purpose of the file is
 *
 * @author Jacob Powell
 */
```

As you can see there is a *@version* tag. For every data structure and file in the project there will the a *@version* tag which will contain the most up to date version of the file / data structure. This allows me to easily keep track of how each element of my project is progressing and evolving. The behavior of this tag is explained in *Section 2.2.1.1*.

2.2.3 File Directory Organization

The file organization for any project is very important as it helps keep not only the project organized but the thought process and progression of the project going strong. So, I have setup multiple sub directories and such to keep the file structure good and well. All of my source code is stored in a folder called src.

```
src/
├── auth/
├── crypto/
├── db/
│   └── sql-queries/
└── layout/
    └── header/
```

```
    ├── information_content/
    └── resource/
        ├── content_xmls/
        ├── css/
        ├── images/
        └── xml-templates/
```

auth This directory contains all the code required for the authentication system in my project. It handles the login and register form.

crypto This has the code for the implementation for the aes algorithm. It also contains tests for the algorithm so I can make sure the algorithm performs the same as what is specified in *FIPS 197*.

db This directory contains all files related to database management and session management. It contains all the table definitions for the Dbo module of Wt.

sql_queries This directory contains any sql queries that I might want to perform multiple times such delete˙all and select˙all operations.

layout This directory contains all files related to display layout and the functionality of how the user interacts with the website.

header The code for the header of the web page is located here.

information_content Here is where all the different sections of cryptography I am trying to teach will be located and loaded from.

resources This is where any non-code related resources will be stored.

content_xmls In this directory xml files will contain the specific information relating to the content that I am trying to teach.

images Any images that I need to use will be stored here.

xml_templates Any xml files that contain layout information are contained here.

2.2.4 Off-Site Backups

When developing a project of this scale it is crucial that you don't just *keep all your eggs in one basket* per say. What I mean by this is that you don't want a single point of failure, if I finished my project and then a week before the submission date my laptop died and I couldn't recover the data from my hard-drive then I would not be able to submit my project and I would fail.

For this reason I will be using Github to store daily backups of my projects code and documentation. This way, even if my laptop dies and I have no way of recovering the data I would have only lost a days worth of work, rather than a few months of work.

This will not actually affect the end result of my project but I do feel it is important to talk about it as it is probably the most important step of all since if something goes wrong, you can recover, if you don't you cant.

2.3 Bespoke Algorithm Design

2.3.1 Question/Answer Algorithm

Since one of the main aspects of this project is to teach a user a new concept and then get them to answer various questions about that topic. There are ? stages to this process, these processes are listen below.

- Stage 1:

  Get the answer selected by the user

- Stage 2:

  Send information based on the users answer to the Database API.

- Stage 3:

  The Database API will then load this information into the db˙user.questions record for the currently logged in user.

- Stage 4:

  The Web Application then queries the Database API for the answer of the question in the questions table.

- Stage 5:

  The Web Application then compares both answers to see if the user is correct.

- Stage 6:

  The Web Application then shows the user whether or not the question is right.

2.3.1.1 Algorithm Block Diagram

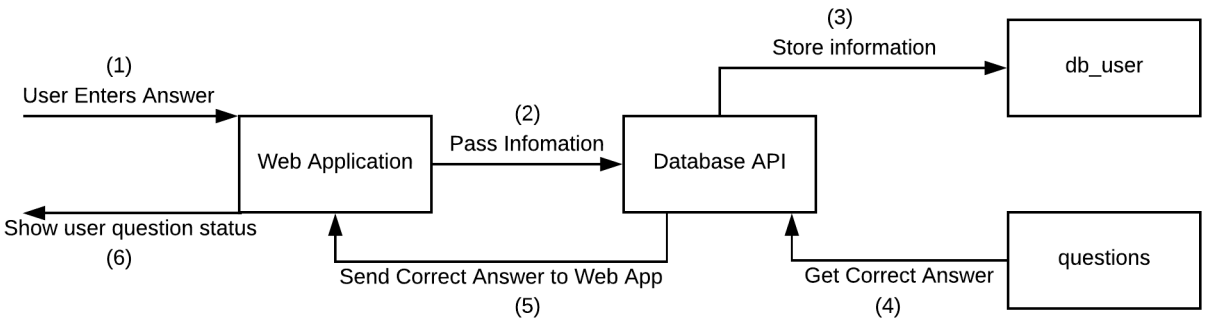


Figure 2.1: The Questions/Answer Algorithm Process

2.3.2 Cryptographic Algorithms

A saying in the cryptographic world is "*Never use your own Crypto!*". This is said for a few reasons but the main one being because it is proven that an algorithm that has been developed in secret and only a few people have seen will be far less secure that a algorithm that has been developed in the light and everyone can see. So for that reason alone I will not be designing any Cryptographic algorithms but rather sticking to use a very relevant algorithm, AES, and implement that instead. The documentation for AES is given in *Section 2.4.2* and for further information you should look up the *FIPS 197* standard.

2.4 Cryptographic Algorithm Design

2.4.1 Galois Fields for AES

In AES, every single operation that is used is based on Finite Fields. This section will give you a brief introduction to what Finite Fields are and how they are used.

2.4.1.1 Introduction to Finite Fields

It should be stated now that the term *Finite Field* means the same thing as the term *Galois Field*. In Abstract Algebra there are 3 basic structures, the group, the ring, and the finite field.

Groups Definition A group is a set of elements G together with an operation $\cdot$ which contains 2 elements of G. A group has the following properties:

1. The group operator $\cdot$ is closed. That is for all $a, b \in G$, it holds that $a \cdot b = c \in G$
2. The group operation is associative. That is, $a \cdot (b \cdot c) \equiv (a \cdot b) \cdot c$ for all $a, b, c \in G$
3. There is an identity element $1 \in G$ such that $a \cdot 1 = 1 \cdot a = a$ for $a \in G$
4. There is an inverse element for all elements in G. That is for $a \in G$, there must be a element $a^{-1} \in G$ such that $a \cdot a^{-1} = a^{-1} \cdot a = 1$
5. A group is Abelian (or commutative) if, $a, b \in G$, $a \cdot b \equiv b \cdot a$

Roughly speaking a group is set with one operation and its corresponding inverse operation. If this operation is addition then the inverse operation will be subtraction, if the operation is multiplication then the inverse will be division. If we need a structure that can hold all 4 operations, that is where we will use Finite Fields (Galois Fields). It should be noted that it is common to denote a Finite Field as FF and a Galois Field as GF.

Finite Field Definition A field F is a set of elements with the following properties:

1. All elements of F for an additive group with the group operation "$+$" and the identity element 0.
2. All elements of F, except the identity element 0, form a multiplicative group with the group operation "$\times$" and the identity element 1.
3. When the two group operations are mixed, the distributivity law holds, e.g. for all $a, b, c \in F : a(b + c) = (ab) + (ac)$

In cryptography we are almost always interested in fields with a finite number of elements which we intuitively name Finite Fields or Galois fields. The number of elements in the field is called the *order* or *cardinality* of the field. It should be noted that a field with order m only exists if m is a prime power, e.g. $m = p^n$ where n is a positive integer and p is a prime number. p is called the characteristic of the finite field. This implies that there are finite fields with 11 elements or 81 elements, since $81 = 3^4$ or with 256 elements (since $256 = 2^8$ and 2 is a prime). There is no finite fields with 12 elements since $12 = 2^2 \cdot 3$ and 12, meaning 12 is not a prime power.

Prime Fields The most intuitive examples of FF are fields of prime order, fields with $n = 1$. Elements of the $GF(p)$ can be represented by the integers $0, 1, ..., p - 1$. The two operations are modular integer addition and multiplication modulo p. This means that if we consider the integer ring $\mathbb{Z}_m$ and m happens to be prime then $\mathbb{Z}_m$ is not only a ring but also a finite field.

In order to do arithmetic in a prime field we have to follow the rules for integer rings: Addition and multiplication are done modulo p, the additive

inverse of any element a is given by $a + (-a) = 0 \bmod p$, and the multiplicative inverse of any non-zero element a is defined as $a \cdot a^{-1} = 1$

We can look at the example of the prime field $GF(2) = \{0, 1\}$, which is the smallest finite field that can exist. The additive and multiplicative tables for the GF are listed below.

+	0	1
0	0	1
1	1	0

Table 2.1: Additive Table for $GF(2)$

×	0	1
0	0	0
1	0	1

Table 2.2: Multiplicative Table for $GF(2)$

This is quite cool as it shows that addition in $GF(2)$, modulo 2 addition, is equivalent to and XOR Gate. It also shows us that multiplication in $GF(2)$ is equivalent to the logical AND Gate. This field, $GF(2)$ is very important for AES.

2.4.1.2 Extension Fields $GF(2^m)$

AES uses a FF of 256 elements and is denoted as $GF(2^8)$. This field was chosen as each of the elements in this field can be represented as exactly one byte. It becomes increasing important as the S-Box and MixColumn transformations treat every byte of the internal data path as an element of the field $GF(2^8)$ and manipulates the data by performing arithmetic in this finite field.

As we said before if the order of a FF is not prime, which is clearly the case here (2^8 is clearly not a prime number) the addition and multiplication operators can not be represented by addition and multiplication of integers modulo 2^8. Fields with $m > 1$ are called *extension fields*. In order for us to have the ability of working with these fields we need the following:

1. A different notation for the field elements
2. Different rules for when we perform arithmetic operations with the elements

So for the elements of these finite fields, we represent them as *polynomials* with coefficients and that when we compute with these we perform a certain type of *polynomial arithmetic*. The polynomials have a maximum degree of $m - 1$, so that there are m coefficients in total for every element. In the field $GF(2^8)$, which is the AES FF, each element $A \in GF(2^8)$ is represented as:

$$A(x) = a_7x^7 + a_6x^6 + ... + a_1x + a_0 \text{ where}$$
$$a_i \in GF(2) = \{0, 1\}$$

It is very important to realise that every element of $GF(2^8)$ can also be stored as an $8 - bit$ vector:

$$A = (a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$$

This means we do not have to store the factors x^7, x^5, etc as it is clear from the bit positions to which power x^i each coefficients belongs.

2.4.1.3 Addition and Subtraction in $GF(2^m)$

This is actually really easy as we just follow the basic polynomial rules of addition and subtraction, we just add or subtract coefficients with equal powers of x. This is mathematically shown below, Let $A(x), B(x) \in GF(2^m)$. The sum of the two elements is then computed as the following:

$$C(x) = A(x) + B(x) = \sum_{i=1}^{m-1} c_i x^i$$

$$\text{where } c_i \equiv a_i + b_i \bmod 2$$

and the difference between the 2 pairs is computed as the following:

$$C(x) = A(x) - B(x) = \sum_{i=1}^{m-1} c_i x^i$$

$$\text{where } c_i \equiv a_i - b_i \equiv a_i + b_i \bmod 2$$

Since we perform modulo 2 addition with the coefficients, addition and subtraction are the same thing. An example of this is given below:

$$A(x) = x^7 + x^6 + x^4 + 1$$
$$B(x) = x^4 + x^2 + 1$$
$$C(x) = A(x) + B(x) = x^7 + x^6 + x^2$$

Note if we were to work out the difference between the two polynomials $A(x)$ and $B(x)$ it would be the same as $C(x)$.

2.4.1.4 Multiplication in $GF(2^m)$

Multiplication in $GF(2^8)$ is the core operation in the MixColumn transformation in AES. Firstly, 2 elements of $GF(2^8)$ are multiplied using standard polynomial rules:

$$A(x) \cdot B(x) = (a_{m-1}x^{m-1} + ... + a_0) \cdot (b_{m-1}x^{-1} + ... + b_0)$$
$$C'(x) = c'_{2m-2}x^{2m-1} + ... + c'_0$$

Where:

$$c'_0 = a_0b_0 \bmod 2$$
$$c'_1 = a_0b_1 + a_1b_0 \bmod 2$$
$$\vdots$$
$$c'_{2m-2} = a_{m-1}b_{m-1} \bmod 2$$

Realize that all coefficients a_i, b_i and c_i are elements of $GF(2)$ and that coefficients arithmetic is performed in $GF(2)$. In general the product polynomial, $C(x)$, will have a degree higher than $m-1$ and will thus have to be reduced. The idea is similar to what we would do in prime fields: in $GF(p)$, we multiply the two integers, divide the result by a prime and consider only the remainder. In extension fields the product polynomial $C(x)$ is divided by a special polynomial and we consider only the remainder after the polynomial division. The special polynomials are called irreducible polynomials and we need them for the module reduction. These polynomials are roughly comparable to prime numbers, basically their only factors are 1 and the polynomial itself. A mathematical description of Extension Field multiplication is given below:

Let $A(x), B(x) \in GF(2^m)$ and let

$$P(x) \equiv \sum_{i=0}^{m} p_i x^i, p_i \in GF(2)$$

be an irreducible polynomial. Multiplication of the two elements $A(x), B(x)$ is given as

$$C(x) \equiv A(x) \cdot B(x) \bmod P(x)$$

This means that we need a irreducible polynomial $P(x)$, of degree m and with coefficients from $GF(2)$, for every field $GF(2^m)$. Not all polynomials are reducible much like not every number is a prime. We only need to know the irreducible polynomial for AES; it is given below

$$P(x) = x^8 + x^4 + x^3 + x + 1$$

This polynomial is a part of the specification for AES.

Putting this all together, lets say that we have the 2 polynomials $A(x)$ and $B(x)$ where $A(x) = x^3 + x^2 + 1$ and $B(x) = x^2 + x$ and we want to multiply them in the $GF(2^4)$. The irreducible polynomial for this GF is given as $P(x) = x^4 + x + 1$.

First we have to work out the plain polynomial multiplication,

$$C'(x) = A(x) \cdot B(x) = x^5 + x^3 + x^2 + x$$

We can now reduce $C'(x)$ using the standard polynomial division method, but it can sometimes be easier to reduce each of the leading terms x^4 and x^5 individually.

$$\begin{aligned} x^4 &= 1 \cdot P(x) + (x + 1) \\ x^4 &\equiv x + 1 \bmod P(x) \\ x^5 &\equiv x^2 + x \bmod P(x) \end{aligned}$$

Now we just substitute that back into our $C'(x)$ expression and we will get our result for the multiplication of $A(x)$ and $B(x)$

$$\begin{aligned} C(x) &\equiv x^5 + x^3 + x^2 + x \bmod P(x) \\ C(x) &\equiv (x^2 + x) + (x^3 + x^2 + x) = X63 \\ A(x) \cdot B(x) &\equiv x^3 \end{aligned}$$

We need to make sure we do not confuse multiplication in $GF(2^m)$ with integer multiplication, especially if we are concerned with software implementations of Galois fields.

2.4.1.5 Inversion in $GF(2^m)$

Inversion in $GF(2^8)$ is the core operation in the Byte Substitution Layer of AES which contains the S-Boxes. For a given FF $GF(2^m)$ and the corresponding irreducible polynomial $P(x)$, the inverse A^{-1} of a nonzero element $A \in GF(2^m)$ is defined as:

$$A^{-1} \cdot A(x) = 1 \bmod P(x)$$

For small fields, in practice fields with 2^16 or fewer elements, look-up tables which contain the precomputed inverses of all finite elements are often used. Figure 2.2 shows the multiplicative inverse for $GF(2^8)$ used in AES.

	Y															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	01	8D	F6	CB	52	7B	D1	E8	4F	29	C0	B0	E1	E5	C7
1	74	B4	AA	4B	99	2B	60	5F	58	3F	FD	CC	FF	40	EE	B2
2	3A	6E	5A	F1	55	4D	A8	C9	C1	0A	98	15	30	44	A2	C2
3	2C	45	92	6C	F3	39	66	42	F2	35	20	6F	77	BB	59	19
4	1D	FE	37	67	2D	31	F5	69	A7	64	AB	13	54	25	E9	09
5	ED	5C	05	CA	4C	24	87	BF	18	3E	22	F0	51	EC	61	17
6	16	5E	AF	D3	49	A6	36	43	F4	47	91	DF	33	93	21	3B
7	79	B7	97	85	10	B5	BA	3C	B6	70	D0	06	A1	FA	81	82
X 8	83	7E	7F	80	96	73	BE	56	9B	9E	95	D9	F7	02	B9	A4
9	DE	6A	32	6D	D8	8A	84	72	2A	14	9F	88	F9	DC	89	9A
A	FB	7C	2E	C3	8F	B8	65	48	26	C8	12	4A	CE	E7	D2	62
B	0C	E0	1F	EF	11	75	78	71	A5	8E	76	3D	BD	BC	86	57
C	0B	28	2F	A3	DA	D4	E4	0F	A9	27	53	04	1B	FC	AC	E6
D	7A	07	AE	63	C5	DB	E2	EA	94	8B	C4	D5	9D	F8	90	6B
E	B1	0D	D6	EB	C6	0E	CF	AD	08	4E	D7	E3	5D	50	1E	B3
F	5B	23	38	34	68	46	03	8C	DD	9C	7D	A0	CD	1A	41	1C

Figure 2.2: The multiplicative inverse table of $GF(2^8)$ for bytes xy used within the AES S-Box

As an alternative to using lookup tables, it is possible to compute the inverses of all the elements. The main algorithm uses for this task is the Extended Euclidean Algorithm (EEA).

2.4.2 AES Internals

2.4.2.1 Structure of AES

It should be noted that the 128-bit data path that runs through the algorithm is split up into 16-bytes, this is why many people would consider AES a byte oriented block cipher as it works in pure bytes. So our structure diagram from Figure 1.3 can be adapted to the structure shown in Figure 2.3. We also noted before that the algorithm operates on a 4×4 matrix of bytes, Figure 1.2, and this remains true so keep that in mind when performing any operations, this becomes especially important in the ShiftRows and MixColumns steps.

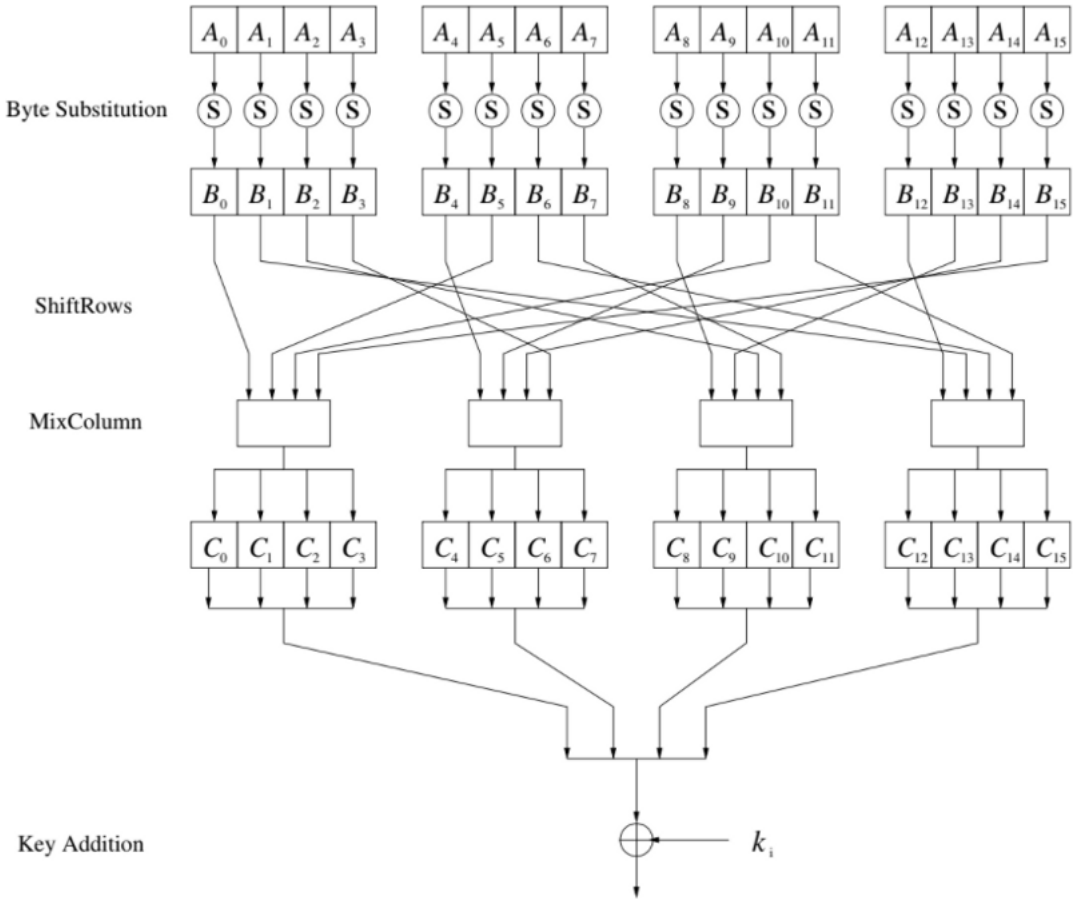


Figure 2.3: AES Byte Level Overview

The rest of this section will be looking into how the different components of each round work, these being the ByteSub Layer, ShiftRows Layer, MixColumns Layer and Key Addition Layer.

2.4.2.2 Byte Substitution Layer

The first layer in AES is the Byte Substitution Layer. There are 16 parallel S-Box which all take 8-bits as input and 8-bits as output. All 16 S-Box are identical with is different to DES as that cipher uses 8 unique S-Boxes. Each state Byte A_i is replaced, substituted, with another Byte, B_i. The process is shown below:

$$S(A_i) = B_i$$

The S-Box is the only non-linear section of AES. This basically means that

$$S(A) + S(B) \neq S(A + B) \text{ With 2 Byte States } A \text{ and } B.$$

The S-Box substitution is a 1-to-1 mapping, basically all of the $2^8 = 256$ elements are mapped to one output element. This is good as it allows us

to uniquely reverse the S-Box, which is key to the decryption process. In software, the S-Box is usually setup as a 256 Byte lookup table with fixed entries. The AES S-Box is given below for an input byte (YX), with (YX) representing its hexadecimal value in each of its respective columns.

	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xa	xb	xc	xd	xe	xf
0x	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1x	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2x	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3x	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4x	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5x	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6x	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7x	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8x	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9x	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
ax	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
bx	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
cx	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
dx	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
ex	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
fx	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figure 2.4: The AES S-Box

So lets look at an example, say we have the Byte State $A_i = (3b)_{hex}$, then we apply the S-Box and get the following result:

$$S(A_i) \equiv S((3b)_{hex}) \equiv (e2)_{hex}$$

If we were to look at this on a bit level, the point of interest for the cryptography of the cipher, then the substitution becomes:

$$S(A_i) \equiv S(00111011) \equiv (11100010)$$

Even though the S-Box has a 1-to-1 mapping, it does not have any fixed points. This means that there are no input values A_i that make $S(A_i) = A_i$. Even the zero-input state is not a fixed point, i.e $S((00)_{hex}) = S(00000000) = (01100011)$.

Mathematical Description of S-Box Unlike the DES S-Box, which are effectively random tables that fulfill certain properties (particularly to help prevent attacks based on differential cryptanalysis), the AES S-Boxes have very strong algebraic structure. The AES S-Box can be thought as a 2 step mathematical process.

$$A_i \longrightarrow GF^{-1}(2^8) \longrightarrow B_i' \longrightarrow \text{Affine Mapping} \longrightarrow B_i$$

The first part of the substitution is a Galois Field inversion. For each element of A_i, the inverse is computed, such that $B_i^{'} = A_i^{-1}$, where both A and B are elements of the field $GF(2^8)$ with the fixed irreducible polynomial $P(x) = x^8 + x^4 + x + 1$.

In the second part of the process, each byte $B_i^{'}$ is multiplied by a constant bit-matrix followed by the addition of a constant $8 - bit$ vector. This is shown below:

$$\begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \\ B_4 \\ B_5 \\ B_6 \\ B_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} B_0^{'} \\ B_1^{'} \\ B_2^{'} \\ B_3^{'} \\ B_4^{'} \\ B_5^{'} \\ B_6^{'} \\ B_7^{'} \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \mod 2$$

This entire process is known as affine mapping.

2.4.2.3 ShiftRows Layer

The ShiftRows Layer does exactly what its name implies. It cyclically shifts the rows of the byte matrix, see Figure 1.2. It shifts the first row by 0 bytes, the second row by 1 byte to the left, the third row 2 bytes to the left and the fourth row by 3 bytes to the left. If we take out original byte matrix, also shown in Figure 1.2, shown below:

b_0	b_4	b_8	b_{12}
b_1	b_5	b_9	b_{13}
b_2	b_6	b_{10}	b_{14}
b_3	b_7	b_{11}	b_{15}

After the ShiftRows Layer, the matrix would look like this, shown below.

b_0	b_4	b_8	b_{12}
b_5	b_9	b_{13}	b_1
b_{10}	b_{14}	b_2	b_6
b_{15}	b_3	b_7	b_{11}

2.4.2.4 MixColumns Layer

The MixColumns Layer is a linear transformation step which mixes the column of the state matrix. The combination of the ShiftRow and MixColumn layers makes it possible that after only three rounds, every byte of the state matrix depends on all 16 plaintext bytes. We denote the MixColumn layer with input matrix B and output matrix C as the following:

MixColumn(B) = C

Where B is the state matrix after the ShiftRows layer. Now we take each column as a vector and is then mulitplied by a fixed 4×4 matrix. This matrix is constant. Multiplication and addition of the coefficients is done in $GF(2^8)$. The computation of the first four output bytes is shown below.

$$\begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} B_0 \\ B_5 \\ B_{10} \\ B_{15} \end{pmatrix}$$

The second column of C, (C_4, C_5, C_6, C_7) is computed by multiplying the same constant matrix by (B_4, B_9, B_{14}, B_3), the third and fourth columns of C ad heed to this as well.

We know that each state byte, C_i and B_i are 8-bit values that represent a value in $GF(2^8)$. All arithmetic done is performed in this layer is formed in $GF(2^8)$. The constants in the matrix are hexadecimal numbers, representing the corresponding values in the Galois Field. So 01_{hex} is 00000001_{bin} which corresponds to the $GF(2^8)$ element 1, 02_{hex} is 00000010_{bin} corresponds to the $GF(2^8)$ polynomial x and 03_{hex} is 00000011_{bin} corresponds to the $GF(2^8)$ polynomial $x+1$.

The additions here are performed in $GF(2^8)$ so are simple XOR operations with the 2 different bytes. For the multiplications, the constants $01, 02, 03$ were chosen as they are very easy to set up in software. Multiplication with 01 is just multiplication by the identity which doesn't require a given operation. Multiplication by 02 and 03 are more complicated, therefore 2 256 byte lookup tables are used. You could do multiplication by 02 as a multiplication by x, which is just a left shift of 1, and then a modular reduction by the irreducible polynomial of $GF(2^8)$, $P(x) = x^8+x^4+x^3+x+1$. Multiplication by 03 is similar, it is just a left shift of 1 and then adding the original value followed by modular reduction of $P(x)$ again.

As an example, lets take the input state array, $B = (25, 25, ..., 25)$. Since we are only doing the first column this only involves 2 multiplications, by 02 and 03.

$$\begin{aligned} 02 \cdot 25 &= x \cdot (x^5 + x^2 + 1) \\ &= x^6 + x^3 + x \\ 03 \cdot 25 &= (x + 1) \cdot (x^5 + x^2 + 1) \\ &= (x^6 + x^3 + x) + (x^5 + x^2 + 1 \\ &= x^6 + x^5 + x^3 + x^2 + x + 1 \end{aligned}$$

Since the order of our intermediate values, $x^6 + x^3 + x$ and $x^6 + x^5 + x^3 + x^2 + x + 1$, do not have an order of above 8 we don't need to perform modular reduction with $P(x)$, the next step is to add together all the components of the multiplication.

$$\begin{aligned} 02 \cdot 25 = &\quad 1x^6 + 0x^5 + 0x^4 + 1x^3 + 0x^2 + 1x + 0 \\ 03 \cdot 25 = &\quad 1x^6 + 1x^5 + 0x^4 + 1x^3 + 1x^2 + 1x + 1 \\ 01 \cdot 25 = &\quad 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 0x + 1 \\ 01 \cdot 25 = &\quad 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 0x + 1 \end{aligned}$$

$$C_i = \quad 0x^6 + 1x^5 + 0x^4 + 0x^3 + 1x^2 + 0x + 1$$

This is where $i = 0, 1, 2, ..., 15$. This means our output state is the value of the $GF(2^8)$ polynomial $x^5 + x^2 + 1$ which is 25. So $C = (25, 25, ..., 25)$.

2.4.2.5 Key Addition Layer

The key addition layer takes 2 inputs, the current 16-byte state matrix and a sub key which is also 16-bytes. Then the 2 inputs are combined using the XOR operation. The particular sub keys that are derived are explained in the next section, *Section 2.4.2.6*.

2.4.2.6 AES Key Schedule

The Key Schedule takes our original 128/192/256 bit key and will then derive the subkeys needed for the algorithm. At the beginning and end of AES a XOR addition is applied to the subkey, this process is sometimes referred to as key-whitening. The number of subkeys required is the number of rounds plus one as we need an extra subkey for the key-whitening. A table with the key lengths, round numbers, and subkeys is shown below.

$N_k(bits)$	N_r	N_{sk}
128	10	11
192	12	13
256	14	15

Table 2.3: Respective Key Lengths, Rounds Numbers, Subkey Numbers

The AES Key Schedule is word oriented, where 1 word = 32-bits. Sub keys are stored in an expansion array called W that consists of words. There are 3 different key schedules for the 3 different key sizes.

128-bit Key Schedule The 11 subkeys are computed and stored in an array with elements $W[0], W[1], ..., W[43]$.. Figure 2.5 shows exactly how the subkeys for AES with a 128-bit key are computed.

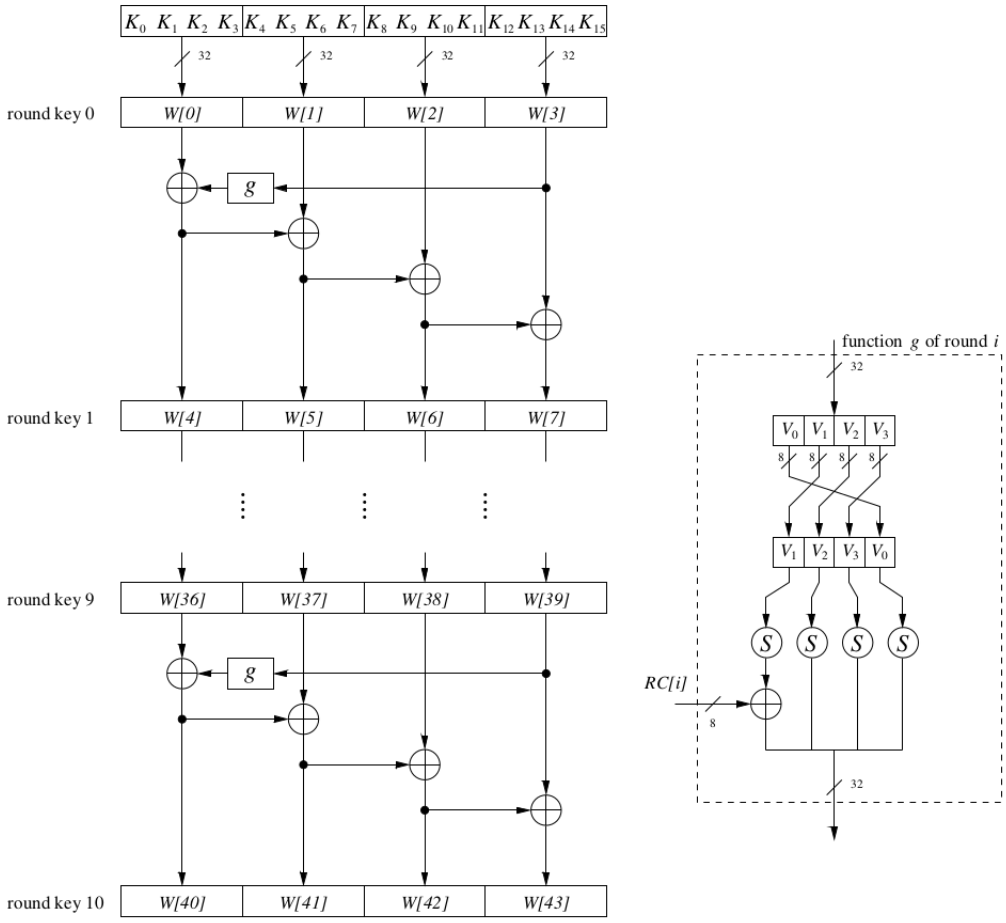


Figure 2.5: 128-bit Key Schedule for AES

The first subkey, k_0, is just the original key used when initializing the algorithm. So elements $W[0], .., W[3]$ contain our original $128 - bit$ key. Then, the left most word of a subkey $W[4i]$, where $i = 1, 2, ..., 10$, is computed using the following formula:

$$W[4i] = W[4(i-1)] \oplus g(W[4i-1])$$

With the function g here being a non-linear function with a $4 - byte$ input and a $4 - byte$ output. The remaining elements of W are calculated using the following formula:

$$W[4i+j] = W[4i+j-1] \oplus W[4(i-1)+j]$$
Where $i = 1, 2, 3, ..., 10$ and $j = 1, 2, 3$.

The function $g()$ rotated our $4 - byte$ input, it applies a S-Box substitution and adds a round coefficient, RC, to it. $RC's$ value is an element from $GF(2^8)$,

i.e. an 8-bit value. This value is only added to the left most byte in the function $g()$. The RC vary from round to round according to the following rule:

$$\begin{aligned} RC[1] &= x^0 = (00000001)_2, \\ RC[2] &= x^1 = (00000010)_2, \\ RC[3] &= x^2 = (00000100)_2, \\ &\vdots \\ RC[10] &= x^9 = (00110110)_2, \end{aligned}$$

This function $g()$ has 2 purposes, its first is to add non-linearity to the key schedule and the second is to remove symmetry from the AES cipher. Both of these are required as it protects from some attacks against block ciphers.

192-bit Key Schedule With a 192-bit key, we need 13 subkeys, which means our array W will have 52 words. The way we compute all of the 13 words, and thus the 52 words, is very similar to the way we compute the subkeys for a 128-bit key, this is shown is Figure 2.6. There are 8 iterations of the Key Schedule, where each iteration computes 6 new words for the subkey array W. We need 8 round coefficients, $RC[i]$, for use in our $g()$ function. These round constants are computed the same way as in a 128-bit key case.

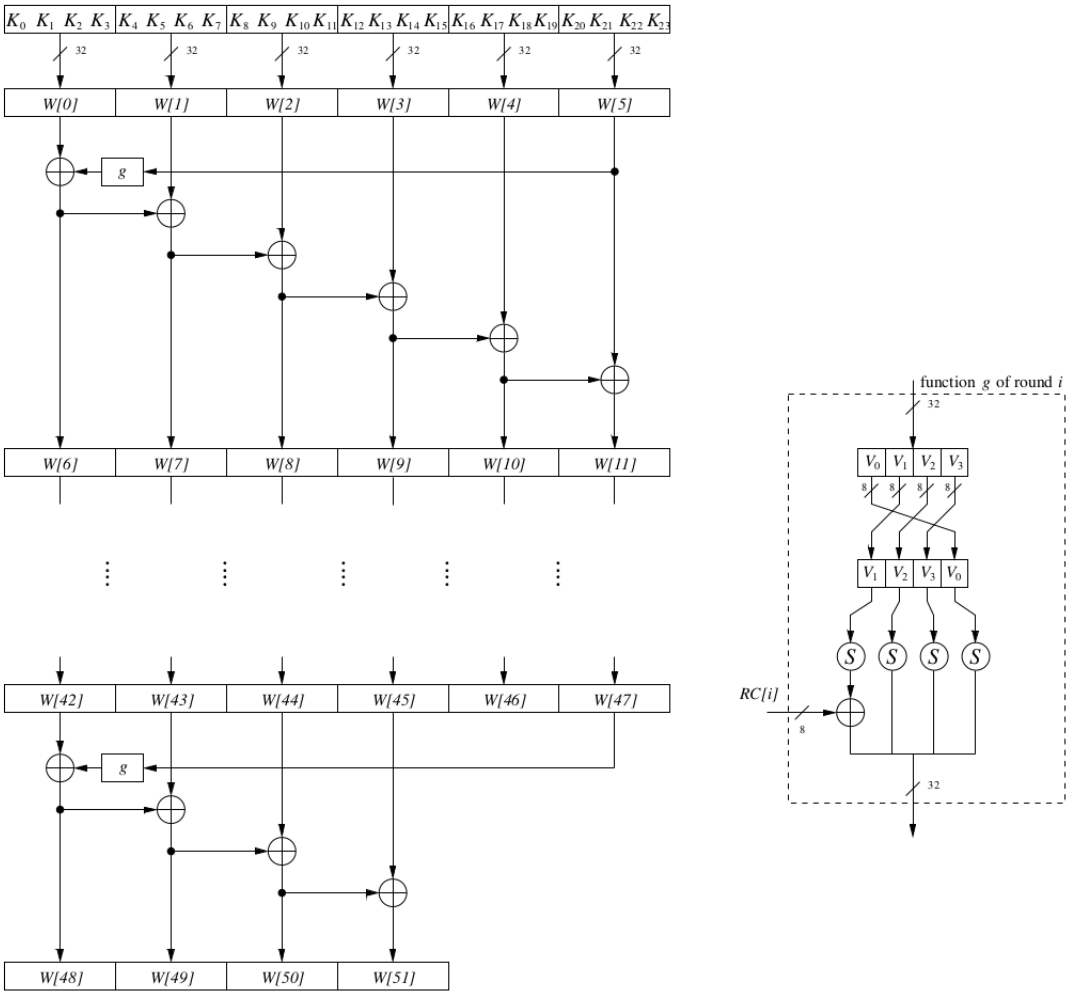


Figure 2.6: 192-bit Key Schedule for AES

256-bit Key Schedule With a 256-bit key. we need 15 subkeys, meaning our array W will contain 60 words. The way we compute all of the 13 words, and thus the 52 words, is very similar to the way we compute the subkeys for a 128-bit key, this is shown is Figure 2.7. There are 7 iterations of the Key Schedule, where each iteration computes 8 new words for the subkey array W. We need 7 round coefficients, $RC[i]$, for use in our $g()$ function. These round constants are computed the same way as in a 128-bit key case. The key schedule also introduces a new function $h()$, which takes a 4-byte input and output. $h()$ applies the S-Box to each input byte.

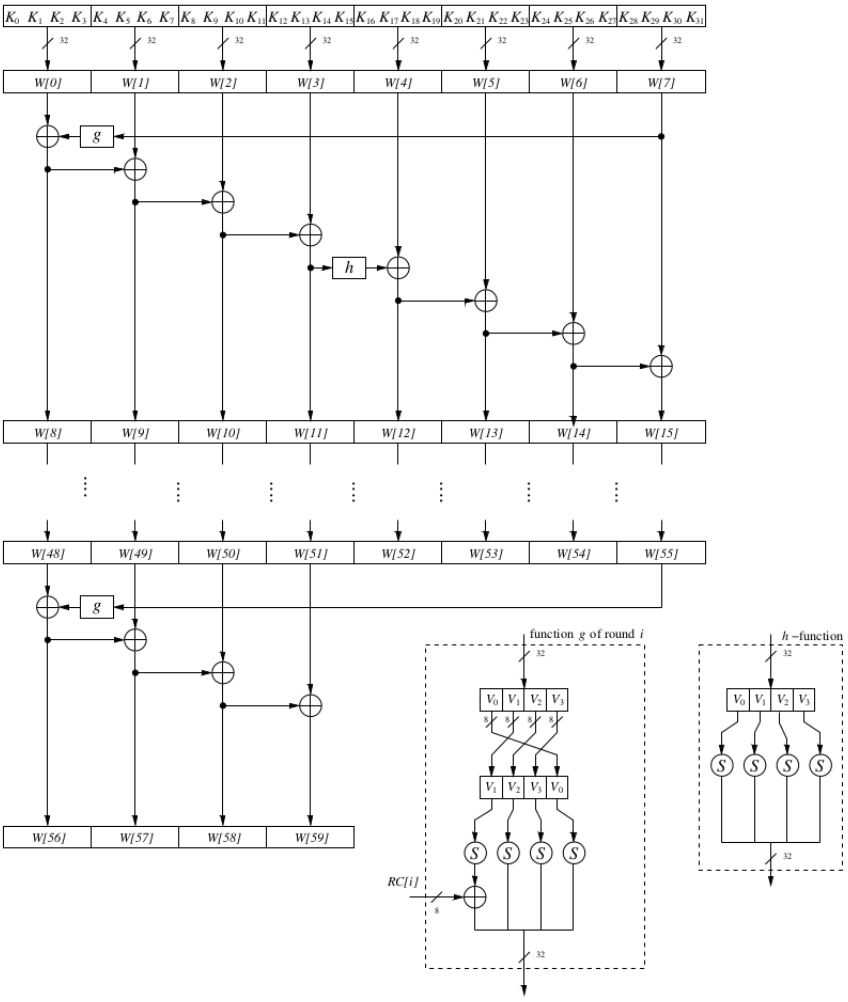


Figure 2.7: 256-bit Key Schedule for AES

2.4.3 Decryption in AES

Unlike DES, because AES is not based on a feistal network each layer needs to be inverted in order to decrypt a message that was encrypted using AES. This means that the S-Box, ShiftRows, MixColumns need to be inverted. For the AddRoundKey, this words slightly differently as for this all we need to do is reverse the order of the key schedule. So for the first round we use the last subkey, the second round we use the second-last subkey. The diagram below, Figure 2.8, illustrated this process.

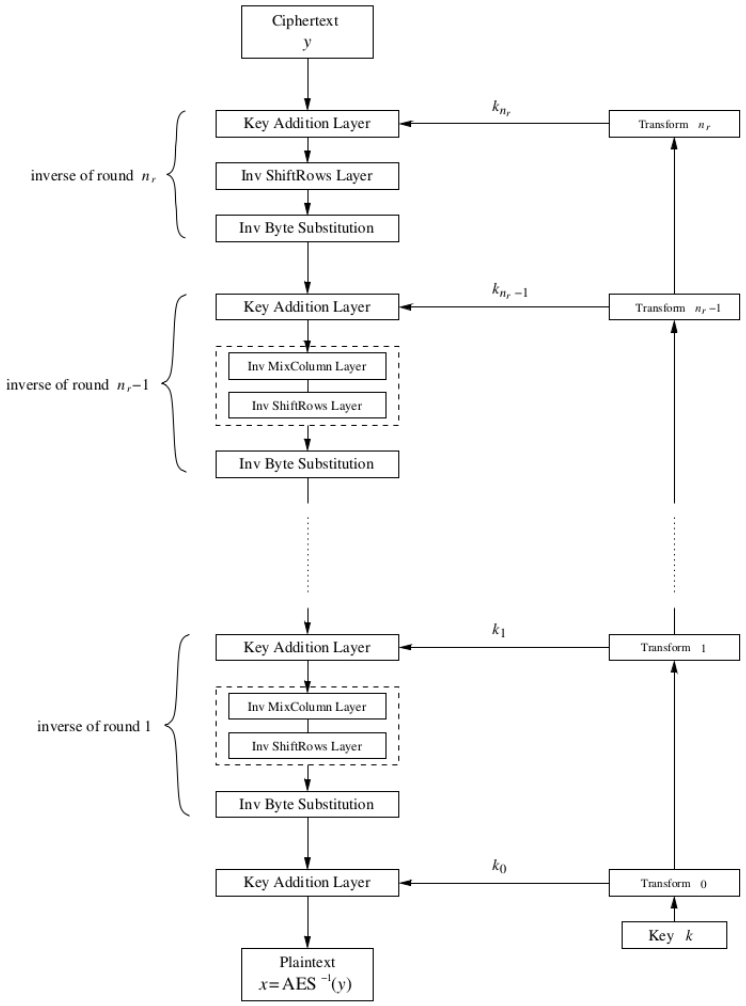


Figure 2.8: The Decryption process for AES

2.4.3.1 Inverse MixColumns Layer

After the addition of the Sub key, the inverse MixColumns step is applied to the state matrix, the exception is on the first decryption round where MixColumns is not applied. We inverse the MixColumn operation by just using the inverse matrix of the MixColumn operation. We have our 4×4 input state (C_0, C_1, C_2, C_3) and we just multiply this by the Inverse Matrix to get our output state (B_0, B_1, B_2, B_3).

$$\begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix}$$

Then in order to calculate (B_4, B_5, B_6, B_7) we just use the input (C_4, C_5, C_6, C_7). B_i and C_i are elements from $GF(2^8)$ and also the constants in our matrix are

elements of $GF(2^8)$. It should also be said that additions in the vector-matrix multiplication are bitwise XOR's.

2.4.3.2 Inverse ShiftRows Layer

In order to reverse the Shift Rows Layer all we have to do is shift the elements in the state matrix the opposite direction. The first row is not affected as it is also not affected in the normal ShiftRows layer. With our state matrix $B = (B_0, B_1, B_2, B_3, ..., B_15)$:

b_0	b_4	b_8	b_{12}
b_1	b_5	b_9	b_{13}
b_2	b_6	b_{10}	b_{14}
b_3	b_7	b_{11}	b_{15}

The Inverse Shift Rows operation would give us the following state matrix:

b_0	b_4	b_8	b_{12}
b_{13}	b_1	b_5	b_9
b_{10}	b_{14}	b_2	b_6
b_7	b_{11}	b_{15}	b_3

2.4.3.3 Inverse ByteSubstitution Layer

The inverse S-Box is applied when decrypting with AES. Since the AES S-Box is bijective, one-to-one mapping, it is possible to create an inverse S-Box such that:

$$A_i = S^{-1}(B_i) = S^{-1}(S(A_i))$$

The values given in the Inverse S-Box are given in Figure 2.9.

		y														
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Figure 2.9: The AES Inverse S-Box

2.4.3.4 Decryption Key Schedule

Since the first round of the decryption round needs the last subkey, the second round of decryption needs the second-last subkey, etc, etc. So we need to supply the subkeys in reverse order for the decryption rounds. This can generally be done by computing the 11 or 13 or 15 subkeys and storing them. This adds some latency to the decryption of the algorithm when compared to the encryption rounds.

2.5 Project Prototypes

2.5.1 Question/Answer Algorithm

Psuedocode for the Question/Answer Algorithm
QuestionAnswer(string question_id, string question_answer)
begin

$dbAPI \leftarrow DatabaseAPI(question\_id, question\_answer)$

$dbAPI.StoreData()$
$dbAPI.GetCorrectAnswer()$
$answer\_result \leftarrow dbAPI.IsAnswerCorrect()$

if $answer\_result$ is true **then**
    Show User Answer Is Correct
else if $answer\_result$ is false **then**
    Show User Answer is Incorrect
else
    Catch Any Errors
end if

end

2.5.2 AES Algorithm

2.5.2.1 Cipher (ENCRYPT)

Psuedocode for the AES Cipher
Cipher(byte input[4*Nb], byte output[4*Nb], word w[Nb*(Nr+1)])
begin
$state \leftarrow byte[4, Nb]$
$state \leftarrow in$

$AddRoundKey(state, w[0, Nb - 1])$
for $round \leftarrow 1$ **to** $Nr - 1$ **do**
    $SubBytes(state)$
    $ShiftRows(state)$
    $MixColumns(state)$
    $AddRoundKey(state, w[round * Nb, (round + 1) * Nb - 1])$
end for

$SubBytes(state)$
$ShiftRows(state)$
$AddRoundKey(state, w[Nr * Nb, (Nr + 1) * Nb - 1])$

$output \leftarrow state$

end

2.5.2.2 Inverse Cipher (DECRYPT)

Psuedocode for the Inverse AES Cipher
InvCipher(byte input[4*Nb], byte output[4*Nb], word w[Nb*(Nr+1)])
begin
$state \leftarrow byte[4, Nb]$
$state \leftarrow in$

$AddRoundKey(state, w[Nr * Nb, (Nr + 1) * (Nb - 1)]$
for $round \leftarrow (Nr - 1)$ **to** 1 **do**
    $InvSubBytes(state)$
    $InvShiftRows(state)$
    $AddRoundKey(state, w[round * Nb, (round + 1) * Nb - 1])$
    $InvMixColumns(state)$
end for

$SubBytes(state)$
$ShiftRows(state)$
$AddRoundKey(state, w[0, Nb - 1)$

$output \leftarrow state$

end

2.6 Bespoke Database Design

2.6.1 User Details

db_user	[table]
id	INTEGER
	auto-incremented
version	INTEGER NOT NULL
user_id	INTEGER NOT NULL
user_role	INTEGER NOT NULL
user_identity	TEXT NOT NULL

Figure 2.10: db_user Database Table

2.6.2 User Answered Questions

user_answered_questions	[table]
id	INTEGER
	auto-incremented
version	INTEGER NOT NULL
user_answered_question_id	INTEGER NOT NULL
user_answered_answer_id	INTEGER NOT NULL
user_answered_question_text	TEXT NOT NULL
user_answered_answer_text	TEXT NOT NULL
user_id	BIGINT

Figure 2.11: user_answered_questions Database Table

2.6.3 Authentication Information

auth_info	[table]
id	INTEGER
	auto-incremented
version	INTEGER NOT NULL
user_id	BIGINT
password_hash	VARCHAR(100) NOT NULL
password_method	VARCHAR(20) NOT NULL
password_salt	VARCHAR(20) NOT NULL
status	INTEGER NOT NULL
failed_login_attempts	INTEGER NOT NULL
last_login_attempt	TEXT
email	VARCHAR(256) NOT NULL
unverified_email	VARCHAR(256) NOT NULL
email_token	VARCHAR(64) NOT NULL
email_token_expires	TEXT
email_token_role	INTEGER NOT NULL

Figure 2.12: auth_information Table

2.6.4 Authentication Identity

auth_identity	[table]
id	INTEGER
	auto-incremented
version	INTEGER NOT NULL
auth_info_id	BIGINT
provider	VARCHAR(64) NOT NULL
"identity"	VARCHAR(512) NOT NULL

Figure 2.13: auth_identity_table Table

2.6.5 Authentication Token

auth_token	[table]
id	INTEGER
	auto-incremented
version	INTEGER NOT NULL
auth_info_id	BIGINT
"value"	VARCHAR(64) NOT NULL
expires	TEXT

Figure 2.14: auth_token Table

2.6.6 Question Details

questions	[table]
id	INTEGER
	auto-incremented
version	INTEGER NOT NULL
question_text	TEXT NOT NULL
question_answer	TEXT NOT NULL
question_option_a	TEXT NOT NULL
question_option_b	TEXT NOT NULL
question_option_c	TEXT NOT NULL

Figure 2.15: questions Table

2.6.7 Database Schema Diagram

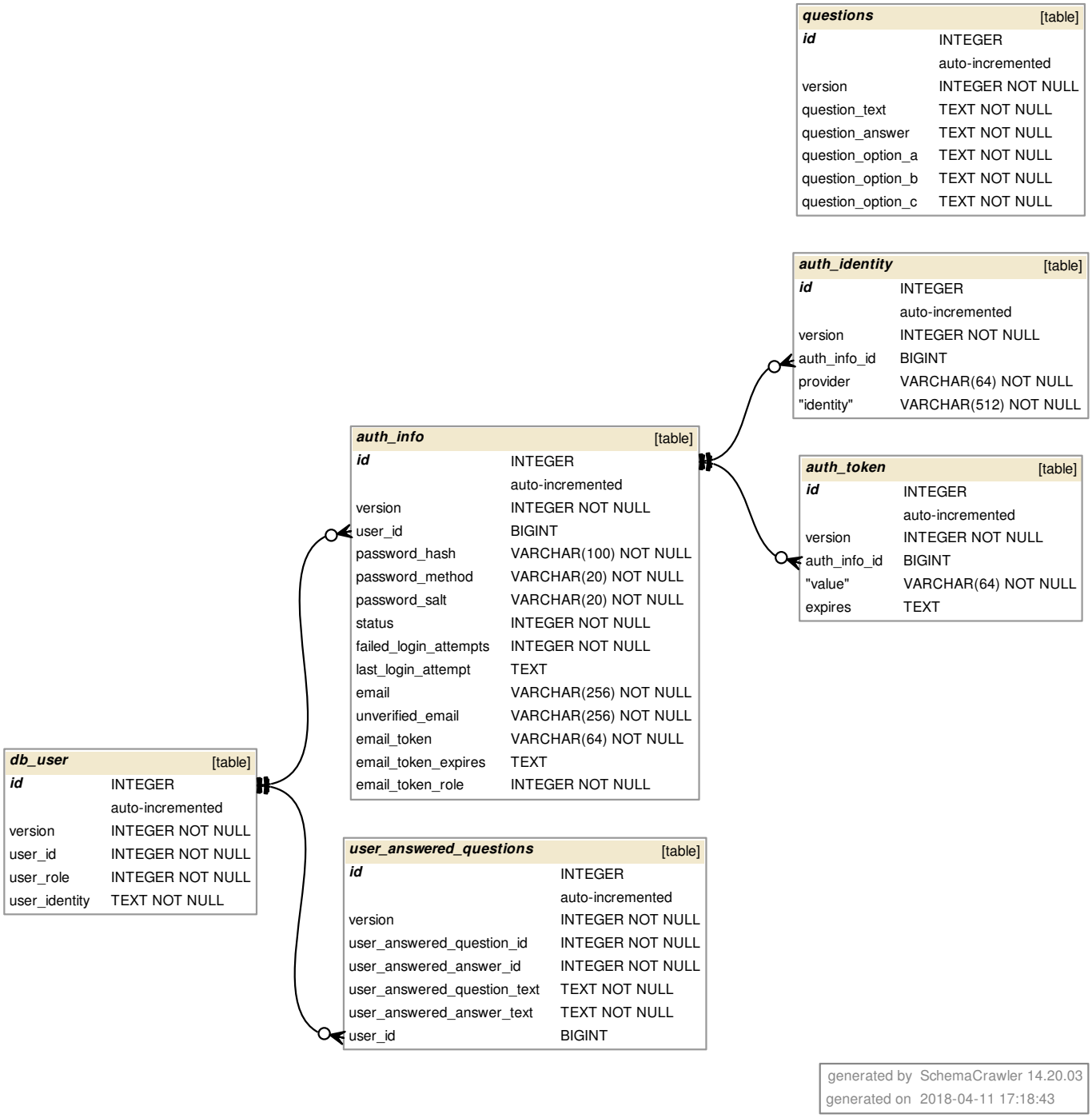


Figure 2.16: data.db schema

2.7 User Interface Design

2.7.1 Header - User not Logged In

2.7.1.1 Left Side of Header

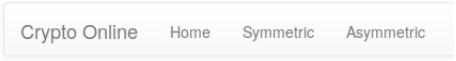


Figure 2.17: Left Side of the Website Header

2.7.1.2 Right Side of Header

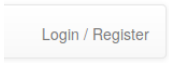


Figure 2.18: Right Side of the Website Header

2.7.2 Header - User Logged In

2.7.2.1 Left Side of Header for a Normal User

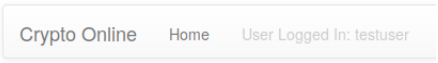


Figure 2.19: Left Side of the Website Header when a Normal User is logged in

2.7.2.2 Left Side of Header for a Admin User

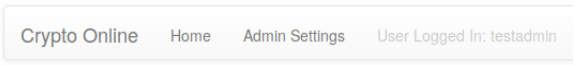


Figure 2.20: Left Side of the Website Header when an Admin User is logged in

2.7.2.3 Right Side of Header

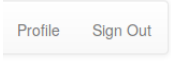


Figure 2.21: Right Side of the Website Header when a User is logged in

2.7.3 Navigation Grid

2.7.3.1 Cryptography Basics

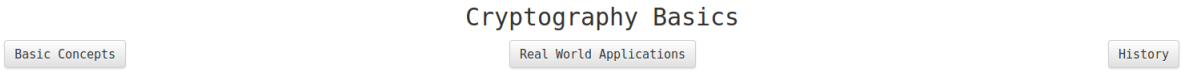


Figure 2.22: Cryptography Basics

2.7.3.2 Symmetric Cryptography

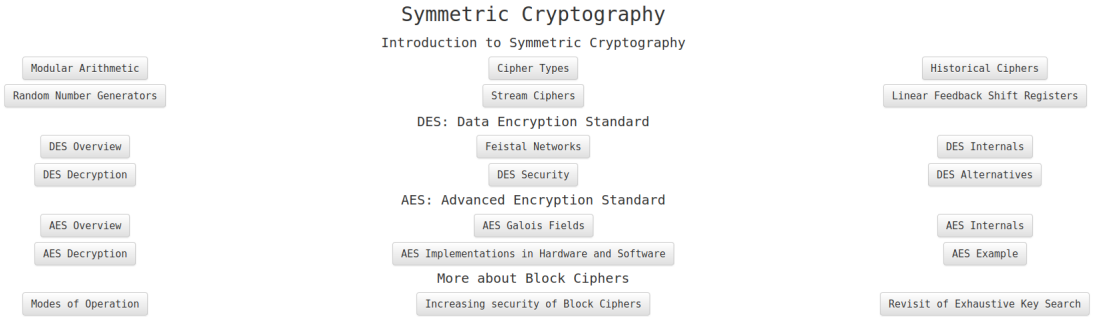


Figure 2.23: Symmetric Cryptography

2.7.3.3 Asymmetric Cryptography

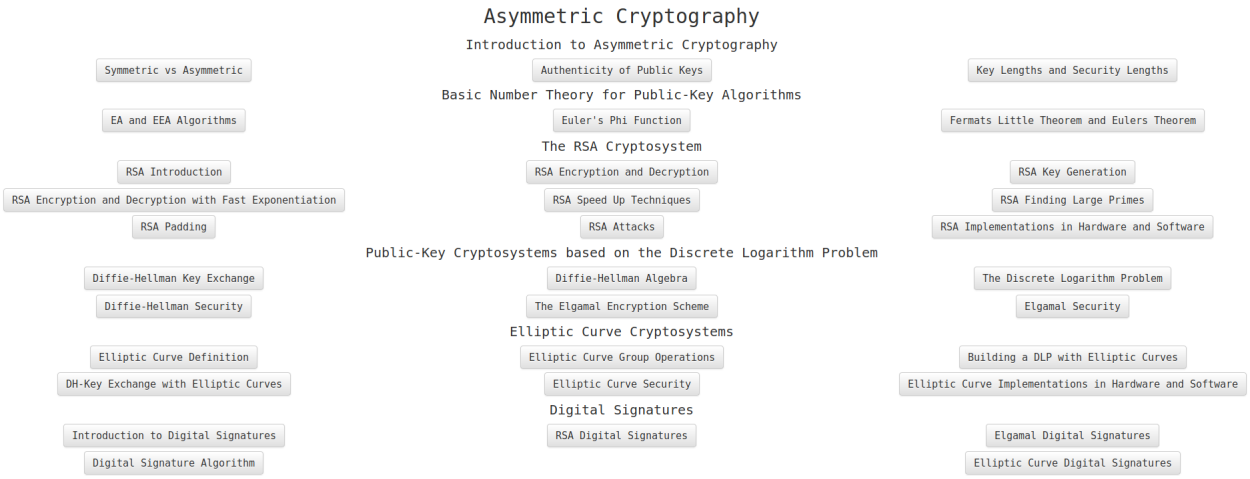


Figure 2.24: Asymmetric Cryptography

2.7.3.4 Protocols

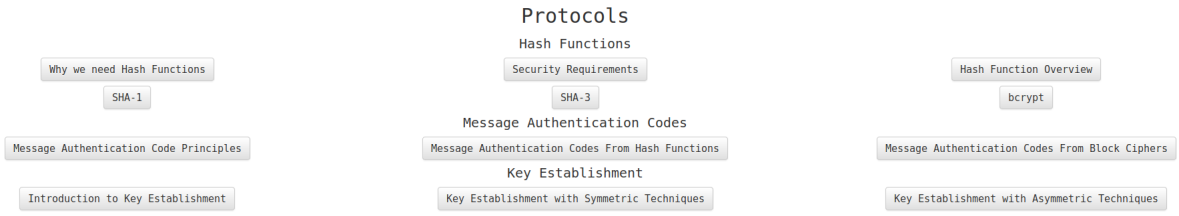


Figure 2.25: Protocols

2.7.4 Information Content

2.7.5 Question Content

2.7.6 Login Form

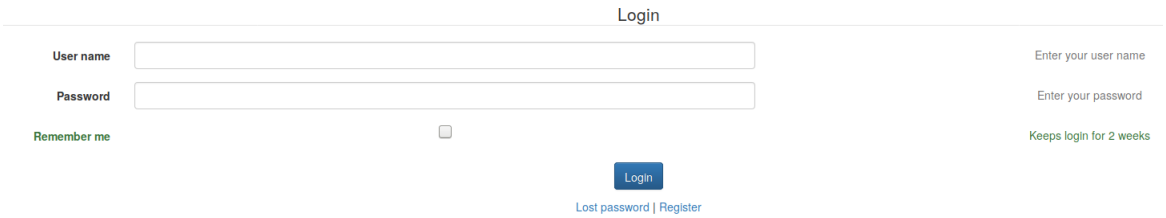


Figure 2.26: Login Form

2.7.7 Register Form

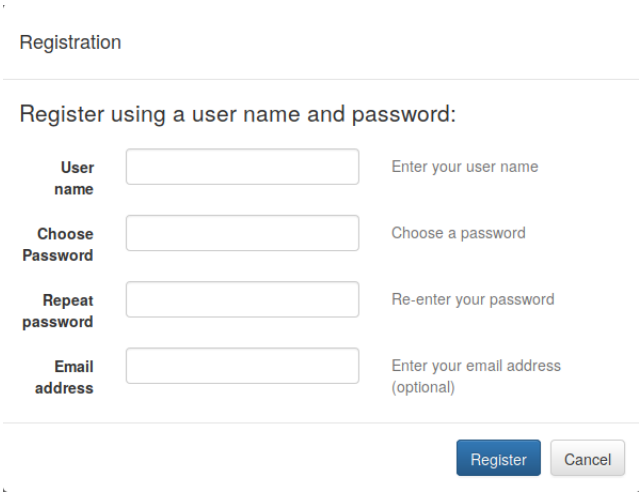


Figure 2.27: Register Form

2.8 System Security

Since my project requires users to sign up that means they need to create a password. That means I need to ensure that all users passwords are secured properly so if at any point my database gets leaked onto the Internet an attacker can get access to all my users accounts and more importantly user account data.

So all passwords are hashed using the bcrypt hash function. There is normally a lot of debate on which hash function to use but bcrypt has been around for over 10 years and no critical security flaws in its design and implementation have been found making a confident choice for me.

For the passwords as well, I have set conditions for password strength to make sure that the password a user chooses is suitable. These conditions mean that a password needs upper/lower case letters, number and special characters to be valid.

The login form also has something called rate-limiting. This means that an attacker cannot try to brute-force a users password as it will add a delay to entering the password after 3 failed attempts. This delay increases after every failed password attempt from here making it practically impossible for an attacker to brute force the password.

2.9 External Library's

2.9.1 Wt

Wt, said *Witty*, is a Web framework for the C++ programming language. It provides the basic widgets and building blocks needed to create A Web Application but also has a lot of other really useful features. This include but are not limited to:

- Client Side Optimized

  By using a signal-slot system, we don't have to worry about dealing with the AJAX requests. We simply connect a widget to the callback function of the server and its done. It also will use whatever technology is available for use in the communication, normally AJAX or WebSockets, but if will fall back into full html page reloads if javascript is not available. This allows a Wt Application to be accessible to any browser

- Built-In Security

  By default, Wt only allows the visible and enable widgets to be interacted with. It also helps prevent CSRF attacks by not storing any of the session information in cookies. Because of the Widget abstraction it discourages the use of raw html in the application which helps to prevent again XSS attacks. SQL Injection attacks are stopped by encouraging the user to use prepared SQL statements when accessing the database. Wt also includes an authentication and registration system with support for OAuth providers like Google and Facebook

- Object Relational Mapping Library

  This maps C++ classes to tables in my database using Wt::Dbo, a ORM that only requires pure C++. It does not depend on preprocessor magic or code generation.

Chapter 3

Technical Solution

3.1 src/

3.1.1 main.cc

```
1  /**
2   * @file main.cc
3   * @date 14/12/2017
4   *
5   * @brief This is the file from which the deployment of the website is ↩
        launched.
6   *
7   * @version 0.01
8   * @author Jacob Powell
9   */
10
11 #include "crypto_online_launcher.h"
12
13
14 using namespace Wt;
15
16 /**
17  * @brief This method returns a live instance of the web application
18  *
19  * @param env The current environment that the application is running ↩
        from.
20  * @return An instance of the starter class which builds the initial ↩
        running of the web app
21  */
22 std::unique_ptr<WApplication> createApplication(const WEnvironment& env)
23 {
24     return cpp14::make_unique<CryptoOnlineLauncher>(env);
25 }
26
27 /**
28  * @brief The main function for the project.
29  * It created the server instance and configures it for the requirements ↩
        of the project. It also handles any errors
```

```
30  * that may stop the flow of the program.
31  *
32  * @param argc The number of command line arguments
33  * @param argv The command line arguments that set the http listen ip and↩
        the port the server will run on. It also sets
34  * the current working directory and the links to the necessary ↩
        directories that contain the style sheets and resources.
35  * @return
36  */
37 int main(int argc, char **argv)
38 {
39     try {
40         WServer server(argc, argv, WTHTTP_CONFIGURATION);
41 
42         server.addEntryPoint(EntryPointType::Application, ↩
               createApplication);
43 
44         Session::configureAuth();
45 
46         if(server.start()){
47             Wt::WServer::waitForShutdown();
48             server.stop();
49         }
50 
51     } catch (WServer::Exception& e) {
52         std::cerr << e.what() << std::endl;
53     } catch (std::exception &e) {
54         std::cerr << "exception: " << e.what() << std::endl;
55     }
56 }
```

3.1.2 crypto_online_launcher.h

```
/**
 * @file crypto_online_launcher.h
 * @date 30/01/2018
 *
 * @file Contains Class Definitions for the CryptoOnlineLauncher class
 * @version 0.01
 * @author Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_LAUNCHER_H
#define CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_LAUNCHER_H

#include "db/session.h"

#include <Wt/WApplication.h>
#include <Wt/WBootstrapTheme.h>
#include <Wt/WContainerWidget.h>
#include <Wt/WServer.h>
#include <Wt/WEnvironment.h>

#include <Wt/Auth/AuthWidget.h>
#include <Wt/Auth/PasswordService.h>

/**
 * @class CryptoOnlineLauncher
 *
 * @brief This class is responsible for how the web application is setup and what the settings and configuration
 * options are set.
 */
class CryptoOnlineLauncher : public Wt::WApplication {
public:
    explicit CryptoOnlineLauncher(const Wt::WEnvironment &environment); /**< Constructor for the CryptoOnlineLauncher Class*/
    void auth_event(); /**< Method that handles any authentication signals */

private:
    void load_css_files(); /**< Loads the required style files */
    void load_message_bundles(); /**< Loads the required message bundles */

    Session _session; /**< Instance that holds the current session */
};


#endif //CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_LAUNCHER_H
```

3.1.3 crypto_online_launcher.cc

```
/**
 * @file crypto_online_launcher.cc
 * @date 30/01/2018
 *
 * @brief Method definitions for the CryptoOnlineLauncher class
 * @version 0.01
 * @author Jacob Powell
 */

#include "crypto_online_launcher.h"
#include "layout/crypto_online_home.h"


#include <Wt/WCssTheme.h>

/**
 * @brief Constructor for the CryptoOnlineLauncher
 * It loads the initial state including the theme, css files and message bundles
 *
 * @param environment current enviroment that the application is loaded with
 */
CryptoOnlineLauncher::CryptoOnlineLauncher(const Wt::WEnvironment &environment)
        : WApplication(environment), _session(appRoot() + "data.db") {
    /**
     * This section gets the current enviroment theme for the project. Then checks if the theme is set to bootstrap3,
     * if it is then it sets the project theme to bootstrap3. If the environment theme is not set to bootstrap3 then
     * It check if the theme is set to bootstrap2, if it is it then makes that the theme. If the environment theme is
     * neither then it just creates a shared WCssTheme and sets that to the theme.
     */
    const std::string *themePtr = environment.getParameter("theme");
    std::string theme;
    if (!themePtr)
        theme = "bootstrap3";
    else
        theme = *themePtr;

    if (theme == "bootstrap3") {
        auto bootstrapTheme = std::make_shared<Wt::WBootstrapTheme>();
        bootstrapTheme->setVersion(Wt::BootstrapVersion::v3);
        bootstrapTheme->setResponsive(true);
        this->setTheme(bootstrapTheme);

        // load the default bootstrap3 (sub)theme
        this->useStyleSheet("resources/themes/bootstrap/3/bootstrap theme.min.css");
```

```
45      } else if (theme == "bootstrap2") {
46          auto bootstrapTheme = std::make_shared<Wt::WBootstrapTheme>();
47          bootstrapTheme->setResponsive(true);
48          this->setTheme(bootstrapTheme);
49      } else
50          this->setTheme(std::make_shared<Wt::WCssTheme>(theme));
51
52      this->setTitle("Cryptology Online");
53
54      this->_session.login().changed().connect(this, &CryptoOnlineLauncher↩
            ::auth_event);
55
56      load_css_files();
57      load_message_bundles();
58
59      /**
60       * Load the initial settings of the website
61       */
62      this->root()->addWidget(Wt::cpp14::make_unique<CryptoOnlineHome>(↩
            _session));
63      this->root()->setContentAlignment(Wt::AlignmentFlag::Center);
64  }
65
66  /**
67   * @brief When a change in the authentication is detected this method is ↩
        called
68   */
69  void CryptoOnlineLauncher::auth_event() {
70      if (_session.login().loggedIn()) {
71          const Wt::Auth::User& user = _session.login().user();
72          std::cout << "User " << user.id()
73                    << " (" << user.identity(Wt::Auth::Identity::LoginName)
74                    << ") Logged in" << std::endl;
75          _session.link_account_to_database(user);
76          this->setInternalPath("/home", true);
77          this->root()->clear();
78          this->root()->addWidget(Wt::cpp14::make_unique<CryptoOnlineHome>(↩
                _session));
79      }else{
80          this->setInternalPath("/home", true);
81          std::cout << "User Logged Out" << std::endl;
82      }
83  }
84
85  /**
86   * @brief This method loads all the required style files
87   */
88  void CryptoOnlineLauncher::load_css_files() {
89
90      this->useStyleSheet("resource/css/CryptoOnlineHeader.css");
91      this->useStyleSheet("resource/css/crypto_online_grid.css");
92      this->useStyleSheet("resource/css/crypto_online_login.css");
93      this->useStyleSheet("resource/css/crypto_online_register.css");
94      this->useStyleSheet("resource/css/crypto_online_profile.css");
```

```
95      this->useStyleSheet("resource/css/crypto_online_learning_content.css");
96      this->useStyleSheet("resource/css/crypto_online_aes_example.css");
97  }
98
99  /**
100  * @brief Load the necessary xml templates and message bundles for the project
101  */
102 void CryptoOnlineLauncher::load_message_bundles() {
103
104     this->messageResourceBundle().use(this->appRoot() + "resource/xml templates/answer_template");
105
106     this->messageResourceBundle().use(this->appRoot() + "resource/content_xmls/modular_arithmetic");
107     this->messageResourceBundle().use(this->appRoot() + "resource/content_xmls/intro_to_cryptography");
108
109 }
```

3.2 src/crypto/

3.2.1 aes_implementation.h

```
1  /**
2   * @file aes_implementation.h
3   * @date 14/01/2018
4   *
5   * @brief This contains the class definition for my AES Implementation
6   *
7   * @version 0.10
8   * @author Jacob Powell
9   */
10
11 #include <cstdint>
12 #include <string>
13
14 typedef unsigned char byte; /**< This is a simple typedef that makes my ↩
        life easier */
15
16 /**
17  * @enum AESKeyLengths
18  *
19  * @version 1.0
20  * @brief This enum contains the possible key lengths that the AES ↩
        algorithm can use.
21  */
22 enum AESKeyLengths{
23     AES128,
24     AES192,
25     AES256
26 };
27
28 /**
29  * @class AESImplementation
30  *
31  * @version 1.0
32  * @brief This class contains my implementation for the AES encryption ↩
        algorithm
33  */
34 class AESImplementation {
35 public:
36     /**
37      * @brief Setting constructor with no arguments to delete
38      */
39     AESImplementation() = default;
40
41     /**
42      * @brief Default constructor which sets the encryption 128 bit key ↩
            to be used
43      *
44      * @param key The encryption key passed into the algorithm
45      */
```

```
46      explicit AESImplementation(AESKeyLengths keyLength);
47
48      /**
49       * @brief This method provides full encryption functionality of the ↩
             encryption rounds for AES
50       *
51       * @param input The full plaintext message
52       * @param output The full ciphertext message
53       * @param key The encryption key
54       * @return A string of the encrypted data
55       */
56      std::string encrypt(const byte input[], byte output[], const byte key↩
             [], size_t message_length);
57
58      /**
59       * @brief This method provides the encryption functionality for a ↩
             block in the AES Algorithm
60       *
61       * @param input The plaintext block that wants to be encrypted
62       * @param output The ciphertext block that has been encrypted
63       * @param key The algorithm key
64       */
65      void encrypt_block(const byte input[], byte output[], const byte key↩
             []);
66
67      /**
68       * @brief This method provides the decryption functionality for a ↩
             block in the AES Algorithm
69       *
70       * @param input The encrypted ciphertext block
71       * @param output The decrypted plaintext block
72       * @param key The algorithm key
73       */
74      void decrypt_block(const byte input[], byte output[], const byte key↩
             []);
75
76  private:
77
78      /**
79       * @brief This provides the implementation for the KeyExpansion part ↩
             of the algorithm
80       *
81       * @param key The original 128 bit key
82       * @param expandedKey The expanded key used for the algorithms round
83       */
84      void KeyExpansion(const byte key[], byte expandedKey[]) const;
85
86      /**
87       * @brief This provides the implementation for the KeyExpansionCore ↩
             for the Key Expansion
88       *
89       * @param roundNumber The current round number
90       * @param keyIn The 4 byte input
91       * @param keyOut The 4 byte output
92       */
```

```
93      static void KeyExpansionCore(byte roundNumber, const byte keyIn[4], ↩
            byte keyOut[4]);
94
95      /**
96       * @brief This XOR's the current state with the rounds subkey
97       *
98       * @param state The current state
99       * @param roundKey The current rounds subkey
100      */
101     void AddRoundKey(byte state[4][4], byte roundKey[4][4]);
102
103     /**
104      * @brief Applies the S Box to each element of the current state
105      *
106      * @param state The current state
107      */
108     void SubBytes(byte state[4][4]);
109
110     /**
111      * @brief This shifts the rows of the current state by the ↩
             appropriate
112      *
113      * @param state The current state
114      */
115     void ShiftRows(byte state[4][4]);
116
117     /**
118      * @brief This applies the MixColumn transformation to the current ↩
             state
119      *
120      * @param state The current state
121      */
122     void MixColumns(byte state[4][4]);
123
124     void InverseSubBytes(byte state[4][4]);
125     void InverseShiftRows(byte state[4][4]);
126     void InverseMixColumns(byte state[4][4]);
127
128     /**
129      * @brief This method outputs what the contents of the current state ↩
             are with a prefix describing what section this
130      * state has been formed in. The primary use for this method is for ↩
             debug information.
131      *
132      * @param prefix This is appended to the start of the current state ↩
             to show where exactly this is
133      * being called from
134      */
135     void outputState(std::string prefix);
136
137     /**
138      * @brief This method outputs the current subkey that will be used ↩
             for the particular round
139      */
140     void outputRoundKey();
```

```
141
142     byte _state[4][4]; /**< This will hold the current state of the ↩
            algorithm */
143     byte _round_key[4][4]; /**< This will hold the round key */
144
145     int _number_of_rounds; /**< This holds the number of rounds the ↩
            algorithm will use */
146     int _block_size; /**< This holds the Block size of the AES Algorithm ↩
            */
147     int _key_size; /**< The chosen key size of the algorithm */
148     byte _n; /**< This holds the initial length of key */
149     byte _b; /**< This holds the length of the expanded key */
150
151     byte _m; /**< This is used when working out how the key schedule ↩
            behaves for certain key lengths */
152
153 };
```

3.2.2 aes_implementation.cc

```
/**
 * @file aes_implementation.cc
 * @date 14/01/2018
 *
 * @brief This file contains my method definitions for my AES Implementation
 *
 * @version 0.10
 * @author Jacob Powell
 */

#include <stdexcept>
#include <cstring>
#include <iostream>
#include <iomanip>
#include <iterator>
#include <sstream>

#include "aes_implementation.h"

#define debug 0

/**< Precomputed Tables and Lookup tables are given below */

static byte sbox[256] = {
        0x63, 0x7C, 0x77, 0x7B, 0xF2, 0x6B, 0x6F, 0xC5, 0x30, 0x01, 0x67, 0x2B, 0xFE, 0xD7, 0xAB, 0x76,
        0xCA, 0x82, 0xC9, 0x7D, 0xFA, 0x59, 0x47, 0xF0, 0xAD, 0xD4, 0xA2, 0xAF, 0x9C, 0xA4, 0x72, 0xC0,
        0xB7, 0xFD, 0x93, 0x26, 0x36, 0x3F, 0xF7, 0xCC, 0x34, 0xA5, 0xE5, 0xF1, 0x71, 0xD8, 0x31, 0x15,
        0x04, 0xC7, 0x23, 0xC3, 0x18, 0x96, 0x05, 0x9A, 0x07, 0x12, 0x80, 0xE2, 0xEB, 0x27, 0xB2, 0x75,
        0x09, 0x83, 0x2C, 0x1A, 0x1B, 0x6E, 0x5A, 0xA0, 0x52, 0x3B, 0xD6, 0xB3, 0x29, 0xE3, 0x2F, 0x84,
        0x53, 0xD1, 0x00, 0xED, 0x20, 0xFC, 0xB1, 0x5B, 0x6A, 0xCB, 0xBE, 0x39, 0x4A, 0x4C, 0x58, 0xCF,
        0xD0, 0xEF, 0xAA, 0xFB, 0x43, 0x4D, 0x33, 0x85, 0x45, 0xF9, 0x02, 0x7F, 0x50, 0x3C, 0x9F, 0xA8,
        0x51, 0xA3, 0x40, 0x8F, 0x92, 0x9D, 0x38, 0xF5, 0xBC, 0xB6, 0xDA, 0x21, 0x10, 0xFF, 0xF3, 0xD2,
        0xCD, 0x0C, 0x13, 0xEC, 0x5F, 0x97, 0x44, 0x17, 0xC4, 0xA7, 0x7E, 0x3D, 0x64, 0x5D, 0x19, 0x73,
        0x60, 0x81, 0x4F, 0xDC, 0x22, 0x2A, 0x90, 0x88, 0x46, 0xEE, 0xB8, 0x14, 0xDE, 0x5E, 0x0B, 0xDB,
        0xE0, 0x32, 0x3A, 0x0A, 0x49, 0x06, 0x24, 0x5C, 0xC2, 0xD3, 0xAC, 0x62, 0x91, 0x95, 0xE4, 0x79,
        0xE7, 0xC8, 0x37, 0x6D, 0x8D, 0xD5, 0x4E, 0xA9, 0x6C, 0x56, 0xF4, 0xEA, 0x65, 0x7A, 0xAE, 0x08,
        0xBA, 0x78, 0x25, 0x2E, 0x1C, 0xA6, 0xB4, 0xC6, 0xE8, 0xDD, 0x74, 0x1F, 0x4B, 0xBD, 0x8B, 0x8A,
```

```
38          0x70, 0x3E, 0xB5, 0x66, 0x48, 0x03, 0xF6, 0x0E, 0x61, 0x35, 0x57,↩
                0xB9, 0x86, 0xC1, 0x1D, 0x9E,
39          0xE1, 0xF8, 0x98, 0x11, 0x69, 0xD9, 0x8E, 0x94, 0x9B, 0x1E, 0x87,↩
                0xE9, 0xCE, 0x55, 0x28, 0xDF,
40          0x8C, 0xA1, 0x89, 0x0D, 0xBF, 0xE6, 0x42, 0x68, 0x41, 0x99, 0x2D,↩
                0x0F, 0xB0, 0x54, 0xBB, 0x16};
41
42  static byte inverse_sbox[256] = {
43          0x52, 0x09, 0x6A, 0xD5, 0x30, 0x36, 0xA5, 0x38, 0xBF, 0x40, 0xA3,↩
                0x9E, 0x81, 0xF3, 0xD7, 0xFB,
44          0x7C, 0xE3, 0x39, 0x82, 0x9B, 0x2F, 0xFF, 0x87, 0x34, 0x8E, 0x43,↩
                0x44, 0xC4, 0xDE, 0xE9, 0xCB,
45          0x54, 0x7B, 0x94, 0x32, 0xA6, 0xC2, 0x23, 0x3D, 0xEE, 0x4C, 0x95,↩
                0x0B, 0x42, 0xFA, 0xC3, 0x4E,
46          0x08, 0x2E, 0xA1, 0x66, 0x28, 0xD9, 0x24, 0xB2, 0x76, 0x5B, 0xA2,↩
                0x49, 0x6D, 0x8B, 0xD1, 0x25,
47          0x72, 0xF8, 0xF6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xD4, 0xA4, 0x5C,↩
                0xCC, 0x5D, 0x65, 0xB6, 0x92,
48          0x6C, 0x70, 0x48, 0x50, 0xFD, 0xED, 0xB9, 0xDA, 0x5E, 0x15, 0x46,↩
                0x57, 0xA7, 0x8D, 0x9D, 0x84,
49          0x90, 0xD8, 0xAB, 0x00, 0x8C, 0xBC, 0xD3, 0x0A, 0xF7, 0xE4, 0x58,↩
                0x05, 0xB8, 0xB3, 0x45, 0x06,
50          0xD0, 0x2C, 0x1E, 0x8F, 0xCA, 0x3F, 0x0F, 0x02, 0xC1, 0xAF, 0xBD,↩
                0x03, 0x01, 0x13, 0x8A, 0x6B,
51          0x3A, 0x91, 0x11, 0x41, 0x4F, 0x67, 0xDC, 0xEA, 0x97, 0xF2, 0xCF,↩
                0xCE, 0xF0, 0xB4, 0xE6, 0x73,
52          0x96, 0xAC, 0x74, 0x22, 0xE7, 0xAD, 0x35, 0x85, 0xE2, 0xF9, 0x37,↩
                0xE8, 0x1C, 0x75, 0xDF, 0x6E,
53          0x47, 0xF1, 0x1A, 0x71, 0x1D, 0x29, 0xC5, 0x89, 0x6F, 0xB7, 0x62,↩
                0x0E, 0xAA, 0x18, 0xBE, 0x1B,
54          0xFC, 0x56, 0x3E, 0x4B, 0xC6, 0xD2, 0x79, 0x20, 0x9A, 0xDB, 0xC0,↩
                0xFE, 0x78, 0xCD, 0x5A, 0xF4,
55          0x1F, 0xDD, 0xA8, 0x33, 0x88, 0x07, 0xC7, 0x31, 0xB1, 0x12, 0x10,↩
                0x59, 0x27, 0x80, 0xEC, 0x5F,
56          0x60, 0x51, 0x7F, 0xA9, 0x19, 0xB5, 0x4A, 0x0D, 0x2D, 0xE5, 0x7A,↩
                0x9F, 0x93, 0xC9, 0x9C, 0xEF,
57          0xA0, 0xE0, 0x3B, 0x4D, 0xAE, 0x2A, 0xF5, 0xB0, 0xC8, 0xEB, 0xBB,↩
                0x3C, 0x83, 0x53, 0x99, 0x61,
58          0x17, 0x2B, 0x04, 0x7E, 0xBA, 0x77, 0xD6, 0x26, 0xE1, 0x69, 0x14,↩
                0x63, 0x55, 0x21, 0x0C, 0x7D};
59
60  static byte galois_mul_2[256] = {
61          0x00,0x02,0x04,0x06,0x08,0x0a,0x0c,0x0e,0x10,0x12,0x14,0x16,0x18↩
                ,0x1a,0x1c,0x1e,
62          0x20,0x22,0x24,0x26,0x28,0x2a,0x2c,0x2e,0x30,0x32,0x34,0x36,0x38↩
                ,0x3a,0x3c,0x3e,
63          0x40,0x42,0x44,0x46,0x48,0x4a,0x4c,0x4e,0x50,0x52,0x54,0x56,0x58↩
                ,0x5a,0x5c,0x5e,
64          0x60,0x62,0x64,0x66,0x68,0x6a,0x6c,0x6e,0x70,0x72,0x74,0x76,0x78↩
                ,0x7a,0x7c,0x7e,
65          0x80,0x82,0x84,0x86,0x88,0x8a,0x8c,0x8e,0x90,0x92,0x94,0x96,0x98↩
                ,0x9a,0x9c,0x9e,
66          0xa0,0xa2,0xa4,0xa6,0xa8,0xaa,0xac,0xae,0xb0,0xb2,0xb4,0xb6,0xb8↩
                ,0xba,0xbc,0xbe,
```

```
67          0xc0,0xc2,0xc4,0xc6,0xc8,0xca,0xcc,0xce,0xd0,0xd2,0xd4,0xd6,0xd8↩
                ,0xda,0xdc,0xde,
68          0xe0,0xe2,0xe4,0xe6,0xe8,0xea,0xec,0xee,0xf0,0xf2,0xf4,0xf6,0xf8↩
                ,0xfa,0xfc,0xfe,
69          0x1b,0x19,0x1f,0x1d,0x13,0x11,0x17,0x15,0x0b,0x09,0x0f,0x0d,0x03↩
                ,0x01,0x07,0x05,
70          0x3b,0x39,0x3f,0x3d,0x33,0x31,0x37,0x35,0x2b,0x29,0x2f,0x2d,0x23↩
                ,0x21,0x27,0x25,
71          0x5b,0x59,0x5f,0x5d,0x53,0x51,0x57,0x55,0x4b,0x49,0x4f,0x4d,0x43↩
                ,0x41,0x47,0x45,
72          0x7b,0x79,0x7f,0x7d,0x73,0x71,0x77,0x75,0x6b,0x69,0x6f,0x6d,0x63↩
                ,0x61,0x67,0x65,
73          0x9b,0x99,0x9f,0x9d,0x93,0x91,0x97,0x95,0x8b,0x89,0x8f,0x8d,0x83↩
                ,0x81,0x87,0x85,
74          0xbb,0xb9,0xbf,0xbd,0xb3,0xb1,0xb7,0xb5,0xab,0xa9,0xaf,0xad,0xa3↩
                ,0xa1,0xa7,0xa5,
75          0xdb,0xd9,0xdf,0xdd,0xd3,0xd1,0xd7,0xd5,0xcb,0xc9,0xcf,0xcd,0xc3↩
                ,0xc1,0xc7,0xc5,
76          0xfb,0xf9,0xff,0xfd,0xf3,0xf1,0xf7,0xf5,0xeb,0xe9,0xef,0xed,0xe3↩
                ,0xe1,0xe7,0xe5};
77
78  static byte galois_mul_3[256] = {
79          0x00,0x03,0x06,0x05,0x0c,0x0f,0x0a,0x09,0x18,0x1b,0x1e,0x1d,0x14↩
                ,0x17,0x12,0x11,
80          0x30,0x33,0x36,0x35,0x3c,0x3f,0x3a,0x39,0x28,0x2b,0x2e,0x2d,0x24↩
                ,0x27,0x22,0x21,
81          0x60,0x63,0x66,0x65,0x6c,0x6f,0x6a,0x69,0x78,0x7b,0x7e,0x7d,0x74↩
                ,0x77,0x72,0x71,
82          0x50,0x53,0x56,0x55,0x5c,0x5f,0x5a,0x59,0x48,0x4b,0x4e,0x4d,0x44↩
                ,0x47,0x42,0x41,
83          0xc0,0xc3,0xc6,0xc5,0xcc,0xcf,0xca,0xc9,0xd8,0xdb,0xde,0xdd,0xd4↩
                ,0xd7,0xd2,0xd1,
84          0xf0,0xf3,0xf6,0xf5,0xfc,0xff,0xfa,0xf9,0xe8,0xeb,0xee,0xed,0xe4↩
                ,0xe7,0xe2,0xe1,
85          0xa0,0xa3,0xa6,0xa5,0xac,0xaf,0xaa,0xa9,0xb8,0xbb,0xbe,0xbd,0xb4↩
                ,0xb7,0xb2,0xb1,
86          0x90,0x93,0x96,0x95,0x9c,0x9f,0x9a,0x99,0x88,0x8b,0x8e,0x8d,0x84↩
                ,0x87,0x82,0x81,
87          0x9b,0x98,0x9d,0x9e,0x97,0x94,0x91,0x92,0x83,0x80,0x85,0x86,0x8f↩
                ,0x8c,0x89,0x8a,
88          0xab,0xa8,0xad,0xae,0xa7,0xa4,0xa1,0xa2,0xb3,0xb0,0xb5,0xb6,0xbf↩
                ,0xbc,0xb9,0xba,
89          0xfb,0xf8,0xfd,0xfe,0xf7,0xf4,0xf1,0xf2,0xe3,0xe0,0xe5,0xe6,0xef↩
                ,0xec,0xe9,0xea,
90          0xcb,0xc8,0xcd,0xce,0xc7,0xc4,0xc1,0xc2,0xd3,0xd0,0xd5,0xd6,0xdf↩
                ,0xdc,0xd9,0xda,
91          0x5b,0x58,0x5d,0x5e,0x57,0x54,0x51,0x52,0x43,0x40,0x45,0x46,0x4f↩
                ,0x4c,0x49,0x4a,
92          0x6b,0x68,0x6d,0x6e,0x67,0x64,0x61,0x62,0x73,0x70,0x75,0x76,0x7f↩
                ,0x7c,0x79,0x7a,
93          0x3b,0x38,0x3d,0x3e,0x37,0x34,0x31,0x32,0x23,0x20,0x25,0x26,0x2f↩
                ,0x2c,0x29,0x2a,
94          0x0b,0x08,0x0d,0x0e,0x07,0x04,0x01,0x02,0x13,0x10,0x15,0x16,0x1f↩
                ,0x1c,0x19,0x1a};
95
```

```
96  static byte galois_mul_9[256] = {
97          0x00,0x09,0x12,0x1b,0x24,0x2d,0x36,0x3f,0x48,0x41,0x5a,0x53,0x6c↩
                ,0x65,0x7e,0x77,
98          0x90,0x99,0x82,0x8b,0xb4,0xbd,0xa6,0xaf,0xd8,0xd1,0xca,0xc3,0xfc↩
                ,0xf5,0xee,0xe7,
99          0x3b,0x32,0x29,0x20,0x1f,0x16,0x0d,0x04,0x73,0x7a,0x61,0x68,0x57↩
                ,0x5e,0x45,0x4c,
100         0xab,0xa2,0xb9,0xb0,0x8f,0x86,0x9d,0x94,0xe3,0xea,0xf1,0xf8,0xc7↩
                ,0xce,0xd5,0xdc,
101         0x76,0x7f,0x64,0x6d,0x52,0x5b,0x40,0x49,0x3e,0x37,0x2c,0x25,0x1a↩
                ,0x13,0x08,0x01,
102         0xe6,0xef,0xf4,0xfd,0xc2,0xcb,0xd0,0xd9,0xae,0xa7,0xbc,0xb5,0x8a↩
                ,0x83,0x98,0x91,
103         0x4d,0x44,0x5f,0x56,0x69,0x60,0x7b,0x72,0x05,0x0c,0x17,0x1e,0x21↩
                ,0x28,0x33,0x3a,
104         0xdd,0xd4,0xcf,0xc6,0xf9,0xf0,0xeb,0xe2,0x95,0x9c,0x87,0x8e,0xb1↩
                ,0xb8,0xa3,0xaa,
105         0xec,0xe5,0xfe,0xf7,0xc8,0xc1,0xda,0xd3,0xa4,0xad,0xb6,0xbf,0x80↩
                ,0x89,0x92,0x9b,
106         0x7c,0x75,0x6e,0x67,0x58,0x51,0x4a,0x43,0x34,0x3d,0x26,0x2f,0x10↩
                ,0x19,0x02,0x0b,
107         0xd7,0xde,0xc5,0xcc,0xf3,0xfa,0xe1,0xe8,0x9f,0x96,0x8d,0x84,0xbb↩
                ,0xb2,0xa9,0xa0,
108         0x47,0x4e,0x55,0x5c,0x63,0x6a,0x71,0x78,0x0f,0x06,0x1d,0x14,0x2b↩
                ,0x22,0x39,0x30,
109         0x9a,0x93,0x88,0x81,0xbe,0xb7,0xac,0xa5,0xd2,0xdb,0xc0,0xc9,0xf6↩
                ,0xff,0xe4,0xed,
110         0x0a,0x03,0x18,0x11,0x2e,0x27,0x3c,0x35,0x42,0x4b,0x50,0x59,0x66↩
                ,0x6f,0x74,0x7d,
111         0xa1,0xa8,0xb3,0xba,0x85,0x8c,0x97,0x9e,0xe9,0xe0,0xfb,0xf2,0xcd↩
                ,0xc4,0xdf,0xd6,
112         0x31,0x38,0x23,0x2a,0x15,0x1c,0x07,0x0e,0x79,0x70,0x6b,0x62,0x5d↩
                ,0x54,0x4f,0x46
113 };
114
115 static byte galois_mul_11[256] = {
116         0x00,0x0b,0x16,0x1d,0x2c,0x27,0x3a,0x31,0x58,0x53,0x4e,0x45,0x74↩
                ,0x7f,0x62,0x69,
117         0xb0,0xbb,0xa6,0xad,0x9c,0x97,0x8a,0x81,0xe8,0xe3,0xfe,0xf5,0xc4↩
                ,0xcf,0xd2,0xd9,
118         0x7b,0x70,0x6d,0x66,0x57,0x5c,0x41,0x4a,0x23,0x28,0x35,0x3e,0x0f↩
                ,0x04,0x19,0x12,
119         0xcb,0xc0,0xdd,0xd6,0xe7,0xec,0xf1,0xfa,0x93,0x98,0x85,0x8e,0xbf↩
                ,0xb4,0xa9,0xa2,
120         0xf6,0xfd,0xe0,0xeb,0xda,0xd1,0xcc,0xc7,0xae,0xa5,0xb8,0xb3,0x82↩
                ,0x89,0x94,0x9f,
121         0x46,0x4d,0x50,0x5b,0x6a,0x61,0x7c,0x77,0x1e,0x15,0x08,0x03,0x32↩
                ,0x39,0x24,0x2f,
122         0x8d,0x86,0x9b,0x90,0xa1,0xaa,0xb7,0xbc,0xd5,0xde,0xc3,0xc8,0xf9↩
                ,0xf2,0xef,0xe4,
123         0x3d,0x36,0x2b,0x20,0x11,0x1a,0x07,0x0c,0x65,0x6e,0x73,0x78,0x49↩
                ,0x42,0x5f,0x54,
124         0xf7,0xfc,0xe1,0xea,0xdb,0xd0,0xcd,0xc6,0xaf,0xa4,0xb9,0xb2,0x83↩
                ,0x88,0x95,0x9e,
```

```
125         0x47,0x4c,0x51,0x5a,0x6b,0x60,0x7d,0x76,0x1f,0x14,0x09,0x02,0x33↩
                ,0x38,0x25,0x2e,
126         0x8c,0x87,0x9a,0x91,0xa0,0xab,0xb6,0xbd,0xd4,0xdf,0xc2,0xc9,0xf8↩
                ,0xf3,0xee,0xe5,
127         0x3c,0x37,0x2a,0x21,0x10,0x1b,0x06,0x0d,0x64,0x6f,0x72,0x79,0x48↩
                ,0x43,0x5e,0x55,
128         0x01,0x0a,0x17,0x1c,0x2d,0x26,0x3b,0x30,0x59,0x52,0x4f,0x44,0x75↩
                ,0x7e,0x63,0x68,
129         0xb1,0xba,0xa7,0xac,0x9d,0x96,0x8b,0x80,0xe9,0xe2,0xff,0xf4,0xc5↩
                ,0xce,0xd3,0xd8,
130         0x7a,0x71,0x6c,0x67,0x56,0x5d,0x40,0x4b,0x22,0x29,0x34,0x3f,0x0e↩
                ,0x05,0x18,0x13,
131         0xca,0xc1,0xdc,0xd7,0xe6,0xed,0xf0,0xfb,0x92,0x99,0x84,0x8f,0xbe↩
                ,0xb5,0xa8,0xa3
132 };
133
134 static byte galois_mul_13[256] = {
135         0x00,0x0d,0x1a,0x17,0x34,0x39,0x2e,0x23,0x68,0x65,0x72,0x7f,0x5c↩
                ,0x51,0x46,0x4b,
136         0xd0,0xdd,0xca,0xc7,0xe4,0xe9,0xfe,0xf3,0xb8,0xb5,0xa2,0xaf,0x8c↩
                ,0x81,0x96,0x9b,
137         0xbb,0xb6,0xa1,0xac,0x8f,0x82,0x95,0x98,0xd3,0xde,0xc9,0xc4,0xe7↩
                ,0xea,0xfd,0xf0,
138         0x6b,0x66,0x71,0x7c,0x5f,0x52,0x45,0x48,0x03,0x0e,0x19,0x14,0x37↩
                ,0x3a,0x2d,0x20,
139         0x6d,0x60,0x77,0x7a,0x59,0x54,0x43,0x4e,0x05,0x08,0x1f,0x12,0x31↩
                ,0x3c,0x2b,0x26,
140         0xbd,0xb0,0xa7,0xaa,0x89,0x84,0x93,0x9e,0xd5,0xd8,0xcf,0xc2,0xe1↩
                ,0xec,0xfb,0xf6,
141         0xd6,0xdb,0xcc,0xc1,0xe2,0xef,0xf8,0xf5,0xbe,0xb3,0xa4,0xa9,0x8a↩
                ,0x87,0x90,0x9d,
142         0x06,0x0b,0x1c,0x11,0x32,0x3f,0x28,0x25,0x6e,0x63,0x74,0x79,0x5a↩
                ,0x57,0x40,0x4d,
143         0xda,0xd7,0xc0,0xcd,0xee,0xe3,0xf4,0xf9,0xb2,0xbf,0xa8,0xa5,0x86↩
                ,0x8b,0x9c,0x91,
144         0x0a,0x07,0x10,0x1d,0x3e,0x33,0x24,0x29,0x62,0x6f,0x78,0x75,0x56↩
                ,0x5b,0x4c,0x41,
145         0x61,0x6c,0x7b,0x76,0x55,0x58,0x4f,0x42,0x09,0x04,0x13,0x1e,0x3d↩
                ,0x30,0x27,0x2a,
146         0xb1,0xbc,0xab,0xa6,0x85,0x88,0x9f,0x92,0xd9,0xd4,0xc3,0xce,0xed↩
                ,0xe0,0xf7,0xfa,
147         0xb7,0xba,0xad,0xa0,0x83,0x8e,0x99,0x94,0xdf,0xd2,0xc5,0xc8,0xeb↩
                ,0xe6,0xf1,0xfc,
148         0x67,0x6a,0x7d,0x70,0x53,0x5e,0x49,0x44,0x0f,0x02,0x15,0x18,0x3b↩
                ,0x36,0x21,0x2c,
149         0x0c,0x01,0x16,0x1b,0x38,0x35,0x22,0x2f,0x64,0x69,0x7e,0x73,0x50↩
                ,0x5d,0x4a,0x47,
150         0xdc,0xd1,0xc6,0xcb,0xe8,0xe5,0xf2,0xff,0xb4,0xb9,0xae,0xa3,0x80↩
                ,0x8d,0x9a,0x97
151 };
152
153 static byte galois_mul_14[256] = {
154         0x00,0x0e,0x1c,0x12,0x38,0x36,0x24,0x2a,0x70,0x7e,0x6c,0x62,0x48↩
                ,0x46,0x54,0x5a,
```

```
155         0xe0,0xee,0xfc,0xf2,0xd8,0xd6,0xc4,0xca,0x90,0x9e,0x8c,0x82,0xa8↩
                ,0xa6,0xb4,0xba,
156         0xdb,0xd5,0xc7,0xc9,0xe3,0xed,0xff,0xf1,0xab,0xa5,0xb7,0xb9,0x93↩
                ,0x9d,0x8f,0x81,
157         0x3b,0x35,0x27,0x29,0x03,0x0d,0x1f,0x11,0x4b,0x45,0x57,0x59,0x73↩
                ,0x7d,0x6f,0x61,
158         0xad,0xa3,0xb1,0xbf,0x95,0x9b,0x89,0x87,0xdd,0xd3,0xc1,0xcf,0xe5↩
                ,0xeb,0xf9,0xf7,
159         0x4d,0x43,0x51,0x5f,0x75,0x7b,0x69,0x67,0x3d,0x33,0x21,0x2f,0x05↩
                ,0x0b,0x19,0x17,
160         0x76,0x78,0x6a,0x64,0x4e,0x40,0x52,0x5c,0x06,0x08,0x1a,0x14,0x3e↩
                ,0x30,0x22,0x2c,
161         0x96,0x98,0x8a,0x84,0xae,0xa0,0xb2,0xbc,0xe6,0xe8,0xfa,0xf4,0xde↩
                ,0xd0,0xc2,0xcc,
162         0x41,0x4f,0x5d,0x53,0x79,0x77,0x65,0x6b,0x31,0x3f,0x2d,0x23,0x09↩
                ,0x07,0x15,0x1b,
163         0xa1,0xaf,0xbd,0xb3,0x99,0x97,0x85,0x8b,0xd1,0xdf,0xcd,0xc3,0xe9↩
                ,0xe7,0xf5,0xfb,
164         0x9a,0x94,0x86,0x88,0xa2,0xac,0xbe,0xb0,0xea,0xe4,0xf6,0xf8,0xd2↩
                ,0xdc,0xce,0xc0,
165         0x7a,0x74,0x66,0x68,0x42,0x4c,0x5e,0x50,0x0a,0x04,0x16,0x18,0x32↩
                ,0x3c,0x2e,0x20,
166         0xec,0xe2,0xf0,0xfe,0xd4,0xda,0xc8,0xc6,0x9c,0x92,0x80,0x8e,0xa4↩
                ,0xaa,0xb8,0xb6,
167         0x0c,0x02,0x10,0x1e,0x34,0x3a,0x28,0x26,0x7c,0x72,0x60,0x6e,0x44↩
                ,0x4a,0x58,0x56,
168         0x37,0x39,0x2b,0x25,0x0f,0x01,0x13,0x1d,0x47,0x49,0x5b,0x55,0x7f↩
                ,0x71,0x63,0x6d,
169         0xd7,0xd9,0xcb,0xc5,0xef,0xe1,0xf3,0xfd,0xa7,0xa9,0xbb,0xb5,0x9f↩
                ,0x91,0x83,0x8d
170 };
171
172 static byte rcon[256] = {
173         0x8d, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36,↩
                0x6c, 0xd8, 0xab, 0x4d, 0x9a,
174         0x2f, 0x5e, 0xbc, 0x63, 0xc6, 0x97, 0x35, 0x6a, 0xd4, 0xb3, 0x7d,↩
                0xfa, 0xef, 0xc5, 0x91, 0x39,
175         0x72, 0xe4, 0xd3, 0xbd, 0x61, 0xc2, 0x9f, 0x25, 0x4a, 0x94, 0x33,↩
                0x66, 0xcc, 0x83, 0x1d, 0x3a,
176         0x74, 0xe8, 0xcb, 0x8d, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40,↩
                0x80, 0x1b, 0x36, 0x6c, 0xd8,
177         0xab, 0x4d, 0x9a, 0x2f, 0x5e, 0xbc, 0x63, 0xc6, 0x97, 0x35, 0x6a,↩
                0xd4, 0xb3, 0x7d, 0xfa, 0xef,
178         0xc5, 0x91, 0x39, 0x72, 0xe4, 0xd3, 0xbd, 0x61, 0xc2, 0x9f, 0x25,↩
                0x4a, 0x94, 0x33, 0x66, 0xcc,
179         0x83, 0x1d, 0x3a, 0x74, 0xe8, 0xcb, 0x8d, 0x01, 0x02, 0x04, 0x08,↩
                0x10, 0x20, 0x40, 0x80, 0x1b,
180         0x36, 0x6c, 0xd8, 0xab, 0x4d, 0x9a, 0x2f, 0x5e, 0xbc, 0x63, 0xc6,↩
                0x97, 0x35, 0x6a, 0xd4, 0xb3,
181         0x7d, 0xfa, 0xef, 0xc5, 0x91, 0x39, 0x72, 0xe4, 0xd3, 0xbd, 0x61,↩
                0xc2, 0x9f, 0x25, 0x4a, 0x94,
182         0x33, 0x66, 0xcc, 0x83, 0x1d, 0x3a, 0x74, 0xe8, 0xcb, 0x8d, 0x01,↩
                0x02, 0x04, 0x08, 0x10, 0x20,
183         0x40, 0x80, 0x1b, 0x36, 0x6c, 0xd8, 0xab, 0x4d, 0x9a, 0x2f, 0x5e,↩
                0xbc, 0x63, 0xc6, 0x97, 0x35,
```

```
184         0x6a, 0xd4, 0xb3, 0x7d, 0xfa, 0xef, 0xc5, 0x91, 0x39, 0x72, 0xe4,↩
                0xd3, 0xbd, 0x61, 0xc2, 0x9f,
185         0x25, 0x4a, 0x94, 0x33, 0x66, 0xcc, 0x83, 0x1d, 0x3a, 0x74, 0xe8,↩
                0xcb, 0x8d, 0x01, 0x02, 0x04,
186         0x08, 0x10, 0x20, 0x40, 0x80, 0x1b, 0x36, 0x6c, 0xd8, 0xab, 0x4d,↩
                0x9a, 0x2f, 0x5e, 0xbc, 0x63,
187         0xc6, 0x97, 0x35, 0x6a, 0xd4, 0xb3, 0x7d, 0xfa, 0xef, 0xc5, 0x91,↩
                0x39, 0x72, 0xe4, 0xd3, 0xbd,
188         0x61, 0xc2, 0x9f, 0x25, 0x4a, 0x94, 0x33, 0x66, 0xcc, 0x83, 0x1d,↩
                0x3a, 0x74, 0xe8, 0xcb, 0x8d
189 };
190
191
192 AESImplementation::AESImplementation(AESKeyLengths keyLength) : _n(0), _b↩
        (0), _number_of_rounds(0), _block_size(0){
193     switch(keyLength){
194         case AES128:
195             this->_block_size = 16;
196             this->_key_size = 128;
197             this->_number_of_rounds = 10;
198             this->_n = 16;
199             this->_b = 176;
200             this->_m = 0;
201             break;
202         case AES192:
203             this->_block_size = 16;
204             this->_key_size = 192;
205             this->_number_of_rounds = 12;
206             this->_n = 24;
207             this->_b = 208;
208             this->_m = 2;
209             break;
210         case AES256:
211             this->_block_size = 16;
212             this->_key_size = 256;
213             this->_number_of_rounds = 14;
214             this->_n = 32;
215             this->_b = 240;
216             this->_m = 3;
217             break;
218         default:
219             throw std::runtime_error("Invalid key length");
220     }
221 }
222
223 std::string AESImplementation::encrypt(const byte input[], byte output[],↩
        const byte key[], const size_t message_length) {
224     std::cout << message_length << std::endl;
225     auto pad_check = (message_length % 16);
226     auto pad_length = 16   pad_check;
227
228     std::cout << pad_check << std::endl;
229
230     if(pad_check == 0){
231         auto *output_padded = new byte[message_length + pad_check];
```

```
232         std::copy(input, (input + std::min(message_length, (↩
                message_length + pad_check))), output_padded);
233         output = output_padded;
234 #if defined(debug) && debug == 1
235         for (byte i = 0; i < (message_length + pad_check); i++) {
236             std::cout << (unsigned) i << ": " << (unsigned) output[i] << ↩
                    std::endl;
237         }
238 #endif
239     }else {
240 
241         auto *output_padded = new byte[message_length + pad_length];
242         std::copy(input, (input + std::min(message_length, (↩
                message_length + pad_length))), output_padded);
243         output = output_padded;
244 
245         for (size_t i = (message_length + pad_length); i >= ↩
                message_length; i  ) {
246             output[i] = pad_length;
247         }
248 #if defined(debug) && debug == 1
249         for (byte i = 0; i < (message_length + pad_length); i++) {
250             std::cout << (unsigned) i << ": " << (unsigned) output[i] << ↩
                    std::endl;
251         }
252 #endif
253     }
254 
255 }
256 
257 void AESImplementation::encrypt_block(const byte input[], byte output[], ↩
    const byte key[]) {
258 
259     std::cout << "Encryption Process Started" << std::endl;
260 
261     for(byte i = 0; i < 4; i++){
262         this->_state[i][0] = input[i];
263         this->_state[i][1] = input[i + 4];
264         this->_state[i][2] = input[i + 8];
265         this->_state[i][3] = input[i + 12];
266     }
267 
268     this->outputState("Current State");
269 
270     byte expanded_key[this->_b];
271     this->KeyExpansion(key, expanded_key);
272 
273     for(byte i = 0; i < 4; i++){
274         this->_round_key[i][0] = expanded_key[i];
275         this->_round_key[i][1] = expanded_key[i + 4];
276         this->_round_key[i][2] = expanded_key[i + 8];
277         this->_round_key[i][3] = expanded_key[i + 12];
278     }
279 
280     this->outputRoundKey();
```

```
281
282     this->AddRoundKey(this->_state, this->_round_key);
283
284     this->outputState("Current State");
285
286     for(byte current_round = 1; current_round <= this->_number_of_rounds;↩
            current_round++){
287 #if defined(debug) && debug == 1
288         std::cout << "Start of Round " << unsigned(current_round) << std↩
               ::endl;
289 #endif
290         for(byte i = 0; i < 4; i++)
291             for(byte j = 0; j < 4; j++) {
292                 this->_round_key[i][j] = expanded_key[(current_round * ↩
                       this->_block_size) + (j * 4) + i];
293
294             }
295
296         this->SubBytes(this->_state);
297         this->ShiftRows(this->_state);
298         if(current_round != this->_number_of_rounds) {
299             this->MixColumns(this->_state);
300         }
301         this->AddRoundKey(this->_state, this->_round_key);
302         this->outputRoundKey();
303 #if defined(debug) && debug == 1
304         std::cout << "End of Round " << unsigned(current_round) << std::↩
               endl;
305 #endif
306     }
307
308     for(byte i = 0; i < 4; i++){
309         for(byte j = 0; j < 4; j++){
310             output[(j * 4) + i] = this->_state[i][j];
311         }
312     }
313     std::cout << "Encryption Process Ended" << std::endl;
314 }
315
316 void AESImplementation::decrypt_block(const byte input[], byte output[], ↩
       const byte key[]) {
317
318     std::cout << "Decryption Process Started" << std::endl;
319
320     for(byte i = 0; i < 4; i++){
321         this->_state[i][0] = input[i];
322         this->_state[i][1] = input[i + 4];
323         this->_state[i][2] = input[i + 8];
324         this->_state[i][3] = input[i + 12];
325     }
326
327     this->outputState("Current State");
328
329     byte expanded_key[this->_b];
330     this->KeyExpansion(key, expanded_key);
```

```
331
332
333     for(byte i = 0; i < 4; i++){
334         this->_round_key[i][0] = expanded_key[this->_b   (16   i)];
335         this->_round_key[i][1] = expanded_key[this->_b   (12   i)];
336         this->_round_key[i][2] = expanded_key[this->_b   (8   i)];
337         this->_round_key[i][3] = expanded_key[this->_b   (4   i)];
338     }
339
340     this->AddRoundKey(this->_state, this->_round_key);
341
342     for(int current_round = (this->_number_of_rounds   1); current_round >= 0; current_round  ){
343 #if defined(debug) && debug == 1
344         std::cout << "Start of Round " << unsigned(current_round) << std::endl;
345         this->outputState("Current State: ");
346 #endif
347         for(byte i = 0; i < 4; i++) {
348             for (byte j = 0; j < 4; j++) {
349                 this->_round_key[i][j] = expanded_key[(current_round * this->_block_size) + (4 * j) + i];
350             }
351         }
352
353         this->outputRoundKey();
354
355         this->InverseShiftRows(this->_state);
356         this->InverseSubBytes(this->_state);
357
358
359         this->outputRoundKey();
360
361         this->AddRoundKey(this->_state, this->_round_key);
362
363         if(current_round != 0){
364             this->InverseMixColumns(this->_state);
365         }
366 #if defined(debug) && debug == 1
367         std::cout << "End of Round " << unsigned(current_round) << std::endl;
368 #endif
369     }
370
371     this->outputState("End of Dec");
372
373     for(byte i = 0; i < 4; i++){
374         for(byte j = 0; j < 4; j++){
375             output[(j * 4) + i] = this->_state[i][j];
376         }
377     }
378
379     std::cout << "Decryption Process Ended" << std::endl;
380 }
381
```

```
382
383 void AESImplementation::KeyExpansion(const byte key[], byte expandedKey[]) const {
384
385 #if defined(debug) && debug == 1
386     std::cout << "Key Expansion Start" << std::endl;
387 #endif
388     memset(expandedKey, 0, this->_b);
389     memcpy(expandedKey, key, this->_n);
390 #if defined(debug) && debug == 1
391     std::cout << "W[0 3]: ";
392     for(byte i = 0; i < this->_n; i++){
393         std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(expandedKey[i]);
394     }
395     std::cout << std::endl;
396 #endif
397
398     byte keySizeIterator = 0;
399     keySizeIterator += this->_n;
400
401     byte t[4];
402     byte u[4];
403
404     for(byte rcon = 1; keySizeIterator < this->_b; rcon++){
405
406         memcpy(t, expandedKey + (keySizeIterator - 4), 4);
407
408 #if defined(debug) && debug == 1
409         std::cout << "PRE 4: ";
410         for(byte i = 0; i < 4; i++)
411             std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(t[i]);
412         std::cout << std::endl;
413 #endif
414
415         this->KeyExpansionCore(rcon, t, u);
416         memcpy(t, expandedKey + (keySizeIterator - this->_n), 4 * sizeof(byte));
417
418 #if defined(debug) && debug == 1
419         std::cout << "W[i - 4]: ";
420         for(byte i = 0; i < 4; i++)
421             std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(u[i]);
422         std::cout << std::endl;
423 #endif
424
425         for(byte i = 0; i < 4; i++)
426             t[i] ^= u[i];
427
428 #if defined(debug) && debug == 1
429         std::cout << "W[i]: ";
430         for(byte i = 0; i < 4; i++)
```

```
431             std::cout << std::hex << std::setfill('0') << std::setw(2) <<↩
                    unsigned(t[i]);
432         std::cout << std::endl;
433 #endif
434 
435         memcpy(expandedKey + keySizeIterator, t, 4 * sizeof(byte));
436         keySizeIterator += 4;
437 
438         for(byte i = 0; i < 3 && (keySizeIterator < this >_b); i++){
439             memcpy(t, expandedKey + (keySizeIterator    4), 4 * sizeof(↩
                    byte));
440 
441 #if defined(debug) && debug == 1
442             std::cout << "temp:  ";
443 #endif
444             for (byte j = 0; j < 4; j++)
445             {
446                 expandedKey[keySizeIterator + j] = t[j] ^ expandedKey[↩
                        keySizeIterator    this >_n + j];
447 #if defined(debug) && debug == 1
448                 std::cout << (unsigned)expandedKey[keySizeIterator + j];
449 #endif
450             }
451 #if defined(debug) && debug == 1
452             std::cout << std::endl;
453 #endif
454             keySizeIterator += 4;
455         }
456 
457         if((this >_key_size == 256) && (keySizeIterator < this >_b)){
458 
459             memcpy(t, expandedKey + (keySizeIterator    4), 4 * sizeof(↩
                    byte));
460             for(byte i = 0; i < 4; i++){
461                 t[i] = sbox[t[i]];
462             }
463 
464             for(byte i = 0; i < 4; i++){
465                 expandedKey[keySizeIterator + i] = t[i] ^ expandedKey[↩
                        keySizeIterator    this >_n + i];
466                 t[i] ^= expandedKey[keySizeIterator    this >_n    4 + i];
467             }
468             //memcpy(expandedKey + (keySizeIterator    3    1), t, 4 * ↩
                    sizeof(byte));
469             keySizeIterator += 4;
470         }
471 
472         for(byte i = 0; (i < this >_m) && (keySizeIterator < this >_b); i↩
                ++){
473             memcpy(t, expandedKey + (keySizeIterator    4), 4 * sizeof(↩
                    byte));
474 #if defined(debug) && debug == 1
475             std::cout << "temp:  ";
476 #endif
477             for (byte j = 0; j < 4; j++)
```

```
478             {
479                 expandedKey[keySizeIterator + j] = t[j] ^ expandedKey[keySizeIterator    this >_n + j];
480 #if defined(debug) && debug == 1
481                 std::cout << (unsigned)expandedKey[keySizeIterator + j];
482 #endif
483             }
484 #if defined(debug) && debug == 1
485             std::cout << std::endl;
486 #endif
487             keySizeIterator += 4;
488         }
489 
490     }
491 
492 #if defined(debug) && debug == 1
493     int count = 0;
494     int roundNumber = 1;
495     std::cout << "Expanded Key: ";
496     for(byte i = 0; i < this >_b; i++){
497         if(count % 16 == 0)
498             std::cout << std::endl;
499 
500         std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(expandedKey[i]);
501         count++;
502         roundNumber++;
503     }
504 
505     std::cout << std::endl;
506     std::cout << "Key Expansion End" << std::endl;
507 #endif
508 }
509 
510 void AESImplementation::KeyExpansionCore(byte roundNumber, const byte keyIn[4], byte keyOut[4]){
511 
512 #if defined(debug) && debug == 1
513     std::cout << "Key Expansion Core Start" << std::endl;
514 
515     std::cout << "KEY OUT: ";
516 
517     for(byte i = 0; i < 4; i++){
518         std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(keyOut[i]);
519     }
520     std::cout << std::endl;
521 
522     std::cout << "KEY IN: ";
523 
524     for(byte i = 0; i < 4; i++){
525         std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(keyIn[i]);
526     }
527     std::cout << std::endl;
```

```
528 #endif
529
530     memcpy(keyOut, keyIn, 4);
531
532     /**
533      * Rotation Step
534      */
535     byte temp = keyOut[0];
536     for(byte i = 0; i < 3; i++){
537         keyOut[i] = keyOut[i+1];
538     }
539     keyOut[3] = temp;
540
541 #if defined(debug) && debug == 1
542     std::cout << "After Rotation: ";
543
544     for(byte i = 0; i < 4; i++){
545         std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
            unsigned(keyOut[i]);
546     }
547     std::cout << std::endl;
548 #endif
549
550     /**
551      * Substitution Step
552      */
553     for(byte i = 0; i < 4; i++){
554         keyOut[i] = sbox[keyOut[i]];
555     }
556
557 #if defined(debug) && debug == 1
558     std::cout << "After Substitution: ";
559     for(byte i = 0; i < 4; i++){
560         std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
            unsigned(keyOut[i]);
561     }
562     std::cout << std::endl;
563 #endif
564
565     keyOut[0] = keyOut[0] ^ rcon[roundNumber];
566
567 #if defined(debug) && debug == 1
568     std::cout << "After XOR With RCON: ";
569     for(byte i = 0; i < 4; i++){
570         std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
            unsigned(keyOut[i]);
571     }
572     std::cout << std::endl;
573
574     std::cout << "Key Expansion Core End" << std::endl;
575 #endif
576 }
577
578 void AESImplementation::AddRoundKey(byte state[4][4], byte roundKey↩
    [4][4]) {
```

```
579     for(byte i = 0; i < 4; i++){
580         auto* key = reinterpret_cast<uint32_t *>(roundKey[i]);
581         auto* row_state = reinterpret_cast<uint32_t *>(state[i]);
582         *row_state ^= *key;
583     }
584     this->outputState("After Key Addition: ");
585 }
586 
587 void AESImplementation::SubBytes(byte state[4][4]) {
588     for(byte i = 0; i < 4; i++)
589         for(byte j = 0; j < 4; j++)
590             state[i][j] = sbox[state[i][j]];
591     this->outputState("After SubBytes: ");
592 }
593 
594 void AESImplementation::ShiftRows(byte state[4][4]) {
595     for(byte i = 1; i < 4; i++){
596         byte row[4];
597         for(byte j = 0; j < 4; j++)
598             row[j] = state[i][j];
599         for(byte j = 0; j < 4; j++)
600             state[i][j] = row[(i + j) % 4];
601     }
602     this->outputState("After ShiftRows: ");
603 }
604 
605 void AESImplementation::MixColumns(byte state[4][4]) {
606 
607     for(byte i = 0; i < 4; i++){
608         byte column[4] = {state[0][i], state[1][i], state[2][i], state↩
                [3][i],};
609 
610         state[0][i] = galois_mul_2[column[0]] ^ galois_mul_3[column[1]] ^↩
                 column[2] ^ column[3];
611         state[1][i] = column[0] ^ galois_mul_2[column[1]] ^ galois_mul_3[↩
                column[2]] ^ column[3];
612         state[2][i] = column[0] ^ column[1] ^ galois_mul_2[column[2]] ^ ↩
                galois_mul_3[column[3]];
613         state[3][i] = galois_mul_3[column[0]] ^ column[1] ^ column[2] ^ ↩
                galois_mul_2[column[3]];
614     }
615     this->outputState("After MixColumns: ");
616 }
617 
618 void AESImplementation::InverseSubBytes(byte state[4][4]) {
619     for(byte i = 0; i < 4; i++)
620         for(byte j = 0; j < 4; j++)
621             state[i][j] = inverse_sbox[state[i][j]];
622     this->outputState("After InverseSubBytes: ");
623 }
624 
625 void AESImplementation::InverseShiftRows(byte state[4][4]) {
626     for(byte i = 1; i < 4; i++){
627         byte row[4];
628         for(byte j = 0; j < 4; j++)
```

```
629             row[j] = state[i][j];
630         for(int k = 3; k >= 0; k  ){
631             state[i][k] = row[(k + (4   i)) % 4];
632         }
633     }
634     this >outputState("After InverseShiftRows: ");
635 }
636
637 void AESImplementation::InverseMixColumns(byte state[4][4]) {
638     for(byte i = 0; i < 4; i++){
639         byte column[4] = {state[0][i], state[1][i], state[2][i], state↩
                [3][i]};
640
641         state[0][i] = galois_mul_14[column[0]] ^ galois_mul_11[column[1]]↩
                ^ galois_mul_13[column[2]] ^ galois_mul_9[column[3]];
642         state[1][i] = galois_mul_9[column[0]] ^ galois_mul_14[column[1]] ↩
                ^ galois_mul_11[column[2]] ^ galois_mul_13[column[3]];
643         state[2][i] = galois_mul_13[column[0]] ^ galois_mul_9[column[1]] ↩
                ^ galois_mul_14[column[2]] ^ galois_mul_11[column[3]];
644         state[3][i] = galois_mul_11[column[0]] ^ galois_mul_13[column[1]]↩
                ^ galois_mul_9[column[2]] ^ galois_mul_14[column[3]];
645     }
646     this >outputState("After InverseMixColumns: ");
647 }
648
649 void AESImplementation::outputState(std::string prefix) {
650 #if defined(debug) && debug == 1
651     std::cout << prefix << std::endl;
652     for(byte i = 0; i < 4; i++){
653         for(byte j = 0; j < 4; j++)
654             std::cout << std::hex << std::setfill('0') << std::setw(2) <<↩
                    unsigned(this >_state[j][i]);
655     }
656     std::cout << std::endl;
657 #endif
658 }
659
660 void AESImplementation::outputRoundKey() {
661 #if defined(debug) && debug == 1
662     std::cout << "Current Round Key" << std::endl;
663     for(byte i = 0; i < 4; i++){
664         for(byte j = 0; j < 4; j++)
665             std::cout << std::hex << std::setfill('0') << std::setw(2) <<↩
                    unsigned(this >_round_key[j][i]);
666     }
667     std::cout << std::endl;
668 #endif
669 }
```

3.2.3 aes_implementation_test.cc

```
/**
 * @file aes_implementation_test.cc
 * @date 28/02/2018
 *
 * @breif This file will contain test methods to verify the functionality of my implementation of the AES Algorithm
 *        All of the test keys and plaintexts are the same as shown in FIPS 197.
 *
 * @version 0.01
 * @author Jacob Powell
 *
 */

#include "aes_implementation.h"

#include <iostream>
#include <iomanip>

void test_aes_256(){
    AESImplementation aes(AES256);
    byte key_256[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f,
                      0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1a, 0x1b, 0x1c, 0x1d, 0x1e, 0x1f};
    byte plaintext[] = {0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff};
    byte ciphertext[16];
    byte plaintext2[16];

    std::cout << "Plaintext: ";
    for(byte i = 0; i < 16; i++){
        std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(plaintext[i]);
    }
    std::cout << std::endl;

    std::cout << "Key: ";
    for(byte i = 0; i < 32; i++){
        std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(key_256[i]);
    }
    std::cout << std::endl;

    aes.encrypt_block(plaintext, ciphertext, key_256);

    std::cout << "Cipher Text: ";
    for(byte i = 0; i < 16; i++){
        std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(ciphertext[i]);
    }
    std::cout << std::endl;
```

```
45
46      aes.decrypt_block(ciphertext, plaintext2, key_256);
47
48      std::cout << "Plaintext: ";
49      for(byte i = 0; i < 16; i++){
50          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(plaintext2[i]);
51      }
52      std::cout << std::endl;
53
54      std::cout << std::endl;
55  }
56
57  void test_aes_192(){
58      AESImplementation aes(AES192);
59      byte key_192[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0↩
            x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f,
60                        0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17};
61      byte plaintext[] = {0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0↩
            x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff};
62      byte ciphertext[16];
63      byte plaintext2[16];
64
65      std::cout << "Plaintext: ";
66      for(byte i = 0; i < 16; i++){
67          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(plaintext[i]);
68      }
69      std::cout << std::endl;
70
71      std::cout << "Key: ";
72      for(byte i = 0; i < 24; i++){
73          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(key_192[i]);
74      }
75      std::cout << std::endl;
76
77      aes.encrypt_block(plaintext, ciphertext, key_192);
78
79      std::cout << "Cipher Text: ";
80      for(byte i = 0; i < 16; i++){
81          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(ciphertext[i]);
82      }
83      std::cout << std::endl;
84
85      aes.decrypt_block(ciphertext, plaintext2, key_192);
86
87      std::cout << "Plaintext: ";
88      for(byte i = 0; i < 16; i++){
89          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(plaintext2[i]);
90      }
91      std::cout << std::endl;
92
```

```
93      std::cout << std::endl;
94  }
95
96  void test_aes_128(){
97      AESImplementation aes(AES128);
98
99      byte key_128[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0↩
            x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f};
100     byte plaintext[] = {0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0↩
            x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff};
101
102     byte ciphertext[16];
103     byte plaintext2[16];
104
105     std::cout << "Plaintext: ";
106     for(byte i = 0; i < 16; i++){
107         std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(plaintext[i]);
108     }
109     std::cout << std::endl;
110
111     std::cout << "Key: ";
112     for(byte i = 0; i < 16; i++){
113         std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(key_128[i]);
114     }
115     std::cout << std::endl;
116
117     aes.encrypt_block(plaintext, ciphertext, key_128);
118
119     std::cout << "Cipher Text: ";
120     for(byte i = 0; i < 16; i++){
121         std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(ciphertext[i]);
122     }
123     std::cout << std::endl;
124
125     aes.decrypt_block(ciphertext, plaintext2, key_128);
126
127     std::cout << "Plaintext: ";
128     for(byte i = 0; i < 16; i++){
129         std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(plaintext2[i]);
130     }
131     std::cout << std::endl;
132
133     std::cout << std::endl;
134
135 }
136
137
138 int main(){
139     test_aes_128();
140     test_aes_192();
141     test_aes_256();
```

```
142 }
```

3.3 src/auth/

3.3.1 crypto_online_auth_widget.h

```
/**
 * @file crypto_online_auth_widget.h
 * @date 31/01/2018
 *
 * @author Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_AUTH_WIDGET_H
#define CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_AUTH_WIDGET_H


#include <Wt/Auth/AuthWidget.h>
#include "../db/db_interface.h"


/**
 * @class CryptoOnlineAuthWidget
 *
 * @brief Basic implementation of the Authentication widget from the Wt Library. It handles basic user registration,
 *        user login, and linking database information.
 */
class CryptoOnlineAuthWidget : public Wt::Auth::AuthWidget {
public:
    CryptoOnlineAuthWidget(const Wt::Auth::AuthService &baseAuth, Wt::Auth::AbstractUserDatabase &users,
                           Wt::Auth::Login &login, Session& session);

    /**
     * @brief This method overrides the default createRegistrationView method for the Wt::Auth::Widget Class
     *        This method loads the registration view form which allows a user to register and create an account
     *        for the website.
     *
     * @param id Identifies what user has just registered
     * @return
     */
    std::unique_ptr<Wt::WWidget> createRegistrationView(const Wt::Auth::Identity& id) override;


private:
    Session& _session;
};


#endif //CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_AUTH_WIDGET_H
```

3.3.2 crypto_online_auth_widget.cc

```
/**
 * @file crypto_online_auth_widget.cc
 * @date 31/01/2018
 *
 * @author Jacob Powell
 */

#include "crypto_online_auth_widget.h"
#include "crypto_online_register_widget.h"

CryptoOnlineAuthWidget::CryptoOnlineAuthWidget(const Wt::Auth::AuthService &baseAuth,
                                               Wt::Auth::AbstractUserDatabase &users,
                                               Wt::Auth::Login &login, Session& session)
        : AuthWidget(baseAuth, users, login),
          _session(session)

{
    this->model()->addPasswordAuth(
            reinterpret_cast<const Wt::Auth::AbstractPasswordService *>(&Session::passwordAuth()));
    //this->model()->addOAuth(Session::oAuth());
    this->setRegistrationEnabled(true);

    this->processEnvironment();
}


std::unique_ptr<Wt::WWidget> CryptoOnlineAuthWidget::createRegistrationView(const Wt::Auth::Identity& id) {
    auto registerWidget = Wt::cpp14::make_unique<CryptoOnlineRegisterWidget>(this, _session);
    auto model = this->createRegistrationModel();
    if(id.isValid()){
        model->registerIdentified(id);
    }
    registerWidget->setModel(std::move(model));
    return std::move(registerWidget);
}
```

3.3.3 crypto_online_register_widget.h

```
1  //
2  // Created by synx on 3/30/18.
3  //
4
5  #ifndef CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_REGISTER_WIDGET_H
6  #define CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_REGISTER_WIDGET_H
7
8
9  #include <Wt/Auth/RegistrationWidget.h>
10 #include "../db/session.h"
11
12 class CryptoOnlineRegisterWidget : public Wt::Auth::RegistrationWidget {
13 public:
14     CryptoOnlineRegisterWidget(Wt::Auth::AuthWidget* parent, Session& ↩
        session);
15
16     void register_new_user(Wt::Auth::User& user);
17
18 private:
19     Session& _session;
20 };
21
22
23 #endif //CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_REGISTER_WIDGET_H
```

3.3.4 crypto_online_register_widget.cc

```
1  //
2  // Created by synx on 3/30/18.
3  //
4
5  #include "crypto_online_register_widget.h"
6  #include <Wt/WLineEdit.h>
7
8  CryptoOnlineRegisterWidget::CryptoOnlineRegisterWidget(Wt::Auth::↩
       AuthWidget *parent, Session& session) :
9          Wt::Auth::RegistrationWidget(parent),
10         _session(session)
11 {
12
13 }
14
15
16 void CryptoOnlineRegisterWidget::register_new_user(Wt::Auth::User &user) ↩
       {
17     _session.new_registered_user(user);
18 }
```

3.4 src/db/

3.4.1 session.h

```
1  /**
2   * @file session.h
3   * @date 30/01/2018
4   *
5   * @brief This file contains the definition for the Session class
6   * @version 0.02
7   *
8   * @author Jacob Powell
9   */
10
11 #ifndef CRYPTO_ONLINE_PROJECT_SESSION_H
12 #define CRYPTO_ONLINE_PROJECT_SESSION_H
13
14 #include "db_user.h"
15
16 #include <Wt/Auth/Login.h>
17 #include <Wt/Auth/Dbo/UserDatabase.h>
18
19 #include <Wt/Dbo/Session.h>
20 #include <Wt/Dbo/ptr.h>
21
22 using UserDatabase = Wt::Auth::Dbo::UserDatabase<Wt::Auth::Dbo::AuthInfo<↩
       DbUser>>;
23
24 /**
25  * @class Session
26  * @inherit Wt::Dbo::Session
27  *
28  * @version 0.1
29  *
30  * @brief This class deals with how we deal with the conept of sessions ↩
        on the website. It deals with how the user
31  *        interacts with and gets information from the connected database↩
        and provides
32  */
33 class Session : public Wt::Dbo::Session {
34 public:
35     explicit Session(const std::string& sqliteDb);
36
37     static void configureAuth();
38
39     static const Wt::Auth::AuthService& auth();
40     static const Wt::Auth::PasswordService& passwordAuth();
41     static const std::vector<const Wt::Auth::OAuthService *>& oAuth();
42
43     void new_registered_user(Wt::Auth::User& user);
44     void link_account_to_database(const Wt::Auth::User& user);
45     bool does_user_exist_in_dbuser(const Wt::Auth::User& user);
46
```

```
47      Wt::Auth::AbstractUserDatabase& users();
48      Wt::Dbo::ptr<AuthInfo> user();
49      Wt::Auth::Login& login(){ return _login; }
50
51  private:
52      std::unique_ptr<Wt::Auth::Dbo::UserDatabase<Wt::Auth::Dbo::AuthInfo<↩
            DbUser>>> _users;
53      Wt::Auth::Login _login;
54  };
55
56
57  #endif //CRYPTO_ONLINE_PROJECT_SESSION_H
```

3.4.2 session.cc

```
/**
 * @file session.cc
 * @date 30/01/2018
 *
 * @brief
 *
 *
 * @author Jacob Powell
 */

#include "session.h"
#include "db_questions.h"

#include <Wt/Dbo/backend/Sqlite3.h>
#include <Wt/Auth/HashFunction.h>
#include <Wt/Auth/PasswordService.h>
#include <Wt/Auth/PasswordStrengthValidator.h>
#include <Wt/Auth/PasswordVerifier.h>
#include <Wt/Auth/GoogleService.h>
#include <Wt/Auth/FacebookService.h>

#include <string>

namespace {
    Wt::Auth::AuthService myAuthService;
    Wt::Auth::PasswordService myPasswordService{myAuthService};
    std::vector<std::unique_ptr<Wt::Auth::OAuthService>> myOAuth;
}

/**
 * @brief This method connects to the database file and maps the various database classes to tables. It then checks if
 *        the table exits and if it doesnt then it creates it. If it does then it just uses the existing database.
 * @param sqliteDb
 */
Session::Session(const std::string &sqliteDb) {
    auto connection = Wt::cpp14::make_unique<Wt::Dbo::backend::Sqlite3>(sqliteDb);
    connection->setProperty("show queries", "true");
    this->setConnection(std::move(connection));

    this->mapClass<DbUser>("db_user");
    this->mapClass<DbUserAnsweredQuestion>("user_answered_questions");
    this->mapClass<DbQuestions>("questions");
    this->mapClass<Wt::Auth::Dbo::AuthInfo<DbUser>>("auth_info");
    this->mapClass<Wt::Auth::Dbo::AuthInfo<DbUser>::AuthIdentityType>("auth_identity");
    this->mapClass<Wt::Auth::Dbo::AuthInfo<DbUser>::AuthTokenType>("auth_token");

    std::cout << "Checking the Database Contents" << std::endl;
```

```
48      try {
49          this->createTables();
50          std::cerr << "Created Database" << std::endl;
51      }catch(Wt::Dbo::Exception &e){
52          std::cerr << e.what() << std::endl;
53          std::cerr << "Using Existing Database" << std::endl;
54      }
55      this->_users = Wt::cpp14::make_unique<Wt::Auth::Dbo::UserDatabase<Wt↩
            ::Auth::Dbo::AuthInfo<DbUser>>>(*this);
56  }
57
58  /**
59   * @return The database containing all users
60   */
61  Wt::Auth::AbstractUserDatabase& Session::users() {
62      return *_users;
63  }
64
65  /**
66   * @brief Setup the authentication for users to login through. This ↩
        includes setting passwords strength validators.
67   *        Adding throttlers to stop brute force attacks on password ↩
        attempts. Adding hashing to user passwords. It also
68   *        adds OAuth services like Google and Facebook 2.
69   *
70   *        It also determines whether or not a user needs to provide an ↩
        email address and whether or not they need to
71   *        prove they are real by activating their account by clicking an ↩
        authentication email.
72   */
73  void Session::configureAuth() {
74      myAuthService.setAuthTokensEnabled(true, "logincookie");
75      myAuthService.setEmailVerificationEnabled(true);
76      myAuthService.setEmailVerificationRequired(false);
77
78      std::unique_ptr<Wt::Auth::PasswordVerifier> verifier =
79              Wt::cpp14::make_unique<Wt::Auth::PasswordVerifier>();
80      verifier->addHashFunction(Wt::cpp14::make_unique<Wt::Auth::↩
            BCryptHashFunction>(7));
81      myPasswordService.setVerifier(std::move(verifier));
82      myPasswordService.setAttemptThrottlingEnabled(true);
83      myPasswordService.setStrengthValidator(Wt::cpp14::make_unique<Wt::↩
            Auth::PasswordStrengthValidator>());
84
85      if (Wt::Auth::GoogleService::configured()) {
86          std::cout << "GOOGLE SERVICE CONFIGURED" << std::endl;
87          myOAuth.push_back(Wt::cpp14::make_unique<Wt::Auth::GoogleService↩
                >(myAuthService));
88      }
89
90      if (Wt::Auth::FacebookService::configured())
91          myOAuth.push_back(Wt::cpp14::make_unique<Wt::Auth::↩
                FacebookService>(myAuthService));
92
93      for (unsigned i = 0; i < myOAuth.size(); i++)
```

```
94          myOAuth[i]->generateRedirectEndpoint();
95  }
96
97  /**
98   * @return The Auth service being used
99   */
100 const Wt::Auth::AuthService& Session::auth() {
101     return myAuthService;
102 }
103
104 /**
105  * @return The Password Auth being used
106  */
107 const Wt::Auth::PasswordService& Session::passwordAuth() {
108     return myPasswordService;
109 }
110
111 const std::vector<const Wt::Auth::OAuthService *>& Session::oAuth() {
112     std::vector<const Wt::Auth::OAuthService *> result;
113     for (auto &auth : myOAuth) {
114         result.push_back(auth.get());
115     }
116     return result;
117 }
118
119 /**
120  * @return The AuthInfo about the current user logged in
121  */
122 Wt::Dbo::ptr<AuthInfo> Session::user() {
123     if(_login.loggedIn()){
124         Wt::Dbo::ptr<AuthInfo> authInfo = _users->find(_login.user());
125         return authInfo;
126     }else{
127         return Wt::Dbo::ptr<AuthInfo>();
128     }
129 }
130
131 /**
132  * @brief This method handles the registration of a new user
133  *        It is called during a transaction
134  *
135  * @param user The new user that just registered on the site
136  */
137 void Session::new_registered_user(Wt::Auth::User &user) {
138     auto authInfo{_users->find(user)};
139     auto foundUser{authInfo->user()};
140
141     if(!foundUser){
142         foundUser = this->add(Wt::cpp14::make_unique<DbUser>());
143         authInfo.modify()->setUser(foundUser);
144     }
145 }
146
147 /**
```

```
148  * First we check if the user exists in the db_user table and if not we ↩
       then
149  * add to the table. If they do already exist this method exits
150  *
151  * @param user The user we are linking to the user database
152  */
153 void Session::link_account_to_database(const Wt::Auth::User& user) {
154     std::cout << "Link Account Called" << std::endl;
155     if(!this->does_user_exist_in_dbuser(user)){
156         std::cout << "Adding user to the db_user database" << std::endl;
157         Wt::Dbo::Transaction transaction(*this);
158 
159         std::unique_ptr<DbUser> new_user{new DbUser()};
160         new_user->user_id = std::stoi(user.id());
161         new_user->user_identity = user.identity(Wt::Auth::Identity::↩
                LoginName);
162         new_user->user_role = Role::User;
163 
164         Wt::Dbo::ptr<DbUser> userPtr = (*this).add(std::move(new_user));
165     }
166 
167 }
168 
169 /**
170  * @brief This methods gets given a User and then goes through the ↩
       db_user table to check to see if that user
171  *        already exists and does not need to be added.
172  *
173  * @param user The user we are looking for in the db_user table
174  * @return True if the user exits in the db_user table. False if the user↩
       does not
175  */
176 bool Session::does_user_exist_in_dbuser(const Wt::Auth::User &user) {
177 
178     Wt::Dbo::Transaction transaction(*this);
179     const std::string& looking_for_this_id = user.id();
180 
181     std::cout << "Checking for a User with ID: " << looking_for_this_id ↩
           << std::endl;
182 
183     Wt::Dbo::collection< Wt::Dbo::ptr<DbUser> > all_users = this->find<↩
           DbUser>();
184 
185     if(!all_users.empty()){
186         for(const Wt::Dbo::ptr<DbUser>& current_user : all_users){
187             long long int id = current_user.id();
188             std::cout << "Identity: " << id << std::endl;
189             if(std::to_string(id) == looking_for_this_id) {
190                 std::cout << "User already exists in Database" << std::↩
                    endl;
191                 return true;
192             }
193         }
194         return false;
195     }
```

```
196 }
```

3.4.3 db_interface.h

```
/*
 * File: db_interface.h
 * Created: 20/12/2017 19:45
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_DATABASE_INTERFACE_H
#define CRYPTO_ONLINE_PROJECT_DATABASE_INTERFACE_H

#include "session.h"
#include "db_roles.h"
#include "db_user.h"
#include "db_questions.h"


class db_interface;

class db_interface {
public:

    explicit db_interface(Session& session) : current_session(session) {}

    Wt::Dbo::ptr<DbUser> get_user(std::string id);

    void add_answer_to_user(const std::string& answer, int question_id);
    bool check_answer(const std::string& answer, int question_id);
    void set_answer_check_flag(int question_id, bool flag);
    int get_total_questions_answered();
    int get_total_correct_question_answered();


private:

    bool does_answer_exist_user(const int& answer_id);
    Wt::Dbo::ptr<DbUserAnsweredQuestion> get_answer(const int& answer_id)
        ;

    Session& current_session;
};


#endif //CRYPTO_ONLINE_PROJECT_DATABASE_INTERFACE_H
```

3.4.4 db_interface.cc

```
/**
 * File: db_interface.cc
 * Created: 20/12/2017 19:45
 * Finished:
 *
 * Description:
 * @version 0.01
 * Author: Jacob Powell
 */

#include "db_interface.h"

/**
 * @brief This method gets given an id for a user and then searches through the DbUser table for the user with that id.
 *        If it finds a user with that id then it returns a Wt::Dbo::ptr<DbUser> object containing all the information
 *        on that user. If it does not find the user it does nothing.
 *
 * @param looking_for_id The id of the user we are looking for
 * @return The user in db_user with id == looking for id
 */
Wt::Dbo::ptr<DbUser> db_interface::get_user(std::string looking_for_id) {

    std::cerr << "Getting Information for User with ID(" << looking_for_id << ")" << std::endl;

    Wt::Dbo::Transaction transaction(current_session);

    std::cerr << "Getting all user data from the DbUser table" << std::endl;
    Wt::Dbo::collection< Wt::Dbo::ptr<DbUser> > all_users = current_session.find<DbUser>();

    if(!all_users.empty()){
        std::cerr << "DbUser Table is not Empty" << std::endl;
        for(const Wt::Dbo::ptr<DbUser>& current_user : all_users){
            long long int id = current_user.id();
            std::cout << "Identity: " << id << std::endl;
            if(std::to_string(id) == looking_for_id)
                return current_user;
            else{
                std::cout << "User not Found" << std::endl;
            }
        }
    }
    std::cout << "Leaving db_interface::db_user" << std::endl;
}

/**
 * @brief When a user answers a question it stores their answer in the database
```

```
47  */
48 void db_interface::add_answer_to_user(const std::string &answer, const int question_id) {
49
50     Wt::Dbo::Transaction transaction(current_session);
51
52     const Wt::Auth::User& u = current_session.login().user();
53     auto current_user = this->get_user(u.id());
54
55     std::cout << question_id << std::endl;
56
57     if(!this->does_answer_exist_user(question_id)){
58         std::cout << "Adding Question for the first time" << std::endl;
59         std::cout << question_id << std::endl;
60         std::unique_ptr<DbUserAnsweredQuestion> question_answer_1{new DbUserAnsweredQuestion()};
61         question_answer_1->question_id = question_id;
62         question_answer_1->answer_text = answer;
63         question_answer_1->user = current_user;
64         Wt::Dbo::ptr<DbUserAnsweredQuestion> userPtr = current_session.add(std::move(question_answer_1));
65     }else{
66         auto question = this->get_answer(question_id);
67         question.modify()->answer_text = answer;
68     }
69
70 }
71
72 /**
73  * @brief This method searches through the questions a user has saved and determines whether or not
74  *        they have already answered said question.
75  *
76  * @param answer_id The id of the answer we are looking for
77  * @return True if the user has answered that question, False if the user has not.
78  */
79 bool db_interface::does_answer_exist_user(const int &answer_id) {
80     Wt::Dbo::Transaction transaction(current_session);
81
82     const std::string &current_user_id = this->current_session.login().user().id();
83     auto current_user = this->get_user(current_user_id);
84
85     if(current_user->questions.empty()){
86         return false;
87     }else{
88         for(const Wt::Dbo::ptr<DbUserAnsweredQuestion>& question : current_user->questions){
89             std::cout << question->question_id << std::endl;
90             if(question->question_id == answer_id)
91                 return true;
92         }
93         std::cout << "Question already exits" << std::endl;
94         return false;
```

```
95      }
96
97  }
98
99  /**
100  * @brief This method searches through the questions the user has answered and then returns an object
101  *        containing the data on the question we are searching for.
102  *
103  * @param answer_id The answer_id of the question we are looking for
104  * @return An object containing the information on the question begin searched for
105  */
106 Wt::Dbo::ptr<DbUserAnsweredQuestion> db_interface::get_answer(const int &answer_id){
107     std::cout << "GET ANSWER CALLED" << std::endl;
108     Wt::Dbo::Transaction transaction(current_session);
109
110     const std::string &current_user_id = this->current_session.login().user().id();
111
112     return this->current_session.find<DbUserAnsweredQuestion>().where("user_answered_question_id = ?").bind(answer_id)
113             .where("user_id = ?").bind(current_user_id);
114 }
115
116 /**
117  * @brief This method gets an answer and a question and then checks to see if the users answer is right
118  *
119  * @param answer The user answer
120  * @param question_id The id of the question being answered
121  * @return Whether or not the question is right
122  */
123 bool db_interface::check_answer(const std::string &answer, int question_id) {
124     std::cout << "CHECK ANSWER CALLED" << std::endl;
125     Wt::Dbo::Transaction transaction(current_session);
126
127     const std::string& current_user_id = this->current_session.login().user().id();
128
129     std::cout << "Getting User Answered Questions" << std::endl;
130     Wt::Dbo::ptr<DbUserAnsweredQuestion> user_answer = this->current_session.find<DbUserAnsweredQuestion>()
131             .where("user_answered_question_id = ?").bind(question_id)
132             .where("user_id = ?").bind(current_user_id);
133
134     std::cout << "Getting  Questions" << std::endl;
135     Wt::Dbo::ptr<DbQuestions> question = this->current_session.find<DbQuestions>()
136             .where("question_id = ?").bind(question_id);
137     std::cout << "Got Questions" << std::endl;
138
139     return (user_answer->answer_text == question->question_answer);
```

```
140 }
141
142 /**
143  * @brief This method works out how many questions the user has answered
144  * @return THe number of questions the user has attempted
145  */
146 int db_interface::get_total_questions_answered() {
147     Wt::Dbo::Transaction transaction(current_session);
148
149     int number_of_questions = 0;
150
151     auto current_user_id = this->current_session.login().user().id();
152     auto current_user = this->get_user(current_user_id);
153
154     for(const Wt::Dbo::ptr<DbUserAnsweredQuestion> question : ↩
        current_user->questions){
155         number_of_questions++;
156     }
157
158     return number_of_questions;
159 }
160
161 /**
162  * This method works out how many of the quesitons the user has answered ↩
     correctly
163  * @return The number of correctly answered questions
164  */
165 int db_interface::get_total_correct_question_answered() {
166     Wt::Dbo::Transaction transaction(current_session);
167
168     int number_of_correct_questions = 0;
169
170     auto current_user_id = this->current_session.login().user().id();
171     auto current_user = this->get_user(current_user_id);
172
173     for(const Wt::Dbo::ptr<DbUserAnsweredQuestion> question : ↩
        current_user->questions){
174         if(question->is_correct)
175             number_of_correct_questions++;
176     }
177     return number_of_correct_questions;
178 }
179
180 /**
181  * @brief This method sets the is_correct flag in the ↩
     user_answered_question table so I can easily work out whether or
182  * not the user got the question correct
183  */
184 void db_interface::set_answer_check_flag(int question_id, bool check) {
185     Wt::Dbo::Transaction transaction(current_session);
186
187     auto question = this->get_answer(question_id);
188     question.modify()->is_correct = check;
189 }
```

3.4.5 db_roles.h

```
1  /**
2   * @file db_roles.h
3   * @date 19/12/2017
4   *
5   * @brief This file contains an enum which holds the possible 'roles' ↩
        that a user registered to a website could be.
6   * By default this value will be assigned to Visitor when the user first ↩
        comes to the site. When they register then
7   * then will become a User and if they need access to the admin stuff ↩
        then they can get assigned the admin role.
8   * For the user to become an admin they can't access it straight from the↩
         site, they need to be in contact with
9   *
10  * @author Jacob Powell
11  */
12
13 #ifndef CRYPTO_ONLINE_PROJECT_DB_ROLES_H
14 #define CRYPTO_ONLINE_PROJECT_DB_ROLES_H
15 /**
16  * @enum Role
17  *
18  * @brief This enum contains the basic roles that a user can be assigned ↩
        to when joining the website
19  */
20 enum class Role{
21     Visitor = 0, /**< The Visitor role, assigned when the user first ↩
           joins the website */
22     User = 1, /**< The User role, assigned when a user on the website ↩
           registers and creates an account*/
23     Admin = 3 /**< The Admin role, only available to people who have been↩
            assigned this role by the creator of the site */
24 };
25
26 #endif //CRYPTO_ONLINE_PROJECT_DB_ROLES_H
```

3.4.6 db_user.h

```
/**
 * @file db_user.h
 * @date 20/12/2017
 *
 * @brief This file contains the class definition for the db_user class. When reading data from the database, this class
 * is used to hold the temporary data of the user details and when the application is done with the data the object is
 * deleted. This class is the intermediary interface between the application and the database. Data goes from the
 * database and into this class and then we can use it by accessing the object holding the data.
 *
 * @author Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_DB_USER_H
#define CRYPTO_ONLINE_PROJECT_DB_USER_H


#include "db_roles.h"
#include "db_user_answered_question.h"

#include <Wt/Dbo/Dbo.h>
#include <Wt/Dbo/Types.h>
#include <Wt/WGlobal.h>
#include <Wt/WString.h>

#include <string>

class DbUser;

using AuthInfo = Wt::Auth::Dbo::AuthInfo<DbUser>;

class DbUser {
public:

    Wt::Dbo::collection< Wt::Dbo::ptr<DbUserAnsweredQuestion> > questions;

    int user_id;
    Role user_role;
    Wt::WString user_identity;

    template<class Action>
    void persist(Action& a) {

        Wt::Dbo::field(a, user_id, "user_id");
        Wt::Dbo::field(a, user_role, "user_role");
        Wt::Dbo::field(a, user_identity, "user_identity");

        Wt::Dbo::hasMany(a, questions, Wt::Dbo::ManyToOne, "user");
```

```
48      }
49
50  private:
51
52  };
53
54
55
56  #endif //CRYPTO_ONLINE_PROJECT_DB_USER_H
```

3.4.7 db_user.cc

```
1  /*
2   * File: db_user.cc
3   * Created: 20/12/2017 22:07
4   * Finished:
5   *
6   * Description:
7   *
8   * Author: Jacob Powell
9   */
10
11 #include "db_user.h"
```

3.4.8 db_questions.h

```
/**
 * @file db_user.h
 * @date 20/12/2017
 *
 * @brief This file contains the class definition for the db_user class. When reading data from the database, this class
 * is used to hold the temporary data of the user details and when the application is done with the data the object is
 * deleted. This class is the intermediary interface between the application and the database. Data goes from the
 * database and into this class and then we can use it by accessing the object holding the data.
 *
 * @author Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_DB_QUESTIONS_H
#define CRYPTO_ONLINE_PROJECT_DB_QUESTIONS_H


#include <Wt/Dbo/Dbo.h>
#include <Wt/Dbo/Types.h>
#include <Wt/WGlobal.h>
#include <Wt/WString.h>

#include <string>

class DbQuestions;

class DbQuestions {
public:

    std::string question_id;
    std::string question_text;
    std::string question_answer;
    std::string question_option_a;
    std::string question_option_b;
    std::string question_option_c;

    template<class Action>
    void persist(Action& a) {

        Wt::Dbo::field(a, question_id, "question_id");
        Wt::Dbo::field(a, question_text, "question_text");
        Wt::Dbo::field(a, question_answer , "question_answer");
        Wt::Dbo::field(a, question_option_a, "question_option_a");
        Wt::Dbo::field(a, question_option_b, "question_option_b");
        Wt::Dbo::field(a, question_option_c, "question_option_c");
    }

private:

```

```
49 };
50
51
52 #endif //CRYPTO_ONLINE_PROJECT_DB_QUESTIONS_H
```

3.4.9 db_user_answered_question.h

```
/**
 * @file db_user.h
 * @date 20/12/2017
 *
 * @brief This file contains the class definition for the db_user class. When reading data from the database, this class
 * is used to hold the temporary data of the user details and when the application is done with the data the object is
 * deleted. This class is the intermediary interface between the application and the database. Data goes from the
 * database and into this class and then we can use it by accessing the object holding the data.
 *
 * @author Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_DB_QUESTION_H
#define CRYPTO_ONLINE_PROJECT_DB_QUESTION_H


#include "db_roles.h"
#include "db_user.h"

#include <Wt/Dbo/Dbo.h>
#include <Wt/Dbo/Types.h>
#include <Wt/WGlobal.h>
#include <Wt/WString.h>

#include <string>

class DbUserAnsweredQuestion;
class DbUser;

using AuthInfo = Wt::Auth::Dbo::AuthInfo<DbUser>;

class DbUserAnsweredQuestion {
public:

    Wt::Dbo::ptr<DbUser> user;

    int question_id;
    int answer_id;
    bool is_correct;

    std::string question_text;
    std::string answer_text;


    template<class Action>
    void persist(Action& a){

        Wt::Dbo::field(a, question_id, "user_answered_question_id");
```

```
49          Wt::Dbo::field(a, answer_id, "user_answered_answer_id");
50          Wt::Dbo::field(a, question_text, "user_answered_question_text");
51          Wt::Dbo::field(a, answer_text, "user_answered_answer_text");
52          Wt::Dbo::field(a, is_correct, "user_answered_is_correct");
53
54          Wt::Dbo::belongsTo(a, user, "user");
55      }
56  };
57
58
59  #endif //CRYPTO_ONLINE_PROJECT_DB_QUESTION_H
```

3.5 src/layout/

3.5.1 crypto_online_home.h

```
1  /**
2   * @file crypto_online_home.cc
3   * @date 19/12/2017
4   *
5   * @brief This file contains the class and method definitions for the ↩
        CryptoOnlineHome class
6   * @version 0.01
7   * @author Jacob Powell
8   */
9
10 #ifndef CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_HOME_H
11 #define CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_HOME_H
12
13 #include "header/crypto_online_header.h"
14 #include "crypto_online_navigation_grid.h"
15 #include "crypto_online_footer.h"
16 #include "../db/db_interface.h"
17 #include "crypto_online_profile.h"
18 #include "../db/session.h"
19 #include "../auth/crypto_online_auth_widget.h"
20
21 #include <Wt/WContainerWidget.h>
22 #include <Wt/WNavigationBar.h>
23 #include <Wt/WMenu.h>
24 #include <Wt/WText.h>
25 #include <Wt/WGridLayout.h>
26 #include <Wt/WTable.h>
27 #include <Wt/WStackedWidget.h>
28
29 #include <string>
30
31 /**
32  * @class CryptoOnlineHome
33  *
34  * @brief This class serves for the purpose of controlling the main state↩
        of the application. All changes that the user makes
35  * while on the website will first start here and then propagate down ↩
        through the related classes and methods
36  */
37 class CryptoOnlineHome : public Wt::WContainerWidget{
38 public:
39     explicit CryptoOnlineHome(Session& session); /**< Constructor for the↩
          CryptoOnlineHome Class */
40
41 private:
42     void load_home_page(); /**< Loads the Home Page */
43     void load_login_page(); /**< Loads the Login Page */
44     void load_profile_page(); /**< Loads the Profile Page */
45
```

```
46      void load_intro_to_cryptography_page(); /**< Loads the Intro to ↩
            Cryptography Page */
47      void load_modular_arithmetic_page(); /**< Loads the Modular ↩
            Arithmetic Page */
48
49      void load_aes_encryption_example(); /**< Loads the aes encryption ↩
            example page */
50
51      void load_database(); /**< Loads and Connects to the Database */
52      void handleInternalPath(const std::string& string); /**< Handles the ↩
            change in the internal path of the application */
53
54      Session& _current_session;
55
56      std::unique_ptr<Wt::WGridLayout> _grid; /**< Holds the state of the ↩
            front end display of the website */
57
58      std::unique_ptr<CryptoOnlineHeader> _header; /**< Holds the state of ↩
            the header of the website */
59      std::unique_ptr<crypto_online_navigation_grid> _navigation_grid; /**<↩
             Holds the state of the navigation grid of the website */
60      std::unique_ptr<crypto_online_footer> _footer; /**< Holds the state ↩
            of the footer of the website */
61      std::unique_ptr<CryptoOnlineProfile> _profile; /**< Holds the state ↩
            of the profile page of the website */
62
63      std::unique_ptr<CryptoOnlineAuthWidget> _login; /**< Holds the state ↩
            of the login / register page of the website */
64
65
66      db_interface database_interface; /**< Allows the interaction between ↩
            the application and the database */
67
68  };
69
70  #endif //CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_HOME_H
```

3.5.2 crypto_online_home.cc

```
1  /**
2   * @file crypto_online_home.cc
3   * @date 19/12/2017
4   *
5   * @brief This file contains the base structure for how the website looks↩
        and how the user interacts with it.
6   * @version 0.01
7   * @author Jacob Powell
8   */
9
10 #include "crypto_online_home.h"
11 #include "information_content/modular_arithmetic.h"
12 #include "information_content/intro_to_cryptography.h"
13 #include "crypto_online_aes_example.h"
14
15 #include <Wt/WApplication.h>
16 #include <Wt/Dbo/backend/Sqlite3.h>
17
18
19 /**
20  * @brief Constructor for the CryptoOnlineHome class.
21  * The constructor handles the initial loading of the home page. It first↩
        calls the home page which contains the
22  * navigation grid to all the internal pages plus the profile stats and ↩
       user information.
23  *
24  * It also handles the behaviour of the website when the internal path ↩
       changes and correctly navigates to the
25  * appropriate page that the user has requested to navigate to.
26  */
27 CryptoOnlineHome::CryptoOnlineHome(Session& session) : _current_session(↩
       session),
28                                                        database_interface↩
                                                             (session){
29
30     load_home_page();
31     load_database();
32
33     this->handleInternalPath(Wt::WApplication::instance()->internalPath()↩
           );
34
35     Wt::WApplication::instance()->internalPathChanged()
36             .connect(this, &CryptoOnlineHome::handleInternalPath);
37 }
38
39 /**
40  * @brief This method handles any internal path changes to the ↩
       application.
41  * It monitors for any change in the internal path and when it does ↩
       change it calls the appropriate method to deal with
42  * internal path change.
43  *
```

```
44  * @param path The new internal path the application has changed to
45  */
46 void CryptoOnlineHome::handleInternalPath(const std::string &path) {
47
48     Wt::WApplication *app = Wt::WApplication::instance();
49
50     std::cout << "PATH CHANGED: " + path << std::endl;
51
52     if (path == "/home") {
53         load_home_page();
54     } else if (path == "/signout") {
55         _current_session.login().logout();
56         load_home_page();
57         app->setInternalPath("/home", true);
58     } else if (path == "/profile"){
59         load_profile_page();
60     } else if(path == "/login") {
61         load_login_page();
62         app->setInternalPath("/login", true);
63     } else if(path == "/symmetric/modular arithmetic") {
64         load_modular_arithmetic_page();
65     } else if(path == "/basics/concepts"){
66         load_intro_to_cryptography_page();
67     } else if(path == "/aes encryption"){
68         load_aes_encryption_example();
69     }
70         /*
71          * TODO: Add navigation to all parts of the website
72          */
73     else
74         Wt::WApplication::instance()->setInternalPath("/home", true);
75 }
76
77 /**
78  * @brief This method loads the home page of the website
79  */
80 void CryptoOnlineHome::load_home_page() {
81     this->clear();
82     this->setHeight(1770);
83
84     this->_grid = Wt::cpp14::make_unique<Wt::WGridLayout>();
85
86     std::cout << "WE GOT HERE" << std::endl;
87     this->_header = Wt::cpp14::make_unique<CryptoOnlineHeader>(↩
        _current_session);
88     std::cout << "WE GOT HERO" << std::endl;
89     this->_navigation_grid = Wt::cpp14::make_unique<↩
        crypto_online_navigation_grid>();
90     this->_footer = Wt::cpp14::make_unique<crypto_online_footer>();
91
92     this->_grid->addItem(std::move(this->_header), 0, 0);
93     this->_grid->addItem(std::move(this->_navigation_grid), 1, 0);
94     this->_grid->addItem(std::move(this->_footer), 2, 0);
95
96     this->setLayout(std::move(this->_grid));
```

```
97  }
98
99  /**
100  * @brief This method loads the login page of the website
101  */
102 void CryptoOnlineHome::load_login_page() {
103     this->clear();
104     this->setHeight(400);
105
106     this->_grid = Wt::cpp14::make_unique<Wt::WGridLayout>();
107
108     this->_header = Wt::cpp14::make_unique<CryptoOnlineHeader>(↩
            _current_session);
109     this->_login = Wt::cpp14::make_unique<CryptoOnlineAuthWidget>(Session↩
            ::auth(), _current_session.users(), _current_session.login(), ↩
            _current_session);
110     this->_footer = Wt::cpp14::make_unique<crypto_online_footer>();
111
112     this->_grid->addItem(std::move(this->_header), 0, 0);
113     this->_grid->addWidget(std::move(this->_login), 1, 0);
114     this->_grid->addItem(std::move(this->_footer), 2, 0);
115
116     this->setLayout(std::move(this->_grid));
117 }
118
119 /**
120  * @brief This method creates a live connection between the web ↩
        application of the website
121  */
122 void CryptoOnlineHome::load_database() {
123     //database_interface.connect_database();
124 }
125
126 /**
127  * @brief This method loads the profile page of the website
128  */
129 void CryptoOnlineHome::load_profile_page() {
130     this->clear();
131     this->setHeight(400);
132
133     this->_grid = Wt::cpp14::make_unique<Wt::WGridLayout>();
134
135     this->_header = Wt::cpp14::make_unique<CryptoOnlineHeader>(↩
            _current_session);
136     this->_profile = Wt::cpp14::make_unique<CryptoOnlineProfile>(↩
            _current_session);
137     this->_footer = Wt::cpp14::make_unique<crypto_online_footer>();
138
139     this->_grid->addItem(std::move(this->_header), 0, 0);
140     this->_grid->addWidget(std::move(this->_profile), 1, 0, Wt::↩
            AlignmentFlag::Center);
141     this->_grid->addItem(std::move(this->_footer), 2, 0);
142
143     this->setLayout(std::move(this->_grid));
144 }
```

```
145
146 /**
147  * @brief This methods loads the Basic Concepts page
148  */
149 void CryptoOnlineHome::load_intro_to_cryptography_page(){
150     this->clear();
151     this->setHeight(4500);
152
153     std::string title_link = "learning.intro to cryptography.title";
154     std::string content_link = "learning.intro to cryptography.content";
155
156     this->_grid = Wt::cpp14::make_unique<Wt::WGridLayout>();
157
158     this->_header = Wt::cpp14::make_unique<CryptoOnlineHeader>(↩
            _current_session);
159     this->_footer = Wt::cpp14::make_unique<crypto_online_footer>();
160
161     this->_grid->addItem(std::move(this->_header), 0, 0);
162     this->_grid->addWidget(Wt::cpp14::make_unique<intro_to_cryptography>(↩
            _current_session , title_link, content_link), 1, 0, Wt::↩
            AlignmentFlag::Center);
163     this->_grid->addItem(std::move(this->_footer), 2, 0);
164
165     this->setLayout(std::move(this->_grid));
166 }
167
168 /**
169  * @brief This method loads the modular arithmetic of the website
170  */
171 void CryptoOnlineHome::load_modular_arithmetic_page() {
172     this->clear();
173     this->setHeight(2700);
174
175     std::string title_link = "learning.modular arithmetic.title";
176     std::string content_link = "learning.modular arithmetic.content";
177     std::string question_link = "learning.modular arithmetic.answer ↩
            template";
178
179     this->_grid = Wt::cpp14::make_unique<Wt::WGridLayout>();
180
181     this->_header = Wt::cpp14::make_unique<CryptoOnlineHeader>(↩
            _current_session);
182     this->_footer = Wt::cpp14::make_unique<crypto_online_footer>();
183
184     this->_grid->addItem(std::move(this->_header), 0, 0);
185     this->_grid->addWidget(Wt::cpp14::make_unique<modular_arithmetic>(↩
            _current_session, title_link, content_link, question_link),
186                            1, 0, Wt::AlignmentFlag::Center);
187     this->_grid->addItem(std::move(this->_footer), 2, 0);
188
189     this->setLayout(std::move(this->_grid));
190
191 }
192
193 /**
```

```
194  * @brief This method loads the aes encryption example page
195  */
196 void CryptoOnlineHome::load_aes_encryption_example() {
197     this->clear();
198     this->setHeight(1000);
199 
200     this->_grid = Wt::cpp14::make_unique<Wt::WGridLayout>();
201 
202     this->_header = Wt::cpp14::make_unique<CryptoOnlineHeader>(↩
        _current_session);
203     this->_footer = Wt::cpp14::make_unique<crypto_online_footer>();
204 
205     this->_grid->addItem(std::move(this->_header), 0, 0);
206     this->_grid->addWidget(Wt::cpp14::make_unique<CryptoOnlineAESExample↩
        >(), 1, 0);
207     this->_grid->addItem(std::move(this->_footer), 2, 0);
208 
209     this->setLayout(std::move(this->_grid));
210 }
```

3.5.3 crypto_online_navigation_grid.h

```
/*
 * File: crypto_online_home.cc
 * Created: 19/12/2017 13:32
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_NAVIGATION_GRID_H
#define CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_NAVIGATION_GRID_H

#include <Wt/WGridLayout.h>
#include <Wt/WText.h>
#include <Wt/WLink.h>
#include <Wt/WAnchor.h>

class crypto_online_navigation_grid : public Wt::WGridLayout {
public:
    crypto_online_navigation_grid();

private:

    void populate_navigation_grid();
    void setup_basic_contents();
    void setup_symmetric_contents();
    void setup_asymmetric_contents();
    void setup_applications_contents();
    void clear_grid();

    std::unique_ptr<Wt::WText> title_cryptography_basics;
    std::unique_ptr<Wt::WText> title_symmetric;
    std::unique_ptr<Wt::WText> title_asymmetric;
    std::unique_ptr<Wt::WText> title_applications;

    std::unique_ptr<Wt::WText> ↩
        subtitle_introduction_to_symmetric_cryptography;
    std::unique_ptr<Wt::WText> subtitle_des;
    std::unique_ptr<Wt::WText> subtitle_aes;
    std::unique_ptr<Wt::WText> subtitle_more_about_block_ciphers;
    std::unique_ptr<Wt::WText> ↩
        subtitle_introduction_to_asymmetric_cryptography;
    std::unique_ptr<Wt::WText> subtitle_basic_number_theory_for_pk;
    std::unique_ptr<Wt::WText> subtitle_rsa_cryptosystem;
    std::unique_ptr<Wt::WText> subtitle_dlp_cryptosystems;
    std::unique_ptr<Wt::WText> subtitle_ec_cryptosystems;
    std::unique_ptr<Wt::WText> subtitle_digital_signatures;
    std::unique_ptr<Wt::WText> subtitle_hash_functions;
    std::unique_ptr<Wt::WText> subtitle_macs;
    std::unique_ptr<Wt::WText> subtitle_key_establishment;

```

```
51      std::unique_ptr<Wt::WAnchor> cryptography_basics_concepts;
52      std::unique_ptr<Wt::WAnchor> ↩
            cryptography_basics_real_world_applications;
53      std::unique_ptr<Wt::WAnchor> cryptography_basics_concepts_history;
54
55      std::unique_ptr<Wt::WAnchor> symmetric_modular_arithmetic;
56      std::unique_ptr<Wt::WAnchor> symmetric_cipher_types;
57      std::unique_ptr<Wt::WAnchor> symmetric_historical_ciphers;
58
59      std::unique_ptr<Wt::WAnchor> symmetric_randomness;
60      std::unique_ptr<Wt::WAnchor> symmetric_stream_ciphers;
61      std::unique_ptr<Wt::WAnchor> ↩
            symmetric_linear_feedback_shift_registers;
62
63      std::unique_ptr<Wt::WAnchor> symmetric_des_overview;
64      std::unique_ptr<Wt::WAnchor> symmetric_des_feistal_networks;
65      std::unique_ptr<Wt::WAnchor> symmetric_des_internals;
66      std::unique_ptr<Wt::WAnchor> symmetric_des_decryption;
67      std::unique_ptr<Wt::WAnchor> symmetric_des_security;
68      std::unique_ptr<Wt::WAnchor> symmetric_des_alternatives;
69
70      std::unique_ptr<Wt::WAnchor> symmetric_aes_overview;
71      std::unique_ptr<Wt::WAnchor> symmetric_aes_galois_fields;
72      std::unique_ptr<Wt::WAnchor> symmetric_aes_internals;
73      std::unique_ptr<Wt::WAnchor> symmetric_aes_decryption;
74      std::unique_ptr<Wt::WAnchor> ↩
            symmetric_aes_implementations_hardware_software;
75      std::unique_ptr<Wt::WAnchor> symmetric_aes_example;
76
77      std::unique_ptr<Wt::WAnchor> ↩
            symmetric_more_about_block_ciphers_modes_of_operation;
78      std::unique_ptr<Wt::WAnchor> ↩
            symmetric_more_about_block_ciphers_increasing_security;
79      std::unique_ptr<Wt::WAnchor> ↩
            symmetric_more_about_block_ciphers_revisit_of_exhaustive_key_search↩
            ;
80
81      std::unique_ptr<Wt::WAnchor> asymmetric_symmetric_vs_asymmetric;
82      std::unique_ptr<Wt::WAnchor> asymmetric_authenticity_of_public_keys;
83      std::unique_ptr<Wt::WAnchor> ↩
            asymmetric_key_lengths_and_security_levels;
84
85      std::unique_ptr<Wt::WAnchor> asymmetric_euclidean_algorithm;
86      std::unique_ptr<Wt::WAnchor> asymmetric_eulers_phi_function;
87      std::unique_ptr<Wt::WAnchor> ↩
            asymmetric_fermats_little_theorem_and_eulers_theorem;
88
89      std::unique_ptr<Wt::WAnchor> asymmetric_rsa_introduction;
90      std::unique_ptr<Wt::WAnchor> asymmetric_rsa_encryption_decryption;
91      std::unique_ptr<Wt::WAnchor> asymmetric_rsa_key_generation;
92      std::unique_ptr<Wt::WAnchor> ↩
            asymmetric_rsa_encryption_decryption_fast_exponentiation;
93      std::unique_ptr<Wt::WAnchor> asymmetric_rsa_speed_up_techniques;
94      std::unique_ptr<Wt::WAnchor> asymmetric_rsa_finding_large_primes;
95      std::unique_ptr<Wt::WAnchor> asymmetric_rsa_padding;
```

```
96      std::unique_ptr<Wt::WAnchor> asymmetric_rsa_attacks;
97      std::unique_ptr<Wt::WAnchor> ↩
            asymmetric_rsa_implementations_in_hardware_software;
98
99      std::unique_ptr<Wt::WAnchor> asymmetric_dh_dhke;
100     std::unique_ptr<Wt::WAnchor> asymmetric_dh_algebra;
101     std::unique_ptr<Wt::WAnchor> asymmetric_dh_dlp;
102     std::unique_ptr<Wt::WAnchor> asymmetric_dh_security;
103     std::unique_ptr<Wt::WAnchor> asymmetric_dh_elgamal_encryption_scheme;
104     std::unique_ptr<Wt::WAnchor> asymmetric_dh_elgamal_security;
105
106     std::unique_ptr<Wt::WAnchor> asymmetric_ec_definition;
107     std::unique_ptr<Wt::WAnchor> asymmetric_ec_group_operations;
108     std::unique_ptr<Wt::WAnchor> asymmetric_ec_dlp;
109     std::unique_ptr<Wt::WAnchor> asymmetric_ec_dhke;
110     std::unique_ptr<Wt::WAnchor> asymmetric_ec_security;
111     std::unique_ptr<Wt::WAnchor> ↩
            asymmetric_ec_implementations_in_hardware_software;
112
113     std::unique_ptr<Wt::WAnchor> asymmetric_ds_introduction;
114     std::unique_ptr<Wt::WAnchor> asymmetric_ds_rsa;
115     std::unique_ptr<Wt::WAnchor> asymmetric_ds_elgamal;
116     std::unique_ptr<Wt::WAnchor> asymmetric_ds_dsa;
117     std::unique_ptr<Wt::WAnchor> asymmetric_ds_ecdsa;
118
119     std::unique_ptr<Wt::WAnchor> protocols_hf_motivation;
120     std::unique_ptr<Wt::WAnchor> protocols_hf_security_requirements;
121     std::unique_ptr<Wt::WAnchor> protocols_hf_overview;
122     std::unique_ptr<Wt::WAnchor> protocols_hf_sha1;
123     std::unique_ptr<Wt::WAnchor> protocols_hf_sha3;
124     std::unique_ptr<Wt::WAnchor> protocols_hf_bcrypt;
125
126     std::unique_ptr<Wt::WAnchor> protocols_macs_principles;
127     std::unique_ptr<Wt::WAnchor> protocols_macs_hash_functions;
128     std::unique_ptr<Wt::WAnchor> protocols_macs_block_cipher;
129
130     std::unique_ptr<Wt::WAnchor> protocols_key_est_introduction;
131     std::unique_ptr<Wt::WAnchor> protocols_key_est_symmetric;
132     std::unique_ptr<Wt::WAnchor> protocols_key_est_asymmetric;
133
134 };
135
136
137 #endif //CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_NAVIGATION_GRID_H
```

3.5.4 crypto_online_navigation_grid.cc

```
/*
 * File: crypto_online_home.cc
 * Created: 19/12/2017 13:32
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#include "crypto_online_navigation_grid.h"

#include <Wt/WPushButton.h>

crypto_online_navigation_grid::crypto_online_navigation_grid() {
    populate_navigation_grid();
}


void crypto_online_navigation_grid::populate_navigation_grid() {
    this->setup_basic_contents();
    this->setup_symmetric_contents();
    this->setup_asymmetric_contents();
    this->setup_applications_contents();
    this->clear_grid();
}

void crypto_online_navigation_grid::setup_basic_contents() {
    this->title_cryptography_basics = Wt::cpp14::make_unique<Wt::WText>("Cryptography Basics");
    this->title_cryptography_basics->setStyleClass("navigation_grid_title");
    this->addWidget(std::move(this->title_cryptography_basics),0,1);

    this->cryptography_basics_concepts = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/basics/concepts"));
    auto button_cryptography_basics_concepts = Wt::cpp14::make_unique<Wt::WPushButton>("Basic Concepts");
    button_cryptography_basics_concepts->setStyleClass("navigation_grid_item");
    this->cryptography_basics_concepts->addWidget(std::move(button_cryptography_basics_concepts));
    this->addWidget(std::move(cryptography_basics_concepts),1,0, Wt::AlignmentFlag::Center);

    this->cryptography_basics_real_world_applications = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/basics/real world applications"));
    auto button_cryptography_basics_real_world_applications = Wt::cpp14::make_unique<Wt::WPushButton>("Real World Applications");
```

```
41      button_cryptography_basics_real_world_applications->setStyleClass("↩
            navigation-grid-item");
42      this->cryptography_basics_real_world_applications->addWidget(std::↩
            move(button_cryptography_basics_real_world_applications));
43      this->addWidget(std::move(this->↩
            cryptography_basics_real_world_applications),1,1, Wt::↩
            AlignmentFlag::Center);
44
45      this->cryptography_basics_concepts_history = Wt::cpp14::make_unique<↩
            Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/basics/↩
            history"));
46      auto button_cryptography_basics_history = Wt::cpp14::make_unique<Wt::↩
            WPushButton>("History");
47      button_cryptography_basics_history->setStyleClass("↩
            navigation-grid-item");
48      this->cryptography_basics_concepts_history->addWidget(std::move(↩
            button_cryptography_basics_history));
49      this->addWidget(std::move(this->cryptography_basics_concepts_history)↩
            ,1,2, Wt::AlignmentFlag::Center);
50  }
51
52  void crypto_online_navigation_grid::setup_symmetric_contents() {
53      this->title_symmetric = Wt::cpp14::make_unique<Wt::WText>("Symmetric ↩
            Cryptography");
54      this->title_symmetric->setStyleClass("navigation-grid-title");
55      this->addWidget(std::move(this->title_symmetric),2,1);
56
57      this->subtitle_introduction_to_symmetric_cryptography = Wt::cpp14::↩
            make_unique<Wt::WText>("Introduction to Symmetric Cryptography");
58      this->subtitle_introduction_to_symmetric_cryptography->setStyleClass(↩
            "navigation-grid-subtitle");
59      this->addWidget(std::move(this->↩
            subtitle_introduction_to_symmetric_cryptography),3,1);
60
61      this->symmetric_modular_arithmetic = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/modular↩
             arithmetic"));
62      auto button_cryptography_symmetric_modular_arithmetic = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Modular Arithmetic");
63      button_cryptography_symmetric_modular_arithmetic->setStyleClass("↩
            navigation-grid-item");
64      this->symmetric_modular_arithmetic->addWidget(std::move(↩
            button_cryptography_symmetric_modular_arithmetic));
65      this->addWidget(std::move(this->symmetric_modular_arithmetic),4,0, Wt↩
            ::AlignmentFlag::Center);
66
67      this->symmetric_cipher_types = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/symmetric/cipher types"));
68      auto button_cryptography_symmetric_cipher_types = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Cipher Types");
69      button_cryptography_symmetric_cipher_types->setStyleClass("↩
            navigation-grid-item");
70      this->symmetric_cipher_types->addWidget(std::move(↩
            button_cryptography_symmetric_cipher_types));
```

```
71      this->addWidget(std::move(this->symmetric_cipher_types),4,1, Wt::↩
            AlignmentFlag::Center);
72
73      this->symmetric_historical_ciphers = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/↩
            historical ciphers"));
74      auto button_cryptography_symmetric_historical_ciphers = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Historical Ciphers");
75      button_cryptography_symmetric_historical_ciphers->setStyleClass("↩
            navigation_grid_item");
76      this->symmetric_historical_ciphers->addWidget(std::move(↩
            button_cryptography_symmetric_historical_ciphers));
77      this->addWidget(std::move(this->symmetric_historical_ciphers),4,2, Wt↩
            ::AlignmentFlag::Center);
78
79      this->symmetric_randomness = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/symmetric/random number ↩
            generators"));
80      auto button_cryptography_symmetric_randomness = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Random Number Generators");
81      button_cryptography_symmetric_randomness->setStyleClass("↩
            navigation_grid_item");
82      this->symmetric_randomness->addWidget(std::move(↩
            button_cryptography_symmetric_randomness));
83      this->addWidget(std::move(this->symmetric_randomness),5,0, Wt::↩
            AlignmentFlag::Center);
84
85      this->symmetric_stream_ciphers = Wt::cpp14::make_unique<Wt::WAnchor>(↩
            Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/stream ciphers")↩
            );
86      auto button_cryptography_symmetric_stream_ciphers = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Stream Ciphers");
87      button_cryptography_symmetric_stream_ciphers->setStyleClass("↩
            navigation_grid_item");
88      this->symmetric_stream_ciphers->addWidget(std::move(↩
            button_cryptography_symmetric_stream_ciphers));
89      this->addWidget(std::move(this->symmetric_stream_ciphers),5,1, Wt::↩
            AlignmentFlag::Center);
90
91      this->symmetric_linear_feedback_shift_registers = Wt::cpp14::↩
            make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/↩
            symmetric/stream ciphers"));
92      auto button_cryptography_symmetric_lfsr = Wt::cpp14::make_unique<Wt::↩
            WPushButton>("Linear Feedback Shift Registers");
93      button_cryptography_symmetric_lfsr->setStyleClass("↩
            navigation_grid_item");
94      this->symmetric_linear_feedback_shift_registers->addWidget(std::move(↩
            button_cryptography_symmetric_lfsr));
95      this->addWidget(std::move(this->↩
            symmetric_linear_feedback_shift_registers),5,2, Wt::AlignmentFlag↩
            ::Center);
96
97      this->subtitle_des = Wt::cpp14::make_unique<Wt::WText>("DES: Data ↩
            Encryption Standard");
98      this->subtitle_des->setStyleClass("navigation_grid_subtitle");
```

```
99      this->addWidget(std::move(subtitle_des),6,1);
100
101     this->symmetric_des_overview = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/des overview"));
102     auto button_cryptography_symmetric_des_overview = Wt::cpp14::make_unique<Wt::WPushButton>("DES Overview");
103     button_cryptography_symmetric_des_overview->setStyleClass("navigation_grid_item");
104     this->symmetric_des_overview->addWidget(std::move(button_cryptography_symmetric_des_overview));
105     this->addWidget(std::move(this->symmetric_des_overview),7,0, Wt::AlignmentFlag::Center);
106
107     this->symmetric_des_feistal_networks = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/feistal networks"));
108     auto button_cryptography_symmetric_des_feistal_networks = Wt::cpp14::make_unique<Wt::WPushButton>("Feistal Networks");
109     button_cryptography_symmetric_des_feistal_networks->setStyleClass("navigation_grid_item");
110     this->symmetric_des_feistal_networks->addWidget(std::move(button_cryptography_symmetric_des_feistal_networks));
111     this->addWidget(std::move(this->symmetric_des_feistal_networks),7,1, Wt::AlignmentFlag::Center);
112
113     this->symmetric_des_internals = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/des internals"));
114     auto button_cryptography_symmetric_des_internals = Wt::cpp14::make_unique<Wt::WPushButton>("DES Internals");
115     button_cryptography_symmetric_des_internals->setStyleClass("navigation_grid_item");
116     this->symmetric_des_internals->addWidget(std::move(button_cryptography_symmetric_des_internals));
117     this->addWidget(std::move(this->symmetric_des_internals),7,2, Wt::AlignmentFlag::Center);
118
119     this->symmetric_des_decryption = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/des decryption");
120     auto button_cryptography_symmetric_des_decryption = Wt::cpp14::make_unique<Wt::WPushButton>("DES Decryption");
121     button_cryptography_symmetric_des_decryption->setStyleClass("navigation_grid_item");
122     this->symmetric_des_decryption->addWidget(std::move(button_cryptography_symmetric_des_decryption));
123     this->addWidget(std::move(this->symmetric_des_decryption),8,0, Wt::AlignmentFlag::Center);
124
125     this->symmetric_des_security = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/des security"));
126     auto button_cryptography_symmetric_des_security = Wt::cpp14::make_unique<Wt::WPushButton>("DES Security");
127     button_cryptography_symmetric_des_security->setStyleClass("navigation_grid_item");
```

```
128     this->symmetric_des_security->addWidget(std::move(↩
            button_cryptography_symmetric_des_security));
129     this->addWidget(std::move(this->symmetric_des_security),8,1, Wt::↩
            AlignmentFlag::Center);
130 
131     this->symmetric_des_alternatives = Wt::cpp14::make_unique<Wt::WAnchor↩
            >(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/des ↩
            alternatives"));
132     auto button_cryptography_symmetric_des_alternatives = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("DES Alternatives");
133     button_cryptography_symmetric_des_alternatives->setStyleClass("↩
            navigation_grid_item");
134     this->symmetric_des_alternatives->addWidget(std::move(↩
            button_cryptography_symmetric_des_alternatives));
135     this->addWidget(std::move(this->symmetric_des_alternatives),8,2, Wt::↩
            AlignmentFlag::Center);
136 
137     this->subtitle_aes = Wt::cpp14::make_unique<Wt::WText>("AES: Advanced↩
             Encryption Standard");
138     this->subtitle_aes->setStyleClass("navigation_grid_subtitle");
139     this->addWidget(std::move(subtitle_aes),9,1);
140 
141     this->symmetric_aes_overview = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/symmetric/aes overview"));
142     auto button_cryptography_symmetric_aes_overview = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("AES Overview");
143     button_cryptography_symmetric_aes_overview->setStyleClass("↩
            navigation_grid_item");
144     this->symmetric_aes_overview->addWidget(std::move(↩
            button_cryptography_symmetric_aes_overview));
145     this->addWidget(std::move(this->symmetric_aes_overview),10,0, Wt::↩
            AlignmentFlag::Center);
146 
147     this->symmetric_aes_galois_fields = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/aes ↩
            galois fields"));
148     auto button_cryptography_symmetric_aes_galois_fields = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("AES Galois Fields");
149     button_cryptography_symmetric_aes_galois_fields->setStyleClass("↩
            navigation_grid_item");
150     this->symmetric_aes_galois_fields->addWidget(std::move(↩
            button_cryptography_symmetric_aes_galois_fields));
151     this->addWidget(std::move(this->symmetric_aes_galois_fields),10,1, Wt↩
            ::AlignmentFlag::Center);
152 
153     this->symmetric_aes_internals = Wt::cpp14::make_unique<Wt::WAnchor>(↩
            Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/aes internals"))↩
            ;
154     auto button_cryptography_symmetric_aes_internals = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("AES Internals");
155     button_cryptography_symmetric_aes_internals->setStyleClass("↩
            navigation_grid_item");
156     this->symmetric_aes_internals->addWidget(std::move(↩
            button_cryptography_symmetric_aes_internals));
```

```
157     this->addWidget(std::move(this->symmetric_aes_internals),10,2, Wt::↩
            AlignmentFlag::Center);
158 
159     this->symmetric_aes_decryption = Wt::cpp14::make_unique<Wt::WAnchor>(↩
            Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/aes decryption")↩
            );
160     auto button_cryptography_symmetric_aes_decryption = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("AES Decryption");
161     button_cryptography_symmetric_aes_decryption->setStyleClass("↩
            navigation_grid_item");
162     this->symmetric_aes_decryption->addWidget(std::move(↩
            button_cryptography_symmetric_aes_decryption));
163     this->addWidget(std::move(this->symmetric_aes_decryption),11,0, Wt::↩
            AlignmentFlag::Center);
164 
165     this->symmetric_aes_implementations_hardware_software = Wt::cpp14::↩
            make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/↩
            symmetric/aes implementations hardware software"));
166     auto button_cryptography_symmetric_aes_implementations = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("AES Implementations in Hardware and ↩
            Software");
167     button_cryptography_symmetric_aes_implementations->setStyleClass("↩
            navigation_grid_item");
168     this->symmetric_aes_implementations_hardware_software->addWidget(std↩
            ::move(button_cryptography_symmetric_aes_implementations));
169     this->addWidget(std::move(this->↩
            symmetric_aes_implementations_hardware_software),11,1, Wt::↩
            AlignmentFlag::Center);
170 
171     this->symmetric_aes_example = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/symmetric/aes example"));
172     auto button_cryptography_symmetric_aes_example = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("AES Example");
173     button_cryptography_symmetric_aes_example->setStyleClass("↩
            navigation_grid_item");
174     this->symmetric_aes_example->addWidget(std::move(↩
            button_cryptography_symmetric_aes_example));
175     this->addWidget(std::move(this->symmetric_aes_example),11,2, Wt::↩
            AlignmentFlag::Center);
176 
177     this->subtitle_more_about_block_ciphers = Wt::cpp14::make_unique<Wt::↩
            WText>("More about Block Ciphers");
178     this->subtitle_more_about_block_ciphers->setStyleClass("↩
            navigation_grid_subtitle");
179     this->addWidget(std::move(this->subtitle_more_about_block_ciphers)↩
            ,12,1);
180 
181     this->symmetric_more_about_block_ciphers_modes_of_operation = Wt::↩
            cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::↩
            InternalPath, "/symmetric/modes of operation"));
182     auto ↩
            button_cryptography_symmetric_more_about_block_ciphers_modes_of_operation↩
             = Wt::cpp14::make_unique<Wt::WPushButton>("Modes of Operation");
183     button_cryptography_symmetric_more_about_block_ciphers_modes_of_operation↩
            ->setStyleClass("navigation_grid_item");
```

```
184     this->symmetric_more_about_block_ciphers_modes_of_operation->addWidget(std::move(button_cryptography_symmetric_more_about_block_ciphers_modes_of_operation));
185     this->addWidget(std::move(this->symmetric_more_about_block_ciphers_modes_of_operation),13,0, Wt::AlignmentFlag::Center);
186
187     this->symmetric_more_about_block_ciphers_increasing_security = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/increasing block cipher security"));
188     auto button_cryptography_symmetric_more_about_block_ciphers_increasing_security = Wt::cpp14::make_unique<Wt::WPushButton>("Increasing security of Block Ciphers");
189     button_cryptography_symmetric_more_about_block_ciphers_increasing_security->setStyleClass("navigation_grid_item");
190     this->symmetric_more_about_block_ciphers_increasing_security->addWidget(std::move(button_cryptography_symmetric_more_about_block_ciphers_increasing_security));
191     this->addWidget(std::move(this->symmetric_more_about_block_ciphers_increasing_security),13,1, Wt::AlignmentFlag::Center);
192
193     this->symmetric_more_about_block_ciphers_revisit_of_exhaustive_key_search = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/symmetric/revisit of exhaustive key search"));
194     auto button_cryptography_symmetric_more_about_block_ciphers_revisit_of_exhaustive_key
          = Wt::cpp14::make_unique<Wt::WPushButton>("Revisit of Exhaustive Key Search");
195     button_cryptography_symmetric_more_about_block_ciphers_revisit_of_exhaustive_key_sea
          ->setStyleClass("navigation_grid_item");
196     this->symmetric_more_about_block_ciphers_revisit_of_exhaustive_key_search->addWidget(std::move(button_cryptography_symmetric_more_about_block_ciphers_revisit_of_exhaustive_key
          ));
197     this->addWidget(std::move(this->symmetric_more_about_block_ciphers_revisit_of_exhaustive_key_search),13,2, Wt::AlignmentFlag::Center);
198 }
199
200 void crypto_online_navigation_grid::setup_asymmetric_contents() {
201     this->title_asymmetric = Wt::cpp14::make_unique<Wt::WText>("Asymmetric Cryptography");
202     this->title_asymmetric->setStyleClass("navigation_grid_title");
203     this->addWidget(std::move(this->title_asymmetric),14,1);
204
205     this->subtitle_introduction_to_asymmetric_cryptography = Wt::cpp14::make_unique<Wt::WText>("Introduction to Asymmetric Cryptography");
206     this->subtitle_introduction_to_asymmetric_cryptography->setStyleClass("navigation_grid_subtitle");
```

```
207     this->addWidget(std::move(this->↩
            subtitle_introduction_to_asymmetric_cryptography),15,1);
208
209     this->asymmetric_symmetric_vs_asymmetric = Wt::cpp14::make_unique<Wt↩
            ::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/↩
            symmetric vs asymmetric"));
210     auto button_cryptography_asymmetric_vs_symmetric = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Symmetric vs Asymmetric");
211     button_cryptography_asymmetric_vs_symmetric->setStyleClass("↩
            navigation_grid_item");
212     this->asymmetric_symmetric_vs_asymmetric->addWidget(std::move(↩
            button_cryptography_asymmetric_vs_symmetric));
213     this->addWidget(std::move(asymmetric_symmetric_vs_asymmetric),16,0, ↩
            Wt::AlignmentFlag::Center);
214
215     this->asymmetric_authenticity_of_public_keys = Wt::cpp14::make_unique↩
            <Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/↩
            authenticity of public keys"));
216     auto button_cryptography_asymmetric_authenticity_of_public_keys = Wt↩
            ::cpp14::make_unique<Wt::WPushButton>("Authenticity of Public Keys↩
            ");
217     button_cryptography_asymmetric_authenticity_of_public_keys->↩
            setStyleClass("navigation_grid_item");
218     this->asymmetric_authenticity_of_public_keys->addWidget(std::move(↩
            button_cryptography_asymmetric_authenticity_of_public_keys));
219     this->addWidget(std::move(asymmetric_authenticity_of_public_keys)↩
            ,16,1, Wt::AlignmentFlag::Center);
220
221     this->asymmetric_key_lengths_and_security_levels = Wt::cpp14::↩
            make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/↩
            asymmetric/key lengths and security levels"));
222     auto button_cryptography_asymmetric_key_lengths_security_levels = Wt↩
            ::cpp14::make_unique<Wt::WPushButton>("Key Lengths and Security ↩
            Lengths");
223     button_cryptography_asymmetric_key_lengths_security_levels->↩
            setStyleClass("navigation_grid_item");
224     this->asymmetric_key_lengths_and_security_levels->addWidget(std::move↩
            (button_cryptography_asymmetric_key_lengths_security_levels));
225     this->addWidget(std::move(asymmetric_key_lengths_and_security_levels)↩
            ,16,2, Wt::AlignmentFlag::Center);
226
227     this->subtitle_basic_number_theory_for_pk = Wt::cpp14::make_unique<Wt↩
            ::WText>("Basic Number Theory for Public Key Algorithms");
228     this->subtitle_basic_number_theory_for_pk->setStyleClass("↩
            navigation_grid_subtitle");
229     this->addWidget(std::move(this->subtitle_basic_number_theory_for_pk)↩
            ,17,1);
230
231     this->asymmetric_euclidean_algorithm = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/ea eea↩
            "));
232     auto button_cryptography_asymmetric_euclidean_algorithm = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("EA and EEA Algorithms");
233     button_cryptography_asymmetric_euclidean_algorithm->setStyleClass("↩
            navigation_grid_item");
```

```
234     this->asymmetric_euclidean_algorithm->addWidget(std::move(↩
            button_cryptography_asymmetric_euclidean_algorithm));
235     this->addWidget(std::move(asymmetric_euclidean_algorithm),18,0, Wt::↩
            AlignmentFlag::Center);
236
237     this->asymmetric_eulers_phi_function = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/euler's↩
            s phi function"));
238     auto button_cryptography_asymmetric_phi_function = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Euler's Phi Function");
239     button_cryptography_asymmetric_phi_function->setStyleClass("↩
            navigation_grid_item");
240     this->asymmetric_eulers_phi_function->addWidget(std::move(↩
            button_cryptography_asymmetric_phi_function));
241     this->addWidget(std::move(asymmetric_eulers_phi_function),18,1, Wt::↩
            AlignmentFlag::Center);
242
243     this->asymmetric_fermats_little_theorem_and_eulers_theorem = Wt::↩
            cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::↩
            InternalPath, "/asymmetric/fermats little theorem eulers theorem")↩
            );
244     auto button_cryptography_fermats_little_theorem_and_eulers_theorem= ↩
            Wt::cpp14::make_unique<Wt::WPushButton>("Fermats Little Theorem ↩
            and Eulers Theorem");
245     button_cryptography_fermats_little_theorem_and_eulers_theorem->↩
            setStyleClass("navigation_grid_item");
246     this->asymmetric_fermats_little_theorem_and_eulers_theorem->addWidget↩
            (std::move(↩
            button_cryptography_fermats_little_theorem_and_eulers_theorem));
247     this->addWidget(std::move(↩
            asymmetric_fermats_little_theorem_and_eulers_theorem),18,2, Wt::↩
            AlignmentFlag::Center);
248
249     this->subtitle_rsa_cryptosystem = Wt::cpp14::make_unique<Wt::WText>("↩
            The RSA Cryptosystem");
250     this->subtitle_rsa_cryptosystem->setStyleClass("↩
            navigation_grid_subtitle");
251     this->addWidget(std::move(this->subtitle_rsa_cryptosystem),19,1);
252
253     this->asymmetric_rsa_introduction = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/rsa ↩
            introduction"));
254     auto button_cryptography_asymmetric_rsa_introduction = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("RSA Introduction");
255     button_cryptography_asymmetric_rsa_introduction->setStyleClass("↩
            navigation_grid_item");
256     this->asymmetric_rsa_introduction->addWidget(std::move(↩
            button_cryptography_asymmetric_rsa_introduction));
257     this->addWidget(std::move(asymmetric_rsa_introduction),20,0, Wt::↩
            AlignmentFlag::Center);
258
259     this->asymmetric_rsa_encryption_decryption = Wt::cpp14::make_unique<↩
            Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/↩
            rsa encryption decryption"));
```

```
260     auto button_cryptography_asymmetric_rsa_encryption_decryption = Wt::↩
            cpp14::make_unique<Wt::WPushButton>("RSA Encryption and Decryption↩
            ");
261     button_cryptography_asymmetric_rsa_encryption_decryption->↩
            setStyleClass("navigation_grid_item");
262     this->asymmetric_rsa_encryption_decryption->addWidget(std::move(↩
            button_cryptography_asymmetric_rsa_encryption_decryption));
263     this->addWidget(std::move(asymmetric_rsa_encryption_decryption),20,1,↩
             Wt::AlignmentFlag::Center);
264
265     this->asymmetric_rsa_key_generation = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/rsa ↩
            key generation"));
266     auto button_cryptography_asymmetric_rsa_key_generation = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("RSA Key Generation");
267     button_cryptography_asymmetric_rsa_key_generation->setStyleClass("↩
            navigation_grid_item");
268     this->asymmetric_rsa_key_generation->addWidget(std::move(↩
            button_cryptography_asymmetric_rsa_key_generation));
269     this->addWidget(std::move(asymmetric_rsa_key_generation),20,2, Wt::↩
            AlignmentFlag::Center);
270
271     this->asymmetric_rsa_encryption_decryption_fast_exponentiation= Wt::↩
            cpp14::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::↩
            InternalPath, "/asymmetric/rsa fast exponentiation"));
272     auto ↩
            button_cryptography_asymmetric_rsa_encryption_decryption_fast_exponentiation↩
             = Wt::cpp14::make_unique<Wt::WPushButton>("RSA Encryption and ↩
            Decryption with Fast Exponentiation");
273     button_cryptography_asymmetric_rsa_encryption_decryption_fast_exponentiation↩
             ->setStyleClass("navigation_grid_item");
274     this->asymmetric_rsa_encryption_decryption_fast_exponentiation->↩
            addWidget(std::move(↩
            button_cryptography_asymmetric_rsa_encryption_decryption_fast_exponentiation↩
            ));
275     this->addWidget(std::move(↩
            asymmetric_rsa_encryption_decryption_fast_exponentiation),21,0, Wt↩
            ::AlignmentFlag::Center);
276
277     this->asymmetric_rsa_speed_up_techniques = Wt::cpp14::make_unique<Wt↩
            ::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/rsa ↩
            speed up techniques"));
278     auto button_cryptography_asymmetric_rsa_speed_up_techniques = Wt::↩
            cpp14::make_unique<Wt::WPushButton>("RSA Speed Up Techniques");
279     button_cryptography_asymmetric_rsa_speed_up_techniques->setStyleClass↩
            ("navigation_grid_item");
280     this->asymmetric_rsa_speed_up_techniques->addWidget(std::move(↩
            button_cryptography_asymmetric_rsa_speed_up_techniques));
281     this->addWidget(std::move(asymmetric_rsa_speed_up_techniques),21,1, ↩
            Wt::AlignmentFlag::Center);
282
283     this->asymmetric_rsa_finding_large_primes = Wt::cpp14::make_unique<Wt↩
            ::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/rsa ↩
            finding large primes"));
```

```
284     auto button_cryptography_asymmetric_rsa_finding_large_primes = Wt::↩
            cpp14::make_unique<Wt::WPushButton>("RSA Finding Large Primes");
285     button_cryptography_asymmetric_rsa_finding_large_primes->↩
            setStyleClass("navigation_grid_item");
286     this->asymmetric_rsa_finding_large_primes->addWidget(std::move(↩
            button_cryptography_asymmetric_rsa_finding_large_primes));
287     this->addWidget(std::move(asymmetric_rsa_finding_large_primes),21,2, ↩
            Wt::AlignmentFlag::Center);
288
289     this->asymmetric_rsa_padding = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/asymmetric/rsa padding"));
290     auto button_cryptography_asymmetric_rsa_padding = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("RSA Padding");
291     button_cryptography_asymmetric_rsa_padding->setStyleClass("↩
            navigation_grid_item");
292     this->asymmetric_rsa_padding->addWidget(std::move(↩
            button_cryptography_asymmetric_rsa_padding));
293     this->addWidget(std::move(asymmetric_rsa_padding),22,0, Wt::↩
            AlignmentFlag::Center);
294
295     this->asymmetric_rsa_attacks = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/asymmetric/rsa finding large↩
             primes"));
296     auto button_cryptography_asymmetric_rsa_attacks = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("RSA Attacks");
297     button_cryptography_asymmetric_rsa_attacks->setStyleClass("↩
            navigation_grid_item");
298     this->asymmetric_rsa_attacks->addWidget(std::move(↩
            button_cryptography_asymmetric_rsa_attacks));
299     this->addWidget(std::move(asymmetric_rsa_attacks),22,1, Wt::↩
            AlignmentFlag::Center);
300
301     this->asymmetric_rsa_implementations_in_hardware_software = Wt::cpp14↩
            ::make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "↩
            /asymmetric/rsa finding large primes"));
302     auto ↩
            button_cryptography_asymmetric_rsa_implementations_in_hardware_software↩
             = Wt::cpp14::make_unique<Wt::WPushButton>("RSA Implementations in↩
             Hardware and Software");
303     button_cryptography_asymmetric_rsa_implementations_in_hardware_software↩
            ->setStyleClass("navigation_grid_item");
304     this->asymmetric_rsa_implementations_in_hardware_software->addWidget(↩
            std::move(↩
            button_cryptography_asymmetric_rsa_implementations_in_hardware_software↩
            ));
305     this->addWidget(std::move(↩
            asymmetric_rsa_implementations_in_hardware_software),22,2, Wt::↩
            AlignmentFlag::Center);
306
307     this->subtitle_dlp_cryptosystems = Wt::cpp14::make_unique<Wt::WText>(↩
            "Public Key Cryptosystems based on the Discrete Logarithm Problem"↩
            );
308     this->subtitle_dlp_cryptosystems->setStyleClass("↩
            navigation_grid_subtitle");
309     this->addWidget(std::move(this->subtitle_dlp_cryptosystems),23,1);
```

```
310
311     this->asymmetric_dh_dhke = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/asymmetric/dh dhke"));
312     auto button_cryptography_asymmetric_dh_dhke = Wt::cpp14::make_unique<↩
            Wt::WPushButton>("Diffie Hellman Key Exchange");
313     button_cryptography_asymmetric_dh_dhke->setStyleClass("↩
            navigation-grid-item");
314     this->asymmetric_dh_dhke->addWidget(std::move(↩
            button_cryptography_asymmetric_dh_dhke));
315     this->addWidget(std::move(asymmetric_dh_dhke),24,0, Wt::AlignmentFlag↩
            ::Center);
316
317     this->asymmetric_dh_algebra= Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/asymmetric/dh algebra"));
318     auto button_cryptography_asymmetric_dh_algebra = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Diffie Hellman Algebra");
319     button_cryptography_asymmetric_dh_algebra->setStyleClass("↩
            navigation-grid-item");
320     this->asymmetric_dh_algebra->addWidget(std::move(↩
            button_cryptography_asymmetric_dh_algebra));
321     this->addWidget(std::move(asymmetric_dh_algebra),24,1, Wt::↩
            AlignmentFlag::Center);
322
323     this->asymmetric_dh_dlp = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/asymmetric/dh dlp"));
324     auto button_cryptography_asymmetric_dh_dlp = Wt::cpp14::make_unique<↩
            Wt::WPushButton>("The Discrete Logarithm Problem");
325     button_cryptography_asymmetric_dh_dlp->setStyleClass("↩
            navigation-grid-item");
326     this->asymmetric_dh_dlp->addWidget(std::move(↩
            button_cryptography_asymmetric_dh_dlp));
327     this->addWidget(std::move(asymmetric_dh_dlp),24,2, Wt::AlignmentFlag↩
            ::Center);
328
329     this->asymmetric_dh_security = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/asymmetric/dh security"));
330     auto button_cryptography_asymmetric_dh_security = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Diffie Hellman Security");
331     button_cryptography_asymmetric_dh_security->setStyleClass("↩
            navigation-grid-item");
332     this->asymmetric_dh_security->addWidget(std::move(↩
            button_cryptography_asymmetric_dh_security));
333     this->addWidget(std::move(asymmetric_dh_security),25,0, Wt::↩
            AlignmentFlag::Center);
334
335     this->asymmetric_dh_elgamal_encryption_scheme = Wt::cpp14::↩
            make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/↩
            asymmetric/dh dlp"));
336     auto button_cryptography_asymmetric_dh_encryption_scheme = Wt::cpp14↩
            ::make_unique<Wt::WPushButton>("The Elgamal Encryption Scheme");
337     button_cryptography_asymmetric_dh_encryption_scheme->setStyleClass("↩
            navigation-grid-item");
338     this->asymmetric_dh_elgamal_encryption_scheme->addWidget(std::move(↩
            button_cryptography_asymmetric_dh_encryption_scheme));
```

```
339     this->addWidget(std::move(asymmetric_dh_elgamal_encryption_scheme)↩
            ,25,1, Wt::AlignmentFlag::Center);
340
341     this->asymmetric_dh_elgamal_security = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/dh dlp↩
            "));
342     auto button_cryptography_asymmetric_dh_elgamal_security  = Wt::cpp14↩
            ::make_unique<Wt::WPushButton>("Elgamal Security");
343     button_cryptography_asymmetric_dh_elgamal_security->setStyleClass("↩
            navigation_grid_item");
344     this->asymmetric_dh_elgamal_security->addWidget(std::move(↩
            button_cryptography_asymmetric_dh_elgamal_security));
345     this->addWidget(std::move(asymmetric_dh_elgamal_security),25,2, Wt::↩
            AlignmentFlag::Center);
346
347     this->subtitle_ec_cryptosystems = Wt::cpp14::make_unique<Wt::WText>("↩
            Elliptic Curve Cryptosystems");
348     this->subtitle_ec_cryptosystems->setStyleClass("↩
            navigation_grid_subtitle");
349     this->addWidget(std::move(this->subtitle_ec_cryptosystems),26,1);
350
351     this->asymmetric_ec_definition = Wt::cpp14::make_unique<Wt::WAnchor>(↩
            Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/ec definition")↩
            );
352     auto button_cryptography_asymmetric_ec_definition  = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Elliptic Curve Definition");
353     button_cryptography_asymmetric_ec_definition->setStyleClass("↩
            navigation_grid_item");
354     this->asymmetric_ec_definition->addWidget(std::move(↩
            button_cryptography_asymmetric_ec_definition));
355     this->addWidget(std::move(asymmetric_ec_definition),27,0, Wt::↩
            AlignmentFlag::Center);
356
357     this->asymmetric_ec_group_operations = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/ec ↩
            group operations"));
358     auto button_cryptography_asymmetric_ec_group_operations = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Elliptic Curve Group Operations");
359     button_cryptography_asymmetric_ec_group_operations->setStyleClass("↩
            navigation_grid_item");
360     this->asymmetric_ec_group_operations->addWidget(std::move(↩
            button_cryptography_asymmetric_ec_group_operations));
361     this->addWidget(std::move(asymmetric_ec_group_operations),27,1, Wt::↩
            AlignmentFlag::Center);
362
363     this->asymmetric_ec_dlp = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/asymmetric/ec dlp"));
364     auto button_cryptography_asymmetric_ec_dlp  = Wt::cpp14::make_unique<↩
            Wt::WPushButton>("Building a DLP with Elliptic Curves");
365     button_cryptography_asymmetric_ec_dlp->setStyleClass("↩
            navigation_grid_item");
366     this->asymmetric_ec_dlp->addWidget(std::move(↩
            button_cryptography_asymmetric_ec_dlp));
367     this->addWidget(std::move(asymmetric_ec_dlp),27,2, Wt::AlignmentFlag↩
            ::Center);
```

```
368
369     this->asymmetric_ec_dhke = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/asymmetric/ec dhke"));
370     auto button_cryptography_asymmetric_ec_dhke = Wt::cpp14::make_unique<↩
            Wt::WPushButton>("DH Key Exchange with Elliptic Curves");
371     button_cryptography_asymmetric_ec_dhke->setStyleClass("↩
            navigation-grid-item");
372     this->asymmetric_ec_dhke->addWidget(std::move(↩
            button_cryptography_asymmetric_ec_dhke));
373     this->addWidget(std::move(asymmetric_ec_dhke),28,0, Wt::AlignmentFlag↩
            ::Center);
374
375     this->asymmetric_ec_security = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/asymmetric/ec security"));
376     auto button_cryptography_asymmetric_ec_security = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Elliptic Curve Security");
377     button_cryptography_asymmetric_ec_security->setStyleClass("↩
            navigation-grid-item");
378     this->asymmetric_ec_security->addWidget(std::move(↩
            button_cryptography_asymmetric_ec_security));
379     this->addWidget(std::move(asymmetric_ec_security),28,1, Wt::↩
            AlignmentFlag::Center);
380
381     this->asymmetric_ec_implementations_in_hardware_software= Wt::cpp14::↩
            make_unique<Wt::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/↩
            asymmetric/ec implementations"));
382     auto ↩
            button_cryptography_asymmetric_ec_implementations_in_hardware_software↩
              = Wt::cpp14::make_unique<Wt::WPushButton>("Elliptic Curve ↩
            Implementations in Hardware and Software");
383     button_cryptography_asymmetric_ec_implementations_in_hardware_software↩
             ->setStyleClass("navigation-grid-item");
384     this->asymmetric_ec_implementations_in_hardware_software->addWidget(↩
            std::move(↩
            button_cryptography_asymmetric_ec_implementations_in_hardware_software↩
            ));
385     this->addWidget(std::move(↩
            asymmetric_ec_implementations_in_hardware_software),28,2, Wt::↩
            AlignmentFlag::Center);
386
387     this->subtitle_digital_signatures = Wt::cpp14::make_unique<Wt::WText↩
            >("Digital Signatures");
388     this->subtitle_digital_signatures->setStyleClass("↩
            navigation-grid-subtitle");
389     this->addWidget(std::move(this->subtitle_digital_signatures),29,1);
390
391     this->asymmetric_ds_introduction= Wt::cpp14::make_unique<Wt::WAnchor↩
            >(Wt::WLink(Wt::LinkType::InternalPath, "/asymmetric/ds ↩
            introduction"));
392     auto button_cryptography_asymmetric_ds_introduction = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Introduction to Digital Signatures")↩
            ;
393     button_cryptography_asymmetric_ds_introduction->setStyleClass("↩
            navigation-grid-item");
```

```
394     this->asymmetric_ds_introduction->addWidget(std::move(↩
            button_cryptography_asymmetric_ds_introduction));
395     this->addWidget(std::move(asymmetric_ds_introduction),30,0, Wt::↩
            AlignmentFlag::Center);
396
397     this->asymmetric_ds_rsa = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/asymmetric/ds rsa"));
398     auto button_cryptography_asymmetric_ds_rsa = Wt::cpp14::make_unique<↩
            Wt::WPushButton>("RSA Digital Signatures");
399     button_cryptography_asymmetric_ds_rsa->setStyleClass("↩
            navigation_grid_item");
400     this->asymmetric_ds_rsa->addWidget(std::move(↩
            button_cryptography_asymmetric_ds_rsa));
401     this->addWidget(std::move(asymmetric_ds_rsa),30,1, Wt::AlignmentFlag↩
            ::Center);
402
403     this->asymmetric_ds_elgamal = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/asymmetric/ds elgamal"));
404     auto button_cryptography_asymmetric_ds_elgamal = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Elgamal Digital Signatures");
405     button_cryptography_asymmetric_ds_elgamal->setStyleClass("↩
            navigation_grid_item");
406     this->asymmetric_ds_elgamal->addWidget(std::move(↩
            button_cryptography_asymmetric_ds_elgamal));
407     this->addWidget(std::move(asymmetric_ds_elgamal),30,2, Wt::↩
            AlignmentFlag::Center);
408
409     this->asymmetric_ds_dsa = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/asymmetric/ds dsa"));
410     auto button_cryptography_asymmetric_ds_dsa = Wt::cpp14::make_unique<↩
            Wt::WPushButton>("Digital Signature Algorithm");
411     button_cryptography_asymmetric_ds_dsa->setStyleClass("↩
            navigation_grid_item");
412     this->asymmetric_ds_dsa->addWidget(std::move(↩
            button_cryptography_asymmetric_ds_dsa));
413     this->addWidget(std::move(asymmetric_ds_dsa),31,0, Wt::AlignmentFlag↩
            ::Center);
414
415     this->asymmetric_ds_ecdsa = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/asymmetric/ec security"));
416     auto button_cryptography_asymmetric_ds_ecdsa = Wt::cpp14::make_unique↩
            <Wt::WPushButton>("Elliptic Curve Digital Signatures");
417     button_cryptography_asymmetric_ds_ecdsa->setStyleClass("↩
            navigation_grid_item");
418     this->asymmetric_ds_ecdsa->addWidget(std::move(↩
            button_cryptography_asymmetric_ds_ecdsa));
419     this->addWidget(std::move(asymmetric_ds_ecdsa),31,2, Wt::↩
            AlignmentFlag::Center);
420
421 }
422
423 void crypto_online_navigation_grid::setup_applications_contents() {
424     this->title_applications = Wt::cpp14::make_unique<Wt::WText>("↩
            Protocols");
425     this->title_applications->setStyleClass("navigation_grid_title");
```

```
426     this->addWidget(std::move(this->title_applications),32,1);
427 
428     this->subtitle_hash_functions = Wt::cpp14::make_unique<Wt::WText>("↩
            Hash Functions");
429     this->subtitle_hash_functions->setStyleClass("↩
            navigation_grid_subtitle");
430     this->addWidget(std::move(this->subtitle_hash_functions),33,1);
431 
432     this->protocols_hf_motivation = Wt::cpp14::make_unique<Wt::WAnchor>(↩
            Wt::WLink(Wt::LinkType::InternalPath, "/protocols/hash function ↩
            motivation"));
433     auto button_cryptography_protocol_hf_motivation = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Why we need Hash Functions");
434     button_cryptography_protocol_hf_motivation->setStyleClass("↩
            navigation_grid_item");
435     this->protocols_hf_motivation->addWidget(std::move(↩
            button_cryptography_protocol_hf_motivation));
436     this->addWidget(std::move(this->protocols_hf_motivation),34,0, Wt::↩
            AlignmentFlag::Center);
437 
438     this->protocols_hf_security_requirements = Wt::cpp14::make_unique<Wt↩
            ::WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/protocols/hash ↩
            function security requirements"));
439     auto button_cryptography_protocol_hf_security_requirements = Wt::↩
            cpp14::make_unique<Wt::WPushButton>("Security Requirements");
440     button_cryptography_protocol_hf_security_requirements->setStyleClass(↩
            "navigation_grid_item");
441     this->protocols_hf_security_requirements->addWidget(std::move(↩
            button_cryptography_protocol_hf_security_requirements));
442     this->addWidget(std::move(this->protocols_hf_security_requirements)↩
            ,34,1, Wt::AlignmentFlag::Center);
443 
444     this->protocols_hf_overview = Wt::cpp14::make_unique<Wt::WAnchor>(Wt↩
            ::WLink(Wt::LinkType::InternalPath, "/protocols/hash function ↩
            overview"));
445     auto button_cryptography_protocol_hf_overview = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Hash Function Overview");
446     button_cryptography_protocol_hf_overview->setStyleClass("↩
            navigation_grid_item");
447     this->protocols_hf_overview->addWidget(std::move(↩
            button_cryptography_protocol_hf_overview));
448     this->addWidget(std::move(this->protocols_hf_overview),34,2, Wt::↩
            AlignmentFlag::Center);
449 
450     this->protocols_hf_sha1 = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/protocols/hash function sha1")↩
            );
451     auto button_cryptography_protocol_hf_sha1 = Wt::cpp14::make_unique<Wt↩
            ::WPushButton>("SHA 1");
452     button_cryptography_protocol_hf_sha1->setStyleClass("↩
            navigation_grid_item");
453     this->protocols_hf_sha1->addWidget(std::move(↩
            button_cryptography_protocol_hf_sha1));
454     this->addWidget(std::move(this->protocols_hf_sha1),35,0, Wt::↩
            AlignmentFlag::Center);
```

```
455
456     this->protocols_hf_sha3 = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/protocols/hash function ↩
            overview"));
457     auto button_cryptography_protocol_hf_sha3=  Wt::cpp14::make_unique<Wt↩
            ::WPushButton>("SHA 3");
458     button_cryptography_protocol_hf_sha3->setStyleClass("↩
            navigation_grid_item");
459     this->protocols_hf_sha3->addWidget(std::move(↩
            button_cryptography_protocol_hf_sha3));
460     this->addWidget(std::move(this->protocols_hf_sha3),35,1, Wt::↩
            AlignmentFlag::Center);
461
462     this->protocols_hf_bcrypt = Wt::cpp14::make_unique<Wt::WAnchor>(Wt::↩
            WLink(Wt::LinkType::InternalPath, "/protocols/hash function ↩
            overview"));
463     auto button_cryptography_protocol_hf_bcrypt = Wt::cpp14::make_unique<↩
            Wt::WPushButton>("bcrypt");
464     button_cryptography_protocol_hf_bcrypt->setStyleClass("↩
            navigation_grid_item");
465     this->protocols_hf_bcrypt->addWidget(std::move(↩
            button_cryptography_protocol_hf_bcrypt));
466     this->addWidget(std::move(this->protocols_hf_bcrypt),35,2, Wt::↩
            AlignmentFlag::Center);
467
468     this->subtitle_macs = Wt::cpp14::make_unique<Wt::WText>("Message ↩
            Authentication Codes");
469     this->subtitle_macs->setStyleClass("navigation_grid_subtitle");
470     this->addWidget(std::move(this->subtitle_macs),36,1);
471
472     this->protocols_macs_principles = Wt::cpp14::make_unique<Wt::WAnchor↩
            >(Wt::WLink(Wt::LinkType::InternalPath, "/protocols/mac principles↩
            "));
473     auto button_cryptography_protocol_mac_principles = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Message Authentication Code ↩
            Principles");
474     button_cryptography_protocol_mac_principles->setStyleClass("↩
            navigation_grid_item");
475     this->protocols_macs_principles->addWidget(std::move(↩
            button_cryptography_protocol_mac_principles));
476     this->addWidget(std::move(this->protocols_macs_principles),37,0, Wt::↩
            AlignmentFlag::Center);
477
478     this->protocols_macs_hash_functions= Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/protocols/mac ↩
            hmac"));
479     auto button_cryptography_protocol_macs_hash_functions = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Message Authentication Codes From ↩
            Hash Functions");
480     button_cryptography_protocol_macs_hash_functions->setStyleClass("↩
            navigation_grid_item");
481     this->protocols_macs_hash_functions->addWidget(std::move(↩
            button_cryptography_protocol_macs_hash_functions));
482     this->addWidget(std::move(this->protocols_macs_hash_functions),37,1, ↩
            Wt::AlignmentFlag::Center);
```

```
483
484     this->protocols_macs_block_cipher = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/protocols/mac ↩
            cbcmac"));
485     auto button_cryptography_protocol_mac_block_cipher = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Message Authentication Codes From ↩
            Block Ciphers");
486     button_cryptography_protocol_mac_block_cipher->setStyleClass("↩
            navigation_grid_item");
487     this->protocols_macs_block_cipher->addWidget(std::move(↩
            button_cryptography_protocol_mac_block_cipher));
488     this->addWidget(std::move(this->protocols_macs_block_cipher),37,2, Wt↩
            ::AlignmentFlag::Center);
489
490     this->subtitle_key_establishment = Wt::cpp14::make_unique<Wt::WText>(↩
            "Key Establishment");
491     this->subtitle_key_establishment->setStyleClass("↩
            navigation_grid_subtitle");
492     this->addWidget(std::move(this->subtitle_key_establishment),38,1);
493
494     this->protocols_key_est_introduction = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/protocols/key est↩
             introduction"));
495     auto button_cryptography_protocol_key_est_introduction = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Introduction to Key Establishment");
496     button_cryptography_protocol_key_est_introduction->setStyleClass("↩
            navigation_grid_item");
497     this->protocols_key_est_introduction->addWidget(std::move(↩
            button_cryptography_protocol_key_est_introduction));
498     this->addWidget(std::move(this->protocols_key_est_introduction),39,0,↩
             Wt::AlignmentFlag::Center);
499
500     this->protocols_key_est_symmetric = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/protocols/key est↩
             symmetric"));
501     auto button_cryptography_protocol_key_est_symmetric = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Key Establishment with Symmetric ↩
            Techniques");
502     button_cryptography_protocol_key_est_symmetric->setStyleClass("↩
            navigation_grid_item");
503     this->protocols_key_est_symmetric->addWidget(std::move(↩
            button_cryptography_protocol_key_est_symmetric));
504     this->addWidget(std::move(this->protocols_key_est_symmetric),39,1, Wt↩
            ::AlignmentFlag::Center);
505
506     this->protocols_key_est_asymmetric = Wt::cpp14::make_unique<Wt::↩
            WAnchor>(Wt::WLink(Wt::LinkType::InternalPath, "/protocols/key est↩
             asymmetric"));
507     auto button_cryptography_protocol_key_est_asymmetric = Wt::cpp14::↩
            make_unique<Wt::WPushButton>("Key Establishment with Asymmetric ↩
            Techniques");
508     button_cryptography_protocol_key_est_asymmetric->setStyleClass("↩
            navigation_grid_item");
509     this->protocols_key_est_asymmetric->addWidget(std::move(↩
            button_cryptography_protocol_key_est_asymmetric));
```

```
510     this->addWidget(std::move(this->protocols_key_est_asymmetric),39,2, ↩
            Wt::AlignmentFlag::Center);
511 
512 }
513 
514 void crypto_online_navigation_grid::clear_grid() {
515 
516 }
```

3.5.5 crypto_online_profile.h

```
/**
 * @file crypto_online_profile.h
 * @date 11/01/2017
 *
 * @brief This file contains the class and method definitions for the CryptoOnlineProfile class
 *
 * @author Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_PROFILE_H
#define CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_PROFILE_H

#include "../db/db_interface.h"
#include "../db/session.h"

#include <Wt/WTable.h>
#include <Wt/WText.h>

/**
 * @class CryptoOnlineProfile
 *
 * @brief This class handles the display and data flow for the profile page of the website.
 *
 */
class CryptoOnlineProfile : public Wt::WTable {
public:
    CryptoOnlineProfile() = delete;

    explicit CryptoOnlineProfile(Session& session);

private:
    /**
     * @brief This method loads the profile page for a user that is currently logged in
     */
    void load_profile_page_logged_in();

    /**
     * @brief This method show an empty page as there is currently no user logged in
     */
    void load_profile_page_logged_out();

    Wt::WText* _username_label; /**< A WText instance to load the username tag into, not the actual username. */
    Wt::WText* _username; /**< A WText instance that holds the username that is loaded into from the database */

    Session& _current_session;

```

```
47      db_interface database_interface; /**< A database_interface instance ↩
            that allows data to be read from the database*/
48
49  };
50
51
52  #endif //CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_PROFILE_H
```

3.5.6 crypto_online_profile.cc

```
/**
 * @file crypto_online_profile.cc
 * @date 11/01/2017
 *
 * @brief This handles the definitions defined in the related header file.
 * It handles the loading of the user profiles and all related user profile data that needs to be shown to the user.
 *
 *
 * @author Jacob Powell
 */

#include "crypto_online_profile.h"
#include "crypto_online_home.h"

/**
 * @brief The Constructor for the CryptoOnlineProfile class
 * The constructor handles the initial connection to the database. Once connected it figures out what user it is that
 * has logged on and collects the user information on that profile loading it into a member of the class. It then
 * loads that information and displays it on the screen.
 *
 * It also checks to see if there is actually a user logged in to the system. If there isn't then it displays to the
 * viewer that there is no user currently logged in.
 */
CryptoOnlineProfile::CryptoOnlineProfile(Session& session) : _current_session(session), database_interface(session)
{
    if(this->_current_session.login().loggedIn())
        load_profile_page_logged_in();
    else{
        load_profile_page_logged_out();
    }
}

void CryptoOnlineProfile::load_profile_page_logged_in() {
    const Wt::Auth::User& user = _current_session.login().user();

    this->_username_label = this->elementAt(0,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>("Username:"));
    this->_username_label->setStyleClass("profile_username");

    this->_username = this->elementAt(0,1)->addWidget(Wt::cpp14::make_unique<Wt::WText>(" " + user.identity(Wt::Auth::Identity::LoginName)));
    this->_username->setStyleClass("profile_username");

    auto questions_answered_label = this->elementAt(1,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>("Questions Answered: "));
```

```
43      auto questions_answered = this->elementAt(1,1)->addWidget(Wt::cpp14::↩
            make_unique<Wt::WText>());
44      auto questions_answered_correct_label = this->elementAt(2,0)->↩
            addWidget(Wt::cpp14::make_unique<Wt::WText>("Questions Correct: ")↩
            );
45      auto questions_answered_correct = this->elementAt(2,1)->addWidget(Wt↩
            ::cpp14::make_unique<Wt::WText>());
46
47      questions_answered_label->setStyleClass("profile_username");
48      questions_answered->setStyleClass("profile_username");
49      questions_answered_correct_label->setStyleClass("profile_username");
50      questions_answered_correct->setStyleClass("profile_username");
51
52      int total_questions = this->database_interface.↩
            get_total_questions_answered();
53      questions_answered->setText(std::to_string(total_questions));
54      int total_questions_correct = this->database_interface.↩
            get_total_correct_question_answered();
55      questions_answered_correct->setText(std::to_string(↩
            total_questions_correct));
56  }
57
58  void CryptoOnlineProfile::load_profile_page_logged_out() {
59      this->_username_label = this->elementAt(0, 0)->addWidget(Wt::cpp14::↩
            make_unique<Wt::WText>("No User Logged In"));
60      this->_username_label->setStyleClass("profile_username");
61  }
```

3.5.7 crypto_online_footer.h

```
/*
 * File: crypto_online_home.cc
 * Created: 20/12/2017 20:16
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#include "crypto_online_footer.h"

#include <Wt/WText.h>

crypto_online_footer::crypto_online_footer() {
    auto footer = Wt::cpp14::make_unique<Wt::WText>("FOOTER LOCATION");
    this->addWidget(std::move(footer),0,0,Wt::AlignmentFlag::Center);
}
```

3.5.8 crypto_online_footer.cc

```
/*
 * File: crypto_online_home.cc
 * Created: 20/12/2017 20:16
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#include "crypto_online_footer.h"

#include <Wt/WText.h>

crypto_online_footer::crypto_online_footer() {
    auto footer = Wt::cpp14::make_unique<Wt::WText>("FOOTER LOCATION");
    this->addWidget(std::move(footer),0,0,Wt::AlignmentFlag::Center);
}
```

3.5.9 crypto_online_aes_example.h

```
/**
 * @file crypto_online_aes_example.h
 * @date 15/04/2018
 *
 * @brief This file contains the class that loads the AES Example page
 *
 * @author Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_AES_EXAMPLE_H
#define CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_AES_EXAMPLE_H

#include "../crypto/aes_implementation.h"

#include <Wt/WTable.h>
#include <Wt/WLineEdit.h>
#include <Wt/WText.h>
#include <Wt/WTextArea.h>
#include <Wt/WPushButton.h>

class CryptoOnlineAESExample : public Wt::WTable {
public:
    CryptoOnlineAESExample();

private:

    void load_page_content();
    void process();
    void hexstring_to_array(std::string hexstring, byte out[], int length);
    uint8_t hex_char_to_string(char c);

    Wt::WLineEdit* plaintext_entry;
    Wt::WLineEdit* key_entry;

    Wt::WText* plaintext_error;
    Wt::WText* key_error;

    Wt::WTextArea* ciphertext_area;

    AESImplementation aes_implementation;
};


#endif //CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_AES_EXAMPLE_H
```

3.5.10 crypto_online_aes_example.cc

```
/**
 * @file crypto_online_aes_example.cc
 * @date 15/04/2018
 *
 * @author Jacob Powell
 */

#include "crypto_online_aes_example.h"

CryptoOnlineAESExample::CryptoOnlineAESExample() {
    load_page_content();
}

void CryptoOnlineAESExample::load_page_content() {
    this->elementAt(0,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>("Plaintext: "));
    this->elementAt(0,0)->setStyleClass("aes_example_text");
    this->elementAt(0,0)->setContentAlignment(Wt::AlignmentFlag::Right);
    this->plaintext_entry = this->elementAt(0,1)->addWidget(Wt::cpp14::make_unique<Wt::WLineEdit>());
    this->elementAt(0,1)->setStyleClass("aes_example_entry");
    this->elementAt(0,1)->setContentAlignment(Wt::AlignmentFlag::Left);
    this->elementAt(0,1)->setPadding(500, Wt::Side::Right);
    this->plaintext_error = this->elementAt(0,2)->addWidget(Wt::cpp14::make_unique<Wt::WText>());
    this->elementAt(0,2)->setContentAlignment(Wt::AlignmentFlag::Left);


    this->elementAt(1,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>("Key: "));
    this->elementAt(1,0)->setStyleClass("aes_example_text");
    this->elementAt(1,0)->setContentAlignment(Wt::AlignmentFlag::Right);
    this->key_entry = this->elementAt(1,1)->addWidget(Wt::cpp14::make_unique<Wt::WLineEdit>());
    this->elementAt(1,1)->setStyleClass("aes_example_entry");
    this->elementAt(1,1)->setContentAlignment(Wt::AlignmentFlag::Left);
    this->elementAt(1,1)->setPadding(500, Wt::Side::Right);
    this->key_error = this->elementAt(0,2)->addWidget(Wt::cpp14::make_unique<Wt::WText>());
    this->elementAt(0,2)->setContentAlignment(Wt::AlignmentFlag::Left);

    auto start_encrypt_process = this->elementAt(2,0)->addWidget(Wt::cpp14::make_unique<Wt::WPushButton>("Encrypt"));
    this->elementAt(2,0)->setContentAlignment(Wt::AlignmentFlag::Right);
    start_encrypt_process->setStyleClass("aes_example_text");
    start_encrypt_process->clicked().connect([=]{
        this->process();
    });

    this->elementAt(3,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>("Ciphertext: "));
    this->elementAt(3,0)->setStyleClass("aes_example_text");
```

```
45      this->elementAt(3,0)->setContentAlignment(Wt::AlignmentFlag::Right);
46      this->ciphertext_area = this->elementAt(3,1)->addWidget(Wt::cpp14::↩
            make_unique<Wt::WTextArea>());
47      this->elementAt(3,1)->setStyleClass("aes_example_entry");
48      this->elementAt(3,1)->setContentAlignment(Wt::AlignmentFlag::Left);
49      this->elementAt(3,1)->setPadding(500, Wt::Side::Right);
50  }
51
52  void CryptoOnlineAESExample::process() {
53
54      std::string plaintext = this->plaintext_entry->text().toUTF8();
55      std::string key = this->key_entry->text().toUTF8();
56      std::string ciphertext = "";
57
58      if(plaintext.length() != 32) {
59          this->plaintext_error->setText("Plaintext needs to be 32 Hex ↩
                Characters");
60          this->plaintext_error->setStyleClass("error_message");
61      }else if(key.length() != 16 && key.length() != 24 && key.length() != ↩
            32){
62          this->key_error->setText("Key needs to be 32, 48 or 64 Hex ↩
                Characters Long");
63          this->key_error->setStyleClass("error_message");
64      }
65      else{
66          byte output[16];
67          aes_implementation.encrypt_block((const byte*)plaintext.data(), ↩
                output, (const byte*)key.data());
68          for(int i = 0; i < 16; i++) {
69              std::cout << unsigned(output[i]);
70              ciphertext += std::to_string(unsigned(output[i]));
71          }
72          std::cout << std::endl;
73          this->ciphertext_area->setText(ciphertext);
74      }
75
76  }
```

3.6 src/layout/information_content/

3.6.1 learning_content_template.h

```
/*
 * File: learning_content_template.h
 * Created: 15/01/2018 14:26
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_LEARNING_CONTENT_TEMPLATE_H
#define CRYPTO_ONLINE_PROJECT_LEARNING_CONTENT_TEMPLATE_H


#include <Wt/WTable.h>
#include <Wt/WLineEdit.h>

#include <string>
#include <Wt/WTemplate.h>
#include "../../db/db_interface.h"

class learning_content_template : public Wt::WTable {
public:
    learning_content_template(Session& session, std::string title, std::string contents_link);
    learning_content_template(Session& session, std::string title, std::string contents_link, std::string questions_link);

    virtual void load_contents() = 0;
    virtual void load_questions() = 0;

    std::unique_ptr<Wt::WTemplate> load_question_section(const std::string& template_name, const int question_id[]);

    Wt::WText* page_title;
    Wt::WText* page_content;

    std::string content_title_link;
    std::string content_resource_link;
    std::string content_question_link;
private:
    db_interface database_api;

};


#endif //CRYPTO_ONLINE_PROJECT_LEARNING_CONTENT_TEMPLATE_H
```

3.6.2 learning_content_template.cc

```
/*
 * File: learning_content_template.cc
 * Created: 15/01/2018 14:26
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#include "learning_content_template.h"

#include <Wt/WPushButton.h>
#include <Wt/WLineEdit.h>

learning_content_template::learning_content_template(Session& session, std::string title_link, std::string contents_link, std::string question_link) :
    database_api(session)
{
    this->content_title_link = std::move(title_link);
    this->content_resource_link = std::move(contents_link);
    this->content_question_link = std::move(question_link);
}

learning_content_template::learning_content_template(Session& session, std::string title, std::string contents_link) : database_api(session) {
    this->content_title_link = std::move(title);
    this->content_resource_link = std::move(contents_link);
}

/**
 * @brief This method loads the functionality of the question section for the given content page
 *
 * @param template_name The template to look up in the xml files
 * @param question_id An array containing the question ids for the pages
 * @return A Wt::WTemplate object containing the question section functionality
 */
std::unique_ptr<Wt::WTemplate>
learning_content_template::load_question_section(const std::string &template_name, const int question_id[]) {
    std::cout << question_id[0] << std::endl;
    auto result = Wt::cpp14::make_unique<Wt::WTemplate>(Wt::WString::tr(template_name));
    const int question_id_temp[] = {question_id[0], question_id[1]};

    auto* answer_entry_1 = new Wt::WLineEdit();
    auto* answer_entry_2 = new Wt::WLineEdit();
    auto* answer_status_1 = new Wt::WText("Answer Status: ");
```

```
45      auto* answer_status_2 = new Wt::WText("Answer Status: ");
46
47      auto process_answer_button_1 = Wt::cpp14::make_unique<Wt::WPushButton↩
            >("Check Answer");
48      auto process_answer_button_2 = Wt::cpp14::make_unique<Wt::WPushButton↩
            >("Check Answer");
49
50      process_answer_button_1->clicked().connect([=]{
51          const std::string &answer_1 = answer_entry_1->text().toUTF8();
52          this->database_api.add_answer_to_user(answer_1, question_id_temp↩
                [0]);
53          if(this->database_api.check_answer(answer_1, question_id_temp[0])↩
                ){
54              std::cout << "WE DEFO GOT HERE HAHAH" << std::endl;
55              answer_status_1->setText("Correct Answer");
56              answer_status_1->setStyleClass("↩
                    learning_content_question_right");
57              this->database_api.set_answer_check_flag(question_id_temp[0],↩
                    true);
58          }else{
59              answer_status_1->setText("Wrong Answer");
60              answer_status_1->setStyleClass("↩
                    learning_content_question_wrong");
61              this->database_api.set_answer_check_flag(question_id_temp[0],↩
                    false);
62          }
63      });
64
65      process_answer_button_2->clicked().connect([=]{
66          const std::string &answer_2 = answer_entry_2->text().toUTF8();
67          this->database_api.add_answer_to_user(answer_2, question_id_temp↩
                [1]);
68          if(this->database_api.check_answer(answer_2, question_id_temp[1])↩
                ){
69              answer_status_2->setText("Correct Answer");
70              this->database_api.set_answer_check_flag(question_id_temp[1],↩
                    true);
71          }else{
72              answer_status_2->setText("Wrong Answer");
73              this->database_api.set_answer_check_flag(question_id_temp[1],↩
                    false);
74          }
75      });
76
77      result->bindWidget(std::to_string(question_id[0]), std::unique_ptr<Wt↩
            ::WLineEdit>(answer_entry_1));
78      result->bindWidget(std::to_string(question_id[1]), std::unique_ptr<Wt↩
            ::WLineEdit>(answer_entry_2));
79      result->bindWidget("result_text_1", std::unique_ptr<Wt::WText>(↩
            answer_status_1));
80      result->bindWidget("result_text_2", std::unique_ptr<Wt::WText>(↩
            answer_status_2));
81      result->bindWidget("submit_button_1", std::move(↩
            process_answer_button_1));
```

```
82      result->bindWidget("submit_button_2", std::move(↩
            process_answer_button_2));
83
84      return result;
85  }
```

3.6.3 intro_to_cryptography.h

```
/*
 * File: intro_to_cryptography.h
 * Created: 24/01/2018 9:00
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_INTRO_TO_CRYPTOGRAPHY_H
#define CRYPTO_ONLINE_PROJECT_INTRO_TO_CRYPTOGRAPHY_H


#include "learning_content_template.h"

class intro_to_cryptography : public learning_content_template {
public:
    intro_to_cryptography(Session& session, std::string title, std::string contents_link);
    void load_contents() override;
    void load_questions() override;

private:

};


#endif //CRYPTO_ONLINE_PROJECT_INTRO_TO_CRYPTOGRAPHY_H
```

3.6.4 intro_to_cryptography.cc

```
/*
 * File: intro_to_cryptography.cc
 * Created: 24/01/2018 9:00
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#include "intro_to_cryptography.h"

#include <Wt/WText.h>


intro_to_cryptography::intro_to_cryptography(Session& session, std::string title, std::string contents_link)
        : learning_content_template(session, std::move(title), std::move(contents_link)) {
    load_contents();

}

void intro_to_cryptography::load_contents() {
    this->page_title = this->elementAt(0,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>(Wt::WString::tr(this->content_title_link)));
    this->elementAt(0,0)->setStyleClass("learning_content_title");

    this->page_content = this->elementAt(1,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>(Wt::WString::tr(this->content_resource_link)));
    this->elementAt(1,0)->setStyleClass("learning_content_content");
    this->elementAt(1,0)->setContentAlignment(Wt::AlignmentFlag::Left);


}

void intro_to_cryptography::load_questions() {

}
```

3.6.5 modular_arithmetic.h

```
/*
 * File: modular_arithmetic.h
 * Created: 15/01/2018 14:35
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_MODULAR_ARITHMETIC_H
#define CRYPTO_ONLINE_PROJECT_MODULAR_ARITHMETIC_H


#include "learning_content_template.h"
#include "../../db/session.h"
#include "../../db/db_interface.h"

#include <Wt/WImage.h>

class modular_arithmetic : public learning_content_template {
public:
    modular_arithmetic(Session& session, std::string &title, std::string ↩
       &contents_link, std::string &question_link);

private:
    void load_contents() override;
    void load_questions() override;


};


#endif //CRYPTO_ONLINE_PROJECT_MODULAR_ARITHMETIC_H
```

3.6.6 modular_arithmetic.cc

```
/**
 * @file modular_arithmetic.cc
 * @date 15/01/2018
 *
 * @brief This file abstracts the Modular arithmetic page from the standard learning_content_template
 *
 * @author Jacob Powell
 */

#include "modular_arithmetic.h"

#include <Wt/WText.h>
#include <Wt/WPushButton.h>

/**
 * @brief This method loads the content to the page and if a user is logged in then it loads the questions
 */
modular_arithmetic::modular_arithmetic(Session& session, std::string &title, std::string &contents_link, std::string &question_link) :
        learning_content_template(session, title, contents_link, question_link){
    load_contents();
    if(session.login().loggedIn())
        load_questions();
}

/**
 * @brief Loading the title and content to the page
 */
void modular_arithmetic::load_contents() {
    this->page_title = this->elementAt(0,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>(Wt::WString::tr(this->content_title_link)));
    this->elementAt(0,0)->setStyleClass("learning_content_title");
    this->page_content = this->elementAt(1,0)->addWidget(Wt::cpp14::make_unique<Wt::WText>(Wt::WString::tr(this->content_resource_link)));
    this->elementAt(1,0)->setStyleClass("learning_content_content");
}

/**
 * @brief Loading the Questions section for the Page
 */
void modular_arithmetic::load_questions() {
    const int question_id[] = {21, 22};
    std::cout << question_id[0] << std::endl;
    auto result = learning_content_template::load_question_section(this->content_question_link, question_id);
    this->elementAt(2,0)->addWidget(std::move(result));
}
```

3.7 src/layout/header/

3.7.1 crypto_online_header.h

```
/**
 * @file crypto_online_home.cc
 * @date 19/12/2017
 *
 * This file contains the class definitions for the CryptoOnlineHeader Class
 *
 * @version 0.01
 * @author Jacob Powell
 */

#ifndef CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_HEADER_H
#define CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_HEADER_H


#include "../../db/db_interface.h"
#include "../../db/db_user.h"
#include "../../db/session.h"

#include <Wt/WHBoxLayout.h>
#include <Wt/WNavigationBar.h>
#include <Wt/WStackedWidget.h>
#include <Wt/WMenu.h>


/**
 * @class CryptoOnlineHeader
 *
 * @brief This Class handles what the appearance and functionality of the website header
 */
class CryptoOnlineHeader : public Wt::WHBoxLayout{
public:
    explicit CryptoOnlineHeader(Session& session); /**< Constructor for the CryptoOnlineHeader Class */

private:
    void create_navigation_bar(); /**< Creates the Inital Navigation Bar that appears in the header */
    void create_user_navigation_bar();

    Session& _current_session; /**< Holds the current session */
    db_interface database_interface; /**< Interface to interact with the database from */

    std::unique_ptr<Wt::WNavigationBar> navigation_bar; /**< Holds the current state of the navigation bar */
    std::unique_ptr<Wt::WMenu> left_menu; /**< Holds the state of the left menu in the navigation bar */
```

```
43      std::unique_ptr<Wt::WMenu> right_menu; /**< Holds the state of the ↩
            Right menu in the navigation bar */
44
45      Wt::WStackedWidget *navigation_bar_contents_stack; /**< Holds the ↩
            contents of the bar in the form a stack */
46
47  };
48
49  #endif //CRYPTO_ONLINE_PROJECT_CRYPTO_ONLINE_HEADER_H
```

3.7.2 crypto_online_header.cc

```
/*
 * File: crypto_online_home.cc
 * Created: 19/12/2017 13:25
 * Finished:
 *
 * Description:
 *
 * Author: Jacob Powell
 */

#include "crypto_online_header.h"
#include "../crypto_online_home.h"


CryptoOnlineHeader::CryptoOnlineHeader(Session& session) : _current_session(session), database_interface(session) {
    if(session.login().loggedIn()){
        create_user_navigation_bar();
    }else{
        create_navigation_bar();
    }
}

/**
 * @brief This method creates the header for the website when no user is logged in
 */
void CryptoOnlineHeader::create_navigation_bar() {
    this->navigation_bar = Wt::cpp14::make_unique<Wt::WNavigationBar>();
    this->navigation_bar->setTitle("Crypto Online", Wt::WLink(Wt::LinkType::InternalPath, "/home"));
    this->navigation_bar->setResponsive(true);
    this->navigation_bar->setStyleClass("navigation_bar_things");
    this->navigation_bar->setMaximumSize(Wt::WLength::Auto, 50);

    this->navigation_bar_contents_stack = this->addWidget(Wt::cpp14::make_unique<Wt::WStackedWidget>());
    navigation_bar_contents_stack->addStyleClass("contents");

    this->left_menu = Wt::cpp14::make_unique<Wt::WMenu>(this->navigation_bar_contents_stack);
    auto left_menu_ = this->navigation_bar->addMenu(std::move(this->left_menu));
    left_menu_->addItem("Home")->setLink(Wt::WLink(Wt::LinkType::InternalPath, "/home"));
    left_menu_->addItem("AES Encryption Form")->setLink(Wt::WLink(Wt::LinkType::InternalPath, "/aes_encryption"));

    this->right_menu = Wt::cpp14::make_unique<Wt::WMenu>();
    auto right_menu_ = this->navigation_bar->addMenu(std::move(this->right_menu), Wt::AlignmentFlag::Right);
```

```
43      right_menu_->addItem("Login / Register")->setLink(Wt::WLink(Wt::↩
            LinkType::InternalPath, "/login"));
44
45      this->addWidget(std::move(this->navigation_bar),1);
46  }
47
48  /**
49   * @brief This method creates a header for when a user is logged in.
50   *        It also handles whether or not the user is an admin and if that↩
         user is then it gives them the
51   *        option to go to admin settings.
52   */
53  void CryptoOnlineHeader::create_user_navigation_bar() {
54      this->navigation_bar = Wt::cpp14::make_unique<Wt::WNavigationBar>();
55      this->navigation_bar->setTitle("Crypto Online", Wt::WLink(Wt::↩
            LinkType::InternalPath, "/home"));
56      this->navigation_bar->setResponsive(true);
57      this->navigation_bar->setStyleClass("navigation_bar_things");
58      this->navigation_bar->setMaximumSize(Wt::WLength::Auto, 50);
59
60      this->navigation_bar_contents_stack = this->addWidget(Wt::cpp14::↩
            make_unique<Wt::WStackedWidget>());
61      navigation_bar_contents_stack->addStyleClass("contents");
62
63      this->left_menu = Wt::cpp14::make_unique<Wt::WMenu>(this->↩
            navigation_bar_contents_stack);
64      auto left_menu_ = this->navigation_bar->addMenu(std::move(this->↩
            left_menu));
65      left_menu_->addItem("Home")->setLink(Wt::WLink(Wt::LinkType::↩
            InternalPath, "/home"));
66
67      // Getting information on the current logged in user
68      const Wt::Auth::User& u = _current_session.login().user();
69      auto current_user = database_interface.get_user(u.id());
70
71      std::cout << "Got user information" << std::endl;
72      if(current_user->user_role == Role::Admin)
73          left_menu_->addItem("Admin Settings")->setLink(Wt::WLink(Wt::↩
              LinkType::InternalPath, "admin"));
74
75      std::cout << "Adding username to header" << std::endl;
76      auto username = left_menu_->addItem("User Logged In: " + u.identity(↩
            Wt::Auth::Identity::LoginName));
77      username->disable();
78
79      std::cout << "ENDED" << std::endl;
80
81      this->right_menu = Wt::cpp14::make_unique<Wt::WMenu>();
82      auto right_menu_ = this->navigation_bar->addMenu(std::move(this->↩
            right_menu), Wt::AlignmentFlag::Right);
83      right_menu_->addItem("Profile")->setLink(Wt::WLink(Wt::LinkType::↩
            InternalPath, "/profile"));
84      right_menu_->addItem("Sign Out")->setLink(Wt::WLink(Wt::LinkType::↩
            InternalPath, "/signout"));
85
```

```
86      this->addWidget(std::move(this->navigation_bar),1);
87  }
```

3.8 src/resource/content_xmls/

3.8.1 intro_to_cryptography.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<messages>

    <message id="learning.intro to cryptography.title">
        <center>
            <h1 style="text-decoration: underline;">Introduction to Cryptography</h1>
        </center>
    </message>

    <message id="learning.intro to cryptography.content">
        <h2 style="text-decoration: underline;">
            Topics covered in this section
        </h2>

        <ul>
            <li>1. Classification</li>
            <li>2. Basics / Setup</li>
            <li>3. Substitution Cipher</li>
            <li>4. Attacks</li>
        </ul>


        <br/>
        <br/>
        <br/>

        <h3 style="text-decoration: underline;">1. Classification</h3>

        <p>
            Cryptography is really only 1 subsection of something called Cryptology. Cryptology is consisted of 2 sub sections, one being Cryptography and the other Cryptanalysis.
            <br/>
            The main goal of Cryptography is to design and develop algorithms / systems that allow a message being being sent by the sender (Alice) to the receiver (Bob) and not allowing and adversary (Oscar) the ability to intercept and read the message.
            <br/>
            The main goal of Cryptanalysis is to try and break the algorithms / systems developed using Cryptography. This is very important as without it we would not know whether or not a given algorithm / system is secure.
            <br/>
            A diagram is given below showing the different sections and subsections of Cryptology
        </p>

```

```
39          <center>
40              <img src="../resource/images/cryptology sections.png" alt="↩
                    Image could not be loaded"/>
41          </center>
42
43          <h3 style="text decoration: underline;">2. Basics / Setup</h3>
44
45          <p>
46              In this example we will be looking at the basic setup for a ↩
                    symmetric cryptosystem. The simple problem we are trying ↩
                    to solve is to provide secure communications over an ↩
                    insecure channel.
47          </p>
48
49          <center>
50              <img src="../resource/images/basic_setup_fv.png" alt="Image ↩
                    showing the basic setup of a symmetric cryptosystem"/>
51          </center>
52
53          <p>
54              Examples of channels that could be used are
55              <ul>
56                  <li>The Internet</li>
57                  <li>Airwaves, e.g. GSM, UMTS, LTE</li>
58                  <li>Insecure public networks</li>
59              </ul>
60
61              In practice you should <b>NEVER</b> use an untested crypto ↩
                    algorithm
62
63          </p>
64
65          <h3 style="text decoration: underline">2.1. Basic Notation</h3>
66
67          <center>
68              <img src="../resource/images/basic_notation_fv.png" alt="↩
                    Basic Notation Diagram for Cryptography"/>
69          </center>
70
71          <h3 style="text decoration: underline">2.2. Kerckhuffs Principle<↩
                /h3>
72
73          A cryptosystem should be secure even if the attacker (Oscar) ↩
                knows all the details about the system, with the exception of ↩
                the secret key.
74
75          <h3 style="text decoration: underline;">3. Substitution Cipher</↩
                h3>
76
77          <p>
78              The substitution cipher is a historical cipher that operates ↩
                    on the letters of the alphabet. <br/>
79              The main idea is that you substitute a plaintext letter with ↩
                    a ciphertext one.
80          </p>
```

```
81
82          <center>
83              <img src="../resource/images/substition_cipher_fv.png" alt="↩
                    Substitution Cipher Example" />
84          </center>
85
86          <br/>
87
88          <p>
89              Question? Is this Cipher Secure and if not how can we attack ↩
                    it?
90          </p>
91
92          <h3 style="text decoration: underline">3.1 Attacks against the ↩
                Substitution Cipher</h3>
93          <h3 style="text decoration: underline">3.1.1 Brute Force Attack ↩
                or  Exhaustive Key Search</h3>
94
95          <p>
96          This is the most boring but the most consistant form of attack on↩
                 any cipher. The only downside of this type of attack is that ↩
                it can take far to long to give back results. So, how does ↩
                this work against the subsitution cipher?
97          </p>
98
99          <center>
100             <img src="../resource/images/↩
                    brute_force_against_sub_cipher_fv.png" alt="Brute force ↩
                    against Sub Ciher"/>
101         </center>
102
103         <br/>
104
105         This shows us that the key space for the substitution cipher is ↩
                to large for us to brute force, its not that its impossible ↩
                but in order to go through all possible keys it would just ↩
                take far to long. So we will need to try another attack..
106
107         <h3 style="text decoration: underline">Letter Frequency Analysis<↩
                /h3>
108
109         By doing this we work out the frequency of the letters in the ↩
                ciphertext and then compare that to how frequent the letters ↩
                of the alphabet occur and then just compare. This allows us to↩
                 effectively work out what ciphertext letter maps to what ↩
                plaintext letter. This works because we have a 1 to 1 ↩
                relationship between the ciphertext letters and the plaintext ↩
                letters. Below is a diagram showing the frequency of the ↩
                letters of the alphabet.
110
111         <center>
112             <img src="../resource/images/letter_frequency_fv.png" alt="↩
                    Letter Frequency Diagram" />
113         </center>
114
```

```
115         <h3 style="text decoration: underline;">4. Classification of Attacks</h3>
116
117         <p>
118             There are often many many different ways to attack a system. Each of theses ways are known as 'Attack Vectors'
119         </p>
120
121         <center>
122             <img src="../resource/images/cryptanalysis_classification_fv.png" alt="The Different Types of Attack Vectors"/>
123         </center>
124
125     </message>
126 </messages>
```

3.8.2 modular_arithmetic.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<messages>

    <message id="learning.modular arithmetic.title">
        <center>
            Modular Arithmetic
        </center>

    </message>

    <message id="learning.modular arithmetic.content">

        <center>
            Modular Arithmetic is probably the most important concept to get your head around when learning about
            Cryptography. It is used in every aspect. The good thing is that it isnt that difficult a topic and
            can be understood after a few examples.

            The idea is that after a certain point you wrap back to zero. A good example of this is a clock. On most modern clock faces the numbers go from 1 12. When it gets to 12 it then goes to 1 again. This is know as Mod 12 arithmetic.

        </center>

        <h2 style="text-decoration: underline"> 1. Modular Arithmetic Basics</h2>

        <h3 style="text-decoration: underline;">1.1 Modular Arithmetic Definition</h3>

            Lets first look at the definition for modular arithmetic calculations:

            <center>
                <img src="../resource/images/mod_arith/def.png" alt="Image could not be loaded" width="50%" height="50%"/>
            </center>

        <h3 style="text-decoration: underline;">Example 1.1</h3>

            <center>
                <img src="../resource/images/mod_arith/example1_1.png" alt="Image could not be loaded" width="30%" height="30%"/>
            </center>

        <h3 style="text-decoration: underline;">1.2 Computation of the Remainder</h3>

```

```
41              <center>
42                  <img src="../resource/images/mod_arith/quot.png" alt="↩
                        Image could not be loaded" width="30%" height="30%"/>
43              </center>
44
45          <h3 style="text decoration: underline;">Example 1.2</h3>
46
47              <center>
48                  <img src="../resource/images/mod_arith/rem_not_uni.png" ↩
                        alt="Image could not be loaded" width="30%" height="↩
                        30%"/>
49              </center>
50
51          <h2 style="text decoration: underline;">2. Equivalence Classes</↩
                h2>
52
53          <h3 style="text decoration: underline"> 2.1 Equivalence classes ↩
                for Mod 5</h3>
54              <center>
55                  <img src="../resource/images/mod_arith/equiv_class.png" ↩
                        alt="Image could not be loaded" width="40%" height="↩
                        40%"/>
56              </center>
57
58          <h3 style="text decoration: underline"> 2.2 Important Application↩
                for Equivilance Classes</h3>
59              <center>
60                  <img src="../resource/images/mod_arith/important_appl.png↩
                        " alt="Image could not be loaded" width="40%" height="↩
                        40%"/>
61              </center>
62
63      </message>
64
65      <message id="learning.modular arithmetic.answer template">
66
67          <center>
68              <h2 style="text decoration: underline">
69                  Questions
70              </h2>
71          </center>
72
73
74          <div>
75
76              <center>
77                  <img src="../resource/images/mod_arith/21_question.png" ↩
                        alt="Image could not be loaded" width="10%" height="↩
                        10%" align="left"/>
78              </center>
79
80              <div style="align: middle; margin left: 40%; margin right: ↩
                    40%" >
81                  ${21}
82              </div>
```

```
83
84              <center>
85                  ${result_text_1}
86                  <br/>
87                  ${submit_button_1}
88              </center>
89
90              <br />
91
92              <center>
93                  <img src="../resource/images/mod_arith/22_question.png" ↩
                        alt="Image could not be loaded" width="10%" height="↩
                        10%" aligh="left"/>
94              </center>
95
96              <div style="align: middle; margin left: 40%; margin right: ↩
                    40%" >
97                  ${22}
98              </div>
99
100             <center>
101                 ${result_text_2}
102                 <br/>
103                 ${submit_button_2}
104             </center>
105
106         </div>
107     </message>
108 </messages>
```

Chapter 4

Testing

4.1 The Web Application

Since my Web Application has many moving parts I will be testing each of the components individually first. Then when I can be sure that the components work individually I will test all of them together.

4.1.1 Web Application Test Table

Test #	Test Description	Test Type	Expected Result	Pass /Fail	Fig No
1	Connecting to the Website	Typical	The user will be able to successfully load the Website	Pass	01
2	Loading the Basic Concepts Page	Typical	The user will be presented with the content for the basic concepts page	Pass	02
3	Loading the Modular Arithmetic Page	Typical	The user will be able to successfully load the content for the Modular Arithmetic Page	Pass	03
4	Loading the Login Page	Typical	The user will be able to successfully load the Login Page	Pass	04
5	Loading the Register Page	Typical	The user will be able to successfully load the Register Page	Pass	05

6	Loading the Profile Page	Typical	The user will be able to successfully load the P rofile Page	Pass	06
7	Check user has to e nter a username w hen registering	Erroneous	The user will not be abl e to register if they don 't enter a username	Pass	07
8	Check the user ha s to enter a passwo rd when registering	Erroneous	The user will not be abl e to register for an acco unt if the don't enter a password	Pass	08
9	Check that the us er has to enter the same password wh en confirming their password to registe r	Erroneous	The user will not be ab le to register if they do n't enter the same pass word	Pass	09
10	Check that the use r doesn't have to e nter a email if they want to register	Typical	The user will be able to register whether or not they enter an email ad dress	Pass	10
11	Check that if the u ser does enter an e mail a confirmation email is sent to the address supplied	Typical	The user will receive a n email providing them with a confirmation lin k to activate their acco unt	Pass	11
12	Check that the pas sword entered mee ts strength require ments	Erroneous	If a user registers with a weak password the u ser will not be able to r egister but if they sign up with a strong passw ord then they will be re gistered	Pass	12
13	Check that all user information is adde d to authentication tables	Typical	The authentication info rmation is added to the authentication tables	Pass	13
14	When a user regist ers a record is also created for them in the db_user tables	Typical	A record for the user i s created in the db_use r table	Pass	14

15	After the user has registered they are then sent to the homepage	Typical	After the user has registered the website will navigate them to the homepage	Pass	15
16	Check the custom header loads for a logged in user	Typical	The user header will be loaded rather than the normal site header navigation bar	Pass	16
17	The admin settings tab should not show for any non-admins	Typical	When a normal user logs in they should not see the admin settings tab in the header	Pass	17
18	Profile page should show when no user is logged in	Typical	The profile page displays that no user is logged in	Pass	19
19	Profile page should load when a user is logged in	Typical	The user will be presented with the profile page for their account	Pass	19
20	Profile page shows total number of questions answered	Typical	The user will be able to see the total number of questions they have answered	Pass	20
21	Profile page shows total number of questions answered correctly	Typical	The profile page will show the total number of questions that have been answered correctly.	Pass	21
22	When a user tries to log in with valid credentials they are logged in	Typical	The user will be logged in	Pass	22
23	When a user tries to log in with invalid credentials they are not logged in	Typical	The user will be logged in	Pass	23
24	After a few attempts to try to log in the user will have to wait to log in again	Typical	The user will be given a time out and will have to wait a fair interval until they can try to login in again	Pass	24

25	When a user answers a question it should store the question in the user_answered_questions table	Typical	The relevant question information is stored in the user_answered_questions table	Pass	25
26	When the user answers the question correctly then it displays that the user got the question correct	Typical	The user is shown that they got the question correct	Pass	26
27	When the user answers the question wrong it displays that they got the question wrong	Typical	The user is shown that they got the question wrong	Pass	27
28	When ever the user answers the question it updates the question in the user_answered_question table	Typical	The users question answer should be updated in the user_answered_question table	Pass	28
29	When a user answers a question correctly it should set the is˙correct field appropriately	Typical	The is˙correct field should be set to 1	pass	29
30	When a user answers a question incorrectly it should set the is˙correct field appropriately	Typical	The is˙correct field should be set to 0	pass	30

4.2 Test Evidence

4.2.1 Test Ref 01

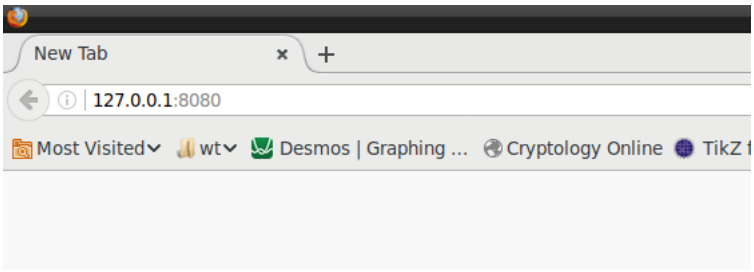


Figure 4.1: Entering the URL of the Web Application

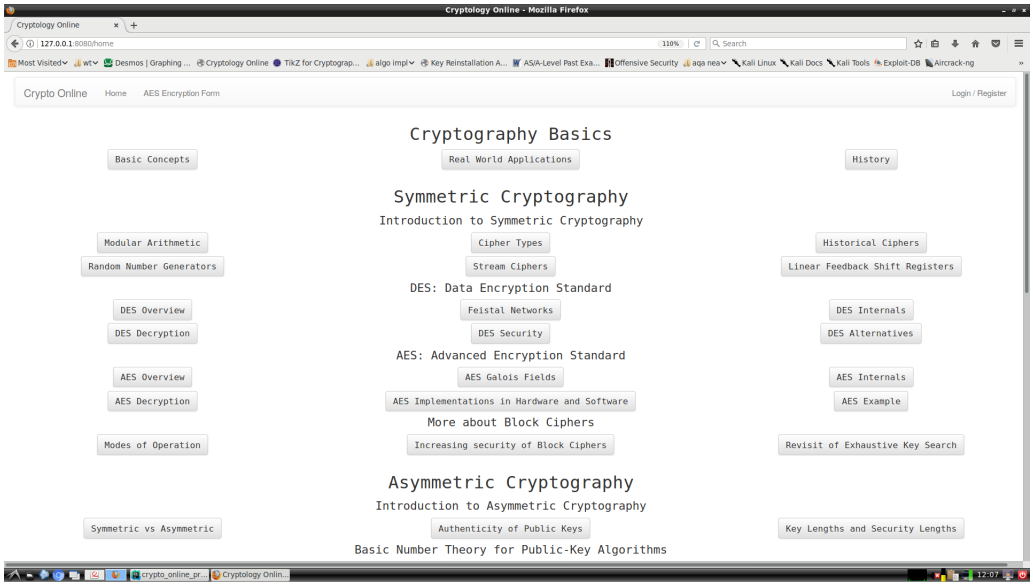


Figure 4.2: Pressing enter the Web Application Loads the Home Page

4.2.2 Test Ref 02

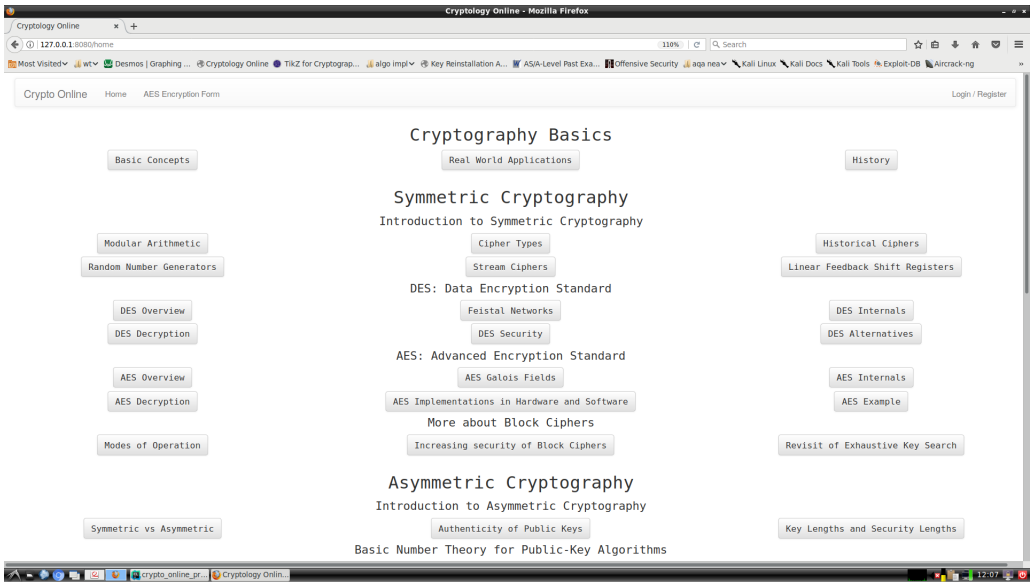


Figure 4.3: Starting at the Homepage and then clicking the Basic Concepts Button

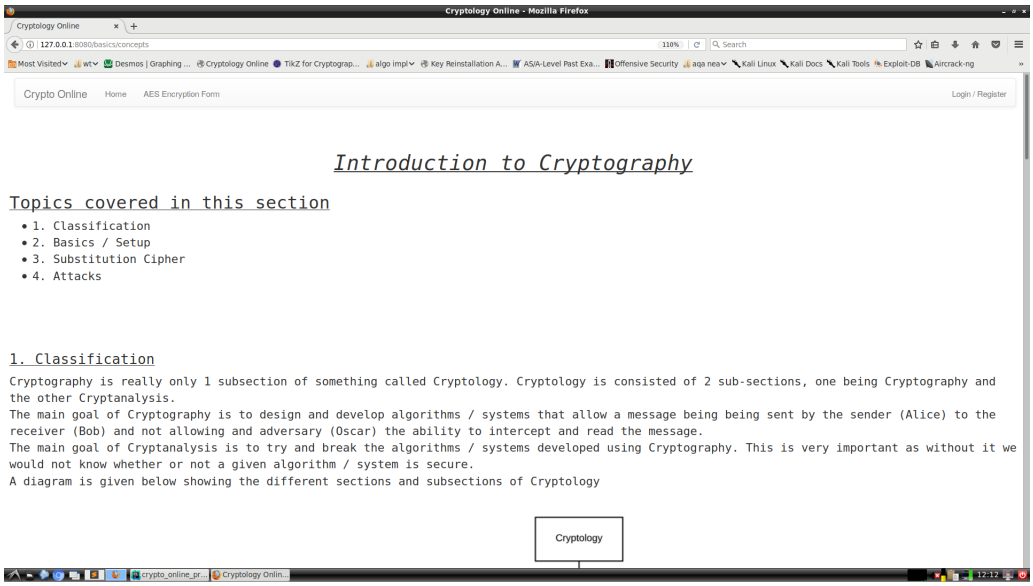


Figure 4.4: After clicking the button the Basic Concepts Page Loads

4.2.3 Test Ref 03

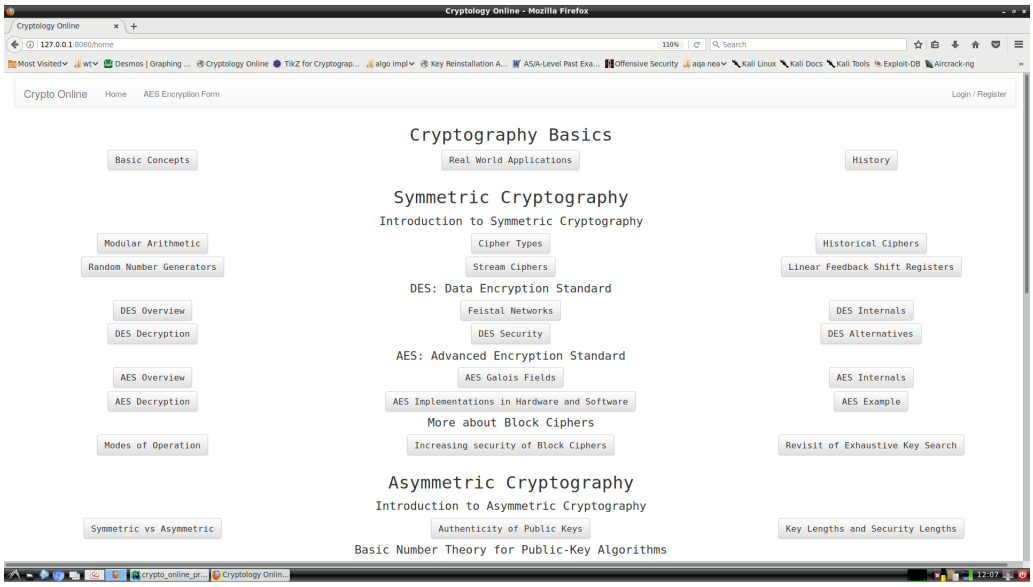


Figure 4.5: Starting at the Homepage and then clicking the Modular Arithmetic Button

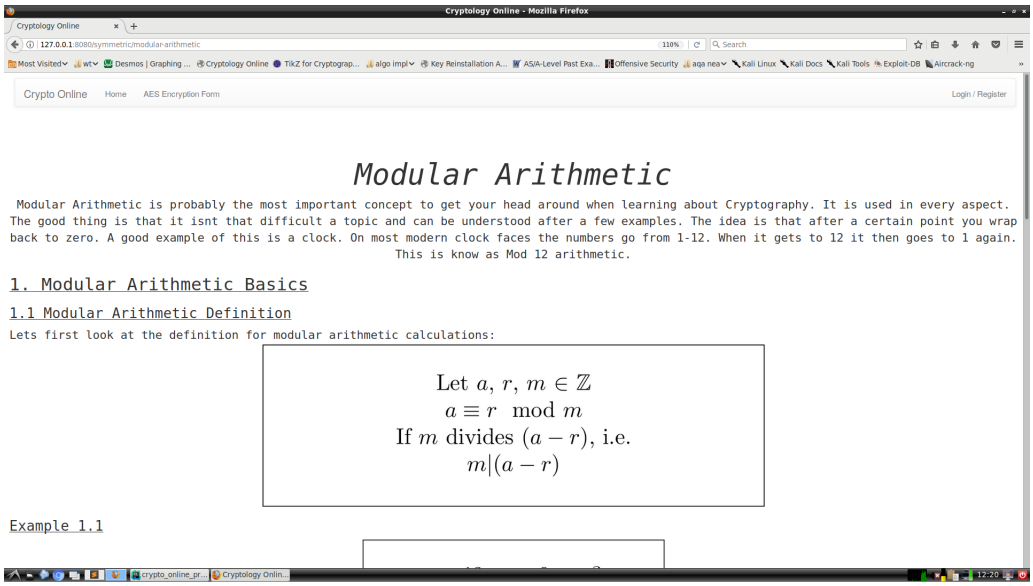


Figure 4.6: After clicking the button the Modular Arithmetic Page Loads

4.2.4 Test Ref 04

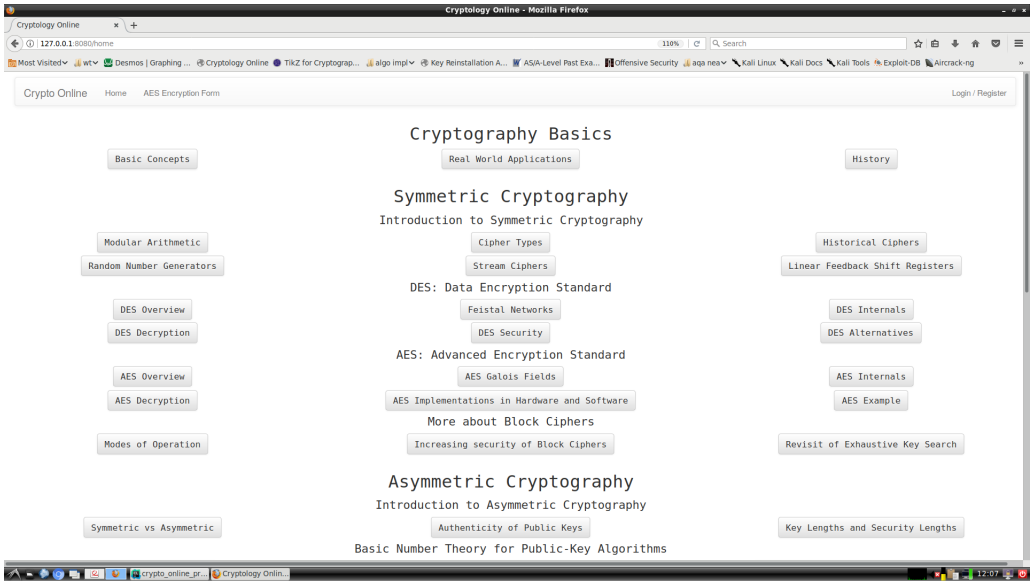


Figure 4.7: Starting at the Homepage and then clicking the Login/Register Button

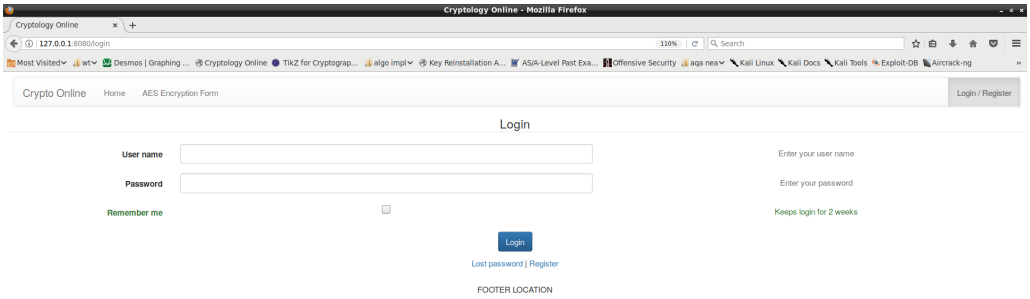


Figure 4.8: After clicking the button the Login/Register Page Loads

4.2.5 Test Ref 05

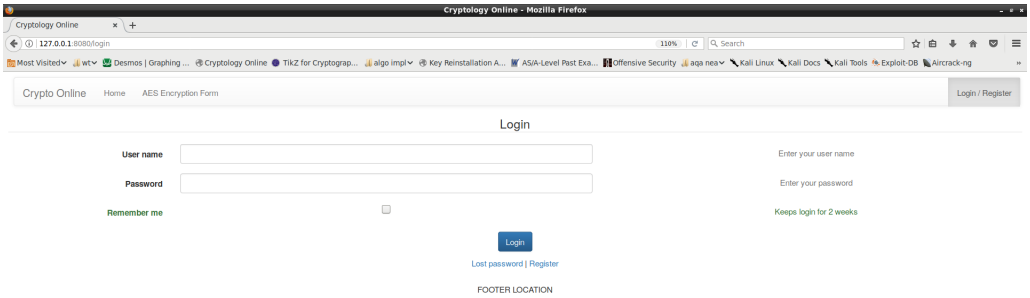


Figure 4.9: Starting at the Login/Register Page and then clicking the Register Button

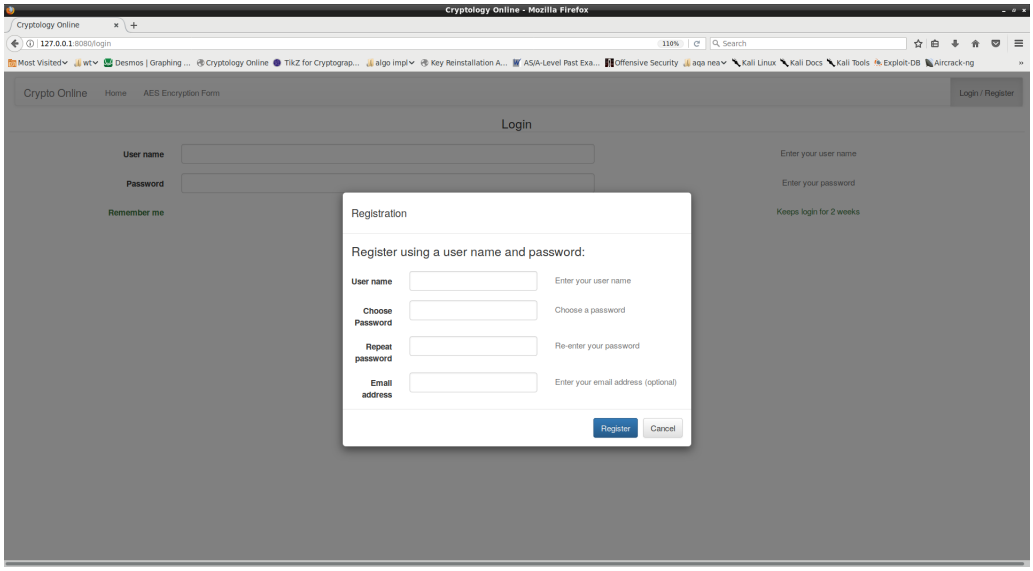


Figure 4.10: After clicking the button the Login/Register Page Loads

4.2.6 Test Ref 06

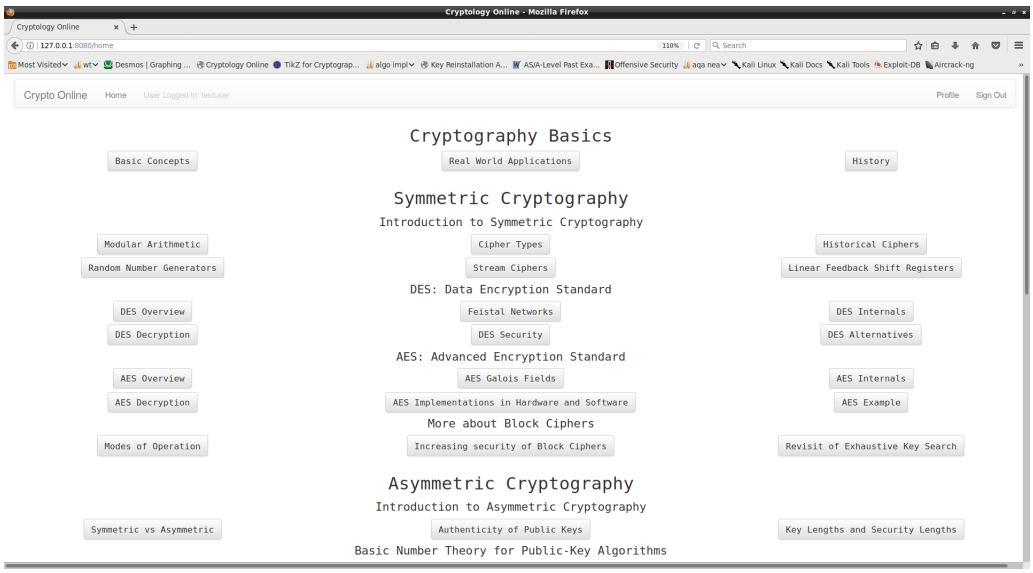


Figure 4.11: Starting at the Home Page and then clicking the Profile Button

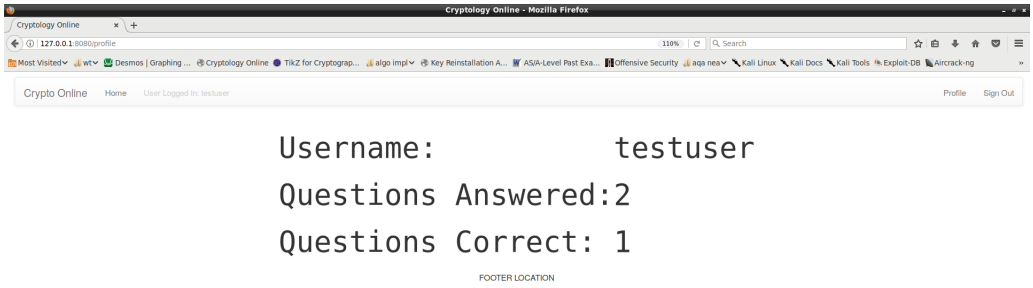


Figure 4.12: After clicking the button Profile Page Loads

4.2.7 Test Ref 07

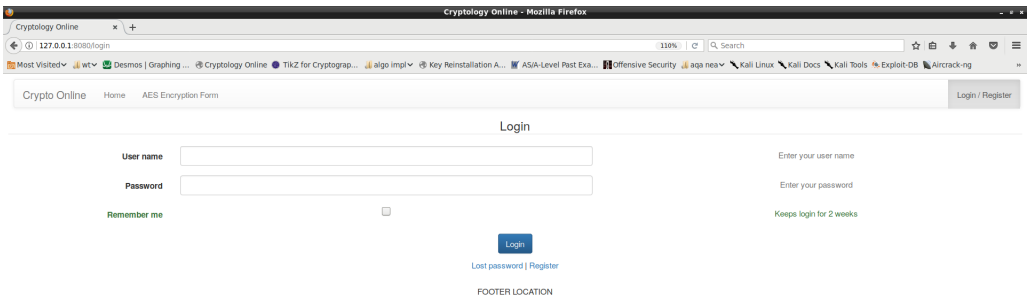


Figure 4.13: Starting at the Register Page and then clicking the Register Button

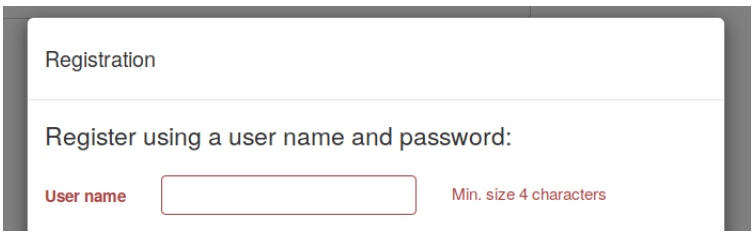


Figure 4.14: The username field states that you need to enter a username

4.2.8 Test Ref 08

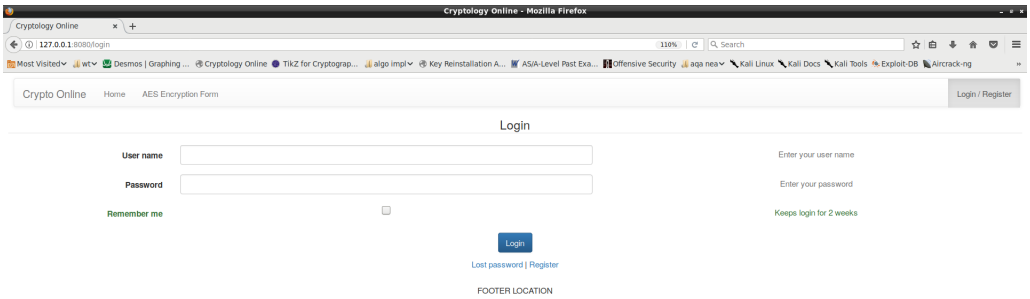


Figure 4.15: Starting at the Register Page and then clicking the Register Button

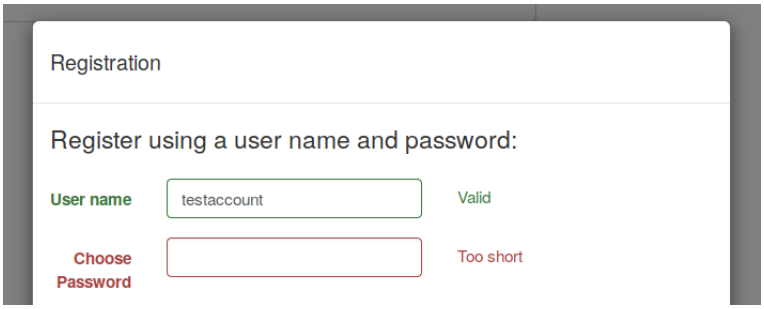


Figure 4.16: The password field states that you need to enter a password

4.2.9 Test Ref 09

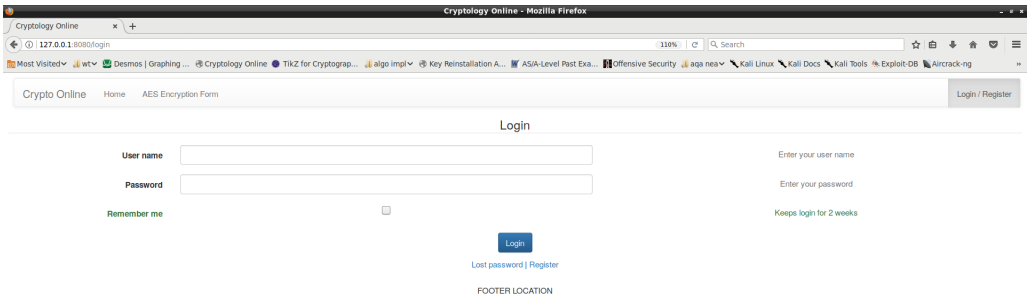


Figure 4.17: Starting at the Register Page and then clicking the Register Button

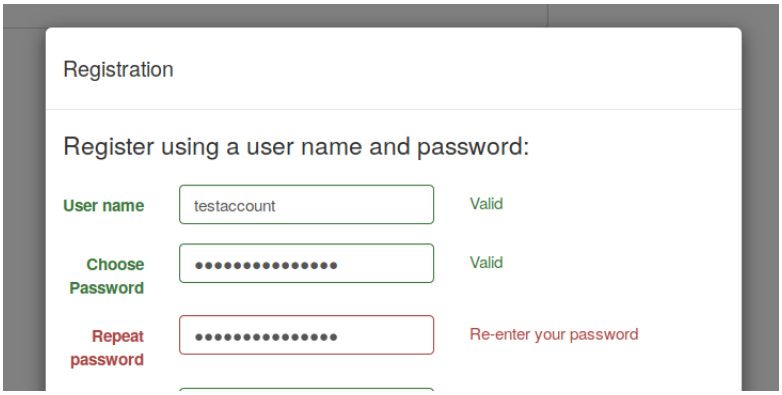


Figure 4.18: Having two different Passwords causes an error

4.2.10 Test Ref 10

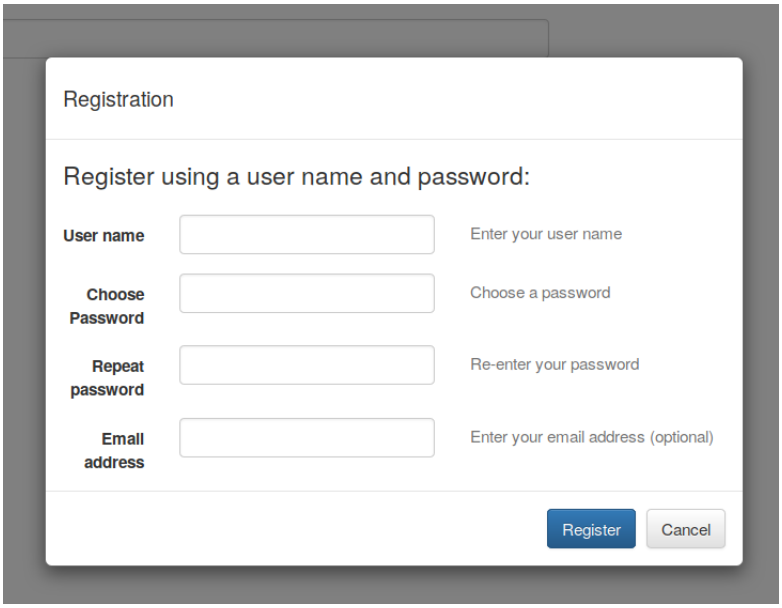


Figure 4.19: Navigating to the Register Form

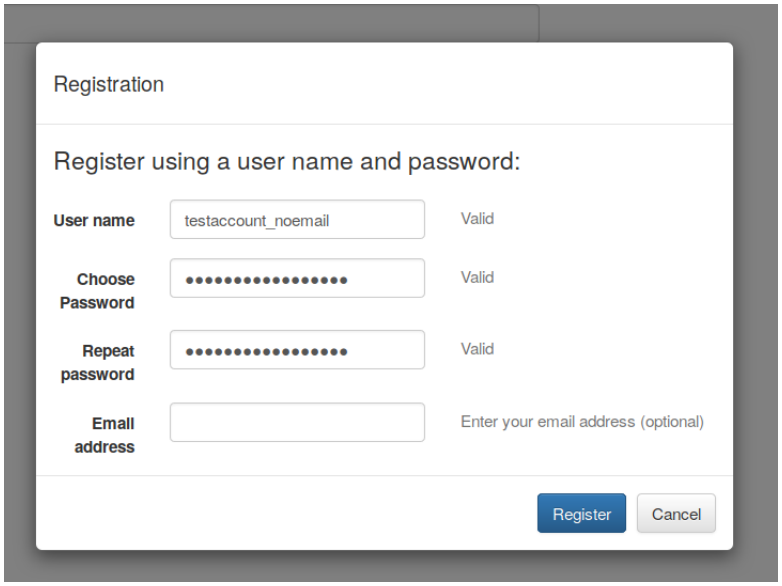


Figure 4.20: Entering Registration information with no email

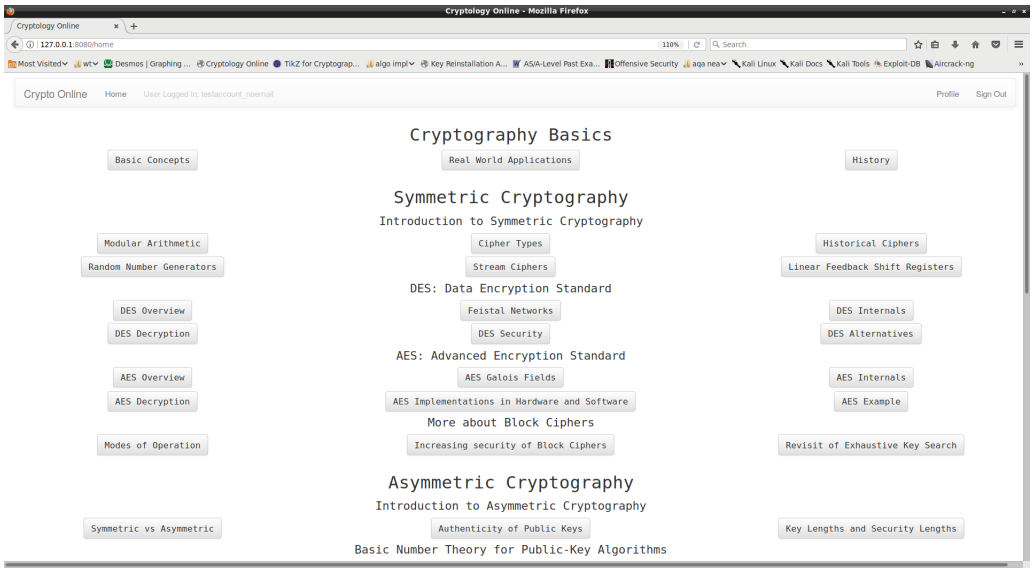


Figure 4.21: User is Registered and Logged in

4.2.11 Test Ref 11

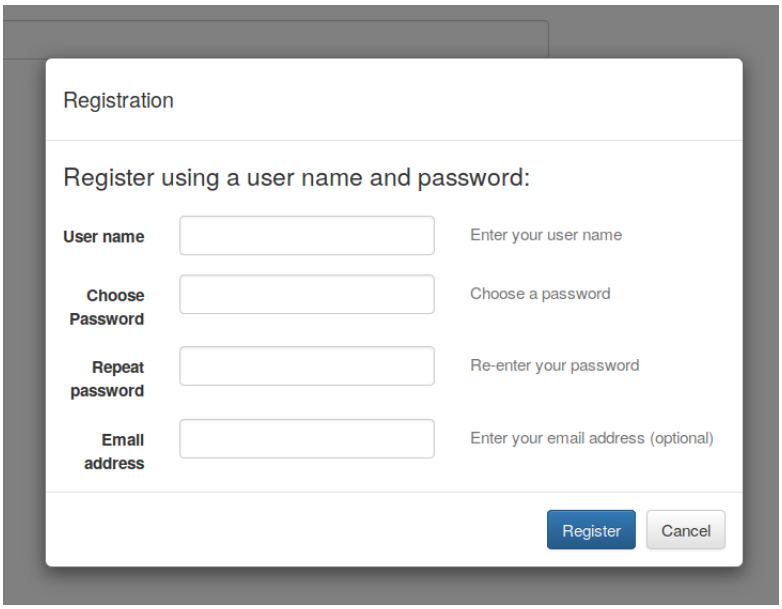


Figure 4.22: Navigating to the Register Form

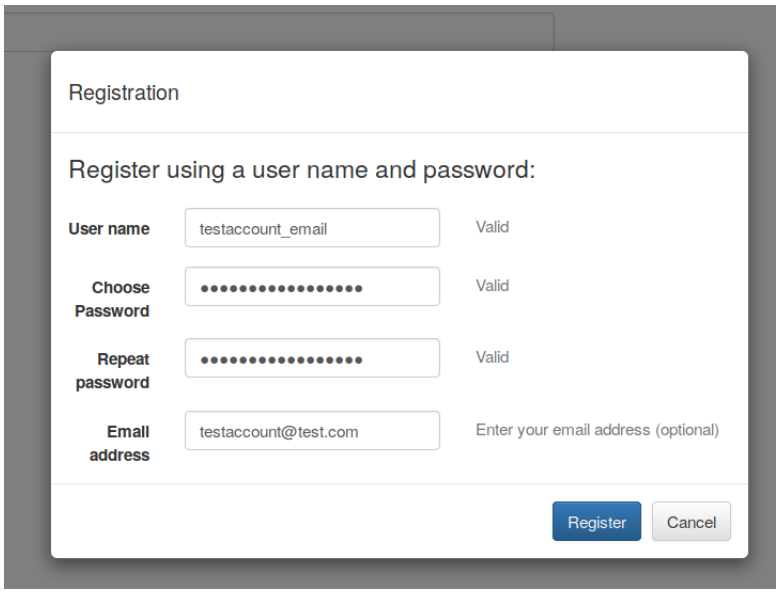


Figure 4.23: Entering Registration information with an email

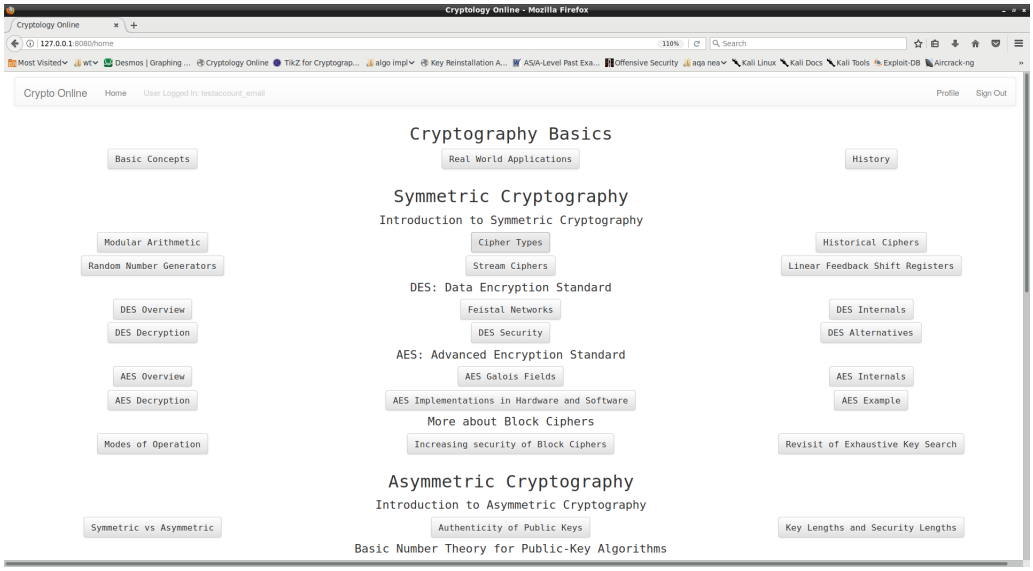


Figure 4.24: User is Registered and Logged in

```
MIME-Version: 1.0
From: "Wt Auth module" <noreply-auth@www.webtoolkit.eu>
Subject: User account activation Link
To: <testaccountsemail@test.com>
Content-Type: multipart/alternative; boundary="--=_epDBkTRHEp-MLrsrcJNbfDetS2Z9GWNK)bnx)h5Awfo7wGjyr-"

----=_epDBkTRHEp-MLrsrcJNbfDetS2Z9GWNK)bnx)h5Awfo7wGjyr-
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: quoted-printable


Hello testaccount,

Thank you for joining!

To complete your registration, please finally confirm your account by
clicking on the following link or copying the URL into your browser.

Please click here to confirm your registration or copy and paste the
following URL into your browser: (Note: be sure to copy the entire
URL, including any part of it which goes onto a second line.)

http://127.0.0.1:8080/auth/mail/wZgcgYApqoOZ0c8xOTUZAGQ4F8ZuJpRy
```

Figure 4.25: If I had a running Email Server on my machine then the user would have received this Email containing an activation link for their account

4.2.12 Test Ref 12

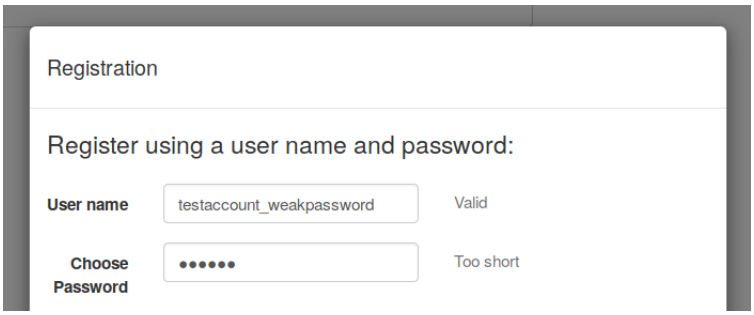


Figure 4.26: Password needs to be at least 7 characters long

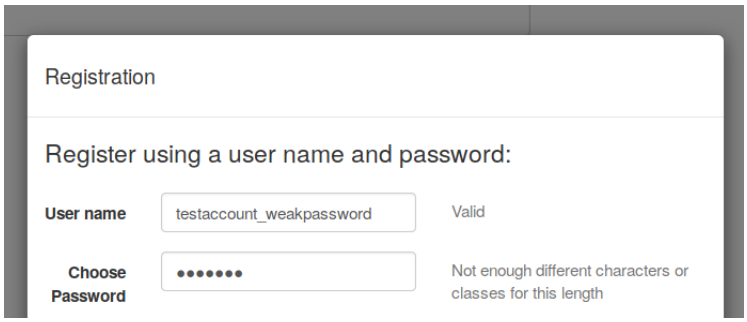


Figure 4.27: Password needs to contain enough characters from different classes

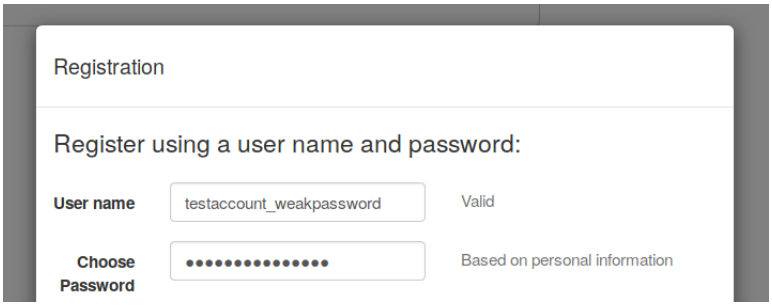


Figure 4.28: Password cannot be based of username

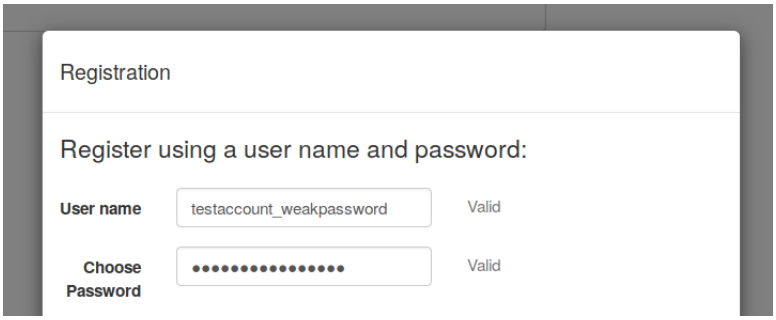


Figure 4.29: Password with suitable strength is met and classed as a valid password

4.2.13 Test Ref 13

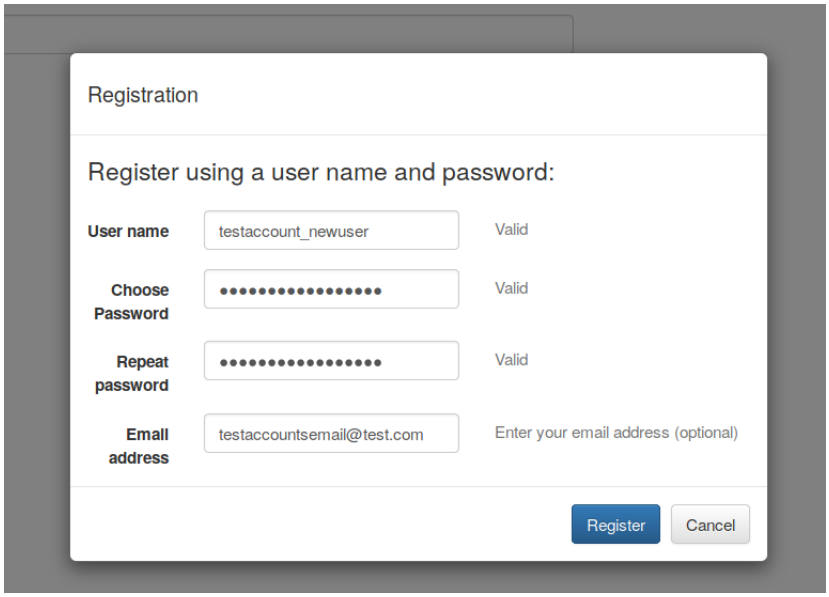


Figure 4.30: New User is about to register

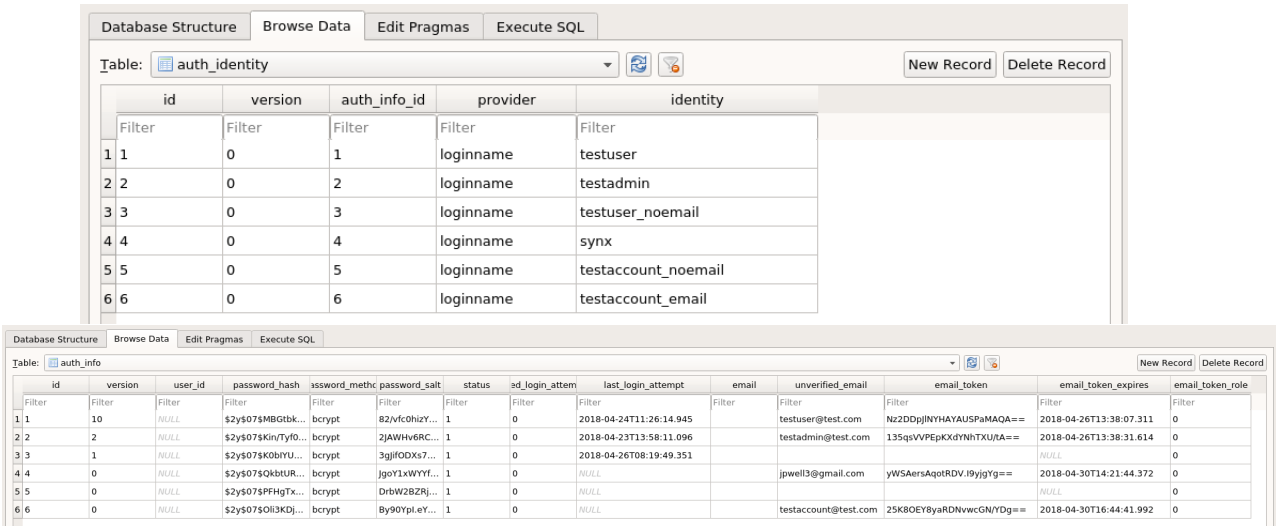


Figure 4.31: Authentication Tables do not contain information on new user

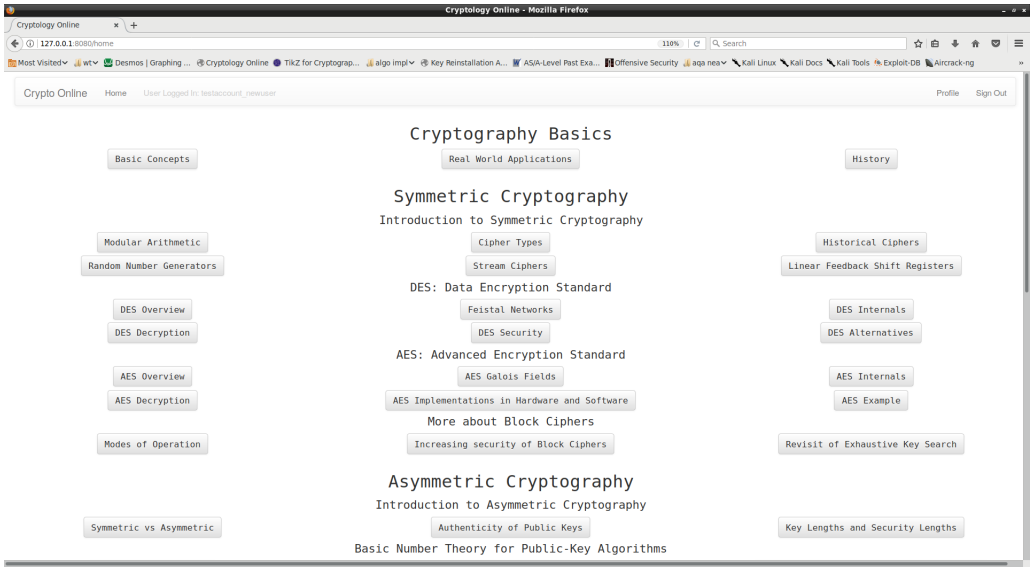


Figure 4.32: New User has registered and is logged in

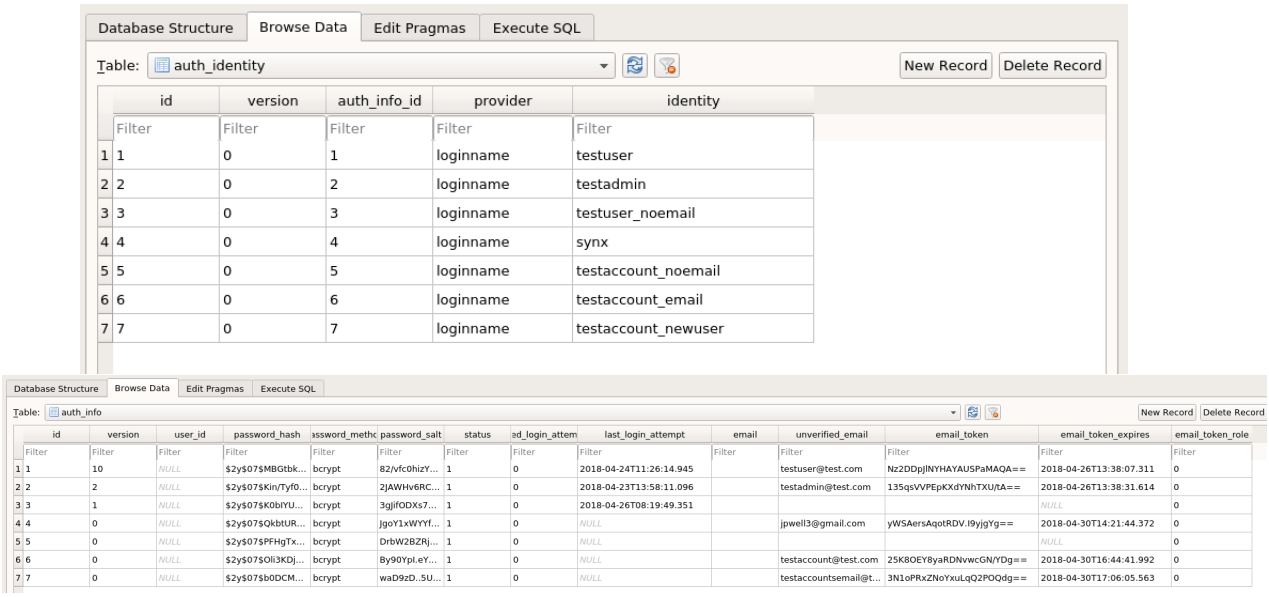


Figure 4.33: New User information is added into authentication tables

4.2.14 Test Ref 14

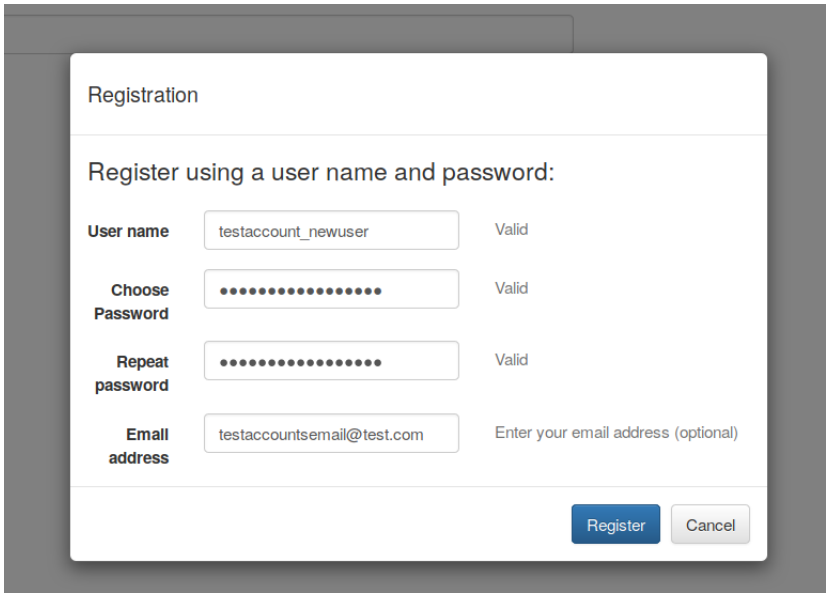
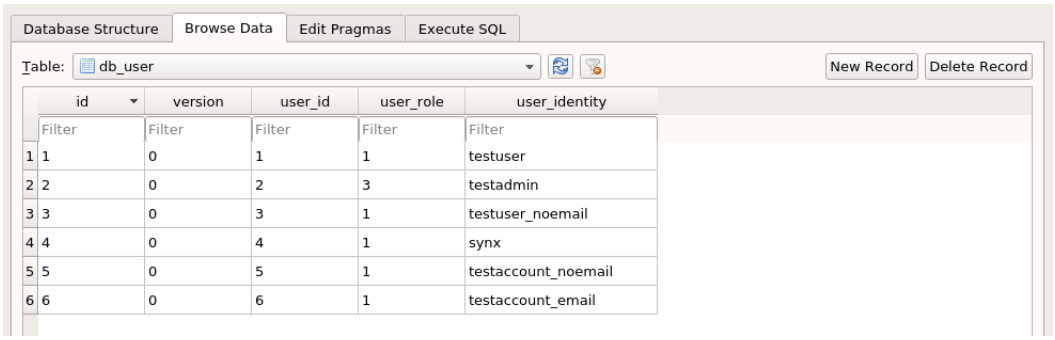


Figure 4.34: New User is about to register



	id	version	user_id	user_role	user_identity
	Filter	Filter	Filter	Filter	Filter
1	1	0	1	1	testuser
2	2	0	2	3	testadmin
3	3	0	3	1	testuser_noemail
4	4	0	4	1	synx
5	5	0	5	1	testaccount_noemail
6	6	0	6	1	testaccount_email

Figure 4.35: db_user Table does not have a record for the new user

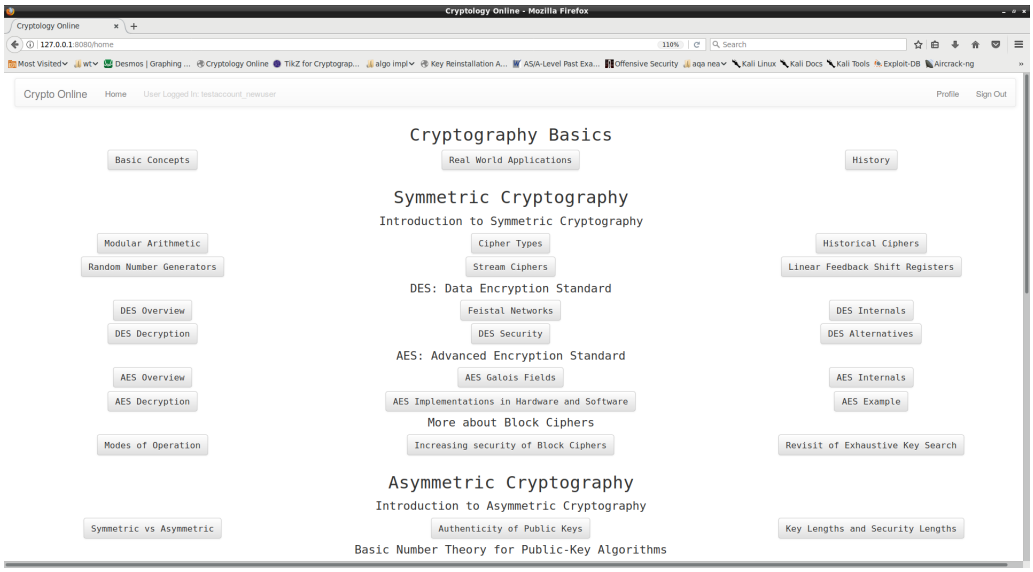
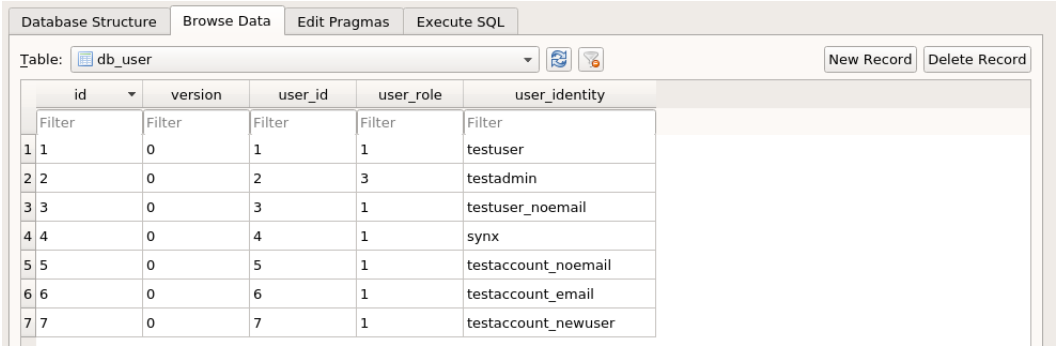


Figure 4.36: New User has registered and is logged in



	id	version	user_id	user_role	user_identity
	Filter	Filter	Filter	Filter	Filter
1	1	0	1	1	testuser
2	2	0	2	3	testadmin
3	3	0	3	1	testuser_noemail
4	4	0	4	1	synx
5	5	0	5	1	testaccount_noemail
6	6	0	6	1	testaccount_email
7	7	0	7	1	testaccount_newuser

Figure 4.37: db_user Table now has a record for the new user

4.2.15 Test Ref 15

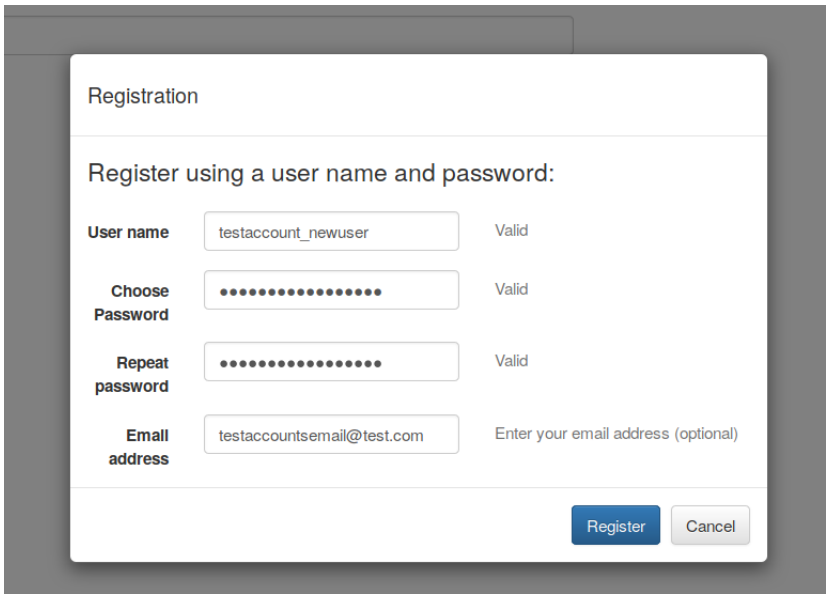


Figure 4.38: New User is about to register

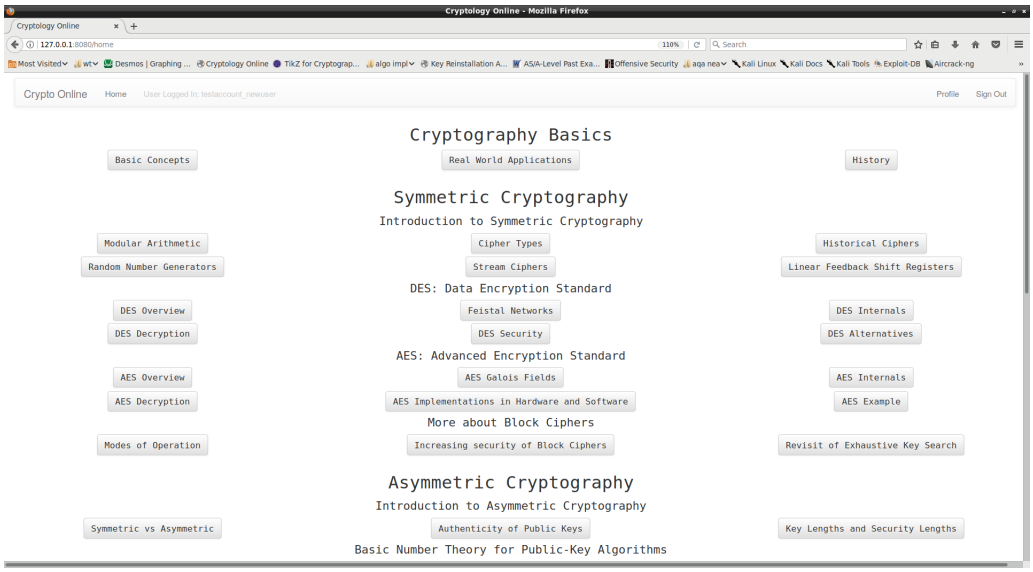


Figure 4.39: New User has registered and is taken straight to the Homepage

4.2.16 Test Ref 16

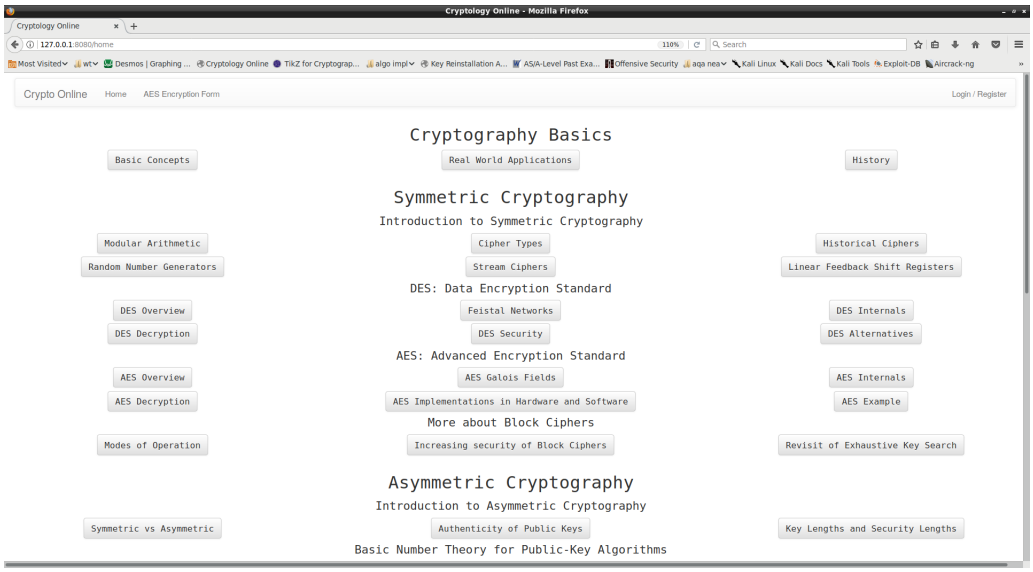


Figure 4.40: With no user logged in the normal header is shown

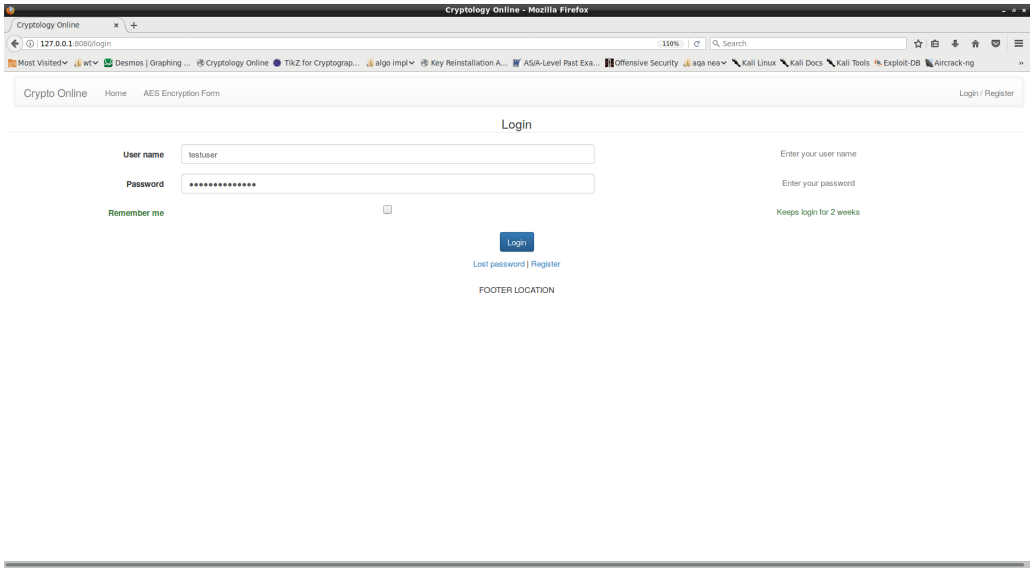


Figure 4.41: A Normal User logs in

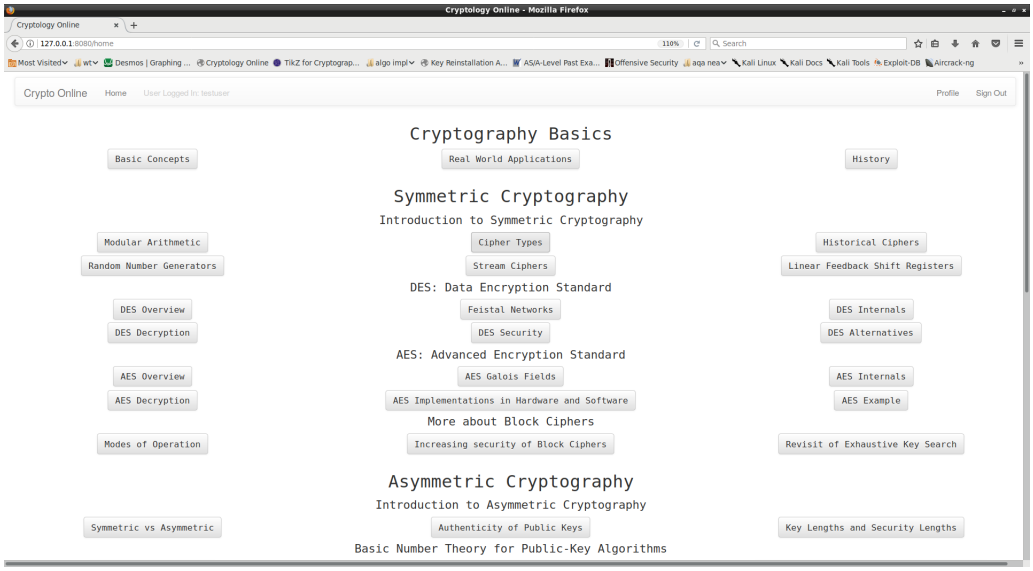


Figure 4.42: The custom page header is shown to the user

4.2.17 Test Ref 17

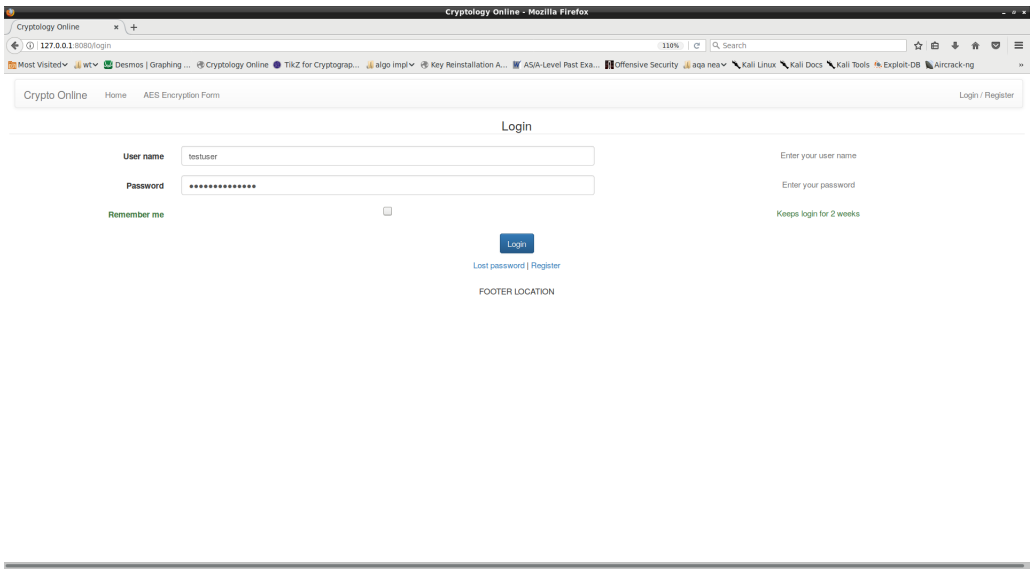


Figure 4.43: A Normal User logs in

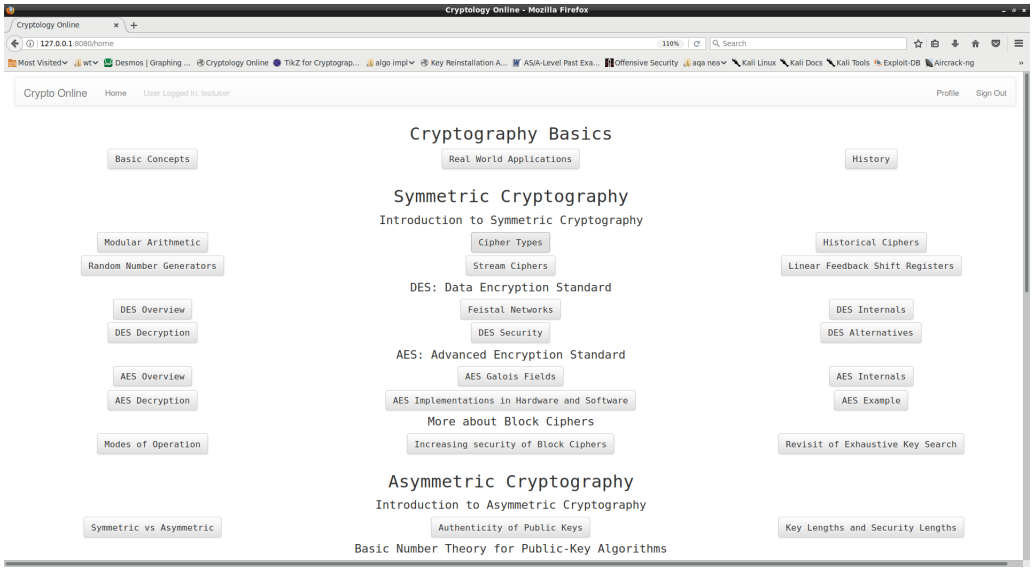


Figure 4.44: The normal user header is loaded, not the admin header

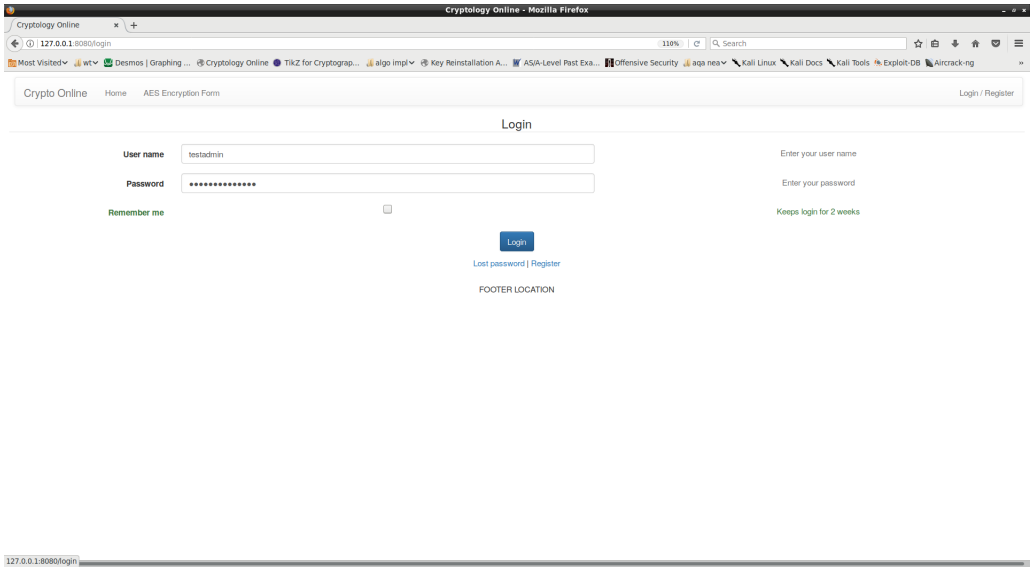


Figure 4.45: A Admin user logs in

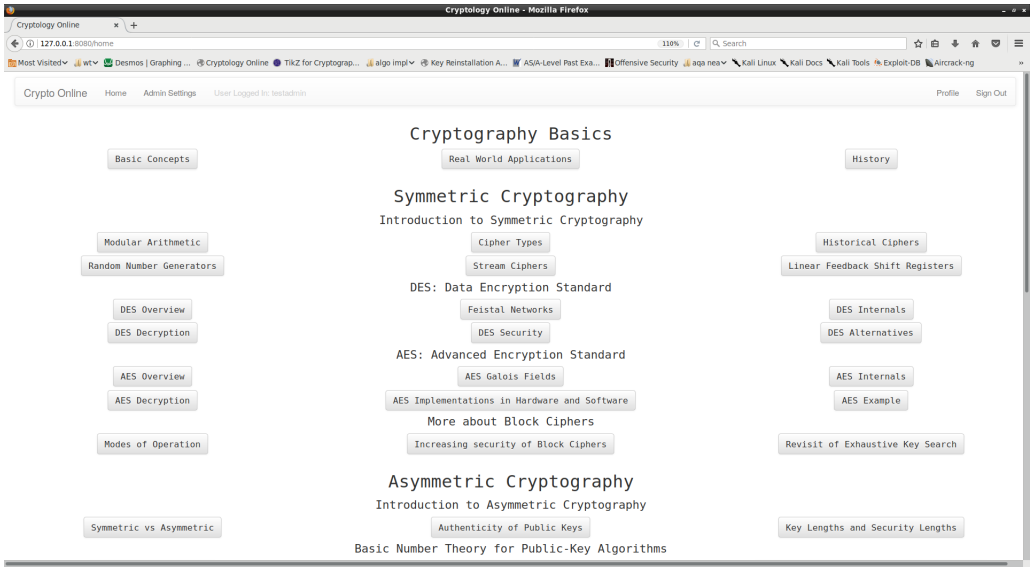


Figure 4.46: The admin user header is loaded

4.2.18 Test Ref 18

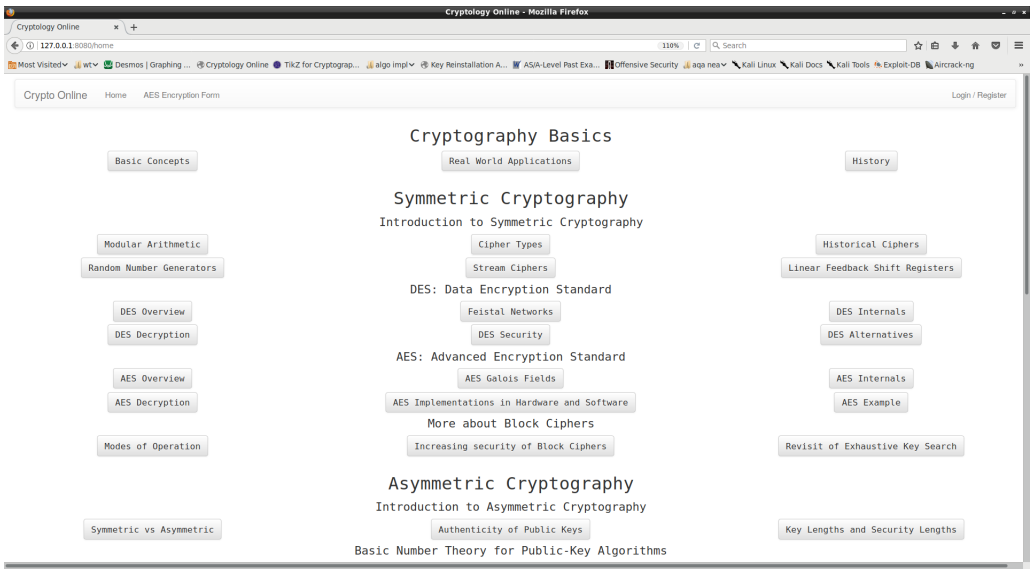


Figure 4.47: No User is logged in

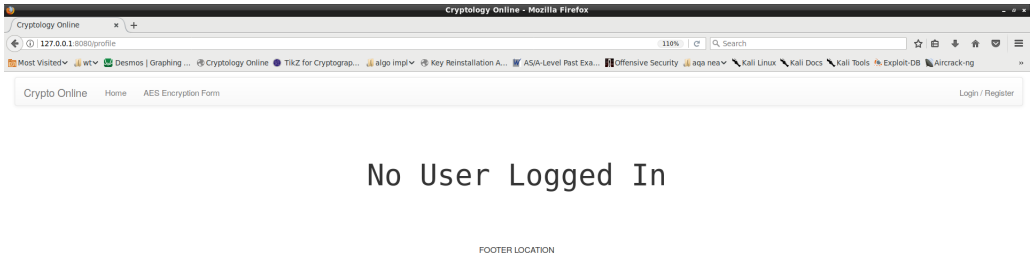


Figure 4.48: Profile page shows no user is logged in

4.2.19 Test Ref 19

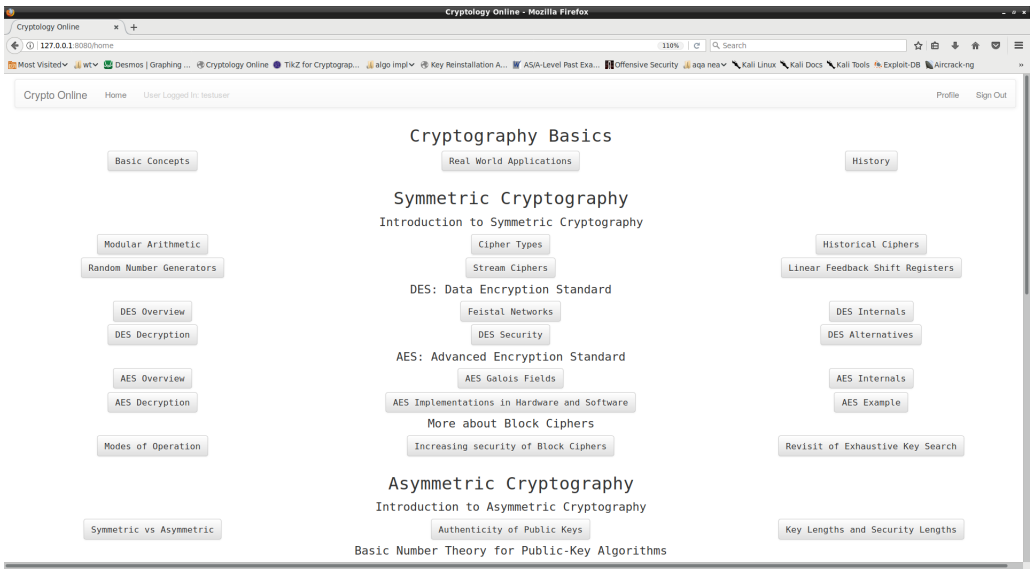


Figure 4.49: A User is logged in

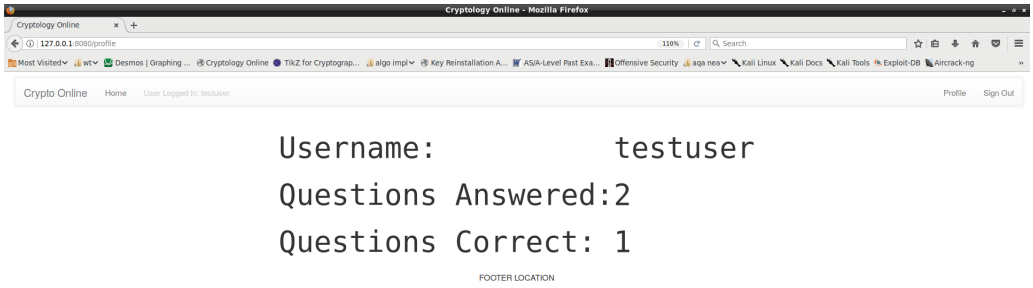


Figure 4.50: Profile page for user is loaded

4.2.20 Test Ref 20

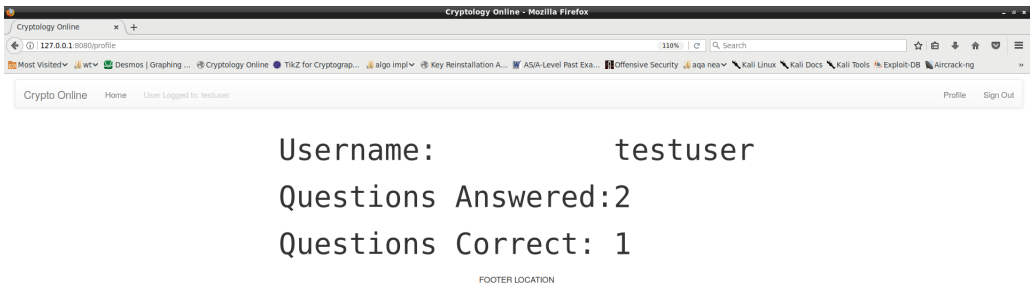


Figure 4.51: Profile page for user shows total questions answered

4.2.21 Test Ref 21

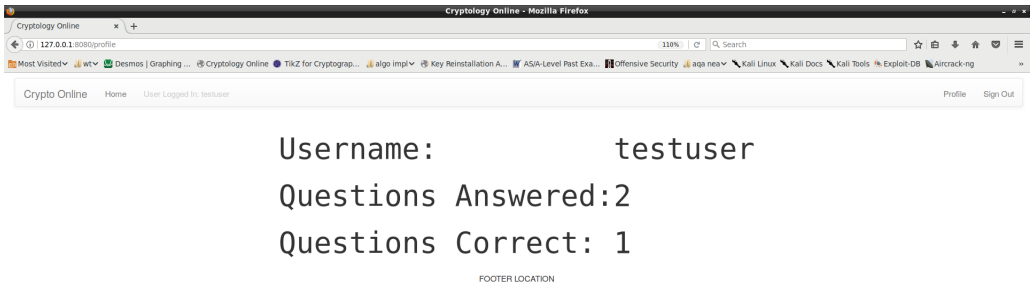


Figure 4.52: Profile page for user shows total questions answered correctly

4.2.22 Test Ref 22

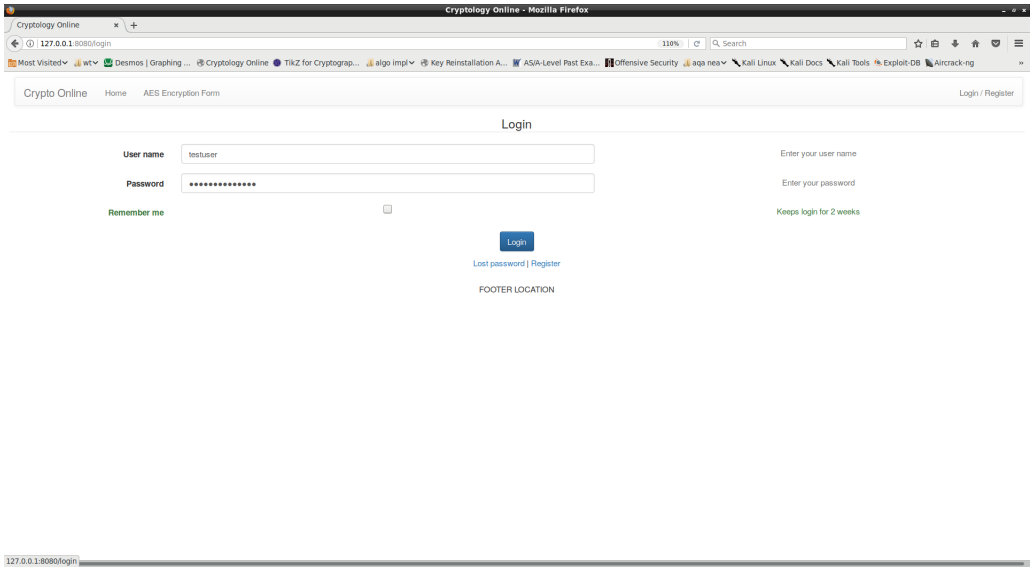


Figure 4.53: User about to login with valid credentials

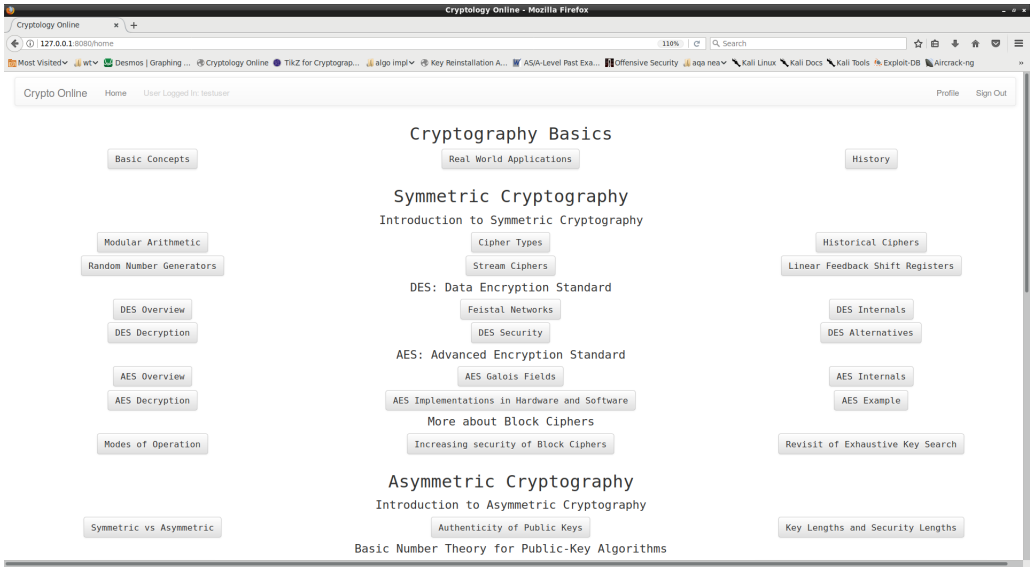


Figure 4.54: The User is logged in

4.2.23 Test Ref 23

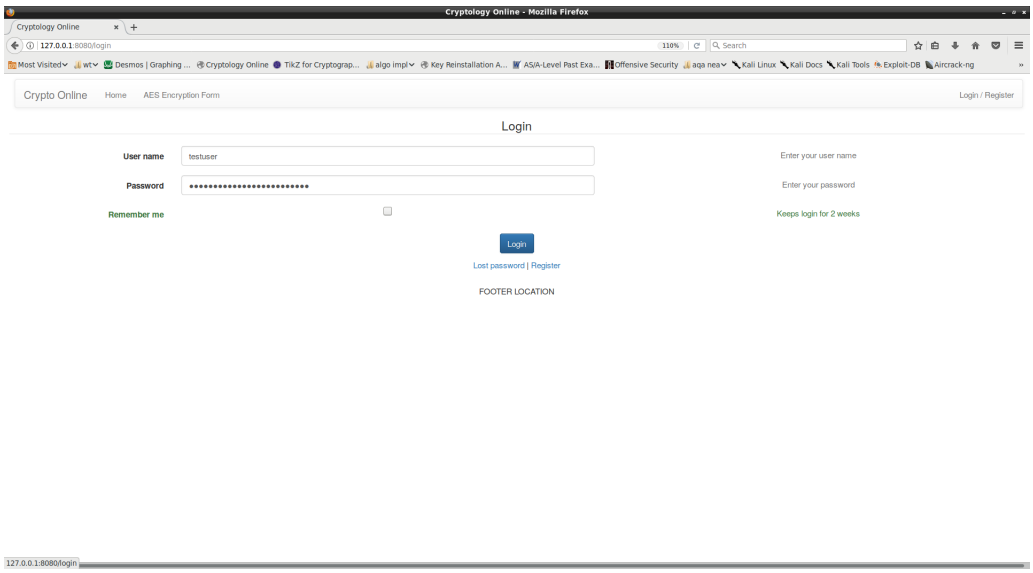


Figure 4.55: User about to login with invalid credentials

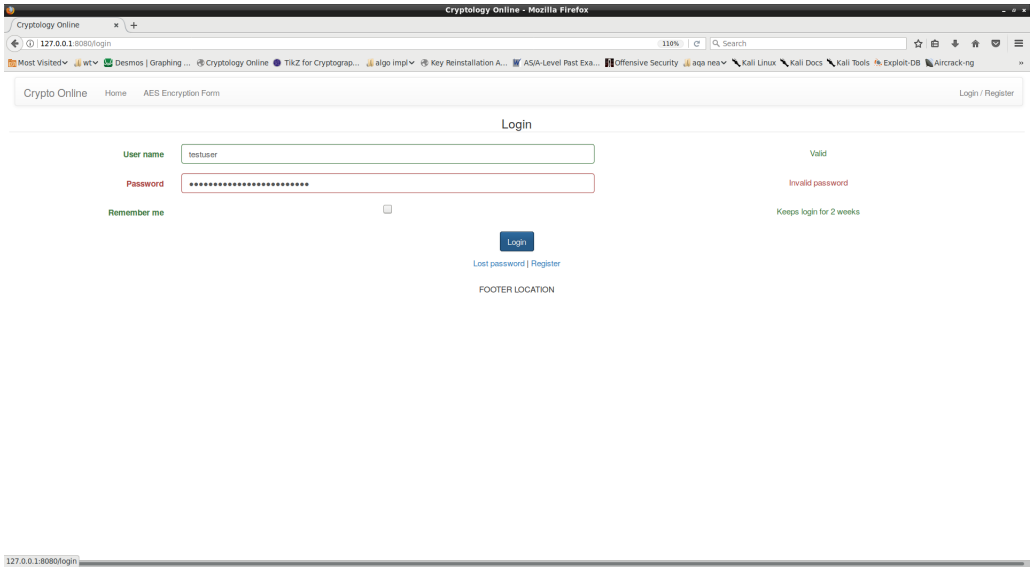


Figure 4.56: The User is not logged in

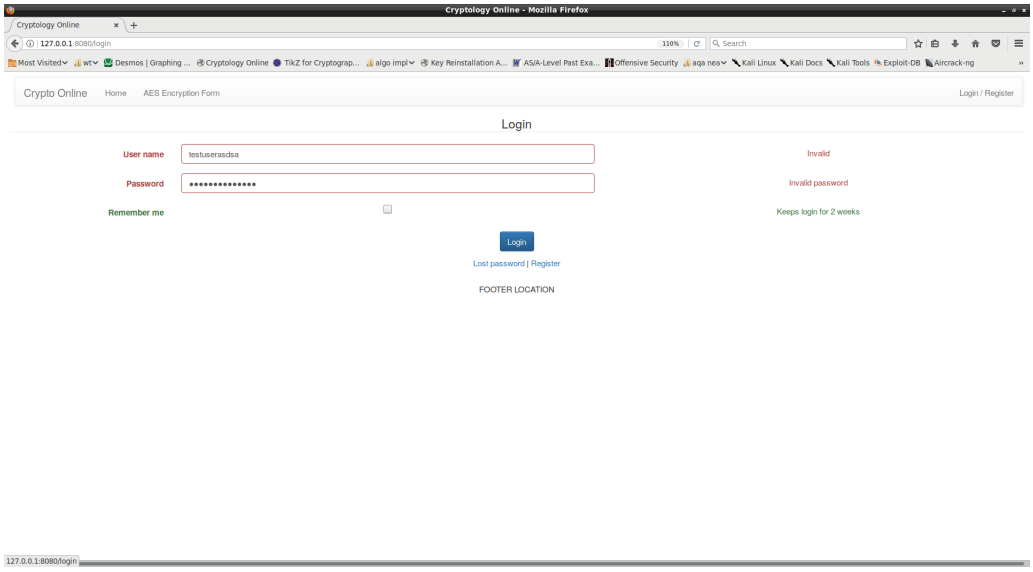


Figure 4.57: It shows the user whether their username or password is wrong

4.2.24 Test Ref 24

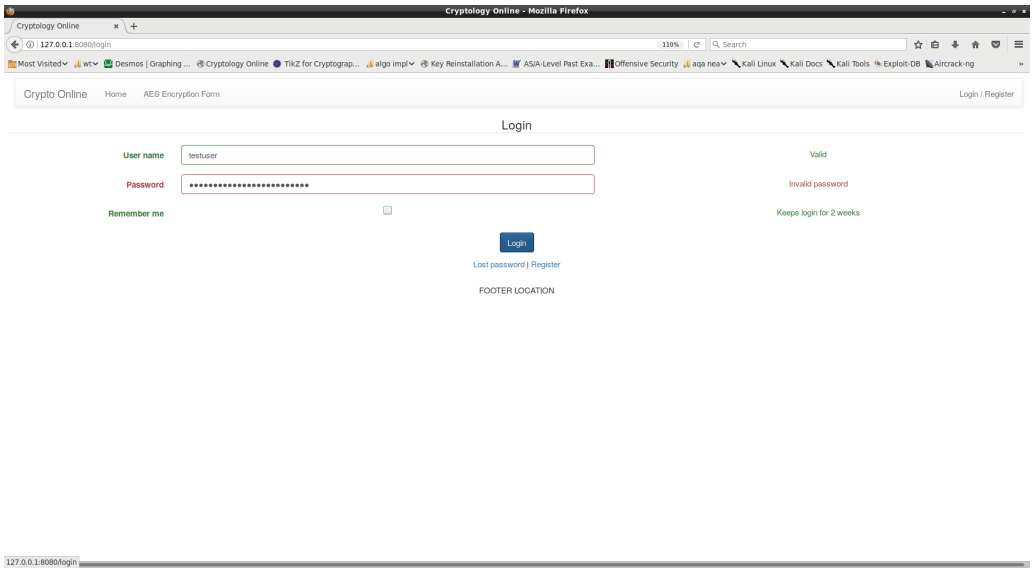


Figure 4.58: User tries to log in with bad credentials

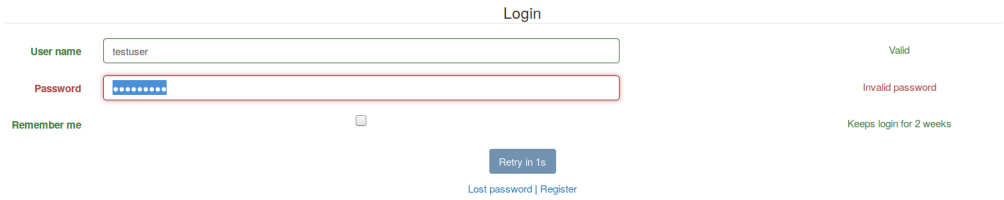


Figure 4.59: The user is given a timeout where they cannot login

4.2.25 Test Ref 25

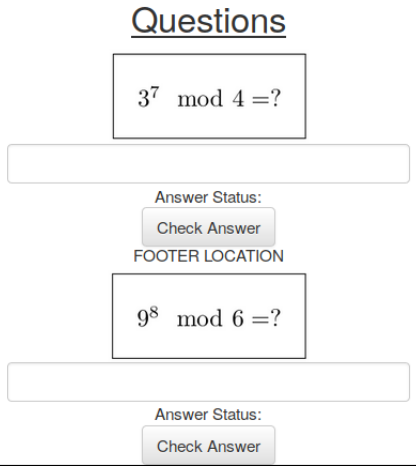


Figure 4.60: User is about to answer a Question

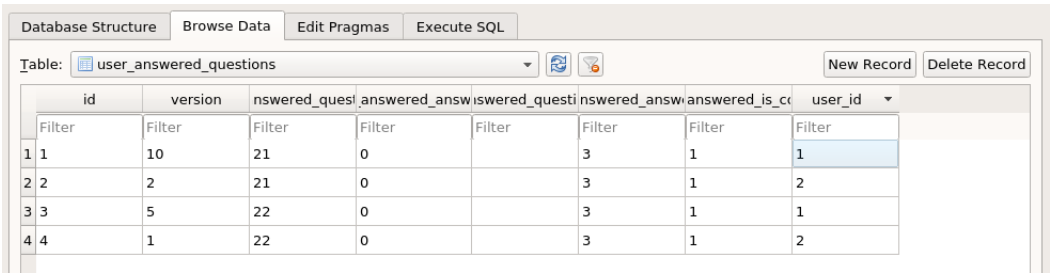


Figure 4.61: User question data is not stored in user_answered_question table

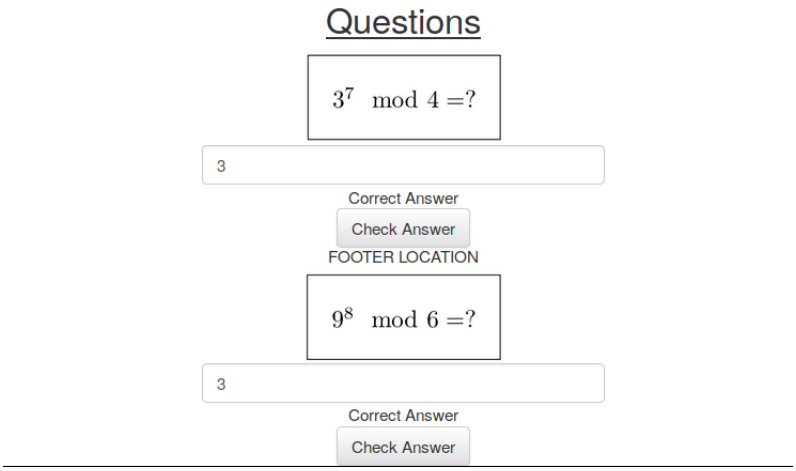


Figure 4.62: User has about to answer a Question

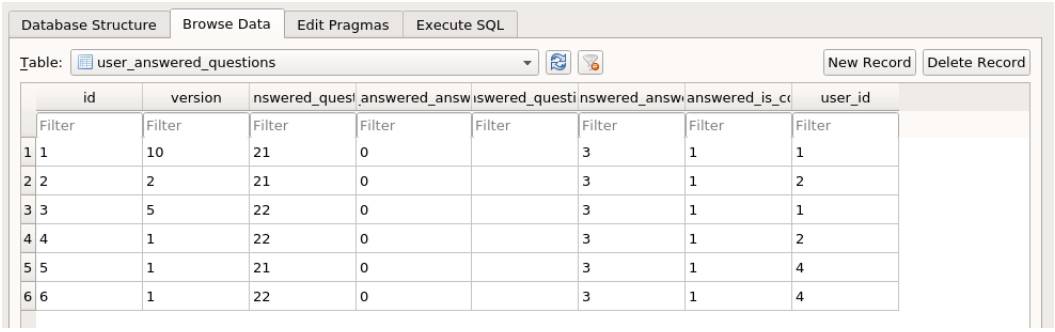


Figure 4.63: User question data is now stored in user_answered_question table

4.2.26 Test Ref 26

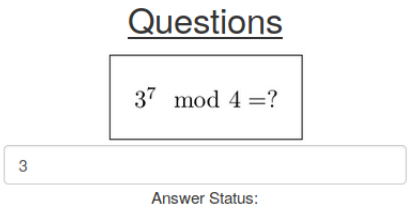


Figure 4.64: User is about to answer a Question Correctly

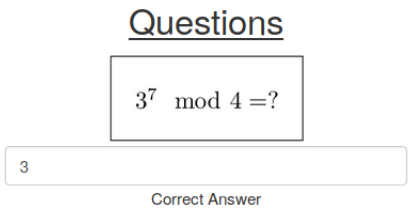


Figure 4.65: It shows the user they got the question correct

4.2.27 Test Ref 27

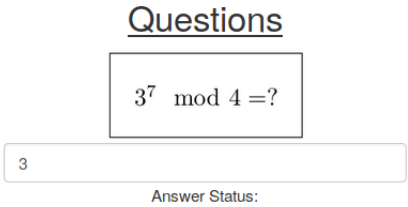


Figure 4.66: User is about to answer a Question Incorrectly

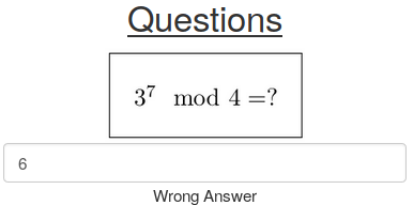


Figure 4.67: It shows the user they got the question incorrect

4.2.28 Test Ref 28

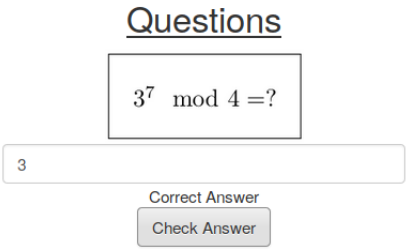


Figure 4.68: User is about to answer a Question they have already answered

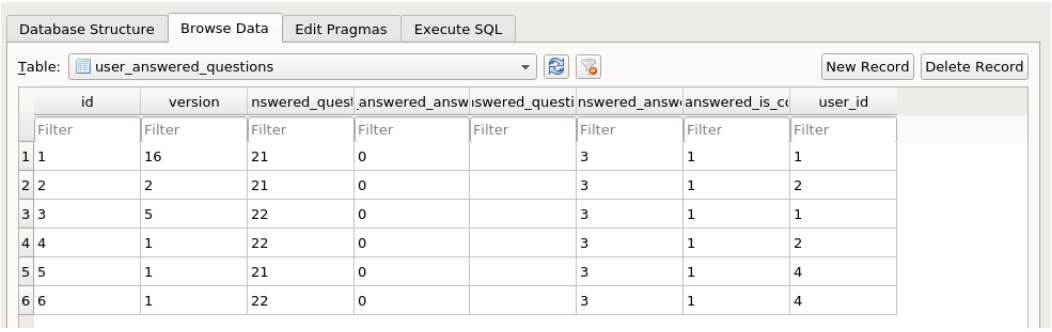


Figure 4.69: The question data is stored in the user_answered_questions Table

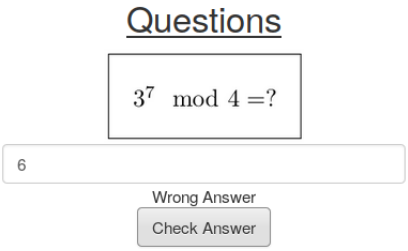


Figure 4.70: User answers the question with a different answer than they did before

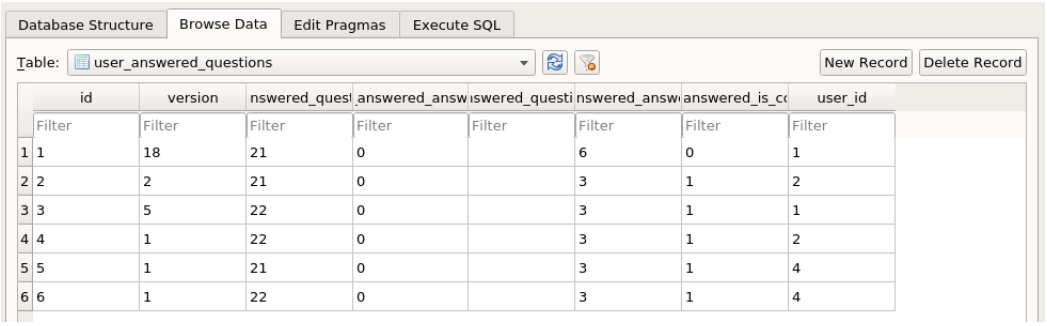


Figure 4.71: The question data stored in the user_answered_questions Table is updated with the user new answer

4.2.29 Test Ref 29

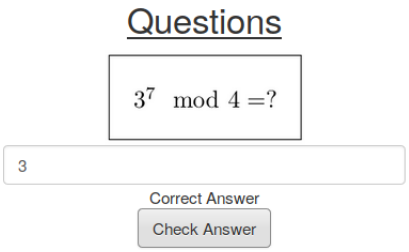


Figure 4.72: User answers the question with the correct answer

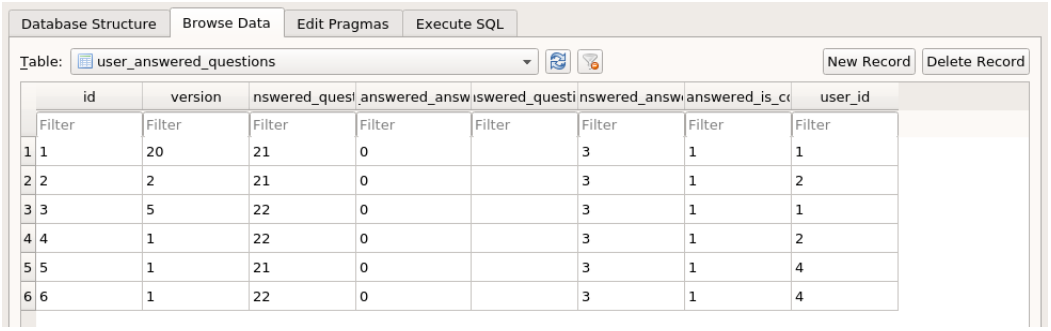


Figure 4.73: The is_correct field is set to 1

4.2.30 Test Ref 30

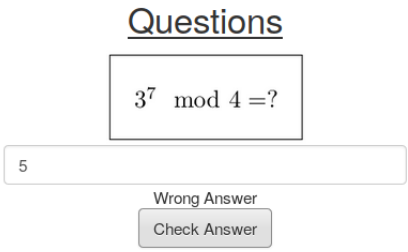


Figure 4.74: User answers the question with the incorrect answer

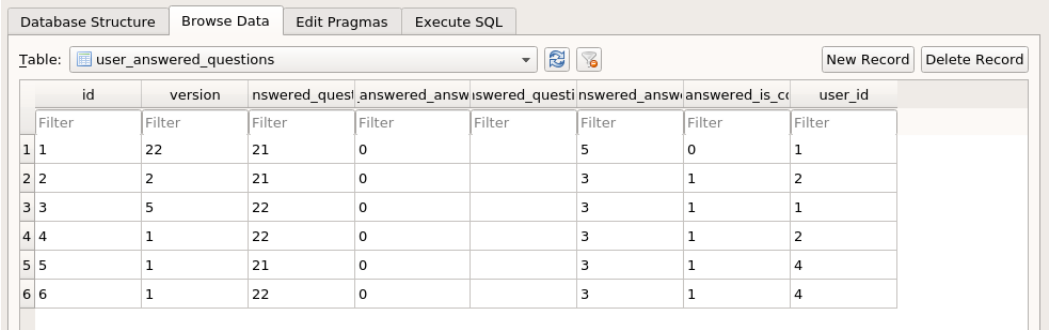


Figure 4.75: The is_correct field is set to 0

4.3 AES Implementation

My AES Implementation has been tested against the Federal Information Processing Standard Publication 197 (FIPS 197) test vectors. All test vectors use hexadecimal notation.

4.3.1 AES Implementation Testing Code

In order to completely test my solution I have written a short program that uses to test vectors from FIPS 197. I can then compare the results from my program and from the FIPS 197 document. If they match then I know my implementation of the algorithm is correct.

```
1  /**
2   * @file aes_implementation_test.cc
3   * @date 28/02/2018
4   *
5   * @breif This file will contain test methods to verify the functionality of my implementation of the AES Algorithm
6   *        All of the test keys and plaintexts are the same as shown in FIPS 197.
7   *
8   * @version 0.01
9   * @author Jacob Powell
10  *
11  */
12
13 #include "aes_implementation.h"
14
15 #include <iostream>
16 #include <iomanip>
17
18 void test_aes_256(){
19     AESImplementation aes(AES256);
20     byte key_256[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f,
21                       0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17, 0x18, 0x19, 0x1a, 0x1b, 0x1c, 0x1d, 0x1e, 0x1f};
22     byte plaintext[] = {0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff};
23     byte ciphertext[16];
24     byte plaintext2[16];
25
26     aes.encrypt_block(plaintext, ciphertext, key_256);
27
28     std::cout << "Cipher Text: ";
29     for(byte i = 0; i < 16; i++){
30         std::cout << std::hex << std::setfill('0') << std::setw(2) << unsigned(ciphertext[i]);
31     }
32     std::cout << std::endl;
33
```

```
34      aes.decrypt_block(ciphertext, plaintext2, key_256);
35
36      std::cout << "Plaintext: ";
37      for(byte i = 0; i < 16; i++){
38          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
            unsigned(plaintext2[i]);
39      }
40      std::cout << std::endl;
41
42  }
43
44  void test_aes_192(){
45      AESImplementation aes(AES192);
46      byte key_192[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0↩
            x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f,
47                        0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0x17};
48      byte plaintext[] = {0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0↩
            x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff};
49      byte ciphertext[16];
50      byte plaintext2[16];
51
52      aes.encrypt_block(plaintext, ciphertext, key_192);
53
54      std::cout << "Cipher Text: ";
55      for(byte i = 0; i < 16; i++){
56          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
            unsigned(ciphertext[i]);
57      }
58      std::cout << std::endl;
59
60      aes.decrypt_block(ciphertext, plaintext2, key_192);
61
62      std::cout << "Plaintext: ";
63      for(byte i = 0; i < 16; i++){
64          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
            unsigned(plaintext2[i]);
65      }
66      std::cout << std::endl;
67
68      std::cout << std::endl;
69  }
70
71  void test_aes_128(){
72      AESImplementation aes(AES128);
73
74      byte key_128[] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0↩
            x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f};
75      byte plaintext[] = {0x00, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0↩
            x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff};
76      byte ciphertext[16];
77      byte plaintext2[16];
78
79      aes.encrypt_block(plaintext, ciphertext, key_128, plaintext);
80
81      std::cout << "Cipher Text: ";
```

```
82      for(byte i = 0; i < 16; i++){
83          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(ciphertext[i]);
84      }
85      std::cout << std::endl;
86
87      aes.decrypt_block(ciphertext, plaintext2, key_128);
88
89      std::cout << "Plaintext: ";
90      for(byte i = 0; i < 16; i++){
91          std::cout << std::hex << std::setfill('0') << std::setw(2) << ↩
                unsigned(plaintext2[i]);
92      }
93      std::cout << std::endl;
94
95  }
96
97
98  int main(){
99      test_aes_128();
100     test_aes_192();
101     test_aes_256();
102 }
```

4.3.2 AES Implementation Testing Code Output

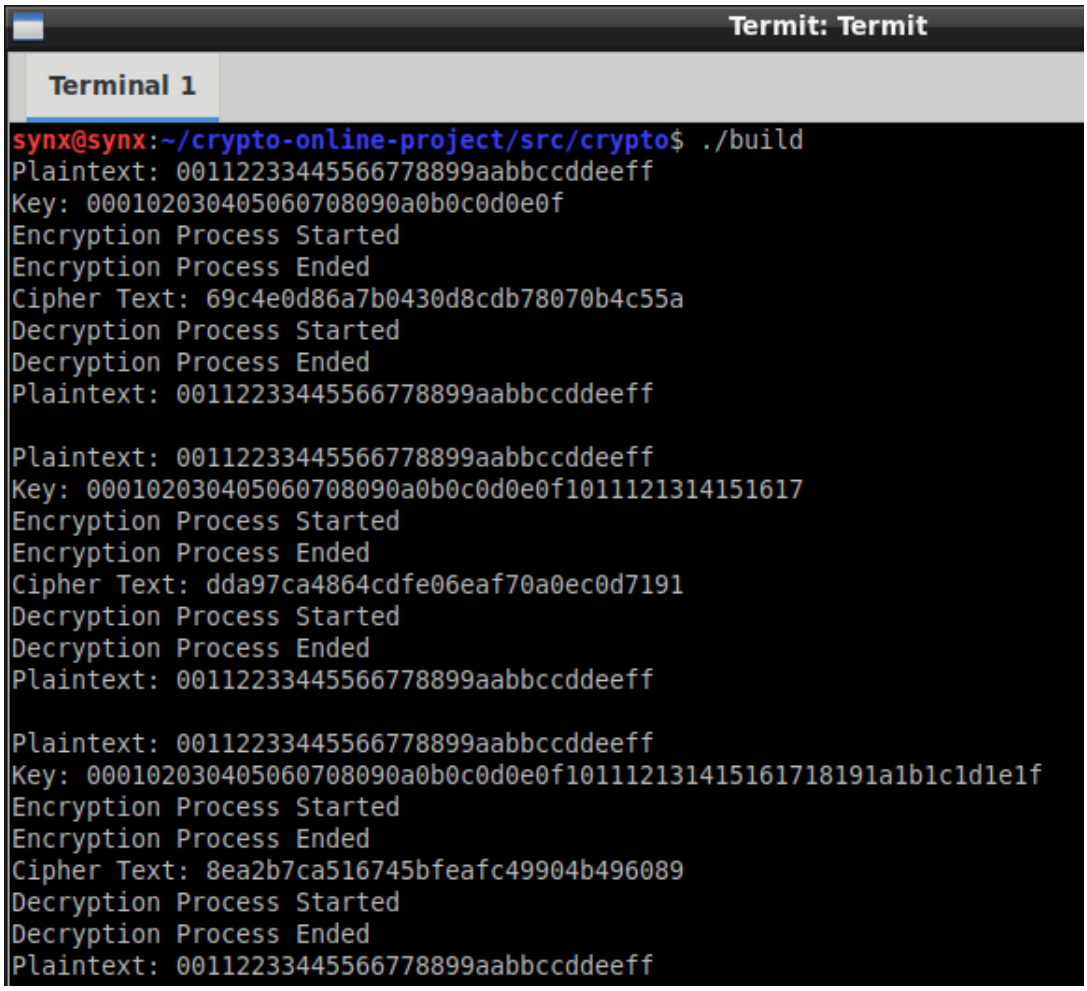


Figure 4.76: The Output from the AES Testing program

4.3.3 AES-128

$$Nk = 4, Nr = 10$$
$$Plaintext = 00112233445566778899aabbccddeeff$$
$$Key = 000102030405060708090a0b0c0d0e0f$$

4.3.3.1 Cipher (ENCRYPT)

round[0].input	00112233445566778899aabbccddeeff
round[0].k_sch	000102030405060708090a0b0c0d0e0f
round[1].start	00102030405060708090a0b0c0d0e0f0
round[1].s_box	63cab7040953d051cd60e0e7ba70e18c

round[1].s_row	6353e08c0960e104cd70b751bacad0e7
round[1].m_col	5f72641557f5bc92f7be3b291db9f91a
round[1].k_sch	d6aa74fdd2af72fadaa678f1d6ab76fe
round[2].start	89d810e8855ace682d1843d8cb128fe4
round[2].s_box	a761ca9b97be8b45d8ad1a611fc97369
round[2].s_row	a7be1a6997ad739bd8c9ca451f618b61
round[2].m_col	ff87968431d86a51645151fa773ad009
round[2].k_sch	b692cf0b643dbdf1be9bc5006830b3fe
round[3].start	4915598f55e5d7a0daca94fa1f0a63f7
round[3].s_box	b59cb73fcd90ee05774222dc067fb68
round[3].s_row	3bd92268fc74fb735767cbe0c0590e2d
round[3].m_col	4c9c1e66f771f0762c3f868e534df256
round[3].k_sch	b6ff744ed2c2c9bf6c590cbf0469bf41
round[4].start	fa636a2825b339c940668a3157244d17
round[4].s_box	2dfb02343f6d12dd09337ec75b36e3f0
round[4].s_row	2d6d7ef03f33e334093602dd5bfb12c7
round[4].m_col	6385b79ffc538df997be478e7547d691
round[4].k_sch	47f7f7bc95353e03f96c32bcfd058dfd
round[5].start	247240236966b3fa6ed2753288425b6c
round[5].s_box	36400926f9336d2d9fb59d23c42c3950
round[5].s_row	36339d50f9b539269f2c092dc4406d23
round[5].m_col	f4bcd45432e554d075f1d6c51dd03b3c
round[5].k_sch	3caaa3e8a99f9deb50f3af57adf622aa
round[6].start	c81677bc9b7ac93b25027992b0261996
round[6].s_box	e847f56514dadde23f77b64fe7f7d490
round[6].s_row	e8dab6901477d4653ff7f5e2e747dd4f
round[6].m_col	9816ee7400f87f556b2c049c8e5ad036
round[6].k_sch	5e390f7df7a69296a7553dc10aa31f6b
round[7].start	c62fe109f75eedc3cc79395d84f9cf5d
round[7].s_box	b415f8016858552e4bb6124c5f998a4c
round[7].s_row	b458124c68b68a014b99f82e5f15554c
round[7].m_col	c57e1c159a9bd286f05f4be098c63439
round[7].k_sch	14f9701ae35fe28c440adf4d4ea9c026
round[8].start	d1876c0f79c4300ab45594add66ff41f
round[8].s_box	3e175076b61c04678dfc2295f6a8bfc0
round[8].s_row	3e1c22c0b6fcbf768da85067f6170495
round[8].m_col	baa03de7a1f9b56ed5512cba5f414d23
round[8].k_sch	47438735a41c65b9e016baf4aebf7ad2
round[9].start	fde3bad205e5d0d73547964ef1fe37f1
round[9].s_box	5411f4b56bd9700e96a0902fa1bb9aa1
round[9].s_row	54d990a16ba09ab596bbf40ea111702f
round[9].m_col	e9f74eec023020f61bf2ccf2353c21c7
round[9].k_sch	549932d1f08557681093ed9cbe2c974e
round[10].start	bd6e7c3df2b5779e0b61216e8b10b689

round[10].s_box	7a9f102789d5f50b2beffd9f3dca4ea7
round[10].s_row	7ad5fda789ef4e272bca100b3d9ff59f
round[10].k_sch	13111d7fe3944a17f307a78b4d2b30c5
round[10].output	69c4e0d86a7b0430d8cdb78070b4c55a

4.3.3.2 Inverse Cipher (DECRYPT)

round[0].iinput	69c4e0d86a7b0430d8cdb78070b4c55a
round[0].ik_sch	13111d7fe3944a17f307a78b4d2b30c5
round[1].istart	7ad5fda789ef4e272bca100b3d9ff59f
round[1].is_box	bdb52189f261b63d0b107c9e8b6e776e
round[1].is_row	bd6e7c3df2b5779e0b61216e8b10b689
round[1].im_col	4773b91ff72f354361cb018ea1e6cf2c
round[1].ik_sch	13aa29be9c8faff6f770f58000f7bf03
round[2].istart	54d990a16ba09ab596bbf40ea111702f
round[2].is_box	fde596f1054737d235febad7f1e3d04e
round[2].is_row	fde3bad205e5d0d73547964ef1fe37f1
round[2].im_col	2d7e86a339d9393ee6570a1101904e16
round[2].ik_sch	1362a4638f2586486bff5a76f7874a83
round[3].istart	3e1c22c0b6fcbf768da85067f6170495
round[3].is_box	d1c4941f7955f40fb46f6c0ad68730ad
round[3].is_row	d1876c0f79c4300ab45594add66ff41f
round[3].im_col	39daee38f4f1a82aaf432410c36d45b9
round[3].ik_sch	8d82fc749c47222be4dadc3e9c7810f5
round[4].istart	b458124c68b68a014b99f82e5f15554c
round[4].is_box	c65e395df779cf09ccf9e1c3842fed5d
round[4].is_row	c62fe109f75eedc3cc79395d84f9cf5d
round[4].im_col	9a39bf1d05b20a3a476a0bf79fe51184
round[4].ik_sch	72e3098d11c5de5f789dfe1578a2cccb
round[5].istart	e8dab6901477d4653ff7f5e2e747dd4f
round[5].is_box	c87a79969b0219bc2526773bb016c992
round[5].is_row	c81677bc9b7ac93b25027992b0261996
round[5].im_col	18f78d779a93eef4f6742967c47f5ffd
round[5].ik_sch	2ec410276326d7d26958204a003f32de
round[6].istart	36339d50f9b539269f2c092dc4406d23
round[6].is_box	2466756c69d25b236e4240fa8872b332
round[6].is_row	247240236966b3fa6ed2753288425b6c
round[6].im_col	85cf8bf472d124c10348f545329c0053
round[6].ik_sch	a8a2f5044de2c7f50a7ef79869671294
round[7].istart	2d6d7ef03f33e334093602dd5bfb12c7
round[7].is_box	fab38a1725664d2840246ac957633931
round[7].is_row	fa636a2825b339c940668a3157244d17

round[7].im_col	fc1fc1f91934c98210fbfb8da340eb21
round[7].ik_sch	c7c6e391e54032f1479c306d6319e50c
round[8].istart	3bd92268fc74fb735767cbe0c0590e2d
round[8].is_box	49e594f755ca638fda0a59a01f15d7fa
round[8].is_row	4915598f55e5d7a0daca94fa1f0a63f7
round[8].im_col	076518f0b52ba2fb7a15c8d93be45e00
round[8].ik_sch	a0db02992286d160a2dc029c2485d561
round[9].istart	a7be1a6997ad739bd8c9ca451f618b61
round[9].is_box	895a43e485188fe82d121068cbd8ced8
round[9].is_row	89d810e8855ace682d1843d8cb128fe4
round[9].im_col	ef053f7c8b3d32fd4d2a64ad3c93071a
round[9].ik_sch	8c56dff0825dd3f9805ad3fc8659d7fd
round[10].istart	6353e08c0960e104cd70b751bacad0e7
round[10].is_box	0050a0f04090e03080d02070c01060b0
round[10].is_row	00102030405060708090a0b0c0d0e0f0
round[10].ik_sch	000102030405060708090a0b0c0d0e0f
round[10].ioutput	00112233445566778899aabbccddeeff

4.3.4 AES-192

$$Nk = 6, Nr = 12$$
$$Plaintext = 00112233445566778899aabbccddeeff$$
$$Key = 000102030405060708090a0b0c0d0e0f$$
$$1011121314151617$$

4.3.4.1 Cipher (ENCRYPT)

round[0].input	00112233445566778899aabbccddeeff
round[0].k_sch	000102030405060708090a0b0c0d0e0f
round[1].start	00102030405060708090a0b0c0d0e0f0
round[1].s_box	63cab7040953d051cd60e0e7ba70e18c
round[1].s_row	6353e08c0960e104cd70b751bacad0e7
round[1].m_col	5f72641557f5bc92f7be3b291db9f91a
round[1].k_sch	1011121314151617584dfd2f95c43f4fe
round[2].start	4f63760643e0aa85aff8c9d041fa0de4
round[2].s_box	84fb386f1ae1ac977941dd70832dd769
round[2].s_row	84e1dd691a41d76f792d389783fbac70
round[2].m_col	9f487f794f955f662afc86abd7f1ab29
round[2].k_sch	544afef55847f0fa4856e2e95c43f4fe
round[3].start	cb02818c17d2af9c62aa64428bb25fd7
round[3].s_box	1f770c64f0b579deaaac432c3d37cf0e
round[3].s_row	1fb5430ef0accf64aa370cde3d77792c
round[3].m_col	b7a53ecbbf9d75a0c40efc79b674cc11
round[3].k_sch	40f949b31cbabd4d48f043b810b7b342
round[4].start	f75c7778a327c8ed8cfebfc1a6c37f53
round[4].s_box	684af5bc0acce85564bb0878242ed2ed
round[4].s_row	68cc08ed0abbd2bc642ef555244ae878
round[4].m_col	7a1e98bdacb6d1141a6944dd06eb2d3e
round[4].k_sch	58e151ab04a2a5557effb5416245080c
round[5].start	22ffc916a81474416496f19c64ae2532
round[5].s_box	9316dd47c2fa92834390a1de43e43f23
round[5].s_row	93faa123c2903f4743e4dd83431692de
round[5].m_col	aaa755b34cffe57cef6f98e1f01c13e6
round[5].k_sch	2ab54bb43a02f8f662e3a95d66410c08
round[6].start	80121e0776fd1d8a8d8c31bc965d1fee
round[6].s_box	cdc972c53854a47e5d64c765904cc028
round[6].s_row	cd54c7283864c0c55d4c727e90c9a465
round[6].m_col	921f748fd96e937d622d7725ba8ba50c
round[6].k_sch	f501857297448d7ebdf1c6ca87f33e3c
round[7].start	671ef1fd4e2a1e03dfdcb1ef3d789b30
round[7].s_box	8572a1542fe5727b9e86c8df27bc1404

round[7].s_row	85e5c8042f8614549ebca17b277272df
round[7].m_col	e913e7b18f507d4b227ef652758acbcc
round[7].k_sch	e510976183519b6934157c9ea351f1e0
round[8].start	0c0370d00c01e622166b8accd6db3a2c
round[8].s_box	fe7b5170fe7c8e93477f7e4bf6b98071
round[8].s_row	fe7c7e71fe7f807047b95193f67b8e4b
round[8].m_col	6cf5edf996eb0a069c4ef21cbfc25762
round[8].k_sch	1ea0372a995309167c439e77ff12051e
round[9].start	7255dad30fb80310e00d6c6b40d0527c
round[9].s_box	40fc5766766c7bcae1d7507f09700010
round[9].s_row	406c501076d70066e17057ca09fc7b7f
round[9].m_col	7478bcdce8a50b81d4327a9009188262
round[9].k_sch	dd7e0e887e2fff68608fc842f9dcc154
round[10].start	a906b254968af4e9b4bdb2d2f0c44336
round[10].s_box	d36f3720907ebf1e8d7a37b58c1c1a05
round[10].s_row	d37e3705907a1a208d1c371e8c6fbfb5
round[10].m_col	0d73cc2d8f6abe8b0cf2dd9bb83d422e
round[10].k_sch	859f5f237a8d5a3dc0c02952beefd63a
round[11].start	88ec930ef5e7e4b6cc32f4c906d29414
round[11].s_box	c4cedcabe694694e4b23bfdd6fb522fa
round[11].s_row	c494bffae62322ab4bb5dc4e6fce69dd
round[11].m_col	71d720933b6d677dc00b8f28238e0fb7
round[11].k_sch	de601e7827bcdf2ca223800fd8aeda32
round[12].start	afb73eeb1cd1b85162280f27fb20d585
round[12].s_box	79a9b2e99c3e6cd1aa3476cc0fb70397
round[12].s_row	793e76979c3403e9aab7b2d10fa96ccc

4.3.4.2 Inverse Cipher (DECRYPTION)

round[0].iinput	dda97ca4864cdfe06eaf70a0ec0d7191
round[0].ik_sch	a4970a331a78dc09c418c271e3a41d5d
round[1].istart	793e76979c3403e9aab7b2d10fa96ccc
round[1].is_box	afd10f851c28d5eb62203e51fbb7b827
round[1].is_row	afb73eeb1cd1b85162280f27fb20d585
round[1].im_col	122a02f7242ac8e20605afce51cc7264
round[1].ik_sch	d6bebd0dc209ea494db073803e021bb9
round[2].istart	c494bffae62322ab4bb5dc4e6fce69dd
round[2].is_box	88e7f414f532940eccd293b606ece4c9
round[2].is_row	88ec930ef5e7e4b6cc32f4c906d29414
round[2].im_col	5cc7aecce3c872194ae5ef8309a933c7
round[2].ik_sch	8fb999c973b26839c7f9d89d85c68c72
round[3].istart	d37e3705907a1a208d1c371e8c6fbfb5

round[3].is_box	a98ab23696bd4354b4c4b2e9f006f4d2
round[3].is_row	a906b254968af4e9b4bdb2d2f0c44336
round[3].im_col	b7113ed134e85489b20866b51d4b2c3b
round[3].ik_sch	f77d6ec1423f54ef5378317f14b75744
round[4].istart	406c501076d70066e17057ca09fc7b7f
round[4].is_box	72b86c7c0f0d52d3e0d0da104055036b
round[4].is_row	7255dad30fb80310e00d6c6b40d0527c
round[4].im_col	ef3b1be1b9b0e64bdcb79f1e0a707fbb
round[4].ik_sch	1147659047cf663b9b0ece8dfc0bf1f0
round[5].istart	fe7c7e71fe7f807047b95193f67b8e4b
round[5].is_box	0c018a2c0c6b3ad016db7022d603e6cc
round[5].is_row	0c0370d00c01e622166b8accd6db3a2c
round[5].im_col	592460b248832b2952e0b831923048f1
round[5].ik_sch	dcc1a8b667053f7dcc5c194ab5423a2e
round[6].istart	85e5c8042f8614549ebca17b277272df
round[6].is_box	672ab1304edc9bfddf78f1033d1e1eef
round[6].is_row	671ef1fd4e2a1e03dfdcb1ef3d789b30
round[6].im_col	0b8a7783417ae3a1f9492dc0c641a7ce
round[6].ik_sch	c6deb0ab791e2364a4055fbe568803ab
round[7].istart	cd54c7283864c0c55d4c727e90c9a465
round[7].is_box	80fd31ee768c1f078d5d1e8a96121dbc
round[7].is_row	80121e0776fd1d8a8d8c31bc965d1fee
round[7].im_col	4ee1ddf9301d6352c9ad769ef8d20515
round[7].ik_sch	dd1b7cdaf28d5c158a49ab1dbbc497cb
round[8].istart	93faa123c2903f4743e4dd83431692de
round[8].is_box	2214f132a896251664aec94164ff749c
round[8].is_row	22ffc916a81474416496f19c64ae2532
round[8].im_col	1008ffe53b36ee6af27b42549b8a7bb7
round[8].ik_sch	78c4f708318d3cd69655b701bfc093cf
round[9].istart	68cc08ed0abbd2bc642ef555244ae878
round[9].is_box	f727bf53a3fe7f788cc377eda65cc8c1
round[9].is_row	f75c7778a327c8ed8cfebfc1a6c37f53
round[9].im_col	7f69ac1ed939ebaac8ece3cb12e159e3
round[9].ik_sch	60dcef10299524ce62dbef152f9620cf
round[10].istart	1fb5430ef0accf64aa370cde3d77792c
round[10].is_box	cbd264d717aa5f8c62b2819c8b02af42
round[10].is_row	cb02818c17d2af9c62aa64428bb25fd7
round[10].im_col	cfaf16b2570c18b52e7fef50cab267ae
round[10].ik_sch	4b4ecbdb4d4dcfda5752d7c74949cbde
round[11].istart	84e1dd691a41d76f792d389783fbac70
round[11].is_box	4fe0c9e443f80d06affa76854163aad0
round[11].is_row	4f63760643e0aa85aff8c9d041fa0de4
round[11].im_col	794cf891177bfd1d8a327086f3831b39
round[11].ik_sch	1a1f181d1e1b1c194742c7d74949cbde

round[12].istart	6353e08c0960e104cd70b751bacad0e7
round[12].is_box	0050a0f04090e03080d02070c01060b0
round[12].is_row	0010203040506070809 0a0b0c0d0e0f0
round[12].ik_sch	000102030405060708090a0b0c0d0e0f
round[12].ioutput	00112233445566778899aabbccddeeff

4.3.5 AES-256

$$Nk = 8, Nr = 14$$
$$Plaintext = 00112233445566778899aabbccddeeff$$
$$Key = 000102030405060708090a0b0c0d0e0f$$
$$101112131415161718191a1b1c1d1e1f$$

4.3.5.1 Cipher (ENCRYPT)

round[0].input	00112233445566778899aabbccddeeff
round[0].k_sch	000102030405060708090a0b0c0d0e0f
round[1].start	00102030405060708090a0b0c0d0e0f0
round[1].s_box	63cab7040953d051cd60e0e7ba70e18c
round[1].s_row	6353e08c0960e104cd70b751bacad0e7
round[1].m_col	5f72641557f5bc92f7be3b291db9f91a
round[1].k_sch	101112131415161718191a1b1c1d1e1f
round[2].start	4f63760643e0aa85efa7213201a4e705
round[2].s_box	84fb386f1ae1ac97df5cfd237c49946b
round[2].s_row	84e1fd6b1a5c946fdf4938977cfbac23
round[2].m_col	bd2a395d2b6ac438d192443e615da195
round[2].k_sch	a573c29fa176c498a97fce93a572c09c
round[3].start	1859fbc28a1c00a078ed8aadc42f6109
round[3].s_box	adcb0f257e9c63e0bc557e951c15ef01
round[3].s_row	ad9c7e017e55ef25bc150fe01ccb6395
round[3].m_col	810dce0cc9db8172b3678c1e88a1b5bd
round[3].k_sch	1651a8cd0244beda1a5da4c10640bade
round[4].start	975c66c1cb9f3fa8a93a28df8ee10f63
round[4].s_box	884a33781fdb75c2d380349e19f876fb
round[4].s_row	88db34fb1f807678d3f833c2194a759e
round[4].m_col	b2822d81abe6fb275faf103a078c0033
round[4].k_sch	ae87dff00ff11b68a68ed5fb03fc1567
round[5].start	1c05f271a417e04ff921c5c104701554
round[5].s_box	9c6b89a349f0e18499fda678f2515920
round[5].s_row	9cf0a62049fd59a399518984f26be178
round[5].m_col	aeb65ba974e0f822d73f567bdb64c877
round[5].k_sch	6de1f1486fa54f9275f8eb5373b8518d
round[6].start	c357aae11b45b7b0a2c7bd28a8dc99fa
round[6].s_box	2e5bacf8af6ea9e73ac67a34c286ee2d
round[6].s_row	2e6e7a2dafc6eef83a86ace7c25ba934
round[6].m_col	b951c33c02e9bd29ae25cdb1efa08cc7
round[6].k_sch	c656827fc9a799176f294cec6cd5598b
round[7].start	7f074143cb4e243ec10c815d8375d54c
round[7].s_box	d2c5831a1f2f36b278fe0c4cec9d0329

round[7].s_row	d22f0c291ffe031a789d83b2ecc5364c
round[7].m_col	ebb19e1c3ee7c9e87d7535e9ed6b9144
round[7].k_sch	3de23a75524775e727bf9eb45407cf39
round[8].start	d653a4696ca0bc0f5acaab5db96c5e7d
round[8].s_box	f6ed49f950e06576be74624c565058ff
round[8].s_row	f6e062ff507458f9be50497656ed654c
round[8].m_col	5174c8669da98435a8b3e62ca974a5ea
round[8].k_sch	0bdc905fc27b0948ad5245a4c1871c2f
round[9].start	5aa858395fd28d7d05e1a38868f3b9c5
round[9].s_box	bec26a12cfb55dff6bf80ac4450d56a6
round[9].s_row	beb50aa6cff856126b0d6aff45c25dc4
round[9].m_col	0f77ee31d2ccadc05430a83f4ef96ac3
round[9].k_sch	45f5a66017b2d387300d4d33640a820a
round[10].start	4a824851c57e7e47643de50c2af3e8c9
round[10].s_box	d61352d1a6f3f3a04327d9fee50d9bdd
round[10].s_row	d6f3d9dda6279bd1430d52a0e513f3fe
round[10].m_col	bd86f0ea748fc4f4630f11c1e9331233
round[10].k_sch	7ccff71cbeb4fe5413e6bbf0d261a7df
round[11].start	c14907f6ca3b3aa070e9aa313b52b5ec
round[11].s_box	783bc54274e280e0511eacc7e200d5ce
round[11].s_row	78e2acce741ed5425100c5e0e23b80c7
round[11].m_col	af8690415d6e1dd387e5fbedd5c89013
round[11].k_sch	f01afafee7a82979d7a5644ab3afe640
round[12].start	5f9c6abfbac634aa50409fa766677653
round[12].s_box	cfde0208f4b418ac5309db5c338538ed
round[12].s_row	cfb4dbedf4093808538502ac33de185c
round[12].m_col	7427fae4d8a695269ce83d315be0392b
round[12].k_sch	2541fe719bf500258813bbd55a721c0a
round[13].start	516604954353950314fb86e401922521
round[13].s_box	d133f22a1aed2a7bfa0f44697c4f3ffd
round[13].s_row	d1ed44fd1a0f3f2afa4ff27b7c332a69
round[13].m_col	2c21a820306f154ab712c75eee0da04f
round[13].k_sch	4e5a6699a9f24fe07e572baacdf8cdea
round[14].start	627bceb9999d5aaac945ecf423f56da5
round[14].s_box	aa218b56ee5ebeacdd6ecebf26e63c06
round[14].s_row	aa5ece06ee6e3c56dde68bac2621bebf
round[14].k_sch	24fc79ccbf0979e9371ac23c6d68de36
round[14].output	8ea2b7ca516745bfeafc49904b496089

4.3.5.2 Inverse Cipher (DECRYPT)

round[0].iinput	8ea2b7ca516745bfeafc49904b496089

round[0].ik_sch	24fc79ccbf0979e9371ac23c6d68de36
round[1].istart	aa5ece06ee6e3c56dde68bac2621bebf
round[1].is_box	629deca599456db9c9f5ceaa237b5af4
round[1].is_row	627bceb9999d5aaac945ecf423f56da5
round[1].im_col	e51c9502a5c1950506a61024596b2b07
round[1].ik_sch	34f1d1ffbfceaa2ffce9e25f2558016e
round[2].istart	d1ed44fd1a0f3f2afa4ff27b7c332a69
round[2].is_box	5153862143fb2595149204030166 95e4
round[2].is_row	516604954353950314fb86e401922521
round[2].im_col	91a29306cc450d0226f4b5eaef5efed8
round[2].ik_sch	5e1648eb384c350a7571b746dc80e684
round[3].istart	cfb4dbedf4093808538502ac33de185c
round[3].is_box	5fc69f53ba4076bf50676aaa669c34a7
round[3].is_row	5f9c6abfbac634aa50409fa766677653
round[3].im_col	b041a94eff21ae9212278d903b8a63f6
round[3].ik_sch	c8a305808b3f7bd043274870d9b1e331
round[4].istart	78e2acce741ed5425100c5e0e23b80c7
round[4].is_box	c13baaeccae9b5f6705207a03b493a31
round[4].is_row	c14907f6ca3b3aa070e9aa313b52b5ec
round[4].im_col	638357cec07de6300e30d0ec4ce2a23c
round[4].ik_sch	b5708e13665a7de14d3d824ca9f151c2
round[5].istart	d6f3d9dda6279bd1430d52a0e513f3fe
round[5].is_box	4a7ee5c9c53de85164f348472a827e0c
round[5].is_row	4a824851c57e7e47643de50c2af3e8c9
round[5].im_col	ca6f71058c642842a315595fdf54f685
round[5].ik_sch	74da7ba3439c7e50c81833a09a96ab41
round[6].istart	beb50aa6cff856126b0d6aff45c25dc4
round[6].is_box	5ad2a3c55fe1b93905f3587d68a88d88
round[6].is_row	5aa858395fd28d7d05e1a38868f3b9c5
round[6].im_col	ca46f5ea835eab0b9537b6dbb221b6c2
round[6].ik_sch	3ca69715d32af3f22b67ffade4ccd38e
round[7].istart	f6e062ff507458f9be50497656ed654c
round[7].is_box	d6a0ab7d6cca5e695a6ca40fb953bc5d
round[7].is_row	d653a4696ca0bc0f5acaab5db96c5e7d
round[7].im_col	2a70c8da28b806e9f319ce42be4baead
round[7].ik_sch	f85fc4f3374605f38b844df0528e98e1
round[8].istart	d22f0c291ffe031a789d83b2ecc5364c
round[8].is_box	7f4e814ccb0cd543c175413e8307245d
round[8].is_row	7f074143cb4e243ec10c815d8375d54c
round[8].im_col	f0073ab7404a8a1fc2cba0b80df08517
round[8].ik_sch	de69409aef8c64e7f84d0c5fcfab2c23
round[9].istart	2e6e7a2dafc6eef83a86ace7c25ba934
round[9].is_box	c345bdfa1bc799e1a2dcaab0a857b728
round[9].is_row	c357aae11b45b7b0a2c7bd28a8dc99fa

round[9].im_col	3225fe3686e498a32593c1872b613469
round[9].ik_sch	aed55816cf19c100bcc24803d90ad511
round[10].istart	9cf0a62049fd59a399518984f26be178
round[10].is_box	1c17c554a4211571f970f24f0405e0c1
round[10].is_row	1c05f271a417e04ff921c5c104701554
round[10].im_col	9d1d5c462e655205c4395b7a2eac55e2
round[10].ik_sch	15c668bd31e5247d17c168b837e6207c
round[11].istart	88db34fb1f807678d3f833c2194a759e
round[11].is_box	979f2863cb3a0fc1a9e166a88e5c3fdf
round[11].is_row	975c66c1cb9f3fa8a93a28df8ee10f63
round[11].im_col	d24bfb0e1f997633cfce86e37903fe87
round[11].ik_sch	7fd7850f61cc991673db890365c89d12
round[12].istart	ad9c7e017e55ef25bc150fe01ccb6395
round[12].is_box	181c8a098aed61c2782ffba0c45900ad
round[12].is_row	1859fbc28a1c00a078ed8aadc42f6109
round[12].im_col	aec9bda23e7fd8aff96d74525cdce4e7
round[12].ik_sch	2a2840c924234cc026244cc5202748c4
round[13].istart	84e1fd6b1a5c946fdf4938977cfbac23
round[13].is_box	4fe0210543a7e706efa476850163aa32
round[13].is_row	4f63760643e0aa85efa7213201a4e705
round[13].im_col	794cf891177bfd1ddf67a744acd9c4f6
round[13].ik_sch	1a1f181d1e1b1c191217101516131411
round[14].istart	6353e08c0960e104cd70b751bacad0e7
round[14].is_box	0050a0f04090e03080d02070c01060b0
round[14].is_row	00102030405060708090a0b0c0d0e0f0
round[14].ik_sch	000102030405060708090a0b0c0d0e0f
round[14].ioutput	00112233445566778899aabbccddeeff

Chapter 5

Evaluation

5.1 My Thoughts

Overall I have been happy with the progress made on the project. Throughout the 6-7 months of work that has gone into the project I feel that my programming skills and knowledge have increased 2, if not 3 fold. I now understand a lot more about how a website core functions operate. My debugging skills have also increased massively. One aspect of this project meant that I had to learn how to use the *Wt* Library and I can say that I have a reasonable understanding of this and will definitely be using it in future projects.

One really positive aspect of this project is that I ended up teaching my self about the core concepts of Cryptography. I followed the online lecture series *Understanding Cryptography* which has taught me all about Symmetric/Asymmetric Cryptography as well as applications of these ideas in MACs and HASH functions.

I do feel however that I definitely *bit of more than I could chew*. In hindsight I definitely would not have been able to complete all the work that I set out to do at the beginning of the project. But the work that I have been able to complete has put me in a very good position to expand to the website as I have programmed it in a way that allows me to easily add new Sections and Questions.

5.2 Objective Analysis

I will now go through each of my objectives and decide on whether or not I have completed to a high degree of quality and success.

5.2.1 Platform Objective Analysis

Objective Number	Feelings about Objective Progress	Did I achieve this Objective
1a	I have definitely met this objective as I have a working website that can be launched from the my Computer	Yes
1b	I did not end up buying a domain for my website to be loaded to. This means that the website will not be public for anyone to use at the end of the project	No
1c	I did not do this as it turns out renting server space for my web server to live is fairly expensive	No
1d	I did not do this either for a similar reason, getting a SSL Certificate to enable HTTPS only connections is expensive. But if I were to put this into the public domain I would have to do this as in this day and age it is a necessity to have HTTPS enabled for your website	No
2a	A Navigation Grid that allows the user to access all the different sections of the website has been implemented	Yes
2b	A Template implementation for all the content I want to show to user has been implemented	Yes
2c	A Template implementation for all the questions that the user can answer to test their knowledge about all the different content sections has been implemented	Yes
2d	Due to time constraints I was not able to implement this. Given more time I definitely feel this would be achievable	No
2e	Without the online code-editor it would not be possible for me to achieve this Objective	No
2f	I was not able to achieve this objective for the same reason I was not able to achieve Objective *2e*	No
3a	I have fully implemented The Login system for the Website	Yes

3b	Authentication for all user is implemented allowing them to verify their accounts with provided email addresses.	Yes
3c	I have fully implemented this objective as every user has a record created in the db_user table which contains relevant information about the user	Yes
3d	Email Verification has been implemented So I have fully achieved this objective	Yes
3e	All answers a user gives are stored in the user_answered_questions Table so I have achieved this objective	yes
3f	The profile page shows the user what questions they have got correct and what questions they have got wrong.	Yes

5.2.2 AES Implementation Objectives

Objective Number	Feelings about Objective Progress	Did I achieve this Objective
1a	I have implemented the AES Key Schedule	Yes
1b	I have implemented the Sub Bytes Routine for AES	Yes
1c	I have implemented the Shift Rows Routine for AES	Yes
1d	I have implemented the Mix Columns Routine for AES	Yes
1e	I have implemented the Add Round Key Routine for AES	Yes
2a	128-bit Keys are supported	Yes
2b	192-bit Keys are supported	Yes
2c	256-bit Keys are supported	Yes
3a	Electronic Code Book Mode has been implemented	Yes

Due to the time constraints I was only able to implement 1 mode of operation, ECB, for the algorithm. Given more time I definitely feel that I would be able to implement the remaining modes of operation for the algorithm. These include CBC, OFB, CFB, etc.

5.3 How I could Extend my Project

Now that I am at the end of the project I have come up with various ways that my project could be extended. I think the first obvious extension would be to streamline the process of answering questions. Right now in order to answer a question you need to create an account, go the the content page, navigate to the bottom of the page where the questions are located and answer them. I feel that if split the site up into different sections that only contained learning content and sections that contained only questions then it would be way easier for a user to just test their knowledge. This would also give me the opportunity to add general topic quizzes. These would contain questions on a broad number of topics about Cryptography.

I would also like to increase the amount of information available to the user when they go to their profile. Right now the user can only see the questions that they have answered and the questions they have answered correctly. I would like to extend this so that for the questions the user got wrong it clearly explains how to get to the correct answer and how to do it more efficiently. Adding a more formal reporting system would also be a good idea. This system would take in all the information about the answers for the user questions and then would correlate and analyze all that information. It would inform the user about what areas they are good at and what areas they need to work on. It would also give specifics on certain areas so that the user can really benefit and learn exactly what they need to learn. Rather than spending time on a topic that they already understand.

Anther extension would be to incorporate more algorithms into the project. Due to the time constraints I was only able to implement AES but I would also like to implement more algorithms. Examples of some algorithms I could implement are

- DES - Data Encryption Standard
- Twofish
- Serpent
- RC6
- SHA-2
- SHA-3
- bcrypt

5.4 Final Thoughts

I definitely feel that this project has been worth my time. I have gained lots of experience and now have various skill sets that I know I will be very useful

for my career. This entire process has been very fun and I have enjoyed every aspect of it. I look forward to any challenge I face in the future that involves any of the information/skills that I have learned throughout this project.

List of Figures

List of Tables