



جامعة قرطبة الخاصة
CORDOBA PRIVATE UNIVERSITY

جامعة قرطبة الخاصة
كلية الهندسة و
الเทคโนโลยية قسم نظم
الحاسوب

مشروع تخرج
تشفيير البيانات بـ AES

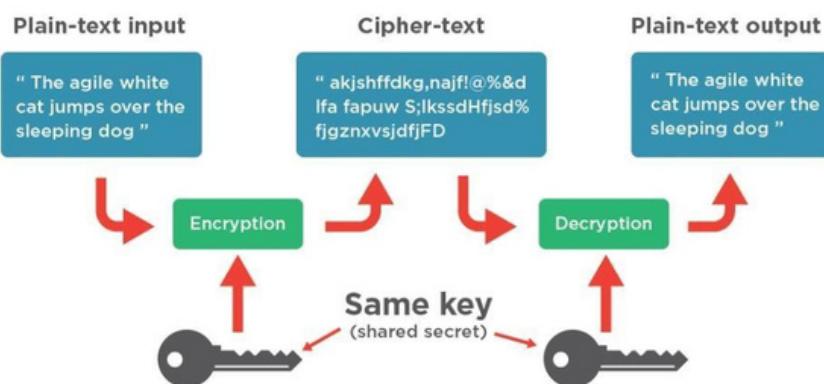
بإشراف الدكتورة
موسى جانات

إعداد الطالب
نورالدين حامد
ابراهيم

2023-2024

جامعة قرطبة الخاصة
كلية الهندسة والتكنولوجيا
قسم نظم الحاسوب

مشروع تخرج
تشفيير البيانات بـ AES

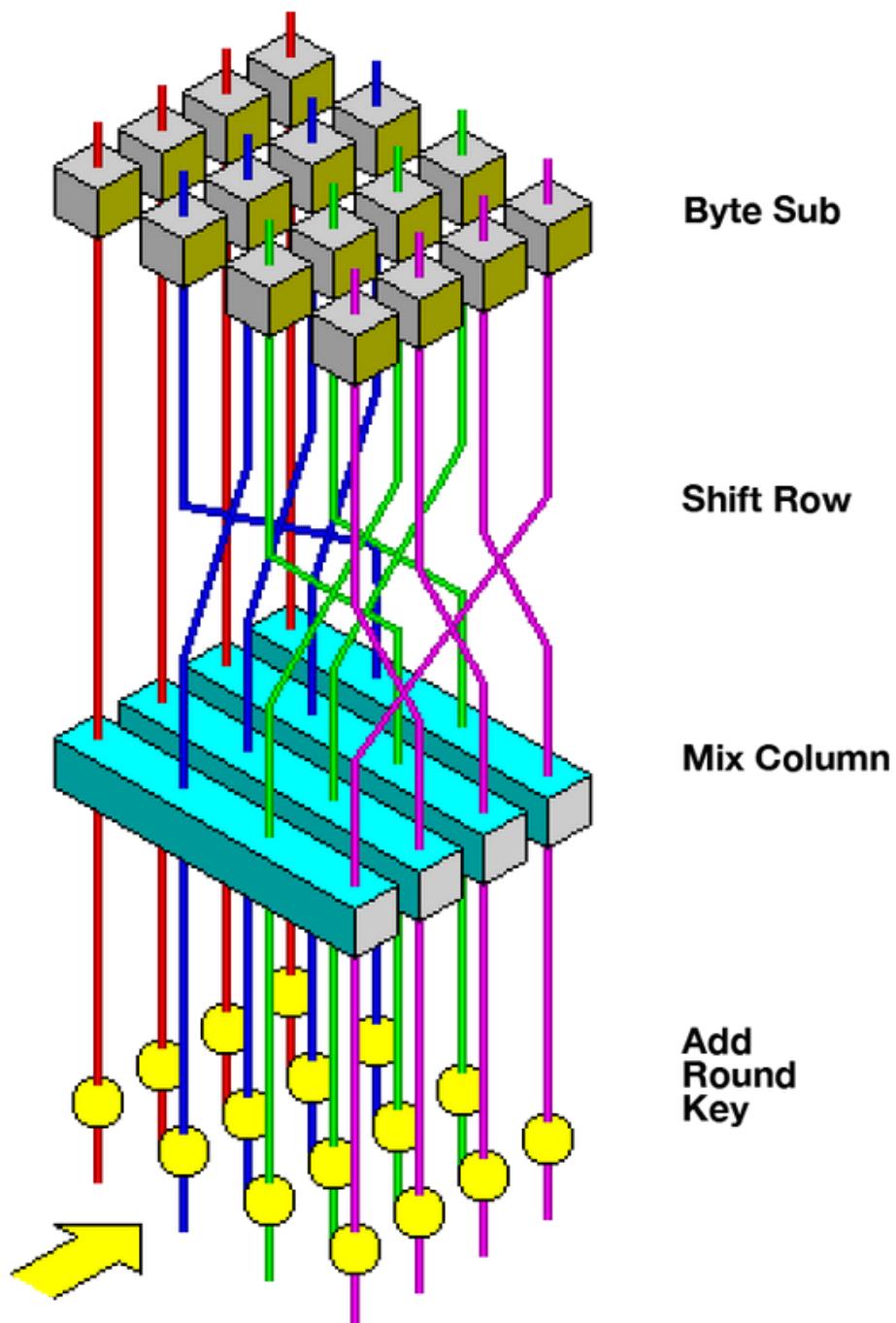


بإشراف الدكتور موسى
جانات

إعداد الطالب: نور الدين حامد ابراهيم

2023-2024

تشفيير البيانات باستخدام خوارزمية AES



الفهرس

7	مقدمة
8	اهمية المشروع
9	هدف المشروع
10	الفصل الاول نظرة عامة عن AES
11-12	المفهوم الاساسي ل AES
13	أنواع التشفير
14-17	كيف يعمل AES
18	مبدأ خوارزمية AES
19	تدفق خوارزمية تشفير AES
20	موثوقية وسلامة خوارزميات AES
21	ملخص الفصل الاول
22	الفصل الثاني-إنشاء مكتبة برمجية لتشفيـر النصوص و الملفات باستخدام C#
32	انشاء كلاس باسم AESAlgorithm
24-26	دالة لتشفيـر وفك تشفيـر مصفوفة من البايتات
26-28	دالة لتشفيـر و لفك تشفيـر النص
28-31	دالة لتشفيـر و فك تشفيـر الملفات
31	ملخص الفصل الثاني
32	الفصل الثالث انشاء تطبيق WinForm
33-42	تصميم الواجهات
43	برمجة حدث الضغط على زر تشفيـر النص
44	برمجة حدث الضغط على زر فك تشفيـر النص
45	برمجة حدث الضغط على زر اختيار مسار
46	برمجة حدث الضغط على زر تشفيـر الملف
47	برمجة حدث الضغط على زر فك تشفيـر الملف
48-51	كود البرنامج كامل

مقدمة

استخدم الإنسان التشفير منذ نحو ألفي عام قبل الميلاد لحماية رسائله السرية، وبلغ هذا الاستخدام ذروته في فترات الحروب؛ خوفاً من وقوع الرسائل الحساسة في أيدي العدو. كان يوليوس قيصر من أشهر القدماء الذين امتهنوا الكتابة المعمقة حيث قام بتطوير خوارزمية تعمية مشفرة سميت باسمه شيفرة قيصر لتأمين اتصالاته وراسلاته مع قادة جيوشة. وظهرت فيما بعد العديد من الآلات التي تقوم بعمليات التشفير، وشكل الكمبيوتر في بدايات ظهوره وسيلةً جديّة للاتصالات الآمنة، وفك تشفير رسائل العدو. واحتكرت الحكومات في فترة السبعينيات حق التشفير وفك التشفير. وفي أواخر السبعينيات، أسست شركة آي بي إم مجموعة تختص بأبحاث التشفير، ونجحت هذه المجموعة في تطوير نظام تشفير أطلق عليه اسم لوسيفَر وكان هذا النظام مثراً للجدل، ورغم تحفظات الحكومة الأمريكية عليه للاعتقاد بها بعدم حاجة الشركات والمؤسسات الخاصة إلى أنظمة التشفير، إلا أنه قد حقق انتشاراً واسعاً في الأسواق. ومنذ ذلك الحين، أخذت العديد من الشركات تقوم بتطوير أنظمة تشفير جديدة، مما أبرز الحاجة إلى وجود معيار لعمليات التشفير ومن أبرز المؤسسات التي أسهمت في هذا المجال، المعهد الوطني للمعايير والتكنولوجيا National Institute of Standards and Technology- NIST الوطني الأمريكي للمعايير U.S. National Bureau of Standards إذ طور هذا المعهد عام 1973 معياراً أطلق عليه معيار تشفير البيانات Data Encryption Standard- DES ويُسْتَنِدُ هذا المعيار إلى خوارزمية Lucifer algorithm التي تستخدِّم مفتاح تشفير بطول 56 بت(bit)، وتشترط أن يكون لكل من المرسل والمُستقبل المفتاح السري ذاته. وقد استخدمت الحكومة هذا المعيار الرسمي عام 1976، واعتمدته البنوك لتشغيل آلات الصراف الآلي.(ATM)

أهمية المشروع

إن منافع الكتابة المشفرة معترف بها بشكل جيد، فالتشفير يمكن أن يحمي الاتصالات وхран المعلومات من الوصول والكشف الغير مخول، كما يحمي المعلومات من التزييف والاختراق، فالتشفير أداة أمن معلومات ضرورية ويجب أن تكون متوفرة بسهولة للمستعملين، ضد الفوضويين السريين، التشفير يحل معظم مشاكل الأمان ويواجه تهديدات المخترقين على سبيل المثال التشفير أن يوقف لصوص الحاسب ، ولكن نظام التشفير يمكن أن يخترق عن طريق برامج تقوم بتخريب عملية التشفير، لذلك نحتاج إلى إدارة وترتيب ،تصميم جيد،سيطرة على الدخول، وتأمين برامج حماية من خلال استخدام العديد من خوارزميات التشفير التي سنقوم باستعراضها.

-الحفاظ على أمن المعلومات وتأمين نقلها بسرية تامة من طرف إلى آخر دون تسرب أي من المعلومات عند تبادلها

-الحفاظ على سرية الملف من قبل مستثمر عادي من خلال القيام بتشغير الملفات وفك تشفيرها عند حاجة المستثمر

هدف المشروع

لقد اصبح أمن المعلومات في غاية الاهمية لالسيما بعد دخول الحاسوب و استخدام الشبكات و وسائل الاتصالات لنقل المعطيات و قد استخدمت الاليات متعددة لتقديم الخدمات الالامنية و لكن الالاساس لمعظم هذه الالاليات هو تقنية التعميمه اننا نعيش اليوم عصر الانترنت و التجارة الالكترونية التي تعتمد بشكل اساسي على امن شبكات المعلومات و التعميمه و التواقيع الرقمية حيث ظهرت تطورات هائلة في تكنولوجيا امن اتصالات الحاسوب ولاسيما خوارزميات التعميمه حيث تعتبر خوارزمية AES من اشهر و اقوى الخوارزميات في مجال التعميمه حاليا لذلك سيكون اهداف المشروع:

المفهوم الالاسي و مبدأ عمل خوارزمية AES-1

2- انشاء مكتبة DLL برمجية تستخدم خوارزمية AES لتشفيير النصوص و الملفات باستخدام لغة C#

استخدام مكتبة WinForm 3- DLL في تطبيق

الفصل الاول

المفهوم الاساسي و مبدأ عمل خوارزمية AES

نظرة عامة

تنقسم خوارزمية التشفير إلى تشفير أحادي الاتجاه وتشفيير ثنائي الاتجاه.

تشفيير أحادي الاتجاه تضمن SHA, MD5 لا يمكن التراجع عن خوارزمية التشفير أحادية الاتجاه ، أي أنه لا يمكن استعادة البيانات المشفرة إلى البيانات الأصلية ما لم يتم اعتماد هجوم التصادم والأساليب الشاملة. مثل تخزين كلمات مرور الحساب المصرفي ، يتم اعتماد طريقة التشفير أحادية الاتجاه بشكل عام.

تشفيير ثنائي الاتجاه إنه قابل للانعكاس ، وهناك مفتاح للنص المشفر ، ويمكن للطرف الذي يحمل النص المشفر فك تشفير النص العادي الأصلي وفقاً للمفتاح ، والذي يُستخدم بشكل عام عندما يمكن كل من المرسل والمستقبل من الحصول على النص العادي من خلال المفتاح. يتضمن التشفير ثنائي الاتجاه التشفير المتماثل والتشفير غير المتماثل. يشمل التشفير المتماثل DES, AES... الخ يشمل التشفير غير المتماثل RSA, ECC... الخ

المفهوم الالاساسي ل AES

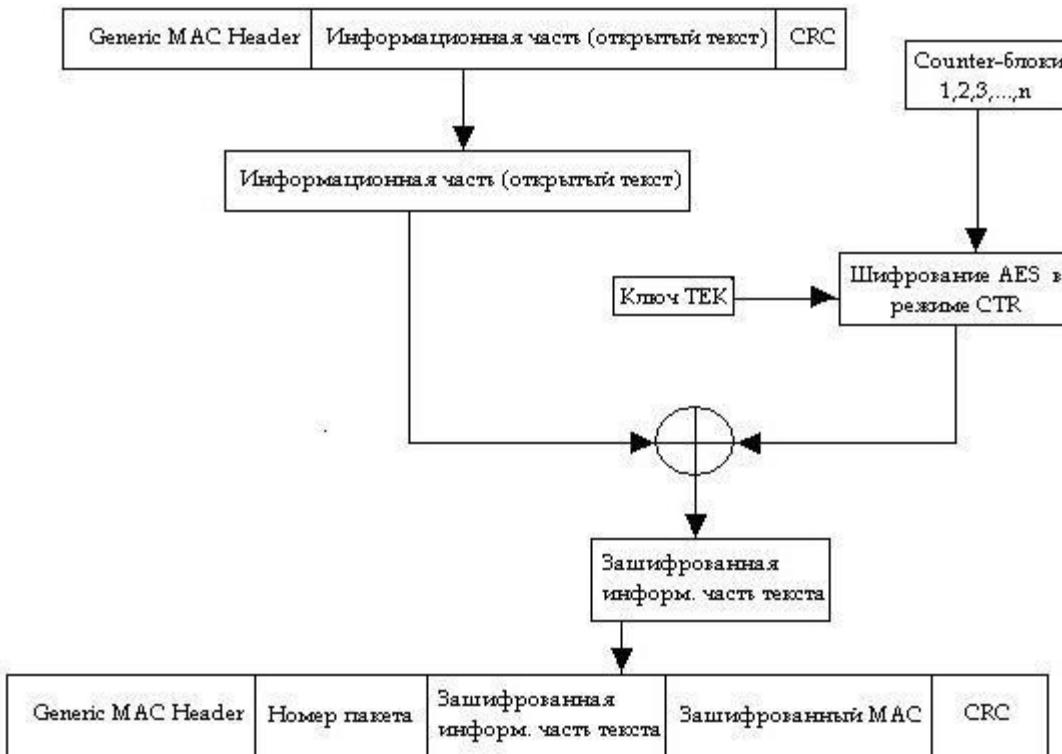
ما هو التشفير AES

بادئ ذي بدء ، فإن النظام نفسه يمثل مجموعة من الخوارزميات التي تجعل من الممكن إخفاء الشكل الأولي لبعض البيانات التي يتم إرسالها أو استلامها من قبل المستخدم أو تخزينها على جهاز الكمبيوتر. في معظم الأحيان يتم استخدامه في تقنيات الإنترنت ، عندما يكون مطلوباً⁼لضمان السرية الكاملة للمعلومات ، ويشير إلى ما يسمى خوارزميات التشفير المتماثلة.

يتم استخدام نوع التشفير AES لتحويل المعلومات إلى طريقة عرض محمية وعكس فك تشفير المفتاح نفسه الذي يعرفه كل من الأطراف المرسلة والمستلمة ، على النقيض من التشفير المتماثل ، والذي يتضمن استخدام مفاتيحين - مغلق ومفتوح. وبالتالي ، من السهل الاستنتاج أنه إذا عرف الطرفان المفتاح الصحيح ، فإن عملية التشفير وفك التشفير عملية بسيطة للغاية.

قليلاً من التاريخ:

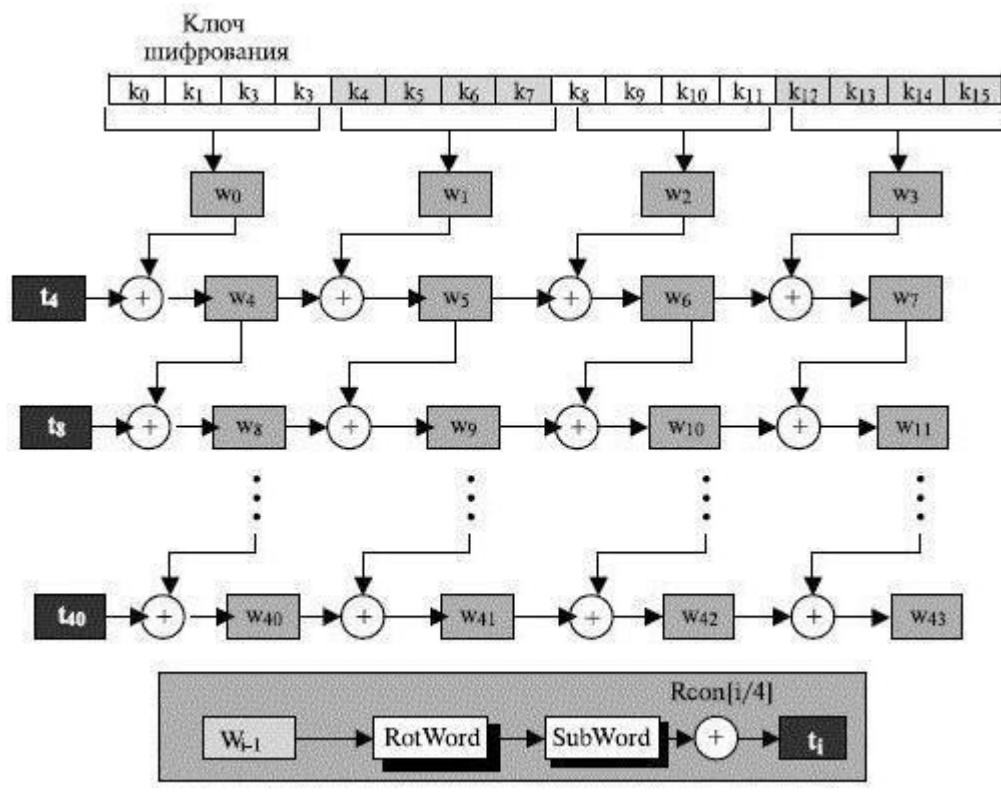
لأول مرة ، تم ذكر التشفير AES مرة أخرى في عام 2000 ، عندما أصبحت خوارزمية Rijndael الفائز في مسابقة الاختيار لخليفة DES ، والتي كانت المعيار في الولايات المتحدة منذ عام 1977.



في عام 2001 ، تم اعتماد نظام AES رسمياً كمعيار فدرالي جديد لتشифير البيانات ومنذ ذلك الحين تم استخدامه عالمياً.

أنواع التشفير AES

وتضمن تطور الخوارزميات عدة المراحل المتوسطة ، والتي ارتبطت بشكل رئيسي بزيادة طول المفتاح. اليوم ، هناك ثلاثة أنواع رئيسية: AES-256 وAES-192 وAES-128-encryption

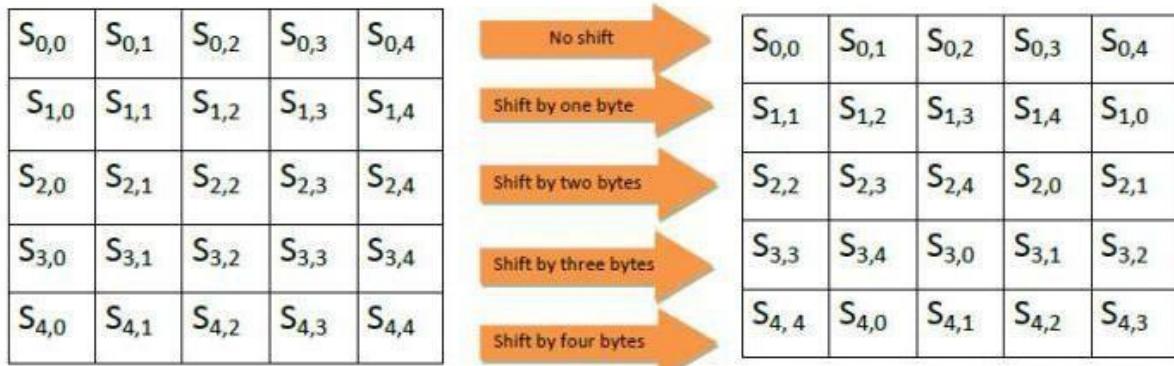


الاسم يتحدث عن نفسه. يقابل التعيين العددي طول المفتاح المستخدم ، معتبراً عنه بذات. بالإضافة إلى ذلك ، يشير تشفير AES إلى نوع فدرة يعمل مباشرةً مع كتل من المعلومات ذات طول ثابت ، يشفّر كل منها ، بخلاف خوارزميات البث التي تعمل مع رموز مفردة لرسالة مفتوحة ، وترجمتها إلى نموذج مشفر. في AES ، يكون طول الكتلة 128 بت.

في المصطلحات العلمية ، نفس الخوارزميات ، التي تستخدم تشفير AES-256 ، تعني عمليات تعتمد على تمثيل متعدد الحدود للعمليات والرموز عند معالجة المصفوفات ثنائية الأبعاد (المصفوفات).

كيف يعمل؟

الخوارزمية معقدة بعض الشيء، يتضمن استخدام العديد من العناصر الأساسية. في البداية ، يتم استخدام مصفوفة ثنائية الأبعاد ، ودورات تحويل (جولات) ، ومفتاح مستدير وجداول بدائل أولية وعكسية.



ت تكون عملية تشفير البيانات من عدة مراحل:

-حساب جميع مفاتيح الدائرية

-استبدال وحدات البايت باستخدام جدول S-Box الرئيسي

-التحول في النموذج باستخدام قيم مختلفة (انظر الشكل أعلاه)

-خلط البيانات داخل كل عمود من المصفوفة (النموذج)

-اضف شكلًا ومتغيرًا مفتاحًا مستديرة.

تتم عملية فك التشفير بالترتيب العكسي ، ولكن بدلًا من جدول S-Box ، يتم تطبيق جدول الإعداد العكسي ، والذي تم ذكره أعلاه.

Key Size	Possible combinations
1-bit	2
2-bit	4
4-bit	16
8-bit	256
16-bit	65536
32-bit	4.2×10^9
56-bit (DES)	7.2×10^{16}
64-bit	1.8×10^{19}
128-bit (AES)	3.4×10^{38}
192-bit (AES)	6.2×10^{57}
256-bit (AES)	1.1×10^{77}

ذا قدّمت مثالاً، إذا كان طول المفتاح 4 لا يتطلب الأمر سوى 16 مرحلة (جولات) ، أي تحتاج إلى التحقق من جميع التوليفات الممكنة ، بدءاً من 0000 وتنتهي بـ 1111. وبطبيعة الحال ، يتم اختراق هذه الحماية بسرعة كافية. ولكن إذا أخذنا المزيد من المفاتيح ، فمن أجل 65 بت ، يلزم توفير 65566 خطوة ، ول 256 بت - 1.1×10^{77} . وكما ذكر المتخصصين الامريكيين ، على حسن الاختيار من مجموعة (مفتاح) سوف يستغرق حوالي 1490000000000000 سنتوات.

لتعلم خوارزمية AES ، يجب أن نفهم أولًا ثلاثة مفاهيم أساسية:

[1- المفتاح](#)

[2-الحشو](#)

[3-الوضع](#)

1-المفتاح

المفتاح AES تدرك الخوارزمية أساسيات التشفير وفك التشفير . خوارزميات التشفير المتماثل متناظرة لأنها تتطلب نفس المفتاح لتشفي وفك تشفير النص العادي.

يدعم AES ثلاثة اطوال رئيسية: 128 بت, 192 بت, 256 بت

عادة ما يستخدم المصطلحات التالية في وصف الخوارزمية AES128 ، AES192 ، AES256 حيث يشير في الواقع إلى استخدام خوارزمية AES لمفاتيح ذات أطوال مختلفة. من منظور الأمان ، فإن AES256 لديها أعلى أمان. من وجهة نظر الأداء ، تتمتع AES128 بأعلى أداء. السبب الأساسي هو أن لديهم جولات معالجة تشفير مختلفة.

2-الحشو

لفهم مفهوم الحشو ، يجب أن نفهم أولًا ميزة تشفير كتلة AES عند تشفير النص العادي ، لا تقوم خوارزمية AES بتشفيـر النص العادي بالكامل إلى نص مشفر بالكامل ، ولكنها تقـسم النص العادي إلى كتل نص عادي مستقلة ، يبلغ طول كل منها 128 بت.

تم معالجة كتل النص المشفر هذه بواسطة مشفر AES لتوليد كتل نص مشفر فردية. يتم تجميع كتل نص التشفير هذه معاً لتشكيل نتيجة التشفير AES النهائية.

ولكن هنا تتطوـي على مشكلة:

إذا كان طول قطعة من النص العادي 192 بت ، إذا تم تقسيمه وفقاً لكتلة نص عادي كل 128 بت ، فإن كتلة النص العادي هي 64 بت فقط ، أقل من 128 بت. ماذا تفعل في هذا الوقت؟ تحتاج إلى حشو كتل النص العادي. تحتوي AES على العديد من خوارزميات الحشو المختلفة في تطبيقات اللغات المختلفة ، وسنقدم فقط قائمة بالحشو النموذجي لتقديمه.

NoPadding : لا يلزم الحشو ، ولكن يجب أن يكون النص العادي عددًا صحيحًا مضاعفًا لـ 16 بايت.

: PKCS5Padding

إذا كانت كتلة النص العادي أقل من 16 بايت (128 بت) ، فسيتم إضافة العدد المقابل من الأحرف في نهاية كتلة النص العادي ، وتساوي قيمة كل بايت عدد الأحرف المفقودة.

: ISO10126Padding

إذا كانت كتلة النص العادي أقل من 16 بايت (128 بت) ، تتم إضافة عدد وحدات البايت المقابلة في نهاية كتلة النص العادي ، فإن قيمة الحرف الألآخر تساوي عدد الأحرف المفقودة ، ويتم ملء الأحرف الأخرى بأرقام عشوائية.

3-الوضع

عكس وضع عمل AES في عملية تشفير كتلة نص عادي إلى كتلة نص مشفر. توفر خوارزمية تشفير AES خمسة أوضاع عمل مختلفة:

ECB、CBC、CTR、CFB、OFB

أفكار السمة بين الأوضاع تقريبية ، وهناك بعض الاختلافات في معالجة التفاصيل. في هذا العدد ، نقدم فقط التعريفات الأساسية لكل نموذج -

وضع ECB افتراضي:

وضع دفتر الكود الالكتروني

-وضع CBC

سلسلة CipherBlock - وضع

نسبة النقر إلى الظهور:

عاده وضع الحاسبة

: CFB وضع

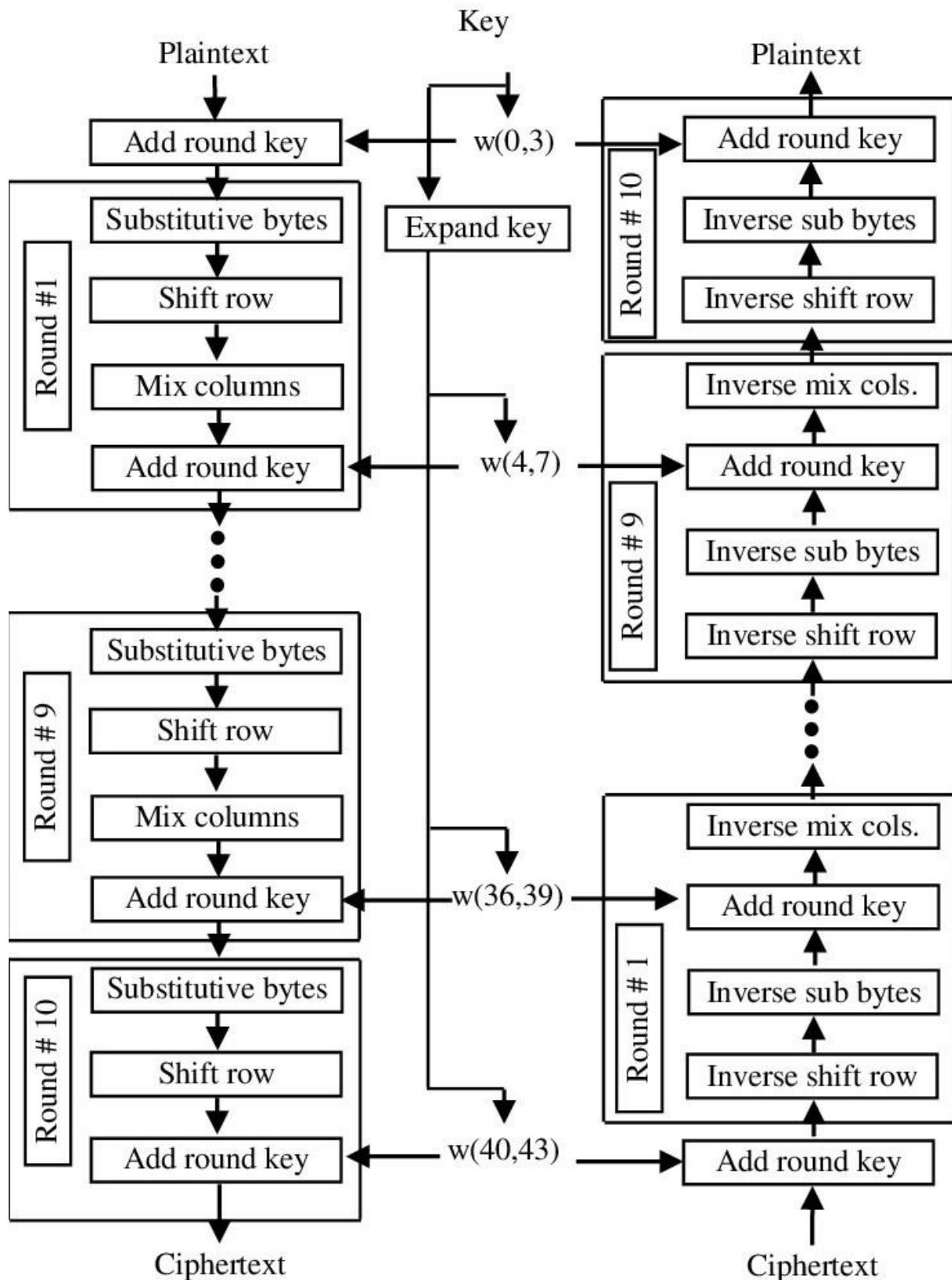
CipherFeedBack

: OFB-

وضع تعليقات الالخراج OutputFeedBack

إذا تم استخدام وضع عمل معين لتشفي AES ، فيجب استخدام نفس وضع العمل لفك التشفير.

مبدأ خوارزمية AES



تدفق خوارزمية تشفير AES

يتضمن تشفير AES بشكل أساسي خطوتين: تمديد مفتاح مع تشفير نص عادي

توسيع المفتاح: يتم توسيع المفاتيح الالادخال 16 بت، 24 بت، 32 بت ويختلف عدد جولات التشفير بعد الحصول على المفاتيح الموسعة وفقاً لطول المفتاح

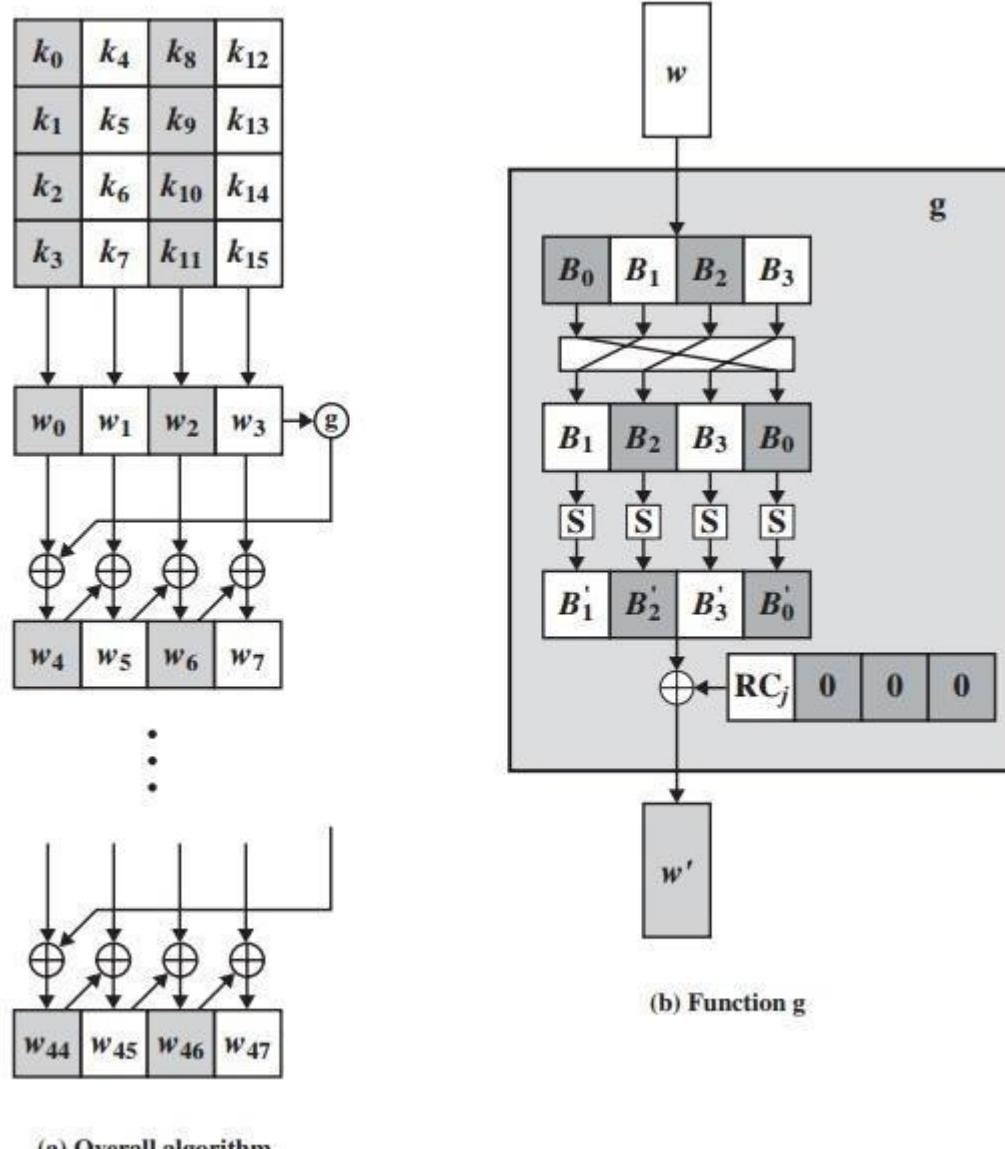


Figure 5.9 AES Key Expansion

موثوقية وسلامة خوارزميات AES

على الرغم من التصريحات الصادقة من المتخصصين ، خوارزميات AES لا تزال من الناحية النظرية ضعيفة ، لأن طبيعة التشفير لها وصف جبري بسيط. هذا ما لاحظه نيلز فيرغسون. وفي عام 2002 ، نشر جوزيف بيتشيك ونيكولاوس كورتوا مقالا يبرر الهجوم المحتمل ل XSL. صحيح أنها سببت الكثير من الجدل في العالم العلمي ، واعتبر البعض أن حساباتهم خطأ.

في عام 2005 ، اقترح أنيمك أن يستخدم الهجوم قنوات تابعة لجهات خارجية ، وليس مجرد حسابات رياضية. في هذه الحالة ، قام أحد الهجمات بحساب المفتاح بعد 800 عملية ، والآخر استلمها بعد 2 .32 العمليات (في الجولة الثامنة).

بدون شك ، لهذا اليوم النظام ويمكن اعتبارها واحدة من أكثرها قدما ، إن لم يكن واحدة ولكن. قبل بضع سنوات ، على شبكة الإنترنت ، وقعت موجة من هجمات الفيروسات ، حيث قام جهاز تشفير الفيروسات (وفي الوقت نفسه أيضا الابتزاز) باختراق أجهزة الكمبيوتر ، مشفراً البيانات بالكامل ، مطالباً بمبلغ نقدى من النقود لفك رموزه. في هذه الرسالة ، لوحظ أن التشفير تم باستخدام الخوارزمية AES1024 ، والتي ، كما كان يعتقد حتى وقت قريب ، لا توجد في الطبيعة.

لذلك أمر لا ، ولكن حتى الأكثر شهرة كان مطورو برامج مكافحة الفيروسات ، بما في ذلك برنامج Kaspersky Lab ، عاجزين عن فك تشفير البيانات. أدرك العديد من الخبراء أن فيروس I Love You I سيء السمعة ، والذي ضرب في وقت واحد الملايين من أجهزة الكمبيوتر في جميع أنحاء العالم ودمر معلومات مهمة عنهم ، مقارنة مع هذا التهديد ، تبين أنه طفولية صبيانية. بالإضافة إلى ذلك ، كان I Love You I تركيزاً على ملفات الوسائط المتعددة ، ولم يتمكن الفيروس الجديد من الوصول إلى المعلومات السرية الخاصة بالشركات الكبرى. ومع ذلك ، للإشارة إلى كل الأدلة التي تشير إلى استخدام تشفير AES-1024 هنا ، لا أحد يأخذها.

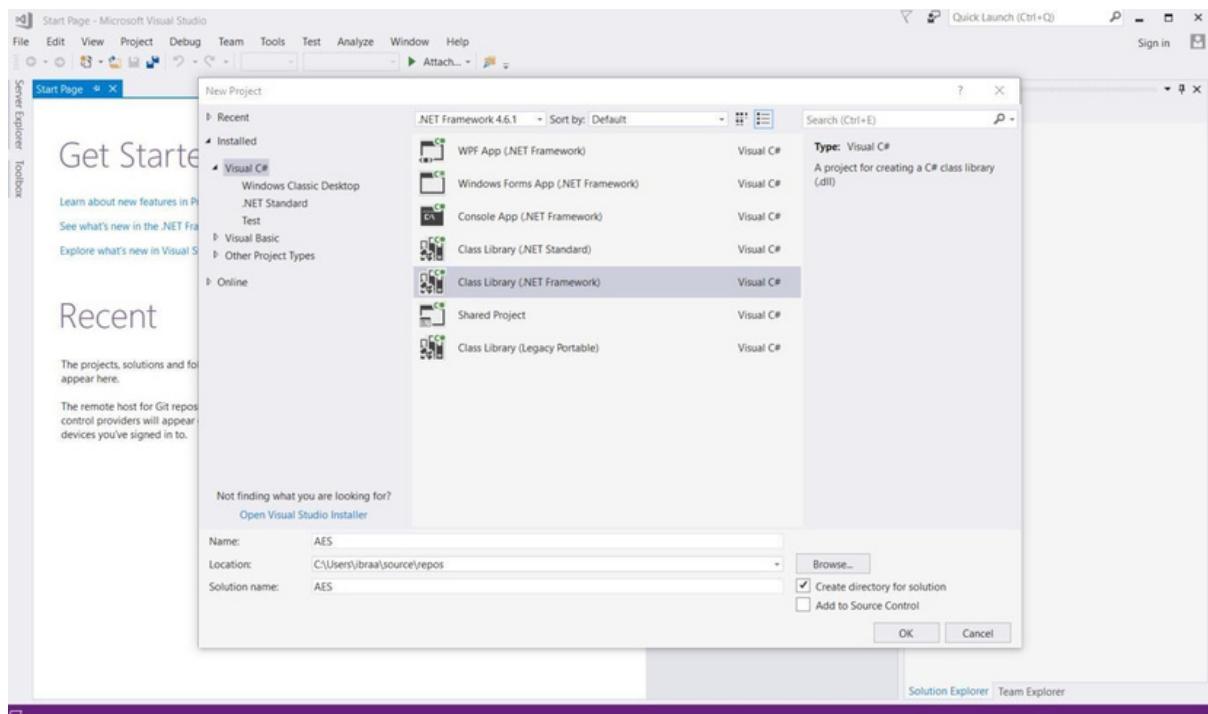
ملخص الفصل الاول:

مكنك ذلک القول بأن AES التشفير هو إلى حد بعيد الألأکثر تقدما وحمايتها، بغض النظر عن طول المفتاح الذي يتم استخدامه. ليس من المستغرب، أن يستخدم هذا المعيار في معظم نظم الترميز، ولها منظور واسع بما فيه الكفاية على تطوير وتحسين في المستقبل المنظور، خصوصاً أن المرجح جداً أن تكون والجمع بين عدة أنواع التشفير في وحدة واحدة (على سبيل المثال، الاستخدام المتزامن للالالمتماثلة وغير المتماثلة أو كتلة وتيار التشفير).

الفصل الثاني

إنشاء مكتبة DLL برمجية لتشفيير النصوص و الملفات باستخدام لغة C#

-نقوم بإنشاء مكتبة DLL بأسم AES



-نقوم بانشاء كلاس باسم AESAlgorithm ونقوم باضافة المكتبات التالية

```
using System;
using System.IO;
using System.Linq;
using System.Security.Cryptography;
using System.Text;
using System.Threading.Tasks;

namespace AES
{
    /// <summary>
    /// تشفير و فك تشفير البيانات
    /// </summary>
    public class AESAlgorithm
```

داخل الكلاس سنقوم بانشاء:

1- دالة لتشفي و دالة لفك تشفير مصفوفة من البايتات

داو لتشفي و دالة لفك تشفير نص-2 (String)

3- دالة لتشفي و دالة لفك تشفير الملفات

اضافة الى ذلك سيكون هناك دوال جانبية تستخدم ضمن دوال التشفير و فك التشفير.

في البداية سنقوم بانشائ دالة لتشفي و دالة لفك تشفير مصفوفة من البايتات

حيث سيعتمد دوال النص على دوال البايتات للتشفي و فك التشفير.

1- دالة لتشفي مصفوفة من البايتات:

حيث تستقبل المصفوفة التي نريد تشفيرها و مفتاح التشفير التي تستخدم لتشفي البيانات.

```

/// <summary>
// دالة لتفصير مصفوفة من البيانات
/// </summary>
/// <param name="bytesToEncrypt">المحفظة التي يجب تشفيرها</param>
/// <param name="key">مفتاح التشفير</param>
/// <returns></returns>
0 references
public async Task<byte[]> EncryptByteArrayAsync(byte[] bytesToEncrypt ,string key)
{
    // ملح لحماية كلمة المرور
    // ملاحظة: على الأقل يجب أن يكون 8 بایتات
    // كلمة سر حجم البيانات زاد الحماية ولكن سيحتاج الى معالجة اكبر
    byte[] saltBytes = new byte[8] { 1, 2, 3, 4, 5, 6, 7, 8 };

    using (MemoryStream ms = new MemoryStream())
    {
        using (RijndaelManaged AES = new RijndaelManaged())
        {

            AES.KeySize = 256;
            AES.BlockSize = 128;

            var Key = new Rfc2898DeriveBytes(key, saltBytes, 1000);
            AES.Key = Key.GetBytes(AES.KeySize / 8);
            AES.IV = Key.GetBytes(AES.BlockSize / 8);

            AES.Mode = CipherMode.CBC;

            byte[] by = new byte[1000000];

            using (var cs = new CryptoStream(ms, AES.CreateEncryptor(), CryptoStreamMode.Write))
            {

                await cs.WriteAsync(bytesToEncrypt, 0, bytesToEncrypt.Length);
                cs.Close();
            }

            return ms.ToArray();
        }
    }
}

```

سنقوم بإنشاء جميع الدوال بشكل غير متزامن Async

```

/// <summary>
// دالة لتفصير مصفوفة من البيانات
/// </summary>
/// <param name="bytesToEncrypt">المحفظة التي يجب تشفيرها</param>
/// <param name="key">مفتاح التشفير</param>
/// <returns></returns>
0 references
public async Task<byte[]> EncryptByteArrayAsync(byte[] bytesToEncrypt ,string key)
{

```

الفائدة من ذلك عند استخدام هذه المكتبة البرمجية في تطبيقات عملية(WinForm) لن يؤدي الى تجميد البرنامج و سيكون البرنامج اكثر استجابة

1- دالة لفك تشفير مصفوفة من البيانات:

ايضاً تستقبل متغيرين المصفوفة التي نريد تشفيرها و مفتاح التشفير التي تستخدم لفك تشفير البيانات.

ملاحظة: يجب ان يكون مفتاح فك التشفير نفس مفتاح التشفير لالاننا نستخدم خوارزمية AES وهي خوارزمية متماثلة حيث تستخدم نفس المفتاح في التشفير و فك التشفير.

```
/// <summary>
/// فك تشفير مصفوفة من البيانات
/// </summary>
/// <param name="bytesToDecrypt">المصفوفة التي سنقوم بفك تشفيرها</param>
/// <param name="keyDecrypt">مفتاح فك التشفير</param>
/// <returns></returns>
0 references
public async Task<byte[]> DecryptByteArrayAsync(byte[] bytesToDecrypt, string keyDecrypt)
{
    :
    ملح لحماية كلمة المرور // ملاحظة: يجب ان يكون نفس المفتاح لفك التشفير البيانات
    byte[] saltBytes = new byte[8] { 1, 2, 3, 4, 5, 6, 7, 8 };

    using (MemoryStream ms = new MemoryStream())
    {
        using (RijndaelManaged AES = new RijndaelManaged())
        {
            AES.KeySize = 256;
            AES.BlockSize = 128;

            var key = new Rfc2898DeriveBytes(Encoding.UTF8.GetBytes(keyDecrypt), saltBytes, 1000);
            AES.Key = key.GetBytes(AES.KeySize / 8);
            AES.IV = key.GetBytes(AES.BlockSize / 8);

            AES.Mode = CipherMode.CBC;

            using (var cs = new CryptoStream(ms, AES.CreateDecryptor(), CryptoStreamMode.Write))
            {
                await cs.WriteAsync(bytesToDecrypt, 0, bytesToDecrypt.Length);
                cs.Close();
            }
        }

        return ms.ToArray();
    }
}
```

ـ دالة لـ تـ شـ فـ يـرـ نـ صـ (String)

في هذه الدالة سنقوم بتحويل النص الى مصفوفة من البايتات باستخدام Encoding.UTF8.GetBytes وسنرسلها لدالة تشفير البايتات الا لكن بما ان نص التشفير سيعطى نفس النتيجة دوما لنفس النص سنقوم بتوليد حرفين بشكل عشوائي و نضيفه للنص المراد تشفيره لنحصل على نتيجة مختلفة

ملاحظة: عند فك تشفير النص يجب حذف اول حرفين

```
private static readonly Random random = new Random();
/// <summary>
/// توليد حرفين عشوائيين و ارجاعهم
/// </summary>
/// <returns></returns>
1 reference
public static string GenerateRandomSaltString()
{
    return new string(Enumerable.Repeat("1234567890-=qwertyuiop[]asdfghjkl;'zxcvbnm,.QWERTYUIOP[]ASDFGHJKL;ZXCVBNM,!?><", 2)
        .Select(s => s[random.Next(s.Length)]).ToArray());
}

/// <summary>
/// دالة تشفير النص
/// </summary>
/// <param name="text">النفع للتشفير</param>
/// <param name="key">مفتاح التشفير</param>
/// <returns></returns>
0 references
public async Task<string> EncryptTextAsync(string text, string key)
{
    return Convert.ToBase64String(await EncryptByteArrayAsync(Encoding.UTF8.GetBytes(GenerateRandomSaltString() + text), key));
}
```

-دالة لفك تشفير نص (String)

```
/// <summary>
/// دالة لفك تشفير النص
/// </summary>
/// <param name="text">النص المشفى</param>
/// <param name="key">مفتاح فك التشفير</param>
/// <returns></returns>
0 references
public async Task<string> DecryptTextAsync(string text, string key)
{
    try
    {
        //Return result
        return Encoding.UTF8.GetString(await DecryptByteArrayAsync(Convert.FromBase64String(text), key)).Remove(0, 2);
    }
    catch
    {
        return "Decrypted Error";
    }
}
```

-دالة لتشفيه الملفات

في حالة تشفير الملف نستطيع استخدام دالة تشفير البايتات الا لكن لا نستطيع تشفير الملفات الكبيرة 2 كيما بايت و اكثر, لذلك سنقوم بانشاء دالة اخري لتشفيه الملفات باي حجم.

في البداية سنحتاج الى دالة لتوليد ملح التشفير بشكل عشوائي

```

/// <summary>
// دالة لارجاع مصفوفة من البيانات
// على الاقل 8 بايتات
/// </summary>
/// <returns></returns>
1reference
private static byte[] GetByteArray()
{
    return new byte[8];
}

/// <summary>
// دالة لتوليد ملح التشفير بشكل عشوائي
// مستخدمة في تشفير وفك تشفير الملفات
/// </summary>
/// <returns></returns>
1reference
public static byte[] GenerateRandomSaltByte()
{
    byte[] data = GetByteArray();

    RNGCryptoServiceProvider.Create().GetBytes(data);

    return data;
}

```

دالة تشفير الملفات:

سيستقبل 3 متغيرات بلشكل التالي

```

/// <summary>
// دالة لتنظير الملفات
/// </summary>
/// <param name="filePathToEncrypt">مسار الملف المراد تشفيره</param>
/// <param name="FolderPathToSave">مسار لحفظ الملف بعد التشفير</param>
/// <param name="key">كلمة المرور</param>
/// <returns></returns>
0 references
public async Task<string> EncryptFileAsync(string filePathToEncrypt, stringFolderPathToSave, string key)
{
    FileInfo file = new FileInfo(filePathToEncrypt);

    //Get salt
    byte[] salt = GenerateRandomSaltByte();

    //Create path for encrypted file
    string EncrypteFilePath = FolderPathToSave + file.Name;

    //Create file stream for crypted file
    using (FileStream fileStreamCrypte = new FileStream(EncrypteFilePath, FileMode.Create))
    {
        using (RijndaelManaged AES = new RijndaelManaged())
        {

            //Set key size
            AES.KeySize = 256;

            //Set block size
            AES.BlockSize = 128;

            AES.Padding = PaddingMode.PKCS7;

            //Create key to encrypt file
            var Key = new Rfc2898DeriveBytes(Encoding.UTF8.GetBytes(key), salt, 50000);

            AES.Key = Key.GetBytes(AES.KeySize / 8);

            AES.IV = Key.GetBytes(AES.BlockSize / 8);

            AES.Mode = CipherMode.CFB;

            fileStreamCrypte.Write(salt, 0, salt.Length);

            using (CryptoStream cryptoStream = new CryptoStream(fileStreamCrypte, AES.CreateEncryptor(), CryptoStreamMode.Write))
            {
                using (FileStream fileStreamToEncrypt = new FileStream(file.FullName, FileMode.Open, FileAccess.Read, FileShare.Read))
                {
                    byte[] buffer = new byte[1048576];
                    int read;

                    while ((read = await fileStreamToEncrypt.ReadAsync(buffer, 0, buffer.Length)) > 0)
                    {
                        await cryptoStream.WriteAsync(buffer, 0, read);
                    }
                }
            }
        }
    }

    return EncrypteFilePath;
}

```

دالة فك تشفير الملفات: سيستقبل 3 متغيرات بلشكل التالي

```
0 references
public async Task<string> DecryptFileAsync(string FilePathToDecrypt, string FolderPathToSave, string key)
{
    //Create salt
    byte[] salt = GetByteArray();

    string fileName = Path.GetFileName(FilePathToDecrypt);

    string DecryptedFilePath = FolderPathToSave + fileName;

    int numberOfFileName = 1;

    //Check from file path
    while (File.Exists(DecryptedFilePath))
    {
        //Set new path
        DecryptedFilePath = FolderPathToSave + numberOfFileName.ToString() + fileName;

        numberOfFileName++;
    }

    using (FileStream EncryptedFilePath = new FileStream(FilePathToDecrypt, FileMode.Open))
    {

        await EncryptedFilePath.ReadAsync(salt, 0, salt.Length);

        using (RijndaelManaged AES = new RijndaelManaged())
        {
            //Set key size
            AES.KeySize = 256;

            //Set block size
            AES.BlockSize = 128;

            //Create key to decrypt
            var Key = new Rfc2898DeriveBytes(Encoding.UTF8.GetBytes(key), salt, 50000);

            //Set AES key
            AES.Key = Key.GetBytes(AES.KeySize / 8);
            AES.IV = Key.GetBytes(AES.BlockSize / 8);
            AES.Padding = PaddingMode.PKCS7;
            AES.Mode = CipherMode.CFB;
```

```

        using (CryptoStream cryptoStream = new CryptoStream(EncryptedFilePath, AES.CreateDecryptor(), CryptoStreamMode.Read))
        {
            using (FileStream fileStreamDecrypted = new FileStream(DecryptedFilePath, FileMode.Create))
            {
                int read;
                byte[] buffer = new byte[1048576];

                while ((read = await cryptoStream.ReadAsync(buffer, 0, buffer.Length)) > 0)
                {
                    await fileStreamDecrypted.WriteAsync(buffer, 0, read);
                }
            }
        }

    }

    return DecryptedFilePath;
}

```

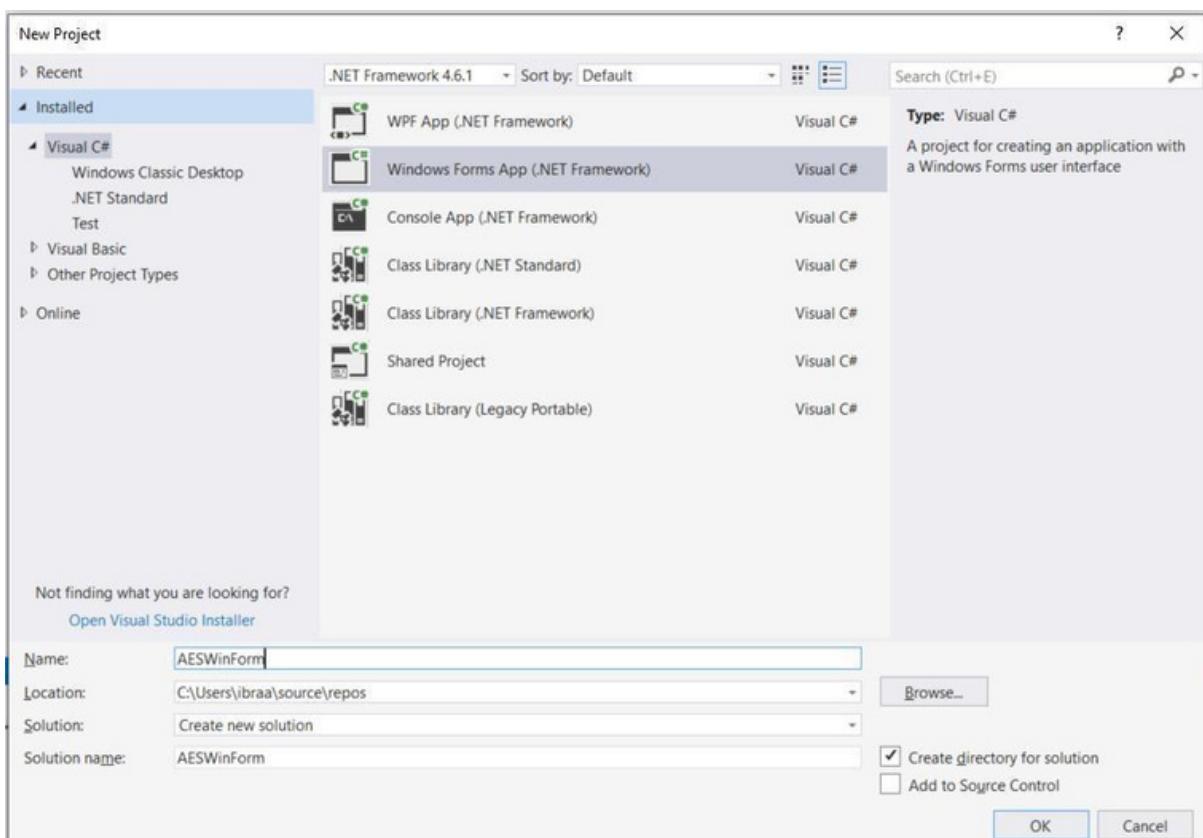
ملخص الفصل الثاني:

في هذا الفصل قمنا بإنشاء مكتبة برمجية تحوي دوال تشفير وفك التشفير للنصوص والملفات و أصبح جاهزة للالاستعمال ضمن تطبيقات عملية.

الفصل الثالث:

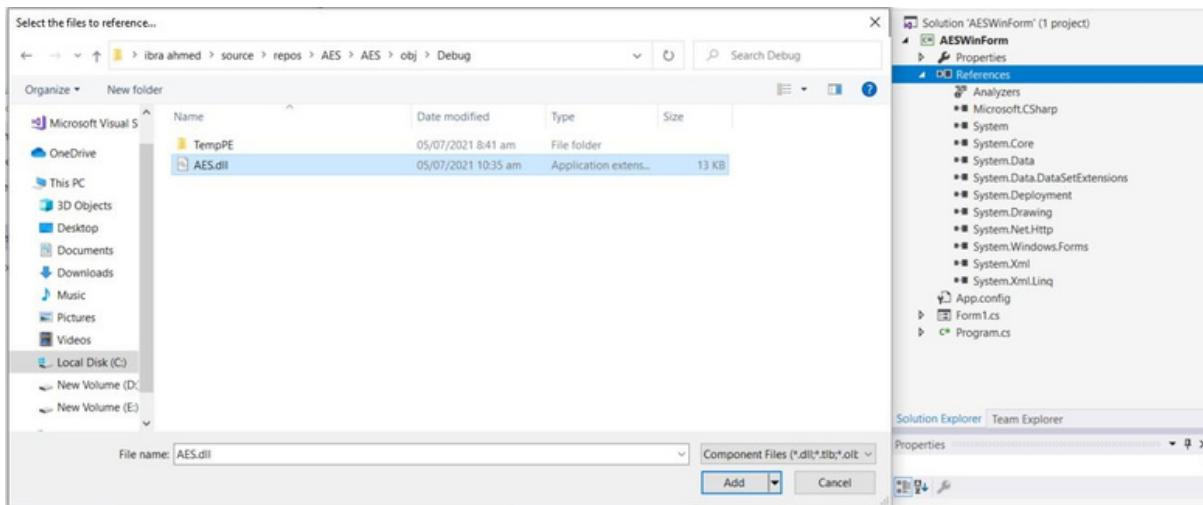
إنشاء تطبيق WinForm لتشفيير وفك التشفير النصوص والملفات

نقوم بإنشاء مشروع جديد من نوع Windows Forms App بأسم AESWinForm



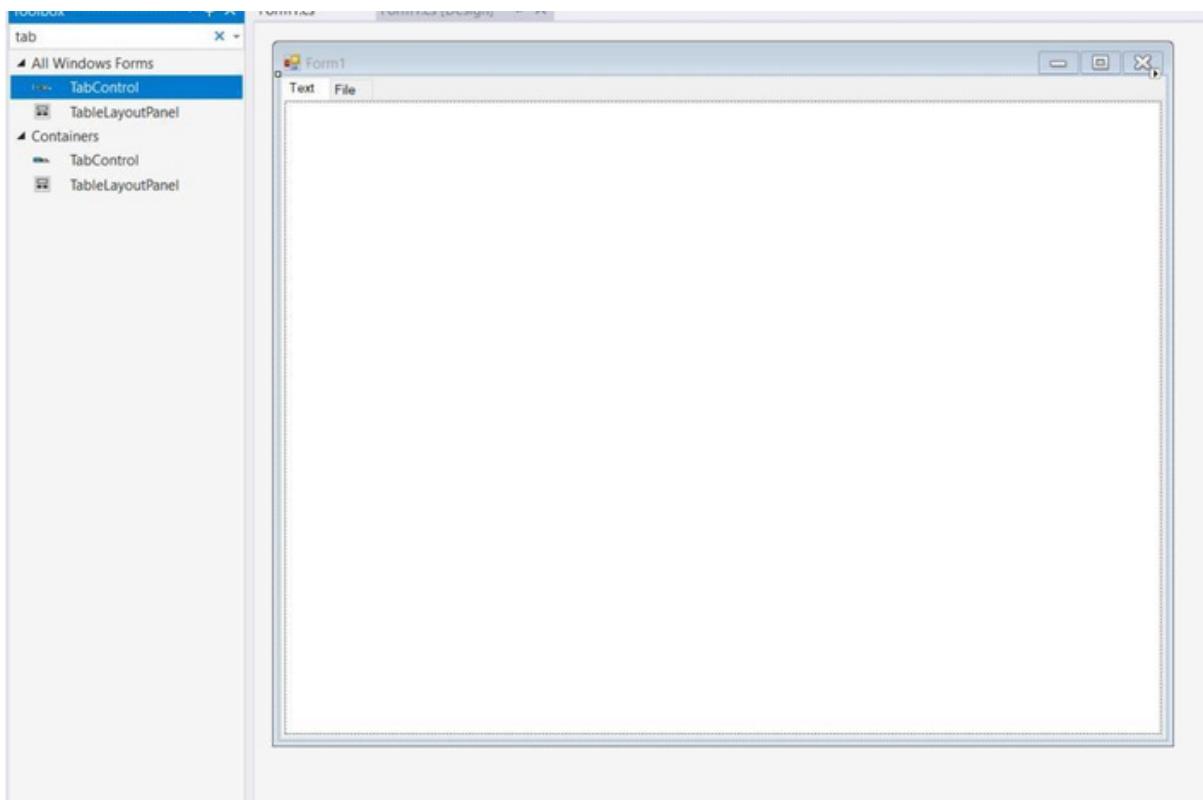
في البداية نقوم بإضافة مكتبة AES.Dll التي قمنا بكتابتها في الفصل الاول الى مراجع المشروع.

-نضع على قائمة References بلزر اليمين و نختار Add References
-و نقوم بالذهاب الى مجلد AES.Dll ونقوم بإضافته

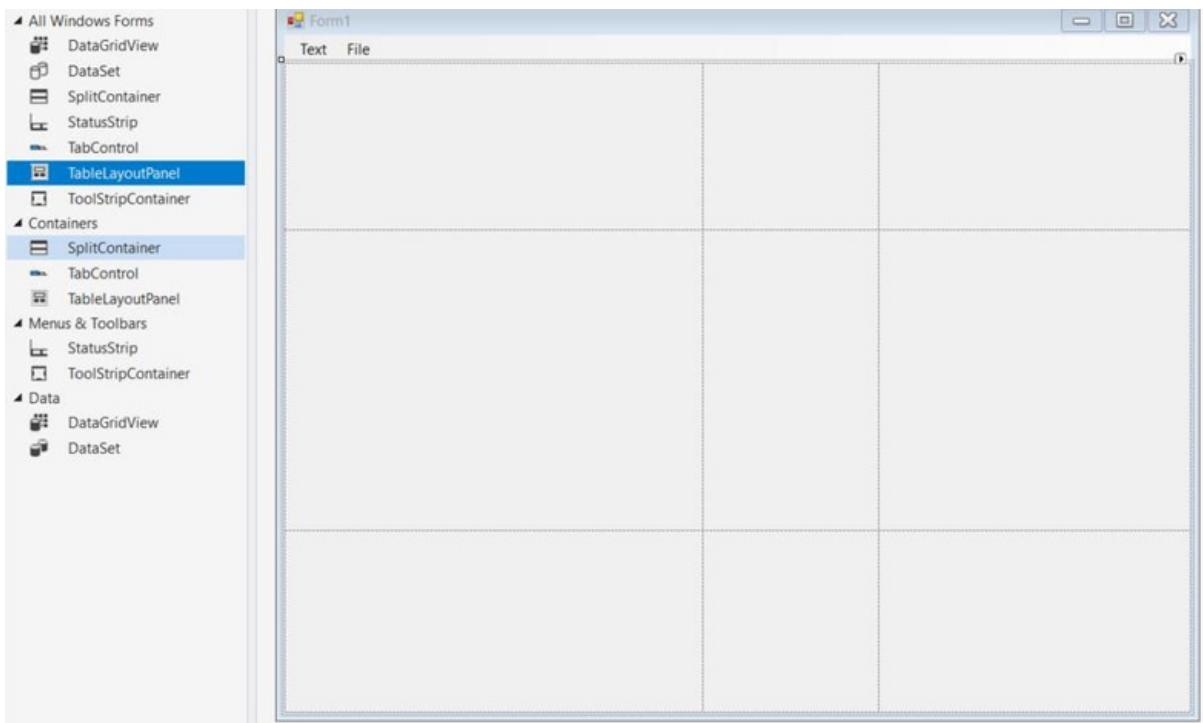


نقوم بتصميم الواجهة بلشكل التالي:

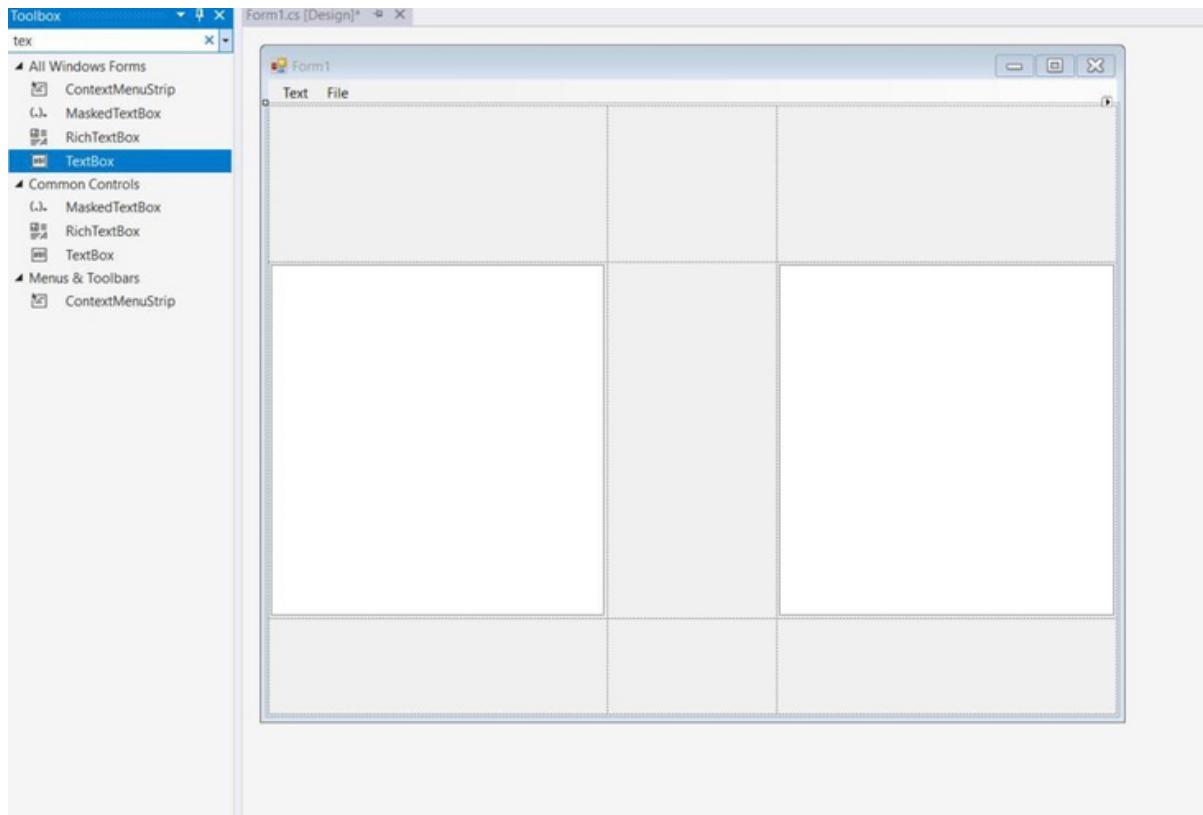
من قائمة **TabControl** نضيف العنصر **TabControl** ونقوم بانشاء قائمتين
Text-1 : لتشفيير وفك تشفير النصوص
File-2 : لتشفيير وفك تشفير الملفات



-نقوم باضافة العنصر TableLayoutPanel الى القائمة لنسطيع ادراج
العناص على لواجهة بشكل متناسق



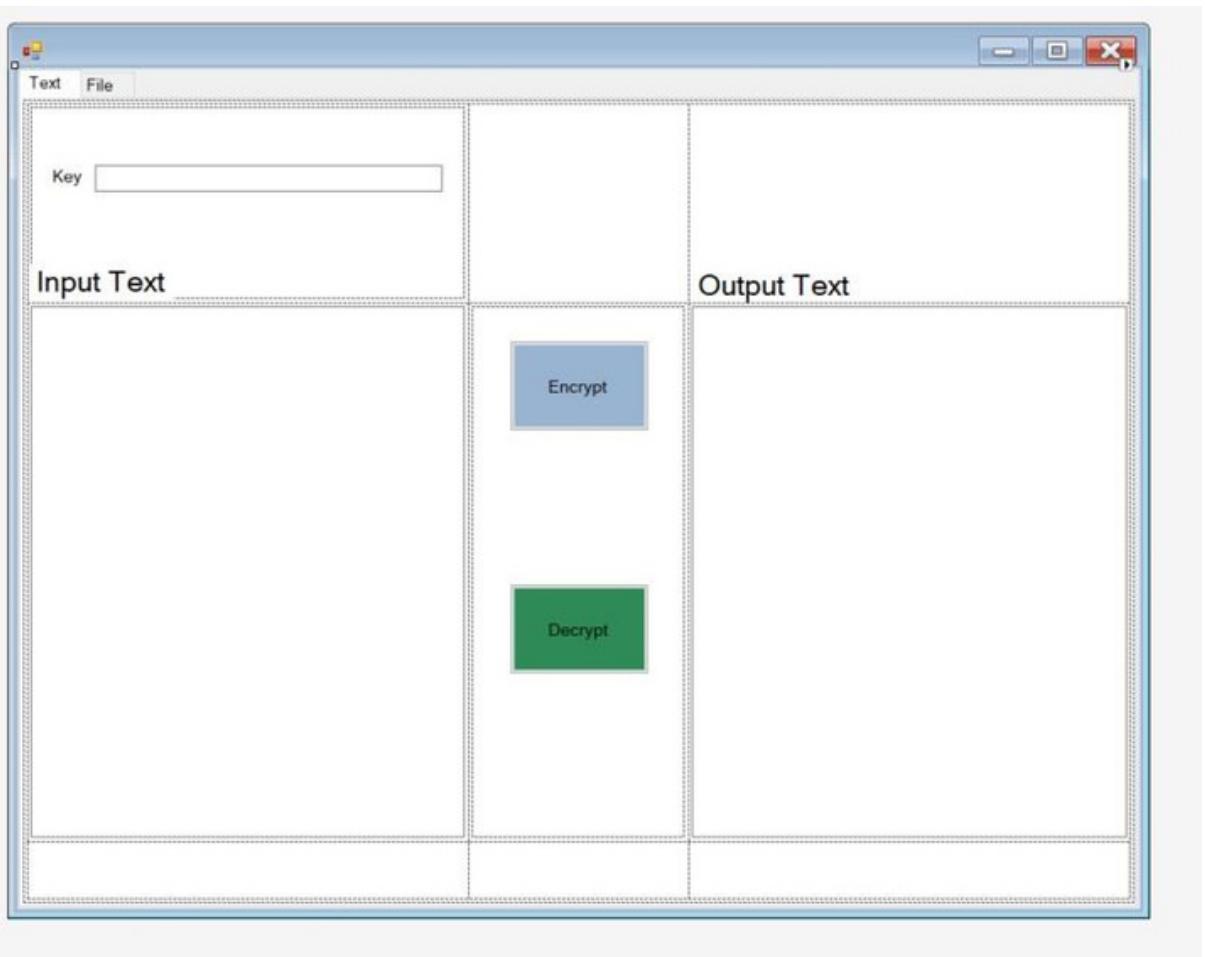
-نقوم الان بادراج عنصر TextBox احدهما لكتابه النص المطلوب و الاخر لاظهار النتيجة المطلوبة



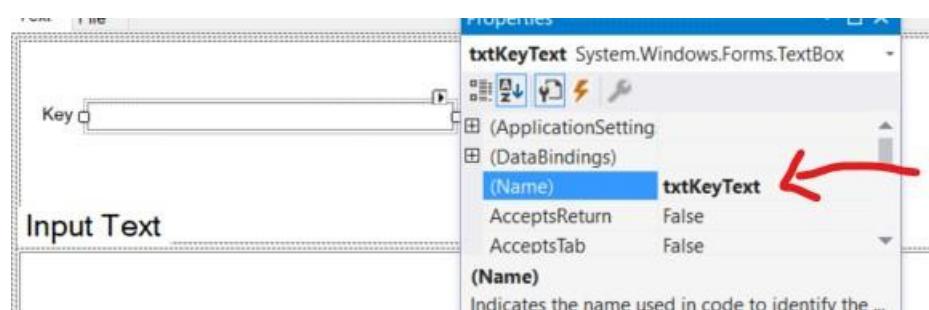
من قائمة الخصائص Properties سنقوم بتسمية كل عنصر txtOutput - النص الالصلي - txtInput - النتيجة ب

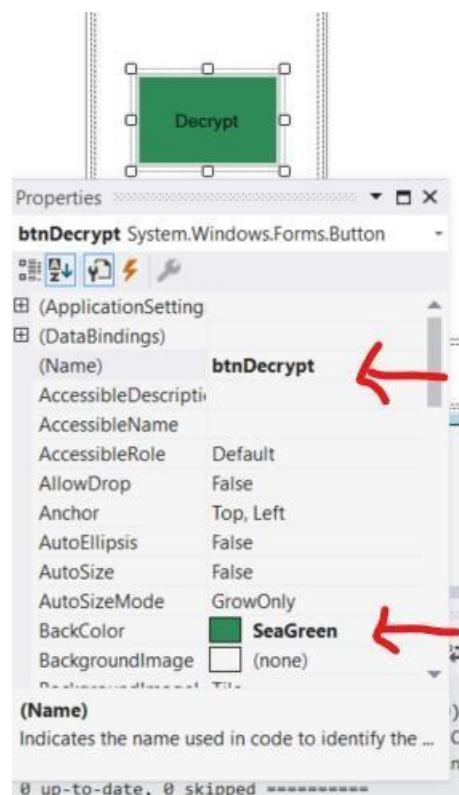
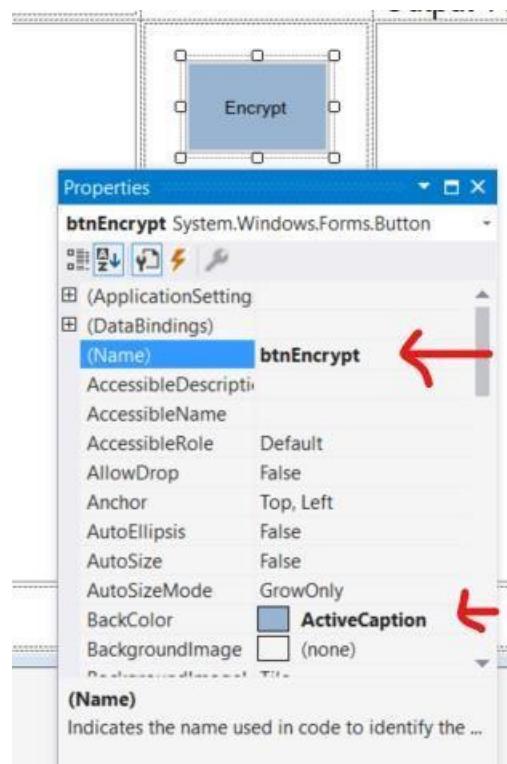
(Name)	txtInput
AcceptsReturn	False
AcceptsTab	False
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
AutoCompleteCustomSource	(Collection)
AutoCompleteMode	None
AutoCompleteSource	None
BackColor	Window
BorderStyle	Fixed3D
CausesValidation	True
Cell	0,1
CharacterCasing	Normal
Column	0
ColumnSpan	1
ContextMenuStrip	(none)
Cursor	IBeam
Dock	Fill
Enabled	True
Font	Microsoft Sans Serif, 7.8pt
ForeColor	WindowText
GenerateMember	True
HideSelection	True
ImeMode	NoControl
Lines	String[] Array
Location	3, 173
Locked	False
Margin	3, 3, 3, 3
MaximumSize	0, 0
MaxLength	32767
MinimumSize	0, 0

-نقوم بإضافة 2 Buttons تقوم بتسمية الاول btnEncrypt والثاني بـ btnDecrypt
-نقوم بإضافة TextBox لالادخال مفتاح للتشفيير او فك التشفير ويكون الشكل النهائي لقائمة Text بلشكل التالي:

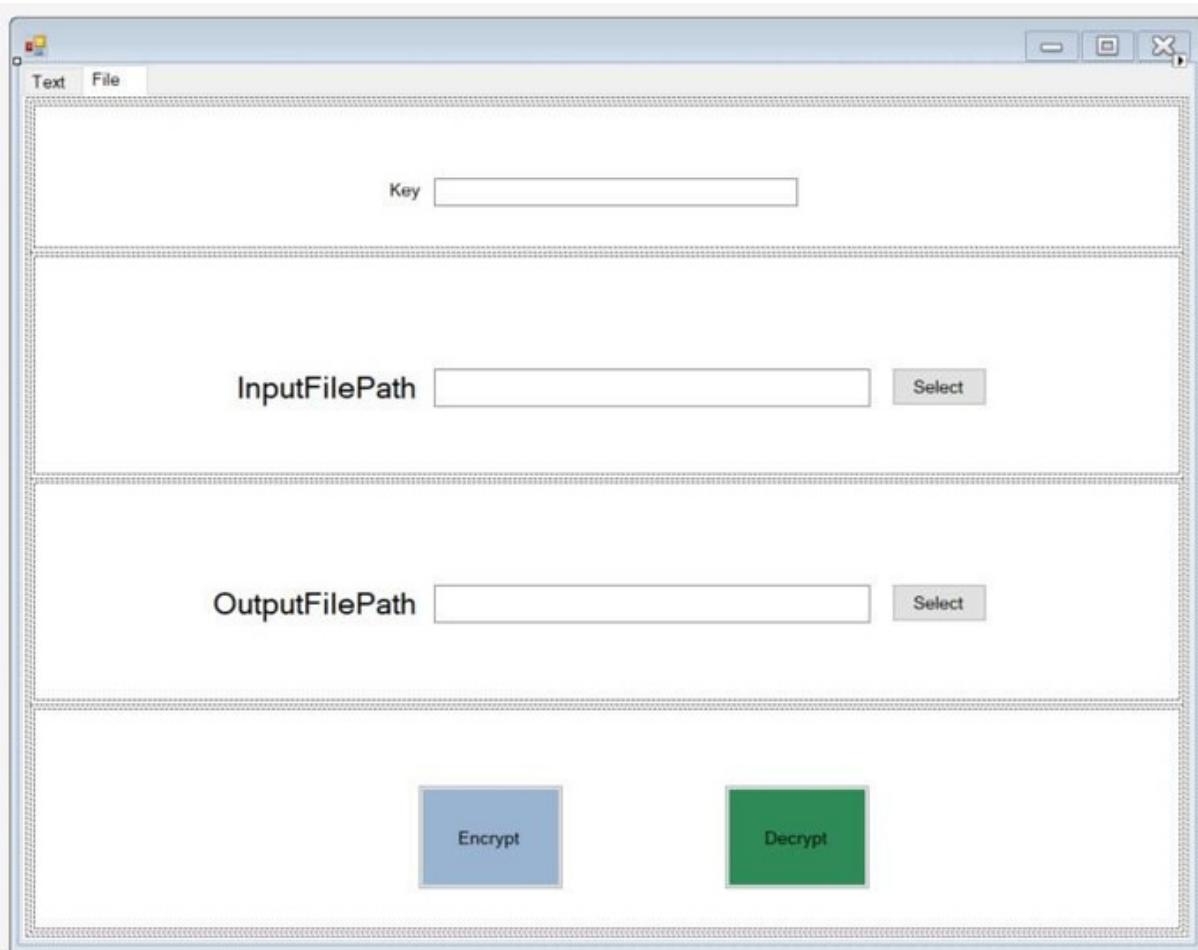


خصائص الألزار و نص المفتاح يكون بلشكل التالي:



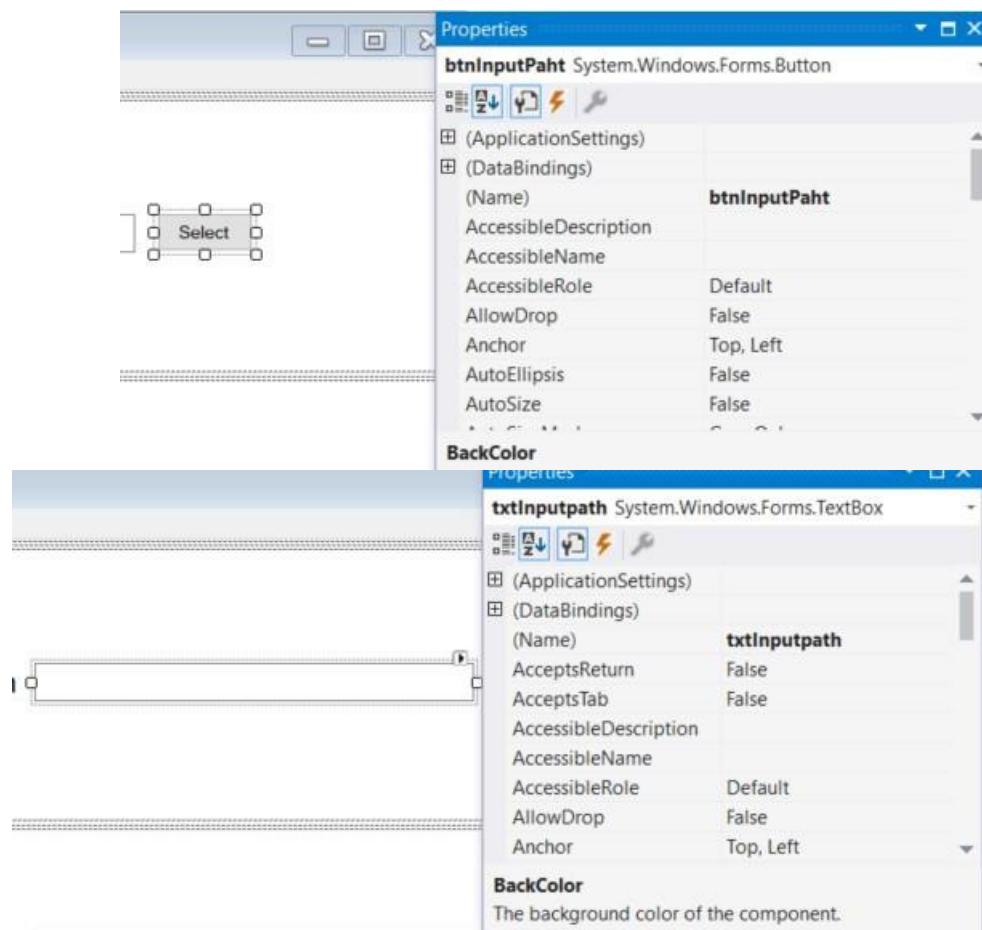


-تصميم قائمة File سيكون بلشكل التالي

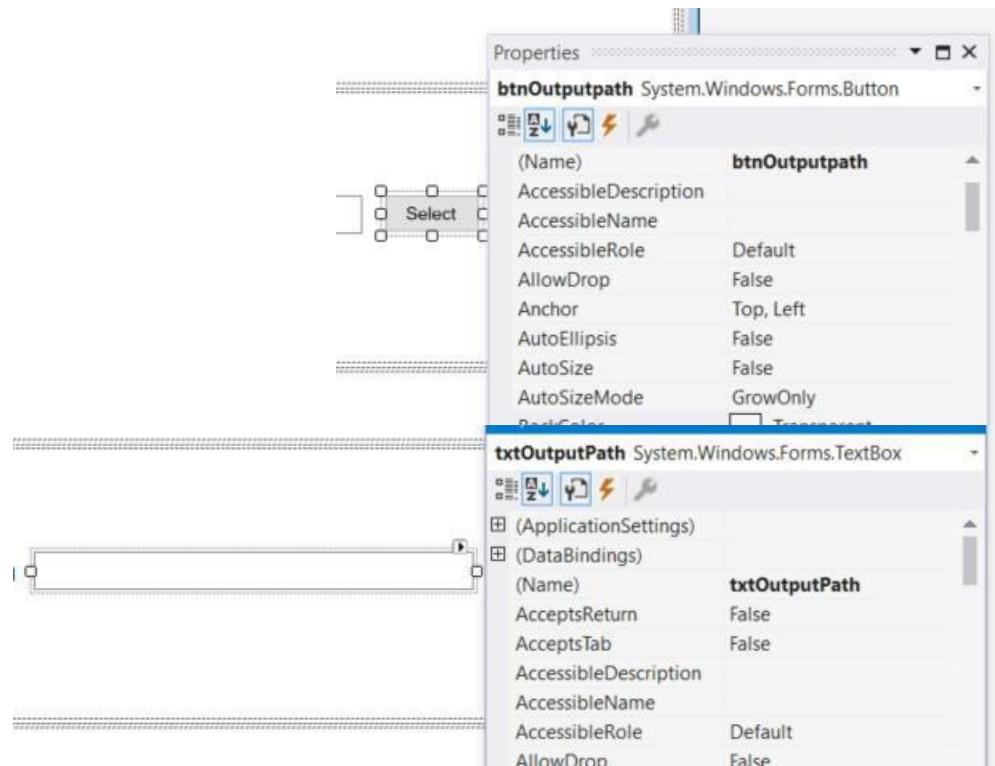


الخصائص:

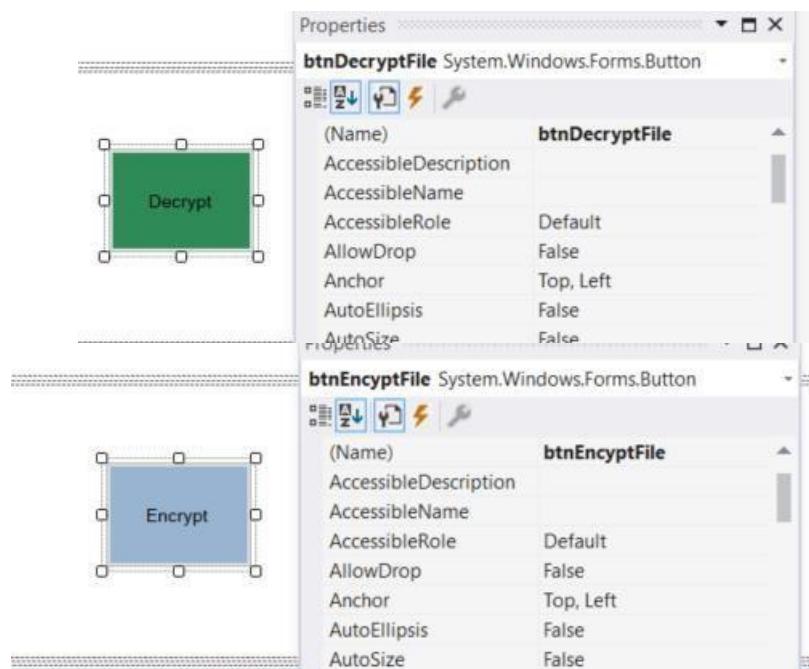
: Input FilePath



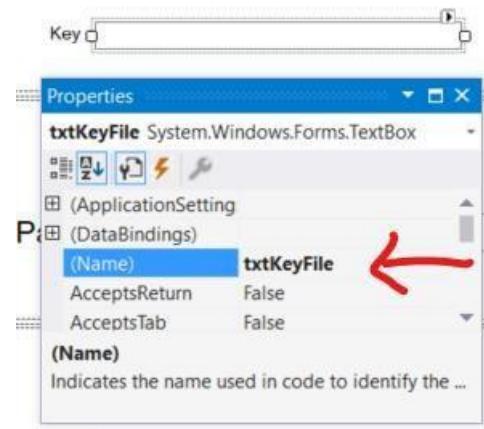
: Out FilePath



: Buttons



Key-



الآن أصبح التصميم جاهزاً للبرمجة
-برمجة قائمة Text

- 1-في قائمة Text نضغط F12 للذهاب الى كود الواجهة
- 2-نقوم بـاستدعاء مجال الالاسماء AES ضمن المكتبة البرمجية التي قمنا بـإنشائها في الفصل الثاني
- 3-نقوم بـلتصریح عن نسخة من الكلاس AESAlgorithm ثم نقوم بـإنشاءه ضمن الـConstructor كما يلي

```

using System;
using System.Windows.Forms;
using AES; ←

namespace AESWinForm
{
    3 references
    public partial class Form1 : Form
    {
        public AESAlgorithm aesAlgorithm; ←
        public Form1()
        {
            InitializeComponent();
            aesAlgorithm = new AESAlgorithm(); ←
        }
    }
}

```

برمجة حدث الضغط على زر تشفير النص

```
/// <summary>
/// حدث فتح الزر لتفعيل النص
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1reference
private async void btnEncrypt_Click(object sender, EventArgs e)
{
    //لتتحقق اذا المستخدم ادخل المفتاح
    if(string.IsNullOrEmpty(txtKeyText.Text.Trim()))
    {
        MessageBox.Show("العنوان", "العنوان", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

        txtKeyText.Focus();

        return;
    }

    //تحقق اذا كان النص المدخل ليس فارغاً
    if (string.IsNullOrEmpty(txtInput.Text.Trim()))
    {
        MessageBox.Show("الرجاء كتابة النص المراد تشفيره", "", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

        txtInput.Focus();

        return;
    }

    try
    {
        //تفعيل النص المدخل و اظهار النتيجة في
        txtOutput.Text= await aesAlgorithm.EncryptTextAsync(txtInput.Text, txtKeyText.Text);
    }
    catch (Exception ex)
    {
        //اظهار رسالة خطأ في حال حدوث خطأ، اثناء محاولة تشفير النص
        MessageBox.Show(ex.Message, "", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
}
}
```

-برمجة

برمجة حدث عند الضغط على زر فك تشفير النص

```
/// <summary>
/// حدث ينفط الزر لفك تشفير النص
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 reference
private async void btnDecrypt_Click(object sender, EventArgs e)
{
    //تحقق اذا المستخدم ادخل المفتاح
    if (string.IsNullOrEmpty(txtKeyText.Text.Trim()))
    {
        MessageBox.Show("الرجاء ادخال المفتاح لفك تشفير البيانات", "العنوان", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        txtKeyText.Focus();
        return;
    }

    //تحقق اذا كان النص المدخل ليس فارغاً
    if (string.IsNullOrEmpty(txtInput.Text.Trim()))
    {
        MessageBox.Show("", "الرجاء كتابة النص المراد فك تشفيره", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        txtInput.Focus();
        return;
    }

    try
    {
        //تشفير النص المدخل و اظهار النتيجة في
        txtOutput.Text = await aesAlgorithm.DecryptTextAsync(txtInput.Text, txtKeyText.Text);
    }
    catch (Exception ex)
    {
        //اظهار رسالة خطأ، في حال حدوث خطأ، هنا، محاولة تشفير النص
        MessageBox.Show(ex.Message, "خطأ", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
}
```

-برمجة

برمجة حدث عند الضغط على زر اختيار مسار الملف المراد تشفيره او فك تشفيره.

```
/// <summary>
/// اختبار المسار لحفظ الملف
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 reference
private void btnInputPaht_Click(object sender, EventArgs e)
{
    OpenFileDialog choofdlog = new OpenFileDialog();
    choofdlog.Filter = "All Files (*.*)|*.*";

    if (choofdlog.ShowDialog() == DialogResult.OK)
    {
        txtInputpath.Text = choofdlog.FileName;
    }
}
```

برمجة حدث عند الضغط على زر اختيار مسار حفظ الملف الناتج من عملية التشفير او فك التشفير.

```
/// <summary>
/// اختبار المسار لحفظ الملف
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 reference
private void btnOutputPaht_Click(object sender, EventArgs e)
{
    var folderBrowserDialog1 = new FolderBrowserDialog();

    if(string.IsNullOrEmpty(txtInputpath.Text.Trim()))
    {
        MessageBox.Show("اختر الملف", "الرجاء اختيار الملف اولاً", MessageBoxButtons.OK);
        return;
    }

    DialogResult result = folderBrowserDialog1.ShowDialog();
    if (result == DialogResult.OK)
    {
        // هنا نقو باختيار المجلد و توليد رقم فريد لاسم الملف و اهتمنا اللائحة
        txtOutputPath.Text = folderBrowserDialog1.SelectedPath + "/" + Path.GetFileName(txtInputpath.Text);
    }
}
```

برمجة حدث عند الضغط على زر تشفير الملف.

```
حدث هنف زر تشفير الملف
/// <summary>
/// <param name="sender"></param>
/// <param name="e"></param>
reference
private async void btnEncryptFile_Click(object sender, EventArgs e)
{
    //لتتحقق اذا المستخدم ادخل الملفات
    if (string.IsNullOrEmpty(txtKeyFile.Text.Trim()))
    {
        MessageBox.Show("الملفات", "الرجاء اختيار مسار الملف المراد تشفيره", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        txtKeyFile.Focus();
        return;
    }

    //تحقق اذا كان النص المدخل ليس فارغاً
    if (string.IsNullOrEmpty(txtInputpath.Text.Trim()))
    {
        MessageBox.Show("الرجاء اختيار مسار حفظ الملف", "", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    try
    {
        btnEncryptFile.Text = "Encrypting..";

        //تشفير النص المدخل و اظهار النتيجة في
        await aesAlgorithm.EncryptFileAsync(txtInputpath.Text,txtOutputPath.Text,txtKeyFile.Text);

        MessageBox.Show("تم تشفير الملف بنجاح", "", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    catch (Exception ex)
    {
        //اظهار رسالة خطأ في حال حدوث خطأ، اتنا، محاولة تشفير النص
        MessageBox.Show(ex.Message, "خطأ", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    finally
    {
        btnEncryptFile.Text = "Encrypt";
    }
}
```

btnDecryptFile_Click: برمجة حدث عند الضغط على زر فك تشفير الملف.

```
/// <summary>
/// حدث ضغط زر لفك تشفير الملف
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
1 reference
private async void btnDecryptFile_Click(object sender, EventArgs e)
{
    //تحقق اذا المستخدم ادخل المفتاح
    if (string.IsNullOrEmpty(txtKeyFile.Text.Trim()))
    {
        MessageBox.Show("الملتخص", "الرجاء اختيار مسار الملف المزدوج لفك تشفيره", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        txtKeyFile.Focus();
        return;
    }

    //تحقق اذا كان النص المدخل ليس فارغا
    if (string.IsNullOrEmpty(txtInputpath.Text.Trim()))
    {
        MessageBox.Show("", "الرجاء اختيار مسار حملة الملف", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
        return;
    }

    try
    {
        btnEncryptFile.Text = "Decrypting..";
        //فك تشفير النص المدخل و الظهار النتيجة في
        await aesAlgorithm.DecryptFileAsync(txtInputpath.Text, txtOutputPath.Text, txtKeyFile.Text);

        MessageBox.Show("تم فك تشفير الملف بنجاح", "", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    catch (Exception ex)
    {
        //اظهار رسالة خطأ في حال حدوث خطأ، هنا محاولة تشفير النص
        MessageBox.Show(ex.Message, "خطأ", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
    finally
    {
        btnEncryptFile.Text = "Decrypt";
    }
}
```

ملخص : هكذا نكون قد انتهينا من قسم البرمجة و اصبح البرنامج جاهز للقيام بعمليات التشفير و فك التشفير

-برمجة

كود البرنامج كامل:

```
using System; using
System.Windows.Forms; using
AES; using System.IO;

namespace AESWinForm
{
public partial class Form1 : Form
{
public AESAlgorithm aesAlgorithm; public
Form1()
{
InitializeComponent();
aesAlgorithm = new AESAlgorithm();
}

/// <summary>
/// </summary> حدث ضغط الزر لتشифر النص ///
/// <param name="sender"></param> ///
<param name="e"></param>
private async void btnEncrypt_Click(object sender, EventArgs e)
{
if(string.IsNullOrEmpty(txtKeyText.Text.Trim()))
{
MessageBox.Show("الرجاء ادخال المفتاح لتشифر البيانات", "المفتاح", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

txtKeyText.Focus();
return;
}

تحقق اذا كان النص المدخل ليس فارغا // 

if (string.IsNullOrEmpty(txtInput.Text.Trim()))
{
MessageBox.Show("", "الرجاء كتابة النص المراد تشفيره", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation);

txtInput.Focus();
return;
} try { //txtOutput تشفير النص المدخل و اظهار النتيجة في txtOutput.Text= await
aesAlgorithm.EncryptTextAsync(txtInput.Text, txtKeyText.Text);
} catch (Exception
ex)
{
اظهار رسالة خطأ في حال حدوث خطاء أثناء محاولة تشفير النص //
MessageBox.Show(ex.Message, "", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
}
```

```

/// <summary>
/// حدث ضغط الزر لف ك تشفير النص
/// </summary>
/// <param name="sender"></param> ///
<param name="e"></param>
private async void btnDecrypt_Click(object sender, EventArgs e)
{
    if (string.IsNullOrEmpty(txtKeyText.Text.Trim()))
    {
        MessageBox.Show("الرجاء ادخال المفتاح لف ك تشفير البيانات", "المفتاح", MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation); txtKeyText.Focus();
        return;
    }

    //تحقق اذا المستخدم ادخل المفتاح //
    if (string.IsNullOrEmpty(txtInput.Text.Trim()))
    {
        MessageBox.Show("الرجاء كتابة النص المراد فك تشفيره", "", MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation);

        txtInput.Focus();
        return;
    }

    //try { تشفير النص المدخل و اظهار النتيجة
    //ف ي
    txtOutput.Text = await aesAlgorithm.DecryptTextAsync(txtInput.Text, txtKeyText.Text);
    }
    catch (Exception ex)
    {
        //اظهار رسالة خطاء ف ي حال حدوث خطاء أثناء محاولة تشفير النص//
        MessageBox.Show(ex.Message, "خطاء", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
    }
}

/// <summary>
/// اختيار الملف المراد تشفيره //
/// </summary>
/// <param name="sender"></param> ///
<param name="e"></param>
private void btnInputPaht_Click(object sender, EventArgs e) {
    OpenFileDialog choofdlog = new OpenFileDialog(); choofdlog.Filter
    = "All Files (*.*)|*.*";

    if (choofdlog.ShowDialog() == DialogResult.OK)
    {
        txtInputpath.Text =
        choofdlog.FileName;
    }
}

/// <summary>
/// اختيار المسار لحفظ الملف //
/// </summary>

```

```

/// <param name="sender"></param> ///
<param name="e"></param>
private void btnOutputpath_Click(object sender, EventArgs e)
{
var folderBrowserDialog1 = new FolderBrowserDialog();

if(string.IsNullOrEmpty(txtInputpath.Text.Trim()))
{
MessageBox.Show("الرجاء اختيار الملف اول ", "اختيار الملف");
return;
}

DialogResult result = folderBrowserDialog1.ShowDialog(); if
(result == DialogResult.OK)
{
    // هنا نقو باختيار المجلد و توليد رقم فريد لسم الملف و اضفنا اللالحقة//g
    txtOutputPath.Text =
    folderBrowserDialog1.SelectedPath+ "//"+Path.GetFileName(txtInputpath.Text);
}
}

/// <summary>
// حدث ضغط زر تشفير الملف
/// </summary>
/// <param name="sender"></param> ///
<param name="e"></param>
private async void btnEncryptFile_Click(object sender, EventArgs e)
{
    // للتشفيـر نتحقق اذا المستخدم ادخل المفتاح//ا
    if (string.IsNullOrEmpty(txtKeyFile.Text.Trim()))
    {
        MessageBox.Show("الرجاء اختيار مسار الملف المراد تشفيره", "المفتاح",
        MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation);

        txtKeyFile.Focus();
        return;
    }

    // نتحقق اذا كان النص المدخل ليس فارغ //ا
    if (string.IsNullOrEmpty(txtInputpath.Text.Trim()))
    {
        MessageBox.Show("الرجاء اختيار مسار حفظ الملف", "", MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation);
        return;
    } try { btnEncryptFile.Text =
    "Encrypting..";
    //txtOutput // تشفـير النص المدخل و اظهـار النتيـجة في
    await
    aesAlgorithm.EncryptFileAsync(txtInputpath.Text,txtOutputPath.Text,txtKeyFile.Text);
    MessageBox.Show("تم تشفـير المـلف بنجـاح", "", MessageBoxButtons.OK,
    MessageBoxIcon.Exclamation);
    } catch (Exception
    ex)
    }

    // اظهـار رسالة خطـاء فـي حال حدـوث خطـاء اثنـاء محاـولة تـشفـير النـص//ا
}

```


المراجع

1- سريما ثوميتا:

codeproject.com/script/Membership/LogOn.aspx?rp=%2fKB%2fcs%2fApplicationAutoUpdate%2fdemo_project.zip&download=true

CodeProject-2

3- المبرمج العربي:

<https://arabicprogrammer.com/article/6888829509>

C# 7.0 in a Nutshell-Joseph Albahari : Book-4