بحث بعنوان:

" تصميم تطبيق لمعالجة الصور "

إعداد الطلاب:

عبد العزيز عباس عبد العزيز عباس

وعد حسين جبرين طلبة

هالة صالح جبرين منزول

إشراف:

د/ ناجي إسحاق محمد

بسم الله الرحمن الرحيم

"اقرأ باسم ربك الذي خلق، خلق الإنسان من علق. اقرأ وربك الأكرم، الذي علم بالقلم، علم الإنسان ما لم يعلم"

[سورة العلق: 1-5]

"وَقُل رَّبِّ زِدْنِي عِلْمًا" [سورة طه: 114]

الإهداء,,,

إلى من كانوا سببًا في دفعي نحو طريق العلم والمعرفة..

إلى والديّ الأعزاء، سندي في الحياة..

إلى أساتذتي الكرام في الكلية الأردنية الذين غرسوا في نفسي حب البحث والتعلم..

إلى أصدقائي وزملائي الذين كانوا خير معين.

أُهدي هذا العمل المتواضع عرفانًا وتقديرًا..

أتوجه بجزيل الشكر والعرفان إلى كل من ساعدنا ووقف بجانبنا خلال إعداد هذا البحث..

إلى المشرف الكريم الذي لم يبخل علينا بالتوجيه والنصح، وإلى كل من مد لنا يد العون بكلمة أو نصيحة أو دعم..

كما لا ننسى أن نشكر الكلية الأردنية على توفير بيئة تعليمية محفزة..

كما نشكر كل من شجعنا وألهمنا للاستمرار حتى يكتمل هذا العمل..

نسأل الله أن يجزيهم عنا خير الجزاء..

المقدمة:

في عالم التكنولوجيا المتسارع، أصبحت معالجة الصور من أهم التطبيقات التي يتم استخدامها في العديد من المجالات. حيث يتم استخدام تقنيات معالجة الصور في الذكاء الاصطناعي والتعليم الطبي والهندسة وغيرها من المجالات .

تهدف هذه الدراسة إلى تصميم تطبيق يعالج الصور البيانية أو الصور الفوتوغرافية.

في هذا البحث، سيتم استعراض أهم الأدوات والتقنيات المستخدمة في تصميم هذا التطبيق، بالإضافة إلى تحليل البيانات واختبار فعالية التطبيق.

مشكلة البحث:

1. التحديات التقنية في معالجة الصور :

- السرعة والدقة :

تواجه العديد من التطبيقات صعوبة في تحقيق التوازن بين السرعة والدقة في معالجة الصور، مما يؤثر على الأداء في التطبيقات الطبية.

- الموارد الحاسوبية :

تتطلب عمليات معالجة الصور المتقدمة قدرة حسابية عالية، مما يؤدي إلى الحاجة لتوفير أجهزة ذات إمكانيات معالجة مرتفعة.

2. تحسين واجهة المستخدم :

- سهولة الاستخدام :

تعتمد العديد من التطبيقات على واجهات استخدام غير سهلة، مما يجعلها صعبة الاستخدام لغير المتخصصين.

- التخصيص:

عدم وجود خيارات كافية لتخصيص التطبيق حسب احتياجات المستخدمين المختلفة يؤدي إلى تجربة أقل فعالية.

3. الأمان والخصوصية:

- حماية البيانات:

معالجة الصور تتضمن بيانات حساسة، مثل الصور الشخصية، التي تتطلب إعدادات أمان مرتفعة لمنع الوصول غير المصرح به أو التسريب.

- الخصوصية:

يجب أن يكون التطبيق متوافقاً مع معايير الخصوصية العالمية لضمان حقوق المستخدمين وعدم انتهاك خصوصيتهم.

4. الاعتماد على الطرق التقليدية:

- الكفاءة:

الطرق التقليدية لمعالجة الصور تكون غير فعالة للمرضى ووقتاً طويلاً، مما يؤدي إلى تجربة مستخدم غير مرضية.

- التحسينات المستقبلية:

عدم القدرة على التكيف مع التغيرات وتحسينات مستقبلية بسهولة بسبب بنية التطبيقات التقليدية.

أهداف البحث:

1. تطوير تطبيق متقدم لمعالجة الصور:

- تصميم تطبيق يستخدم أحدث تقنيات الذكاء الاصطناعي والتعلم العميق لتحليل ومعالجة الصور بدقة وسرعه عالية.
- ضمان أن يكون التطبيق قادرا على التعامل مع أنواع مختلفة من الصور بما في ذلك الصور الثابتة والفيديو.

2. تحسين الكفاءة التشغيلية للتطبيق:

- تقليل الوقت المستغرق في معالجة الصور من خلال تحسين الخوارزميات المستخدمة وزيادة سرعة المعالجة.
- تقليل استخدام الموارد الحاسوبية دون التأثير على جودة معالجة الصور.

3. توفير ميزات متقدمة للمستخدمين:

- توفير تقنيات مثل التعرف على الوجه، والتعرف على الأشياء، وتحسين جودة الصور في تطبيقات الهواتف الذكية، مما يسهم في تحسين تجربة المستخدم.

4. تحليل البيانات المستخلصة من الصور:

- استخراج بيانات من الصور الملتقطة وتخزينها لتحليلها لاحقاً، مثل تحليل سلوك المستخدمين أو مراقبة الحالة الصحية، مما يسهم في اتخاذ قرارات أفضل بناءً على المعلومات المستخلصة من الصور.

5. ضمان الخصوصية والأمان:

- توفير خيارات لتأمين الصور والمعالجة بسرية تامة لضمان حماية خصوصية المستخدمين.
- تنفيذ إجراءات لمنع الوصول غير المصرح به إلى الصور أو البيانات المستخلصة.

منهجية البحث:

منهجية البحث المقترحة لتصميم تطبيق معالجة الصور:

1. تحديد المشكلة وأهداف البحث:

- تحديد مشكلة معالجة الصور المقترحة وتحديد الأسباب المؤدية لها.
- توضيح الأهداف المرجوة من البحث وتحديد الفائدة المرجوة من تطوير تطبيق المعالجة.

2. مراجعة الأدبيات:

- استعراض الدراسات والأبحاث السابقة المتعلقة بمعالجة الصور والتقنيات المستخدمة فيها.
- تحليل النماذج والتقنيات المستخدمة حالياً وتحديد نقاط القوة والضعف فيها.

3. تصميم النموذج الأولي للتطبيق:

- تصميم النموذج الأولي للتطبيق بناءً على الأهداف والمواصفات المطلوبة.
- تحديد الأدوات والبرمجيات اللازمة لتطوير النموذج، مثل لغة البرمجة وأطر العمل.

4. تطوير التطبيق:

- بناء الكود الأساسي وتطوير الوحدات المختلفة للتطبيق.
- تنفيذ خوارزميات معالجة الصور واختبارها لتحسين أدائها

5. اختبار التطبيق:

- إجراء اختبارات شاملة للتطبيق للتحقق من كفاءة ودقة في معالجة الصور.
- استخدام بيانات حقيقية لاختبار التطبيق في سيناريوهات مختلفة وتقييم الأداء.

6. تحليل البيانات:

- جمع البيانات الناتجة من اختبارات التطبيق وتحليلها.
- تحديد المشاكل المحتملة واقتراح التحسينات للتطبيق.

7. تحسين وتطوير النظام:

- تحسين الخوارزميات والأدوات المستخدمة بناءً على نتائج التحليل.
- تطوير واجهة المستخدم لتسهيل استخدام المستخدمين وتحسين تجربة الاستخدام.

8. توثيق البحث:

- إعداد تقرير نهائي يشمل جميع المراحل التي تم تنفيذها في البحث.
- يتضمن التقرير التوصيات والمقترحات المستقبلية لتحسين البحث.

9. نشر النتائج:

- نشر النتائج في مجلات علمية محكمة أو تقديمها في مؤتمرات علمية.
- توظيف النتائج في خدمة المجتمع العلمي للاستفادة من الأفكار والملاحظات.

أدوات البحث:

1. البرمجيات وأطر العمل.
2. بيئات التطوير المتكاملة (IDE).
3. مكتبات ومعالجات الصور.
4. أدوات التحليل الإحصائي.
5. أدوات اختبار وتحليل الأداء.
6. بيانات التدريب والتقييم.
7. أدوات إدارة المشاريع والتعاون.

تصميم البرنامج:

الهيكل العام للبرنامج:

أ/ الواجهة الأمامية (Frontend):

- واجهة مستخدم جذابة وسهلة الاستخدام تتيح للمستخدمين تحميل الصور ومعالجتها.
- أدوات تحكم لتطبيق الفلاتر والتأثيرات ومعاينة النتائج الفورية.

ب/ الواجهة الخلفية (Backend):

- خادم يستقبل طلبات المستخدم لمعالجة الصور ويعيد النتائج.
- قواعد بيانات لتخزين الصور والبيانات المتعلقة بها.

المكونات الأساسية:

أ. تحميل الصور:

- يتم تحميل الصور من الكمبيوتر الشخصي أو الأجهزة المحمولة.
- دعم أنواع ملفات الصور المختلفة (JPEG، PNG، وغيرها).

ب. معالجة الصور:

- خوارزميات تحسين الصور: تحسين الإضاءة، تعديل الألوان، إزالة الضوضاء.
- خوارزميات التعرف على الأشياء: تحديد الكائنات في الصور.
- خوارزميات التعرف على الوجوه: تحديد الوجوه في الصور.

ج/ تطبيق الفلاتر والتأثيرات:

- مجموعة متنوعة من الفلاتر لتحسين وتجميل الصور.
- أدوات تحرير مثل القص، التدوير، التعديل على الألوان.

د/ تحليل الصور:

- استخراج الميزات والمعلومات من الصور المعالجة.
- استخدام خوارزميات مستخدمة بناءً على البيانات المستخدمة.

التقنيات المستخدمة:

أ/ لغات البرمجة:

- Python: لمعالجة الصور وتنفيذ الخوارزميات.
- JavaScript: لبناء واجهة المستخدم التفاعلية.

ب/ أطر العمل والمكتبات:

- OpenCV: لمعالجة الصور بواسطة الحاسوب.
- TensorFlow أو PyTorch: لبناء نماذج التعلم العميق.
- React أو Vue.js: لبناء واجهة المستخدم التفاعلية.

الأمان والخصوصية:

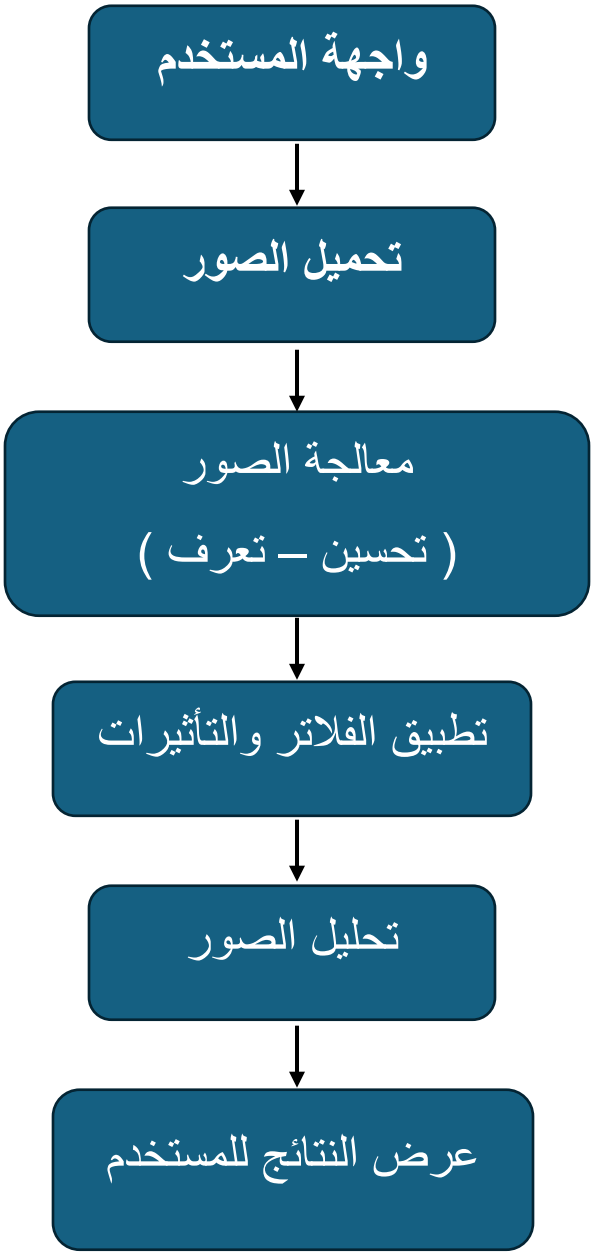
أ/ تأمين البيانات:

- تشفير الصور والبيانات أثناء النقل والتخزين.
- تنفيذ إجراءات أمان قوية لحماية بيانات المستخدمين.

ب/ الامتثال للمعايير العالمية:

- ضمان أن يكون البرنامج متوافقاً مع المعايير العالمية للخصوصية وحماية البيانات.

الرسم البياني للتصميم:



تقييم الأداء:

تقييم الأداء يعتبر جزءاً أساسياً من منهجية البحث، حيث يمكن لك التحقق من كفاءة ودقة نماذج التصنيف المستخدمة، فيما يلي بعض المقاييس المستخدمة لتقييم أداء التطبيق:

1. دقة المعالجة:

- **معدل الدقة (Accuracy Rate)**

قياس عدد الصور المعالجة بشكل صحيح مقارنة بعدد الصور الإجمالي.

- **المقاييس التقليدية للتصنيف (Precision)، (Recall, F1-Score)**

تستخدم لتقييم نماذج التصنيف في استرجاع الميزات الصحيحة داخل الصور.

2. سرعة المعالجة:

- **زمن الاستجابة (Response Time):**

قياس الوقت المستغرق في معالجة الصورة من لحظة إدخالها حتى إتمام المعالجة.

- **عدد الصور المعالجة في الثانية (Throughput):**

قياس قدرة التطبيق على معالجة عدد كبير من الصور في وقت محدد.

3. استهلاك الموارد:

- **استهلاك الذاكرة (Memory Usage):**

قياس كمية الذاكرة المستخدمة أثناء معالجة الصور.

- **استهلاك وحدة المعالجة المركزية (CPU Usage):**

قياس معدل استهلاك وحدة المعالجة المركزية أثناء عمليات المعالجة.

4. الجودة البصرية:

- التحليل البصري (Visual Inspection):

فحص الصور المعالجة بصريا للتأكد من أنها تلبى متطلبات الجودة.

- المقاييس الرقمية للجودة (PSNR,SSIM)

استخدام مقاييس رقمية مثل نسبة الإشارة الى الضوضاء القصوى (PSNR) ومؤشر التشابه الهيكلي لتقييم جودة الصور المعالجة (SSIM).

5. تجربة المستخدم:

- اختبارات القابلية للاستخدام (Usability Testing):

الحصول على ملاحظات من المستخدمين النهائيين حول سهولة استخدام التطبيق وفعاليته.

- اختبارات رضا المستخدم (User Satisfaction Surveys):

عمل استبيانات من المستخدمين لقياس مدى رضاهم عن أداء التطبيق والخدمات المتاحة.

6. الأمان والخصوصية:

- اختبارات الأمان (Security Testing):

التحقق من وجود ثغرات أمنية ومعالجتها لتوفير بيئة آمنة.

- اختبارات حماية الخصوصية (Privacy Testing):

التأكد من حماية بيانات المستخدمين الشخصية وفقاً للمعايير العالمية للخصوصية.

الخلاصة:

في هذا البحث، تم استعراض الخطوات والمراحل الأساسية لتطوير تطبيق معالجة الصور باستخدام التقنيات الحديثة مثل الذكاء الاصطناعي، والتعلم العميق. تمت مناقشة الأهداف التي تشمل تطوير تطبيق فعال، وتحسين دقة المعالجة، وسرعة الاستجابة، مع مراعاة سهولة الاستخدام والخصوصية للمستخدمين.

تم تناول المشاكل والتحديات التي تواجه تطبيقات معالجة الصور، مثل التعرف على الأنماط، واكتشاف الكائنات، وتحسين جودة الصور، وتم اقتراح حلول تقنية تعتمد على الشبكات العصبية العميقة، وخوارزميات التعلم الآلي، بالإضافة إلى اختبار الأداء والتقييم المستمر.

باستخدام هذه المنهجية والأدوات، يمكن تحقيق نتائج معالجة صور متقدم وفعال يلبي احتياجات المستخدمين في مختلف المجالات.

المراجع:

1. كتب ومراجع علمية:

- Digital Image Processing by Rafael C. Gonzalez and Richard E. Woods
- Computer Vision: Algorithms and Applications by Richard Szeliski
- Deep Learning for Computer Vision by Rajalingappaa Shanmugamani

2. مقالات علمية وبحوث:

- IEEE Xplore Digital Library (ieeexplore.ieee.org)
- ResearchGate (researchgate.net)
- SpringerLink (link.springer.com)

3. مواقع ومنصات تعليمية:

- Coursera (coursera.org): دورات حول معالجة الصور والذكاء الاصطناعي
- edX (edx.org): دورات تعليمية مفتوحة عبر الإنترنت في مجال معالجة الصور
- Udacity (udacity.com) – دورات تدريبية على التعلم العميق ومعالجة الصور

4. أدوات ومكتبات مفتوحة المصدر:

- OpenCV Documentation (docs.opencv.org)
- TensorFlow Documentation (tensorflow.org)
- PyTorch Documentation (pytorch.org)

5. مؤتمرات وندوات علمية:

- CVPR (Computer Vision and Pattern Recognition)
- ICCV Conference (International Conference on Computer Vision)
- ECCV Conference (European Conference on Computer Vision)

الدراسات السابقة:

شهدت السنوات الأخيرة تطوراً كبيراً في تطبيقات تعديل الصور، وذلك بسبب الانتشار الواسع للهواتف الذكية ووسائل التواصل الاجتماعي، مما أدى إلى زيادة الطلب على أدوات تتيح للمستخدمين تعديل صورهم بسهولة وسرعة دون الحاجة إلى مهارات احترافية في التصميم..

دراسة عبد الرحمن (2018): تصميم محرر صور بسيط باستخدام Canvas API

ركزت هذه الدراسة على بناء تطبيق بسيط يسمح بتعديل الصور داخل المتصفح باستخدام واجهة Canvas التابعة لـ HTML5.قدم الباحث أموراً مثل:

- البحث والفيديو
- تغيير الألوان إلى أبيض وأسود
- إضافة نصوص على الصور

نتائج الدراسة: أظهرت نتائج الدراسة من حيث أداء التطبيق أنه استطاع أن يظهر النتائج المطلوبة، وتم اختبار التطبيق على متصفحي كروم وفايرفوكس.

دراسة سليمان (2019): فعالية تطبيقات تعديل الصور في تحسين جودة الصور الرقمية

تناولت هذه الدراسة تطبيقات مثل Photoshop Express وSnapseed، وركزت على واجهة المستخدم وسهولة الاستخدام، وخلصت إلى أن هذه التطبيقات تُحسن جودة الصور بشكل ملحوظ.

دراسة أحمد وأكون (2020): فعالية استخدام تطبيقات ويب لتصميم القصص الرقمية باستخدام HTML5 وJavaScript

ناقشت الدراسة استخدام أدوات مثل Fabric.js وStoryboard That، وأبرزت دورها في تصميم القصص الرقمية وتأثيرها على تحفيز الطلاب وتحسين نتائج التعلم.

دراسة الشمري (2021) بعنوان: مقارنة بين أدوات تعديل الصور في تطبيقات الويب

تناولت هذه الدراسة مقارنة بين أدوات تعديل الصور مثل Cropper.js وPhotopea وPintura، حيث تتميز هذه الأدوات بواجهات بسيطة وتحقق تفاعلاً فعالاً دون الحاجة إلى خادم.

دراسة الهاشمي (2022) بعنوان: تطوير تطبيق ويب تفاعلي لتعديل الصور باستخدام إطار React.js

ركزت هذه الدراسة على تطوير تطبيق ويب لتعديل الصور باستخدام إطار React.js، حيث يوفر للمستخدمين طريقة سهلة وتفاعلية لتعديل الصور.

من بين الميزات التي تم تضمينها:

- قص الصور وقلبها
- استخدام مرشحات CSS
- حفظ الصورة المعدلة بصيغة PNG

التوصيات: أوصى الباحث بتوسيع الوظائف في المستقبل لتشمل الذكاء الاصطناعي، مثل إزالة الخلفيات أو تحسين الصور تلقائيًا.

<u>والآن سنتحدث بالتفصيل عن هذه الدراسات:</u>

دراسة عبد الرحمن (2018): "تصميم محرر صور بسيط باستخدام Canvas API"

تناولت هذه الدراسة إنشاء تطبيق ويب بسيط لتحرير الصور باستخدام واجهة Canvas API الخاصة بـ HTML5، دون الاعتماد على خادم خارجي. وركزت الدراسة على تصميم واجهة مستخدم بسيطة وسهلة الاستخدام، بالإضافة إلى توفير وظائف تحرير الصور الأساسية مثل القص والتدوير وتغيير الحجم.

<u>المنهج المستخدم:</u>

اعتمد الباحث على المنهج التطبيقي من خلال تصميم وتنفيذ محرر صور باستخدام لغات HTML، CSS، وJavaScript مع التركيز على استخدام عنصر <canvas> لرسم الصورة وتطبيق التعديلات عليها.

<u>أبرز وظائف التطبيق:</u>

1. **أدوات تعديل أساسية:**

تتضمن الوظائف الأساسية: التدوير، تغيير الحجم (Resize) للصورة، بالإضافة إلى إمكانية القص (Crop).

2. **التأثيرات اللونية:**

قدم التطبيق مجموعة من التأثيرات اللونية مثل الرمادي (Grayscale)، والفلاتر الدافئة (Warm Filter)، ومعالجة بكسل.

3. إمكانية الحفظ:

أتاح التطبيق حفظ الصورة المعدلة على الجهاز بصيغة PNG باستخدام canvas.toDataURL().

نتائج الدراسة:

أظهرت التجربة أن التطبيق يعمل بكفاءة على معظم المتصفحات الحديثة (مثل Chrome وFirefox)، كما أنه لا يتطلب اتصالاً بالإنترنت، مما يتيح للطلاب العمل في بيئة آمنة.

توصيات الدراسة:

يوصى الباحث باستخدام تقنيات الويب القياسية مثل Canvas API في المشاريع التعليمية والتدريبية، نظراً لسهولة استخدامها وسهولة التعامل معها، مع إمكانية تطويرها لاحقاً لاستخدام ميزات إضافية.

دراسة سليمان (2019): فعالية تطبيقات تعديل الصور في تحسين جودة الرقمية

هدفت هذه الدراسة إلى تقييم فعالية بعض تطبيقات تعديل الصور الشائعة في تحسين جودة الصور الرقمية. استخدم الباحث المنهج الوصفي التحليلي، حيث قام بجمع صور رقمية قبل وبعد تعديلها باستخدام تطبيقات مختلفة، ثم قام بتحليل الفروقات في الجودة باستخدام معايير محددة.

استخدم الباحث المنهج الوصفي التحليلي، حيث قام بجمع صور رقمية قبل وبعد تعديلها باستخدام تطبيقات مختلفة، ثم قام بتحليل الفروقات في الجودة باستخدام معايير محددة.

من التطبيقات التي تم تحليلها في الدراسة:

Photoshop Express، Snapseed، VSCO لتعديل مجموعة من الصور المختلفة.

أهم المحاور التي ركزت عليها الدراسة:

1. سهولة الاستخدام:

أظهرت النتائج أن غالبية المستخدمين فضلوا التطبيقات التي تحتوي على واجهة مستخدم بسيطة ومنظمة، وكان Snapseed هو الأعلى تقييمًا من حيث سهولة الوصول للأدوات وتوضيح طريقة استخدامها.

2. جودة التعديلات:

بيّنت الدراسة أن الأدوات المتعلقة بتعديل الإضاءة والحدة (Sharpness) كانت الأكثر استخدامًا، وأوضح عدد كبير من المستخدمين أن هذه الأدوات ساهمت في تحسين مظهر الصورة بشكل ملحوظ، خاصة الصور التي تم التقاطها في ظروف إضاءة ضعيفة.

3. حفظ الصور:

تفوق تطبيق VSCO من حيث سرعة التطبيق أثناء حفظ الصور، بينما احتاج تطبيق Photoshop Express وقتًا أطول عند حفظ الصور، خاصة عند زيادة حجم الصورة أو دقتها.

4. رضا المستخدم:

خلصت الدراسة إلى أن المستخدمين يفضلون التطبيقات التي توازن بين سهولة الاستخدام وفعالية التعديلات دون الحاجة إلى مهارات متقدمة، كما أن الأداء السريع يلعب دوراً مهماً في رفع معدل الرضا.

توصيات الدراسة:

أوصى الباحث بضرورة تبسيط واجهات تطبيقات تعديل الصور مع الحفاظ على درجة التعيين للاستخدام، كما يجب أن تكون جاهزة وتقدم تجربة استخدام سلسة، خاصة في تطبيقات الويب التي تستهدف المستخدمين المبتدئين.

دراسة أحمد وآخرون (2020): تطبيق ويب لتعديل الصور باستخدام تقنيات HTML5 و JavaScript

هدفت هذه الدراسة إلى تطوير تطبيق ويب بسيط لتعديل الصور باستخدام تقنيات الويب الحديثة مثل HTML5 وJavaScript. تم استخدام مكتبة Fabric.js في تطوير واجهة المستخدم. ركز المشروع على تعديل الصور بطريقة سهلة وسريعة باستخدام واجهة مستخدم بسيطة.

المنهج المستخدم:

اعتمد الباحثون على المنهج التطبيقي، حيث تم تصميم وتطوير التطبيق باستخدام JavaScript ومكتبة Fabric.js، وتم التركيز على سهولة الاستخدام في واجهة المستخدم.

أبرز الوظائف التي وفرها التطبيق:

1. **القص والتدوير:**

تم توفير أدوات لتحديد جزء من الصورة وقصه، بالإضافة إلى إمكانية تدوير الصورة بزوايا مختلفة.

2. **التحكم في السطوع والتباين:**

تم تطوير واجهة تسمح للمستخدم بالتحكم بدرجة السطوع والتباين للصورة باستخدام بحركات مرئية.

3. **إضافة فلاتر بصرية:**

تم توفير فلاتر بصرية مثل الأبيض والأسود، وتلاشي (Fade)، والتي تم تطبيقها باستخدام أكواد Canvas API الخاصة بـ HTML5.

نتائج الدراسة:

أظهرت نتائج التجارب أن التطبيق يعمل بشكل فعال على الصفحات المختلفة، ويتيح للمستخدمين التفاعل مع الصور بطريقة مرئية وسهلة، مما يوفر تجربة مستخدم محسنة.

مزايا التطبيق:

تميز التطبيق ببساطة التصميم وسرعة التنفيذ، مما جعله مناسبًا للاستخدام في بيئات تعليمية وتدريبية.

توصيات للمطورين:

أوصى الباحث باستخدام مكتبات JavaScript مفتوحة المصدر لتقليل وقت التطوير وتحسين الأداء، مع ضرورة الحفاظ على واجهة استخدام سهلة ومناسبة للمستخدمين المبتدئين.

دراسة الشمري (2021):

تناولت الدراسة مقارنة بين أدوات تعديل الصور في تطبيقات الويب، مثل Photopea وCropper.js وPintura، حيث تم وصف هذه الأدوات بأنها مفتوحة المصدر ويمكن دمجها بسهولة في تطبيقات الويب لتعزيز قدرات تعديل الصور.

منهج الدراسة:

اتبع الباحث المنهج المقارن، حيث تم تحليل كل مكتبة بناءً على مجموعة من المعايير شملت:

- واجهة المستخدم
- استجابة التطبيق
- جودة التعديلات
- التوافق مع المتصفحات

نتائج المقارنة:

1. Cropper.js

مكتبة متخصصة في أدوات القص، وقد تميزت بسهولة دمجها في صفحات الويب، لكنها محدودة من حيث الوظائف، إذ تتيح فقط الفلاتر أو التحكم في الألوان.

2. Pintura

مكتبة احترافية تحتوي على أدوات متقدمة لتحرير الصور لكنها مدفوعة وغير مفتوحة المصدر مما يقلل من إمكانية استخدامها في مشاريع المصدر المفتوح.

3. Photopea

يعد تطبيقاً متقدماً يحاكي بيئة Photoshop داخل المتصفح، وهو مجاني ويدعم ملفات PSD، لكنه أكثر تعقيداً من التطبيقات السابقة ويحتاج إلى خبرة أكثر.

توصيات الدراسة:

ينصح الباحث باستخدام التطبيقات المجانية مفتوحة المصدر لتصميم صور تخرج إلى تطبيقات الويب، مثل تطبيق Photopea، وذلك لتوفير التكاليف، كما ينصح باستخدام مكتبة Pintura في المشاريع التجارية، لما تحتويه من أدوات احترافية تسهل من إدخال الصور وتحريرها.

دراسة الهاشمي (2022): تطوير تطبيق تفاعلي لتعديل الصور باستخدام React.js

هدفت هذه الدراسة إلى تصميم وبناء تطبيق ويب تفاعلي لتعديل الصور باستخدام إطار العمل الحديث React.js، مع التركيز على تحسين تجربة المستخدم من خلال واجهة تفاعلية مرنة وسريعة الاستجابة.

المنهج المستخدم:

اعتمد الباحث على المنهج التجريبي، حيث تم بناء التطبيق باستخدام React.js مع دمج بعض مكتبات التعديل مثل react-image-editor وdropzone لتسهيل عملية رفع الصور وتعديلها.

أهم خصائص التطبيق:

1. واجهة تفاعلية سهلة:

تم بناء التطبيق باستخدام مكونات (Components) من React لبناء واجهة منظمة تتيح للمستخدم سهولة الاستخدام وإظهارها عند عملية التعديل على الصور.

2. فلاتر تعديل جاهزة:

تم إضافة فلاتر تعتمد على CSS مثل الألوان الحادة، وتغبيش الصورة (Blur)، ويمكن للمستخدم تطبيقها مباشرة على الصورة.

3. الحفظ والمعاينة المباشرة:

يتيح التطبيق للمستخدم معاينة الصورة قبل حفظها، مع إمكانية تحميل الصورة المعدلة بصيغة PNG أو JPEG.

نتائج الدراسة:

أظهرت نتائج الدراسة أن استخدام React.js ساهم في تحسين تجربة المستخدم وسرعة استجابة الواجهة، كما نجح التطبيق في معالجة التحديات اللحظية في واجهة الاستخدام.

يوصي الباحث بتوسيع التطبيق ليشمل ميزات إضافية مثل إزالة الخلفية وتحسين الجودة باستخدام تقنيات الذكاء الاصطناعي.

كما يقترح استخدام التطبيق في مجالات أخرى مثل التصوير الطبي وتحليل الشيفرة الوراثية.

تاريخ معالجة الصور:

المقدمه:

معالجة الصور هي تقنية تستخدم لتحليل وتعديل وتحسين الصور الرقمية باستخدام الخوارزميات والبرمجيات. تعد جزءًا من الذكاء الاصطناعي ورؤية الحاسوب، وتُستخدم في العديد من المجالات مثل التصوير الطبي، والتعرف على الوجوه، وتحسين جودة الصور الفوتوغرافية، وأتمتة عمليات التصوير الصناعي.

تشمل معالجة الصور عدة خطوات، مثل:

1. تحويل الصورة: يمكن تحويل الصور إلى درجات رمادية أو تنسيق معين يسهل معالجتها.
2. تحسين الصورة: يشمل تحسين التباين، إزالة الضوضاء، وضبط الألوان.
3. استخراج الميزات: مثل تحديد الحواف، التعرف على الأشكال، وتحليل المكونات داخل الصورة.
4. التعرف على الأنماط: يستخدم في التطبيقات الذكية مثل التعرف على الأوجه أو النصوص في الصور.

تاريخ معالجة الصور:

تعود معالجة الصور إلى بدايات القرن العشرين، حيث بدأت باستخدام تقنيات التصوير الضوئي والكهربائي. في عام 1920، تم إرسال الصور رقميًا عبر الكابلات البحرية بين لندن ونيويورك، مما كان أحد التطبيقات الأولى للصور الرقمية (1) ومع تطور الحواسيب في الأربعينيات، بدأت معالجة الصور تأخذ منحى أكثر تطورًا، حيث تم تحسين الصور باستخدام تقنيات رقمية.

في الستينيات، استخدمت معالجة الصور في التطبيقات الفضائية، مثل تحسين الصور المرسلة من المركبات الفضائية، كما حدث مع مركبة *Ranger 7* عام 1964وفي السبعينيات، دخلت معالجة الصور مجال التصوير الطبي، حيث تم تطوير تقنية التصوير المقطعي المحوسب (CT)، والتي حصل مبتكروها على جائزة نوبل عام1979 (1)

اليوم، أصبحت معالجة الصور جزءًا أساسيًا من الذكاء الاصطناعي ورؤية الحاسوب، وتستخدم في مجالات مثل التعرف على الوجوه، التصوير الطبي، وتحليل الصور الصناعية.

أساليب معالجة الصور:

معالجة الصور تتضمن العديد من الأساليب والتقنيات لتحسين الصور أو تحليلها. إليك بعض الأساليب الشائعة:

1. تحسين الصور: يشمل ضبط التباين، السطوع، التشبع، وإزالة الضوضاء لتحسين جودة الصورة.
2. تحويل الصور: مثل تغيير الحجم، التدوير، القص، وتحويل الصور إلى تدرجات رمادية.

3. استخراج الميزات: مثل اكتشاف الحواف باستخدام خوارزميات مثل Canny، أو التعرف على الأنماط.
4. تقنيات الفلترة: مثل الفلاتر التمويجية (Gaussian blur) لإزالة الضوضاء، أو الفلاتر الحادة لتحسين التفاصيل.
5. تحليل الصور: يشمل التقسيم (Segmentation) لتحديد الأجزاء المهمة في الصورة، والتعرف على الأشكال.
6. معالجة الصور بالألوان: تعديل القنوات اللونية واستخدام نماذج ألوان مثل RGB أو HSV لتحليل الصورة.
7. التعلم العميق: مثل الشبكات العصبية لاكتشاف الأنماط والتعرف على الأشياء داخل الصور.

أحدث التقنيات في معالجة الصور:

هناك العديد من التقنيات الحديثة التي تُحدث ثورة في مجال معالجة الصور، ومن أبرزها:

1. الذكاء الاصطناعي والتعلم العميق: تستخدم الشبكات العصبية العميقة مثل CNN وGAN لتحليل الصور، تحسينها، وإنشاء صور واقعية تماماً.
2. التصوير الفوتوغرافي بالذكاء الاصطناعي: تعتمد الكاميرات الحديثة على الذكاء الاصطناعي لضبط الإعدادات تلقائيًا وتحسين جودة الصور بناءً على المشهد.
3. التعرف الذكي على المشهد: تقنية تمكن الكاميرات من فهم المشهد تلقائيًا وضبط الإعدادات مثل الإضاءة والتركيز للحصول على أفضل صورة ممكنة
4. تحليل الصور الطبية يستخدم الذكاء الاصطناعي في تحليل صور الأشعة السينية والرنين المغناطيسي لتشخيص الأمراض بدقه وسرعة.
5. التصوير بالطائرات بدون طيار: يتيح التقاط صور جوية مذهلة باستخدام الذكاء الاصطناعي لتحديد الأهداف وتتبعها تلقائيًا (2).
6. تحسين الصور باستخدام الذكاء الاصطناعي: يمكن إزالة الضوضاء، تحسين الألوان، وضبط التفاصيل تلقائيًا باستخدام تقنيات الذكاء الاصطناعي المتقدمة (1).

دور الذكاء الاصطناعي في معالجة الصور:

يلعب الذكاء الاصطناعي دورًا محوريًا في معالجة الصور، حيث يتيح تحسين الجودة، التعرف على الأنماط، وتحليل المحتوى بطرق أكثر دقة وكفاءة، إليك بعض أبرز تطبيقاته:

1. التعرف على الصور وتصنيفها: تستخدم خوارزميات الذكاء الاصطناعي التعلم العميق لفهم الأنماط داخل الصور، مما يساعد في تصنيف الأشياء بدقة عالية.
2. تحسين جودة الصور: يمكن للذكاء الاصطناعي تحسين الصور منخفضة الدقة باستخدام تقنيات مثل الدقة الفائقة وتقليل الضوضاء، وهو أمر بالغ الأهمية في التصوير الطبي وصور الأقمار الصناعية

3. التعرف على الوجه والقياسات الحيوية: تُستخدم هذه التقنية في الأمن والمراقبة، حيث يمكن للذكاء الاصطناعي التعرف على الأفراد بدقة عالية.
4. تقطيع الصورة: يساعد الذكاء الاصطناعي في تقسيم الصور إلى أجزاء ذات معنى، مما يسهل تحليلها في تطبيقات مثل التصوير الطبي والمركبات ذاتية القيادة.
5. الشبكات العصبية التوليدية (GANs): تتيح هذه التقنية إنشاء صور واقعية غير موجودة في العالم الحقيقي، مما يفتح آفاقً جديده والتصميم وإنشاء المحتوى.
6. *المعالجة في الوقت الحقيقي*: مع تطور الأجهزة والخوارزميات، أصبح بالإمكان معالجة الصور بسرعة فائقة، مما يفيد في تطبيقات مثل بث الفيديو والمراقبة والواقع المعزز.

الفئات الرئيسية للمستخدمين المحتملين:

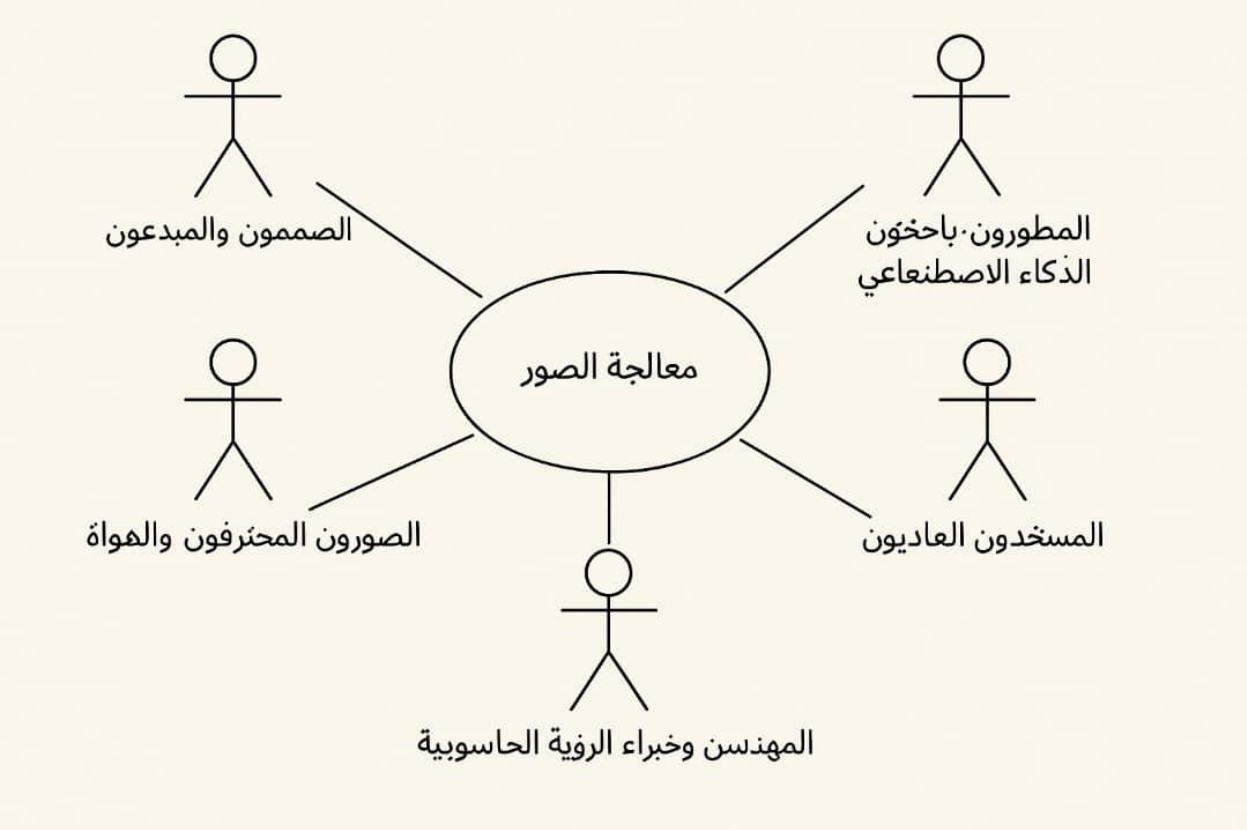
1. **المصممون والمبدعون**
   - يستخدمون التطبيق لتحرير الصور وتحسين جودتها .
   - يطبقون الفلاتر والتأثيرات لإنشاء صور جذابة .

2. **المصورون المحترفون والهواة**
   - يقومون بتعديل الصور لضبط الألوان والإضاءة .
   - يحتاجون إلى أدوات متقدمة مثل تصحيح الحواف وتحسين التفاصيل .

3. **المطورون والباحثون في الذكاء الاصطناعي**
   - يستخدمون التطبيق لتدريب نماذج تعلم الآلة، مثل تصنيف الصور وتحليلها .
   - يدمجون تقنيات مثل *GANs* و *CNNs* لمعالجة الصور الذكية .

4. **المهندسون وخبراء الرؤية الحاسوبية**
   - يستخدمون التطبيق في تطبيقات متقدمة مثل التعرف على الأجسام والكشف عن الأنماط .
   - يمكن أن يكون جزءًا من مشاريع تحليل الصور الطبية أو الصناعية .

5. **المستخدمون العاديون**
   - يعدلون الصور الشخصية بسهولة، مثل إزالة الخلفية أو تحسين الجودة.
   - يضيفون تأثيرات ممتعة لمشاركتها عبر وسائل التواصل.



عند فتح البرنامج تمر العملية عادةً بعدة خطوات منظمة لتقديم تجربة مستخدم فعالة. إليك تسلسلًا مبسطًا للخطوات التي يقوم بها التطبيق:

مخطط خطوات فتح البرنامج:

1. **تهيئة الواجهة (Initialize UI):**
   - يتم تحميل العناصر الرسومية مثل الأزرار والقوائم.

2. **تحميل الموارد (Load Resources):**
   - يتأكد البرنامج من توفر الأدوات، الفلاتر، والنماذج) إن وُجدت مثل GAN أو.(CNN

3. **تسجيل الدخول أو التعريف بالمستخدم (Authentication):**
- إذا كان التطبيق يتطلب حسابًا، يتم التحقق من بيانات المستخدم .
- بناءً على نوع المستخدم (مصمم، مطور، زائر... إلخ) ، يتم تخصيص الأدوات المتاحة.

4. **فتح الصورة أو البدء بمشروع جديد (Image Selection):**
- يمكن للمستخدم اختيار صورة من الجهاز أو البدء بملف فارغ.

5. **إظهار أدوات التحرير والمعالجة (Editing Tools Ready):**
- كأدوات الاقتصاص، الفلاتر، تحسين الألوان، اكتشاف الحواف... وغيرها.

6. **تفعيل الأحداث والعمليات (Activate Event Listeners):**
- مثل السحب، الحفظ، الإلغاء، معاينة النتائج، التراجع.

7. **جاهزية التعديل (Ready to Edit!):**
- ينتظر البرنامج تفاعل المستخدم.


عند فتح البرنامج
تهيئة الواجهة
تسجيل الموارد
العريف بالمستخدم
ق فتح الصوره أو
بابدء بمشروع جديد
إفهار ادوات التحرير
والمعالجة
Editing Tools
Ready
جاهزية التعديل!

لتوضيح الفرق في الأهداف والوظائف التي يؤديها كل منهما:

- **المصممون**

الهدف: إنتاج أعمال مرئية احترافية وإبداعية عالية الجودة.

الأنشطة:

- تصميم صور دعائية، إعلانات، شعارات أو منشورات سوشال ميديا.
- استخدام الطبقات (Layers) والأقنعة (Masks) لتعديل أجزاء محددة من الصورة.
- التحكم المتقدم بالألوان، الإضاءة، والتدرجات.
- إنشاء ملفات قابلة للطباعة أو التصدير بصيغ عالية الدقة) مثلTIFF ، .(PSD
- استخدام أدوات متقدمة مثل البن تول(Pen Tool) ، التسييح(Liquify) ، وتصحيح العدسة.

الصلاحيات:

- الوصول الى أدوات احترافية مخصصة
- إمكانية العمل بدقة بكسلية عالية والتحكم الكامل بالتنسيقات.

- **المستخدمون العاديون:**

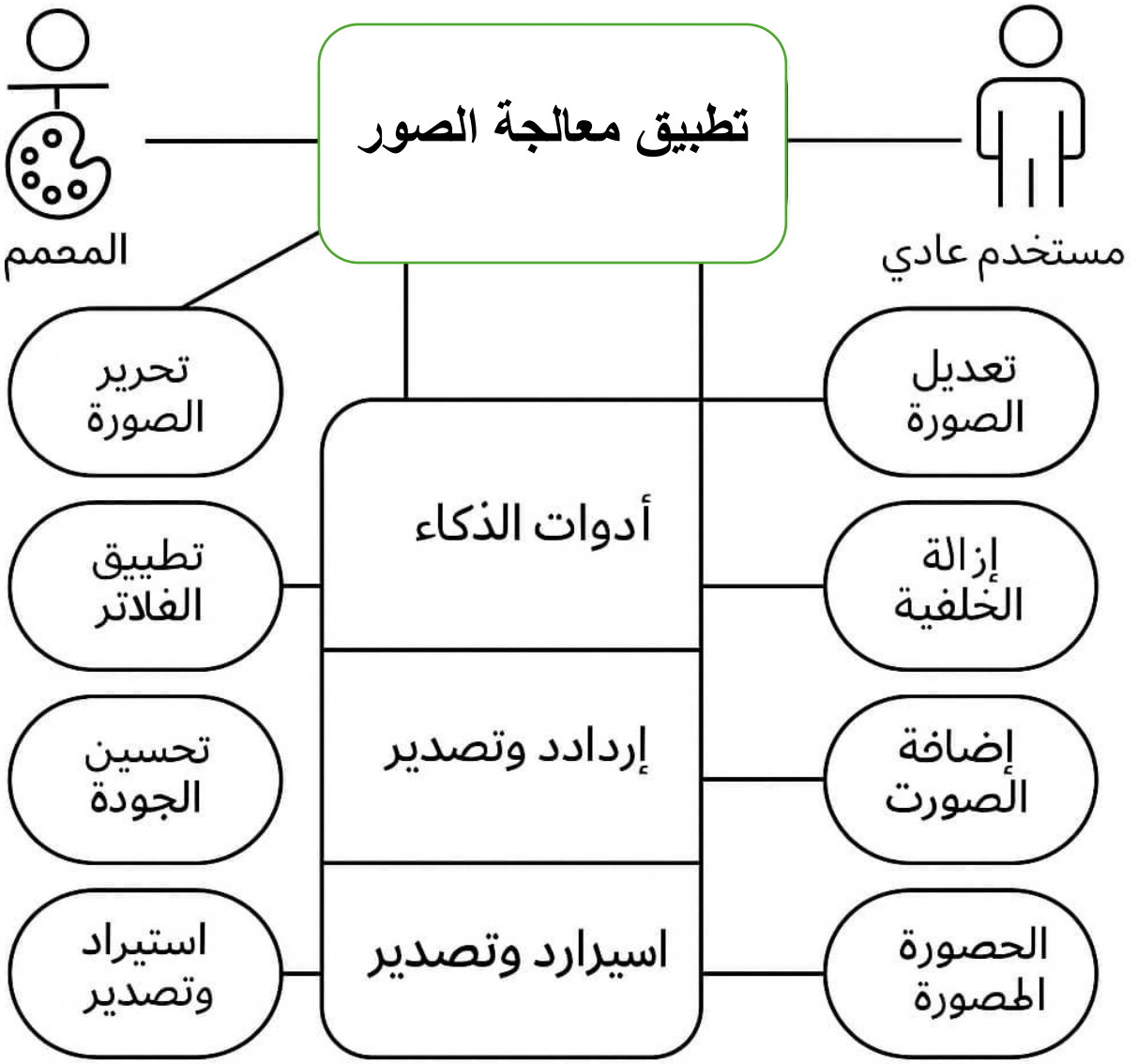
الهدف: تحسين الصور الشخصية أو اليومية بطريقة سريعة وبسيطة.

الأنشطة:

- تطبيق فلاتر جاهزة وتحسين الصورة تلقائيًا.
- تعديل الألوان السطحية مثل السطوع والتباين والتشبع.
- إزالة الخلفية أو العيوب البسيطة.
- قص وتدوير الصور وتغيير الأبعاد.
- مشاركة الصور مباشرة على وسائل التواصل الاجتماعي.

الصلاحيات:

- استخدام واجهات سهلة وسريعة الاستجابة .
- وصول محدود للأدوات المعقدة لكن مدعوم بخيارات ذكية أو تلقائية.



- **استخدام كل أداة أساسية داخل التطبيق مقسّمة حسب الفئة الوظيفية:**

أدوات التعديل الأساسية:

- *القص * :(Crop) لتعديل أبعاد الصورة أو التركيز على جزء معيّن.
- *التدوير / الانعكاس * :(Rotate / Flip) لتصحيح اتجاه الصورة أو إبداع تأثير بصري مختلف.
- *السطوع والتباين * :(Brightness & Contrast) للتحكم في الإضاءة والظلال.
- *التشبع والحدة * :(Saturation & Sharpness) لتحسين الألوان ووضوح التفاصيل.

أدوات التصميم والتحسين الفني:

- *الفلاتر *:(Filters) مجموعة تأثيرات جمالية جاهزة.
- *الفرشاة *:(Brush) لتلوين أو تعديل مناطق معينة يدويًا.
- *الطبقات *:(Layers) تُستخدم لتنظيم العناصر وتعديلها بدون التأثير على باقي الصورة.
- *أدوات النص *:(Text Tool) لإضافة كلمات أو شعارات مع خيارات الخط والتنسيق.

أدوات المعالجة الذكية:(AI Tools)

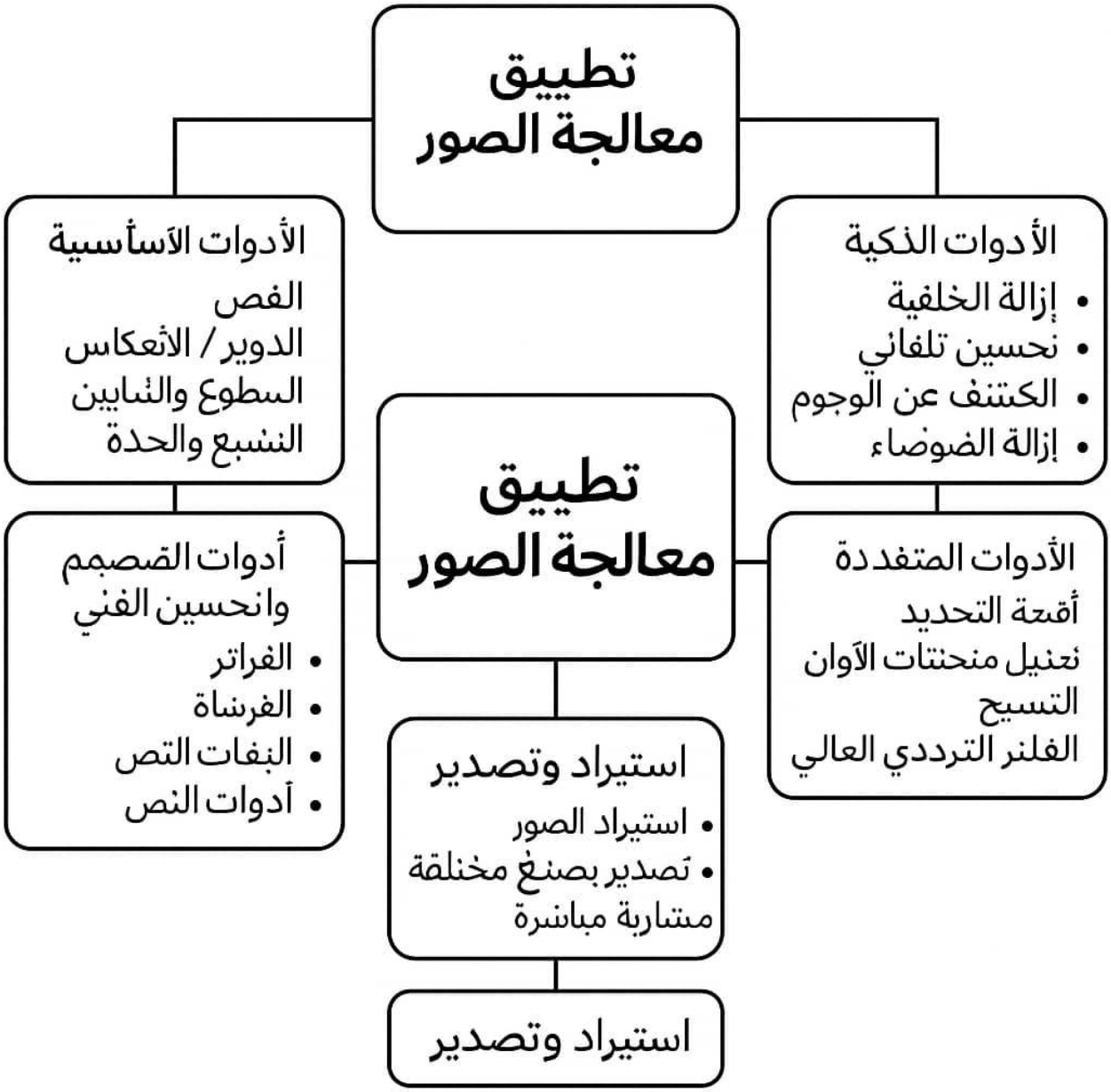
- *إزالة الخلفية *:(Background Remover) تحدد وتزيل الخلفيات تلقائيًا.
- *تحسين تلقائي *:(Auto Enhance) الذكاء الاصطناعي يضبط الألوان والإضاءة تلقائيًا.
- *الكشف عن الوجوه / العيون:* يُستخدم لتعديل البشرة أو تلميع العيون.
- *إزالة الضوضاء *:(Denoise) يحسّن جودة الصور ذات الإضاءة المنخفضة.

أدوات متقدمة للمحترفين:

- *أقنعة التحديد *:(Masks) لتطبيق تأثيرات فقط على أجزاء محددة من الصورة.
- *تعديل منحنيات الألوان *:(Curves) تحكم دقيق في توزيع الإضاءة والألوان.
- *التسييح *:(Liquify) لتغيير شكل العناصر في الصورة بسلاسة.
- *الفلتر الترددي العالي *:(High-Pass Filter) لزيادة الحدة بدون فقد التفاصيل.

أدوات الاستيراد والتصدير:

- *استيراد الصور من الكاميرا أو السحابة *.
- *تصدير بصيغ مختلفةJPG) ، PNG، PSD، *(TIFF
- *مشاركة مباشرة على وسائل التواصل مثل Instagram أو*.Facebook


تطبيق
معالجة الصور
الأدوات الأساسية
الفص
الدوير / الأنعكاس
السطوع والتباين
التشبع والحدة
الأدوات الذكية
• إزالة الخلفية
• تحسين تلقائي
• الكشف عن الوجوم
• إزالة الضوضاء
تطبيق
معالجة الصور
أدوات التصحيم
وانحسين الفني
• الفراتر
• الفرشاة
• النفات التص
• أدوات النص
الأدوات المتفددة
أقمة التحديد
تعديل منحنيات الألوان
التسيح
الفلتر الترددي العالي
استيراد وتصدير
• استيراد الصور
• تصدير بصنغ مختلفة
متتاربة مباشرة
استيراد وتصدير

التصميم والتطبيق العملي لتطبيق معالجة الصور:

مقدمة :

يتناول هذا الباب تحليلًا تفصيليًا لتطبيق عملي تم تطويره باستخدام بيئة *Android Studio* ولغة *Java*، بهدف معالجة الصور رقمياً من خلال واجهة تفاعلية. يتيح التطبيق للمستخدم تعديل خصائص الصورة مثل السطوع، التباين، التشبع، التدوير، إضافة النصوص، وتطبيق الفلاتر، بالإضافة إلى حفظ الصورة المعدلة في معرض الجهاز.

وصف واجهة المستخدم:

تم تصميم واجهة التطبيق لتكون تفاعلية وسهلة الاستخدام، وتحتوي على العناصر التالية:

- لعرض الصورة الأصلية والمعدلة :ImageView .
- للتحكم في السطوع، التباين، التشبع، والتدوير:SeekBars
- لاختيار نوع الفلتر (None، Grayscale،Sipa) :Spinner
- لإضافة نص وحفظ الصورة :Buttons

الشكل (1): واجهة تعديل الخصائص البصرية

الصورة توضح واجهة التطبيق التي تحتوي على منزلقات للتحكم في السطوع(Brightness) ، التباين (Contrast)، والتشبع(Saturation) ، بالإضافة إلى قائمة الفلاتر. تظهر أيضًا أيقونات الحفظ والتراجع في الأسفل.

الوظائف الأساسية للتطبيق:

تحميل الصورة

يتم تحميل صورة من مجلد الموارد (drawable) باستخدامBitmapFactory ، وتُخزن في متغيرين : originalBitmapكمرجع ثابت، و `currentBitmap`لتطبيق التعديلات عليه.

التدوير

يتم تدوير الصورة باستخدام مصفوفة Matrix بناءً على زاوية يحددها المستخدم عبر منزلق(SeekBar) ، مما يسمح بتعديل اتجاه الصورة.

تعديل السطوع والتباين والتشبع

يتم تعديل هذه الخصائص باستخدامColorMatrix ، حيث يتم ضرب مصفوفة لونية في بيانات الصورة لتغيير الإضاءة والحدة وكثافة الألوان.

الشكل (2): واجهة التحكم في التعديلات

الصورة توضح منزلقات التعديل التي تسمح للمستخدم بتغيير القيم البصرية للصورة بشكل مباشر، مع عرض حي للتأثيرات.

تطبيق الفلاتر

يتم تطبيق مصفوفات لونية جاهزة لتحويل الصورة إلى نمط رمادي (Grayscale) أو بني كلاسيكي (Sepia)، مما يضفي طابعًا فنيًا على الصورة.

إضافة النص

يُدخل المستخدم نصًا ويختار لونه عبر منزلقاتRGB ، ثم يُرسم النص على الصورة باستخدام Canvas و`Paint`.

الشكل (3): نافذة إضافة النص

الصورة توضح نافذة منبثقة بعنوان"Add Text" ، تحتوي على حقل إدخال نص ودائرة اختيار اللون، بالإضافة إلى زري "OK" و."CANCEL"

حفظ الصورة

تُحفظ الصورة المعدلة في معرض الجهاز باستخدامMediaStore.Images.Media.insertImage ، مما يُمكّن المستخدم من الاحتفاظ بالتعديلات ومشاركتها.

منطق المعالجة

يعتمد التطبيق على استخدام ColorMatrix لتعديل خصائص الصورة، وهي تقنية تعتمد على ضرب مصفوفة لونية في بيانات الصورة. أما التدوير، فتم باستخدام Matrix لتغيير زاوية العرض. الفلاتر تم تطبيقها عبر مصفوفات جاهزة تحاكي أنماط لونية محددة.

قابلية التوسع

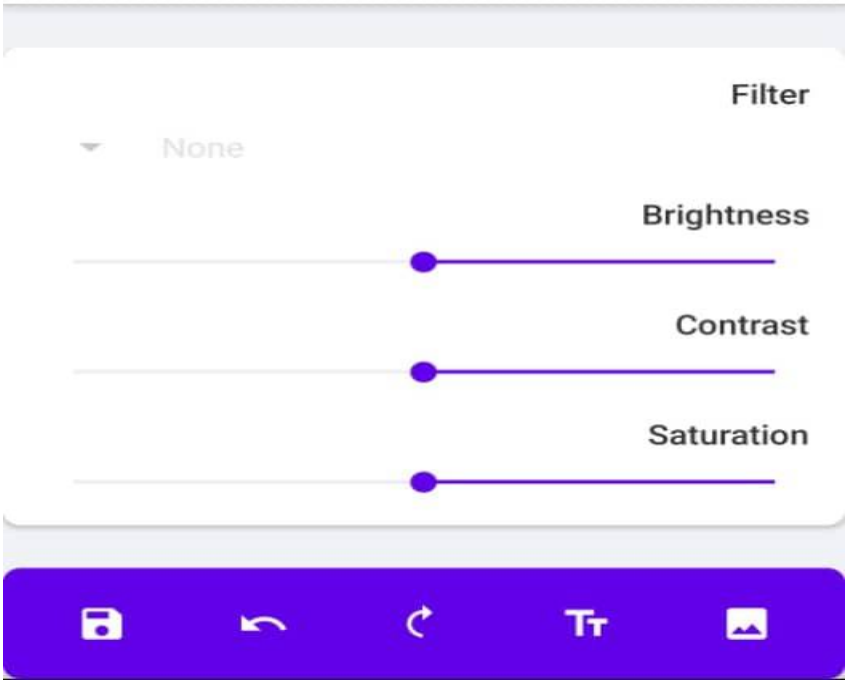
تم تصميم التطبيق بطريقة تسمح بإضافة وظائف مستقبلية بسهولة، مثل:

- إضافة فلاتر متقدمة(Blur ، Sharpen)
- تحديد مكان النص بالسحب
- تحليل الصورة باستخدام خوارزميات ذكاء اصطناعي
- حفظ الصورة بصيغ متعددة(PNG ، JPEG)

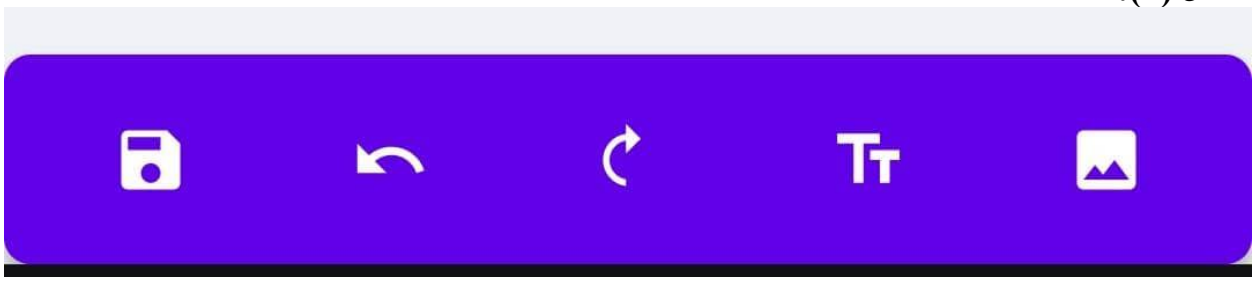
خاتمة الباب

يعكس تصميم هذا التطبيق فهمًا عمليًا لمبادئ معالجة الصور الرقمية، ويُظهر كيف يمكن دمج الوظائف التقنية في واجهة سهلة الاستخدام. تم بناء التطبيق بأسلوب مرن وقابل للتطوير، مما يجعله نموذجًا مناسبًا للأبحاث التطبيقية في مجال الصور.

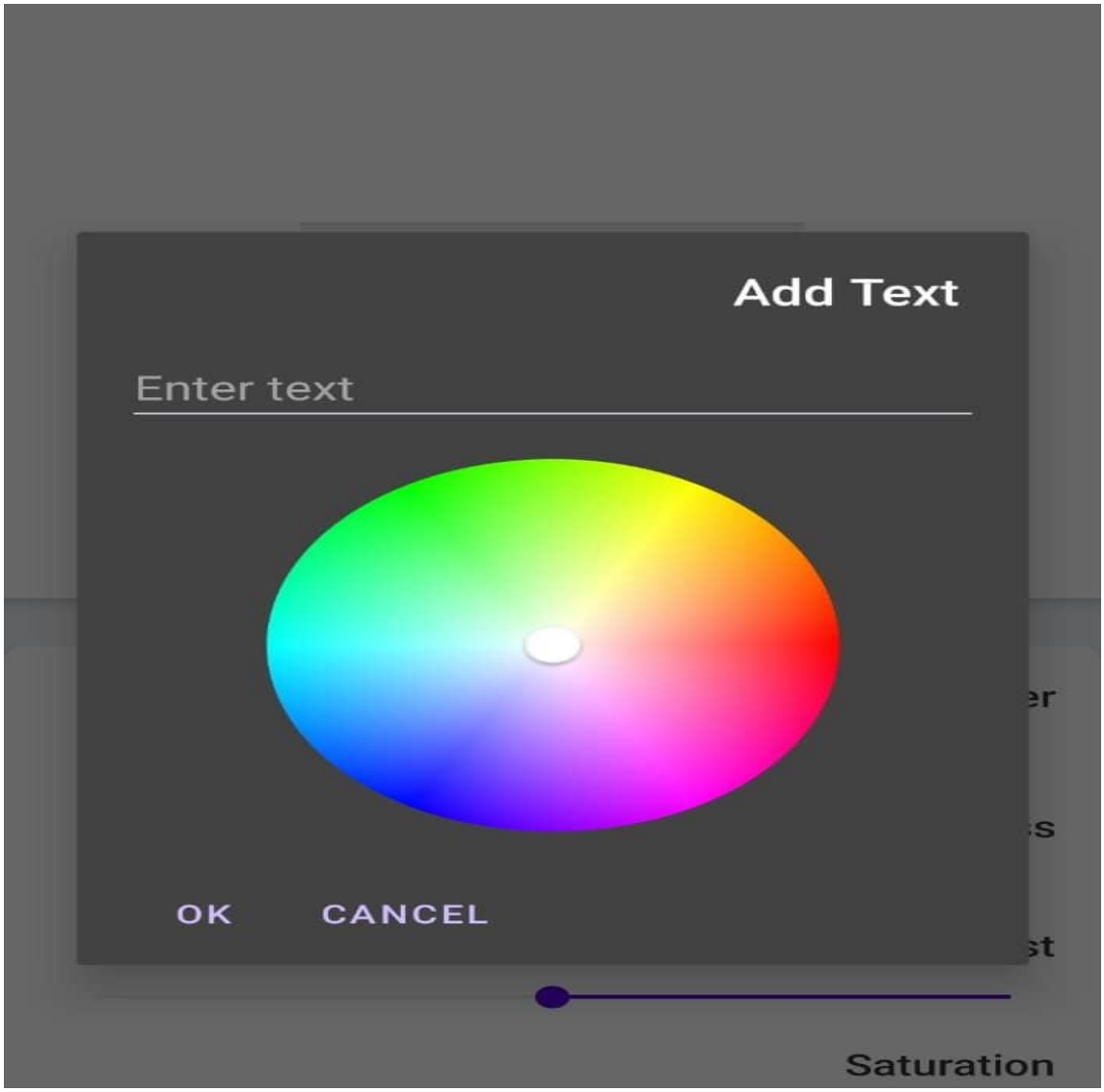
- **الشكل (1):**



- **الشكل (2):**



- **الشكل (3):**



النتائج والتوصيات:

النتائج

في ضوء نتائج تطبيق عمليات تحويل الصور بهدف إلى تقييم أدوات تحرير متقدمة مشابهة لبرنامج الفوتوشوب، تم التوصل إلى النتائج التالية:

١. أظهرت نتائج تطبيق عمليات تحويل الصور أن هذه العمليات أثبتت فعاليتها في اختيار الأدوات المناسبة لتحرير الصور، مما يسهم في تحسين جودة الصور.

٢. بناءً على نتائج التطبيق، تم التوصل إلى أن الأدوات التالية هي الأكثر فعالية في تحرير الصور:

- أداة القص (Crop Tool) لتحسين التكوين العام للصورة.
- أداة التعديل اللوني (Color Adjustment Tool) لتحسين جودة الألوان.
- أداة إزالة العيوب (Healing Tool) لإزالة العيوب والشوائب من الصور.

٣. **واجهة المستخدم:** أظهرت نتائج التقييم أن واجهة المستخدم كانت سهلة الاستخدام، مما يسهل على المستخدمين التعامل مع الأدوات المختلفة.

٤. **الأداء:** أظهرت نتائج التقييم أن الأداء العام للأدوات كان جيدًا، حيث تم تنفيذ العمليات بسرعة وكفاءة.

٥. **التوافق:** أظهرت نتائج التقييم أن الأدوات كانت متوافقة مع أنظمة التشغيل المختلفة، مثل: ويندوز، ماك.

التوصيات:

بناءً على النتائج السابقة، تم التوصل إلى التوصيات التالية:

١. التوصية للاحتفاظ بالأدوات الأكثر فعالية بناءً على نتائج التقييم، مثل أداة القص، أداة التعديل اللوني، وأداة إزالة العيوب.

٢. التوصية بتحسين واجهة المستخدم لتكون أكثر تفاعلية وسهولة في الاستخدام.

٣. التوصية بتحسين الأداء العام للأدوات لتقليل الوقت المستغرق في تنفيذ العمليات.

٤. التوصية بدعم توافق الأدوات مع أنظمة التشغيل المختلفة، مثل macOS و Windows.

مقترحات للدراسات المستقبلية:

بناءً على نتائج الدراسة الحالية، يمكن اقتراح الدراسات التالية:

١. دراسة مقارنة بين أدوات تحرير الصور المختلفة من حيث الفعالية وسهولة الاستخدام.

٢. دراسة تأثير استخدام أدوات تحرير الصور على جودة الصور في مجالات مختلفة، مثل الإعلام، التعليم، والتسويق.

٣. دراسة إمكانية تطوير أدوات تحرير الصور باستخدام تقنيات الذكاء الاصطناعي لتحسين الأداء والجودة

٤. **إضافة ميزات ذكية**: دمج تقنيات الذكاء الاصطناعي لتحسين أدوات مثل إزالة الخلفية أو تصحيح الألوان تلقائيًا لتوفير وقت المستخدم.

٥. **دعم مختلف اللغات:** إضافة دعم للغات متعددة في الواجهة لتجنب مستخدمين من مختلف أنحاء العالم.

٦. **اختبارات موسعة:** إجراء اختبارات إضافية على نطاق أوسع لضمان التوافق مع أجهزة متنوعة وتحسين الأداء في ظروف الاستخدام المكثف.

الملاحق:

```
package com.example.photoeditorapp2;

import android.Manifest;
import android.app.AlertDialog;
import android.content.ContentResolver;
import android.content.ContentValues;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.ColorMatrix;
import android.graphics.ColorMatrixColorFilter;
import android.graphics.Matrix;
import android.graphics.Paint;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.EditText;
```

```
import android.widget.ImageButton;
import android.widget.SeekBar;
import android.widget.Spinner;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import com.skydoves.colorpickerview.ColorPickerView;
import com.skydoves.colorpickerview.listeners.ColorListener;
import ja.burhanrashid52.photoeditor.PhotoEditor;
import ja.burhanrashid52.photoeditor.PhotoEditorView;
import java.io.IOException;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {

private static final int PICK_IMAGE_REQUEST = 1;
private static final int STORAGE_PERMISSION_CODE = 100;
private PhotoEditorView photoEditorView;
private PhotoEditor photoEditor;
private Bitmap originalBitmap;
private float brightness = 0f;
private float contrast = 1f;
private float saturation = 1f;
```

```
    private Spinner filterSpinner;
    private List<Bitmap> undoStack = new ArrayList<>();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        photoEditorView = findViewById(R.id.photoEditorView);
        photoEditor = new PhotoEditor.Builder(this, photoEditorView).build();

        ImageButton selectImageButton = findViewById(R.id.selectImageButton);
        ImageButton addTextButton = findViewById(R.id.addTextButton);
        ImageButton rotateButton = findViewById(R.id.rotateButton);
        ImageButton undoButton = findViewById(R.id.undoButton);
        ImageButton saveButton = findViewById(R.id.saveButton);
        SeekBar brightnessSeekBar = findViewById(R.id.brightnessSeekBar);
        SeekBar contrastSeekBar = findViewById(R.id.contrastSeekBar);
        SeekBar saturationSeekBar = findViewById(R.id.saturationSeekBar);
        filterSpinner = findViewById(R.id.filterSpinner);

        // إعداد (Spinner) لاختيار الفلاتر
        String[] filters = {"None", "Grayscale", "Sepia", "Vintage", "Warm", "Cool",
                "Invert",
                "Posterize"};
        ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
                android.R.layout.simple_spinner_item, filters);
```

```
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
filterSpinner.setAdapter(adapter);

// تطبيق الفلتر تلقائيا عند اختيار (Spinner)
filterSpinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
@Override
public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
applySelectedFilter();
}

@Override
public void onNothingSelected(AdapterView<?> parent) {}
});

selectImageButton.setOnClickListener(v -> checkStoragePermission());
addTextButton.setOnClickListener(v -> showTextInputDialog());
rotateButton.setOnClickListener(v -> rotateImage());
undoButton.setOnClickListener(v -> undo());
saveButton.setOnClickListener(v -> saveImage());

brightnessSeekBar.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
@Override
```

```
public void onProgressChanged(SeekBar seekBar, int progress, boolean
fromUser) {
brightness = (progress - 100) / 100.0f;
applyAdjustments();
}

@Override
public void onStartTrackingTouch(SeekBar seekBar) {}
@Override
public void onStopTrackingTouch(SeekBar seekBar) {}
});

contrastSeekBar.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener()
{
@Override
public void onProgressChanged(SeekBar seekBar, int progress, boolean
fromUser) {
contrast = progress / 100.0f;
applyAdjustments();
}

@Override
public void onStartTrackingTouch(SeekBar seekBar) {}
@Override
public void onStopTrackingTouch(SeekBar seekBar) {}
});
```

```
saturationSeekBar.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener() {
@Override
public void onProgressChanged(SeekBar seekBar, int progress, boolean
fromUser) {
saturation = progress / 100.0f;
applyAdjustments();
}

@Override
public void onStartTrackingTouch(SeekBar seekBar) {}
@Override
public void onStopTrackingTouch(SeekBar seekBar) {}
});
}

private void checkStoragePermission() {
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.TIRAMISU) {
if (ContextCompat.checkSelfPermission(this,
Manifest.permission.READ_MEDIA_IMAGES) !=
PackageManager.PERMISSION_GRANTED) {
ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.READ_MEDIA_IMAGES},
STORAGE_PERMISSION_CODE);
} else {
openImagePicker();
}
} else {
```

```
            if (ContextCompat.checkSelfPermission(this,
Manifest.permission.READ_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED) {
ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.READ_EXTERNAL_STORAGE},
STORAGE_PERMISSION_CODE);
            } else {
openImagePicker();
            }
        }
    }

    private void openImagePicker() {
Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT);
intent.addCategory(Intent.CATEGORY_OPENABLE);
intent.setType("image/*");
startActivityForResult(intent, PICK_IMAGE_REQUEST);
    }

    private void showTextInputDialog() {
        if (originalBitmap == null) {
Toast.makeText(this, "Please select an image first",
Toast.LENGTH_SHORT).show();
            return;
        }

AlertDialog.Builder builder = new AlertDialog.Builder(this);
```

```
LayoutInflater inflater = getLayoutInflater();
View dialogView = inflater.inflate(R.layout.dialog_text_input, null);
builder.setView(dialogView);

EditText textInput = dialogView.findViewById(R.id.textInput);
ColorPickerView colorPickerView =
dialogView.findViewById(R.id.colorPickerView);
final int[] selectedColor = {0xFFFFFFFF}; // اللون الافتراضي أبيض

colorPickerView.setColorListener((ColorListener) (color, fromUser) ->
selectedColor[0] =
color);

builder.setTitle("Add Text")
.setPositiveButton("OK", (dialog, which) -> {
String text = textInput.getText().toString();
if (!text.isEmpty()) {
photoEditor.addText(text, selectedColor[0]);
// حفظ الحالة بعد إضافة نص
saveCurrentState();
} else {
Toast.makeText(this, "Text cannot be empty",
Toast.LENGTH_SHORT).show();
}
})
.setNegativeButton("Cancel", null)
.show();
}
```

```
private void rotateImage() {
if (originalBitmap != null) {
// حفظ الحالة الحالية قبل التدوير
saveCurrentState();
Matrix matrix = new Matrix();
matrix.postRotate(90);
originalBitmap = Bitmap.createBitmap(originalBitmap, 0, 0,
originalBitmap.getWidth(), originalBitmap.getHeight(), matrix, true);
photoEditorView.getSource().setImageBitmap(originalBitmap);
} else {
Toast.makeText(this, "Please select an image first",
Toast.LENGTH_SHORT).show();
}
}

private void applySelectedFilter() {
if (originalBitmap == null) {
Toast.makeText(this, "Please select an image first",
Toast.LENGTH_SHORT).show();
return;
}

// حفظ الحالة الحالية قبل تطبيق الفلتر
saveCurrentState();

String selectedFilter = filterSpinner.getSelectedItem().toString();
```

```
Bitmap adjustedBitmap = Bitmap.createBitmap(originalBitmap.getWidth(),
originalBitmap.getHeight(), originalBitmap.getConfig());
Canvas canvas = new Canvas(adjustedBitmap);
Paint paint = new Paint();
ColorMatrix colorMatrix = new ColorMatrix();

switch (selectedFilter) {
case "Grayscale":
colorMatrix.setSaturation(0);
break;
case "Sepia":
colorMatrix.set(new float[] {
0.393f, 0.769f, 0.189f, 0, 0,
0.349f, 0.686f, 0.168f, 0, 0,
0.272f, 0.534f, 0.131f, 0, 0,
0, 0, 0, 1, 0
});
break;
case "Vintage":
colorMatrix.set(new float[] {
0.9f, 0.5f, 0.1f, 0, 0,
0.3f, 0.8f, 0.1f, 0, 0,
0.2f, 0.3f, 0.5f, 0, 0,
0, 0, 0, 1, 0
});
break;
case "Warm":
```

```
colorMatrix.set(new float[] {
                1.2f, 0, 0, 0, 20,
                0, 1.1f, 0, 0, 10,
                0, 0, 0.9f, 0, 0,
                0, 0, 0, 1, 0
                });
                break;
            case "Cool":
colorMatrix.set(new float[] {
                0.9f, 0, 0, 0, 0,
                0, 1.0f, 0, 0, 0,
                0, 0, 1.2f, 0, 20,
                0, 0, 0, 1, 0
                });
                break;
            case "Invert":
colorMatrix.set(new float[] {
                -1, 0, 0, 0, 255,
                0, -1, 0, 0, 255,
                0, 0, -1, 0, 255,
                0, 0, 0, 1, 0
                });
                break;
            case "Posterize":
colorMatrix.set(new float[] {
                0.5f, 0, 0, 0, 0,
                0, 0.5f, 0, 0, 0,
```

```
                0, 0, 0.5f, 0, 0,
                0, 0, 0, 1, 0
            });
            break;
        default:
            // "None" – لا تطبيق فلتر
            canvas.drawBitmap(originalBitmap, 0, 0, null);
            photoEditorView.getSource().setImageBitmap(adjustedBitmap);
            return;
    }

    paint.setColorFilter(new ColorMatrixColorFilter(colorMatrix));
    canvas.drawBitmap(originalBitmap, 0, 0, paint);
    photoEditorView.getSource().setImageBitmap(adjustedBitmap);
    originalBitmap = adjustedBitmap; // تحديث (OriginalBitmap) ليعكس الفلتر
}

private void applyAdjustments() {
    if (originalBitmap == null) {
        return;
    }

    // حفظ الحالة الحالية قبل تطبيق التعديلات
    saveCurrentState();

    Bitmap adjustedBitmap = Bitmap.createBitmap(originalBitmap.getWidth(),
            originalBitmap.getHeight(), originalBitmap.getConfig());
```

```
Canvas canvas = new Canvas(adjustedBitmap);
Paint paint = new Paint();
ColorMatrix colorMatrix = new ColorMatrix();
colorMatrix.set(new float[] {
contrast, 0, 0, 0, brightness * 255,
0, contrast, 0, 0, brightness * 255,
0, 0, contrast, 0, brightness * 255,
0, 0, 0, 1, 0
});
ColorMatrix saturationMatrix = new ColorMatrix();
saturationMatrix.setSaturation(saturation);
colorMatrix.postConcat(saturationMatrix);
paint.setColorFilter(new ColorMatrixColorFilter(colorMatrix));
canvas.drawBitmap(originalBitmap, 0, 0, paint);
photoEditorView.getSource().setImageBitmap(adjustedBitmap);
originalBitmap = adjustedBitmap; // تحديث (OriginalBitmap) ليعكس التعديلات
}

private void saveCurrentState() {
if (originalBitmap != null) {
// إنشاء نسخة من الصورة الحالية
Bitmap currentState = Bitmap.createBitmap(originalBitmap);
undoStack.add(currentState);
// الحد من الحجم المكدس لتجنب استهلاك الذاكرة
if (undoStack.size() > 10) {
undoStack.remove(0);
}
```

```
        }
    }

    private void undo() {
        // التراجع عن التعديلات (PhotoEditor) مثل النصوص
        if (photoEditor.undo()) {
            return; // إذا نجح التراجع عن النص لاحاجة للتراجع عن تعديلات أخرى
        }

        // التراجع عن تعديلات يدوية (فلاتر, تدوير, سطوع, تباين, تشبع)
        if (!undoStack.isEmpty()) {
            originalBitmap = undoStack.remove(undoStack.size() - 1);
            photoEditorView.getSource().setImageBitmap(originalBitmap);
            // إعادة تعيين الفلتر إلى (None) إذا لم يكن هناك فلتر نشط
            filterSpinner.setSelection(0);
        } else {
            Toast.makeText(this, "No more actions to undo",
                    Toast.LENGTH_SHORT).show();
        }
    }

    private void saveImage() {
        if (originalBitmap == null) {
            Toast.makeText(this, "Please select an image first",
                    Toast.LENGTH_SHORT).show();
            return;
        }
```

```
// إنشاء (BitMap) جديدة بنفس أبعاد الصورة الأصلية
Bitmap bitmapToSave = Bitmap.createBitmap(
        photoEditorView.getWidth(),
        photoEditorView.getHeight(),
        Bitmap.Config.ARGB_8888
);
Canvas canvas = new Canvas(bitmapToSave);
photoEditorView.draw(canvas); // رسم محتوى يشمل النصوص (PhotoEditorView)

// إستخدام (MediaStore) لحفظ الصورة في مجلد Pictures/PhotoEditorApp2
ContentResolver resolver = getContentResolver();
ContentValues contentValues = new ContentValues();
contentValues.put(MediaStore.Images.Media.DISPLAY_NAME,
        "EditedImage_" +
        System.currentTimeMillis() + ".png");
contentValues.put(MediaStore.Images.Media.MIME_TYPE, "image/png");
contentValues.put(MediaStore.Images.Media.RELATIVE_PATH,
        "Pictures/PhotoEditorApp2");
contentValues.put(MediaStore.Images.Media.IS_PENDING, 1);

Uri imageUri =
resolver.insert(MediaStore.Images.Media.EXTERNAL_CONTENT_URI,
        contentValues);
if (imageUri == null) {
    Toast.makeText(this, "Failed to create image entry",
            Toast.LENGTH_SHORT).show();
    return;
```

```
                }

            try (OutputStream outputStream = resolver.openOutputStream(imageUri)) {
                if (outputStream != null) {
                    bitmapToSave.compress(Bitmap.CompressFormat.PNG, 100,
                            outputStream);
                    outputStream.flush();
                } else {
                    Toast.makeText(this, "Failed to open output stream",
                            Toast.LENGTH_SHORT).show();
                    resolver.delete(imageUri, null, null);
                    return;
                }
                contentValues.clear();
                contentValues.put(MediaStore.Images.Media.IS_PENDING, 0);
                resolver.update(imageUri, contentValues, null, null);
                Toast.makeText(this, "Image saved to Pictures/PhotoEditorApp2",
                        Toast.LENGTH_LONG).show();
            } catch (Exception e) {
                Toast.makeText(this, "Failed to save image: " + e.getMessage(),
                        Toast.LENGTH_LONG).show();
                e.printStackTrace();
                resolver.delete(imageUri, null, null); // حذف الإدخال إذا فشل الحفظ
            } finally {

                if (bitmapToSave != null && !bitmapToSave.isRecycled()) {
                    bitmapToSave.recycle();
```

```
}
}
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
super.onActivityResult(requestCode, resultCode, data);
if (requestCode == PICK_IMAGE_REQUEST && resultCode ==
RESULT_OK && data !=
null) {
Uri imageUri = data.getData();
try {
originalBitmap =
MediaStore.Images.Media.getBitmap(getContentResolver(),
imageUri);
photoEditorView.getSource().setImageBitmap(originalBitmap);
undoStack.clear(); // إعادة تعيين مكدس التراجع عند تحميل صورة جديدة
filterSpinner.setSelection(0); // إعادة تعيين الفلتر إلى (None)
} catch (IOException e) {
Toast.makeText(this, "Error loading image: " + e.getMessage(),
Toast.LENGTH_SHORT).show();
e.printStackTrace();
}
}
}

@Override
```

```
public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions,
@NonNull int[] grantResults) {
super.onRequestPermissionsResult(requestCode, permissions, grantResults);
if (requestCode == STORAGE_PERMISSION_CODE &&
grantResults.length > 0) {
boolean allPermissionsGranted = true;
for (int result : grantResults) {
if (result != PackageManager.PERMISSION_GRANTED) {
allPermissionsGranted = false;
break;
}
}
if (allPermissionsGranted) {
openImagePicker();
} else {
String permission = Build.VERSION.SDK_INT >=
Build.VERSION_CODES.TIRAMISU ?
Manifest.permission.READ_MEDIA_IMAGES :
Manifest.permission.READ_EXTERNAL_STORAGE;
if (!shouldShowRequestPermissionRationale(permission)) {
showPermissionSettingsDialog();
} else {
Toast.makeText(this, "Storage permission is required to select
images",
Toast.LENGTH_LONG).show();
}
}
```

```
        }
    }

    private void showPermissionSettingsDialog() {
        new AlertDialog.Builder(this)
            .setTitle("Permission Required")
            .setMessage("Storage permission is required to select images. Please enable it in
app settings.")
            .setPositiveButton("Go to Settings", (dialog, which) -> {
                Intent intent = new
Intent(android.provider.Settings.ACTION_APPLICATION_DETAILS_SETTINGS);
                intent.setData(Uri.parse("package:" + getPackageName()));
                startActivity(intent);
            })
            .setNegativeButton("Cancel", null)
            .show();
    }
}
```

Contents

المقدمة: 4
مشكلة البحث: 4
أهداف البحث: 5
منهجية البحث: 6
أدوات البحث: 8
تصميم البرنامج: 8
الهيكل العام للبرنامج: 8
المكونات الأساسية: 8
التقنيات المستخدمة: 9
الأمان والخصوصية: 9
الرسم البياني للتصميم: 10
تقييم الأداء: 11
الخلاصة: 13
المراجع: 13
الدراسات السابقة: 15
دراسة عبد الرحمن (2018): تصميم محرر صور بسيط باستخدام Canvas API 15
دراسة سليمان (2019): فعالية تطبيقات تعديل الصور في تحسين جودة الصور الرقمية 15
دراسة أحمد وأكون (2020): فعالية استخدام تطبيقات ويب لتصميم القصص الرقمية باستخدام HTML5 وJavaScript 15
دراسة الشمري (2021) بعنوان: مقارنة بين أدوات تعديل الصور في تطبيقات الويب.....15
دراسة الهاشمي (2022) بعنوان: تطوير تطبيق ويب تفاعلي لتعديل الصور باستخدام إطار React.js 16
دراسة عبد الرحمن (2018): "تصميم محرر صور بسيط باستخدام Canvas API" 16
دراسة سليمان (2019): فعالية تطبيقات تعديل الصور في تحسين جودة الرقمية 17
دراسة أحمد وآخرون (2020): تطبيق ويب لتعديل الصور باستخدام تقنيات HTML5 و JavaScript 18
دراسة الشمري (2021): 19
دراسة الهاشمي (2022): تطوير تطبيق تفاعلي لتعديل الصور باستخدام React.js 20
تاريخ معالجة الصور: 22
المقدمه: 22
الفئات الرئيسية للمستخدمين المحتملين:.....24

التصميم والتطبيق العملي لتطبيق معالجة الصور:.....32
مقدمة : 32
وصف واجهة المستخدم: 32
الوظائف الأساسية للتطبيق:.....32
تحميل الصورة 32
التدوير.....32
تعديل السطوع والتباين والتشبع.....33
تطبيق الفلاتر 33
إضافة النص 33
حفظ الصورة 33
منطق المعالجة 33
قابلية التوسع 34
خاتمة الباب.....34
النتائج والتوصيات:.....37
النتائج.....37
التوصيات:.....37
مقترحات للدراسات المستقبلية:.....38
الملاحق: 39