**مقدمة**

يوما بعد يوم تتوسع الخدمات التي تقدم من التطبيقات الالكترونية لتسهل شؤون حياتنا وقد ساهم تطور تكنولوجيا الهواتف المحمولة وأنظمة التشغيل في توسيع دائرة تلك التطبيقات في السنوات الأخيرة بشكل ملحوظ خصوصا مع ارتفاع عدد مستخدمي الهواتف الذكية كما ساهمت في تسهيل الأعمال وتسهيل التواصل والتعلم والتسلية وأصبحت ضرورة لبعض الشركات والبنوك وغيرها من المؤسسات لإنهاء أعمالهم بأسرع وقت وأسهل طريقة.

هناك العديد من الفوائد للتطبيقات الإلكترونية مثل جذب العديد من الزبائن بسهولة إلى المنتجات والخدمات دون عناء أو تكلفة كبيرة خصوصا وأن التعامل معها سهل جدًا كما تساعد التطبيقات في إيجاد الصلة المستمرة مع الزبائن والتواصل المباشر كما يمكن تأسيس العلامات التجارية المميزة وكسب ثقة العملاء.

**1- الفكرة العامة :**

يهدف هذا المشروع إلى توفير متجر إلكتروني لبيع القطع المستعملة، وذلك لتلبية الاحتياجات المتزايدة للمستهلكين في هذا المجال. ويمكن تحقيق هذا الهدف بعدة طرق منها:

* توفير فرصة للمستهلكين للوصول إلى القطع المستعملة بأسعار مناسبة: يعاني العديد من المستهلكين من صعوبة الحصول على القطع المستعملة بأسعار مناسبة، ولذلك يمثل توفير متجر إلكتروني لبيع القطع المستعملة فرصة لهم للحصول على هذه القطع بأسعار تنافسية
* توفير وسيلة للبيع والشراء بسهولة: يتيح متجر الكتروني لبيع القطع المستعملة للمستخدمين البيع والشراء بسهولة وفي أي وقت، دون الحاجة إلى الذهاب إلى مكان محدد للشراء أو البيع

خلاصة:

بناء متجر إلكتروني لبيع القطع المستعملة يعد هدفاً مهماً يستحق الاهتمام، حيث يساعد في تلبية الاحتياجات المتزايدة للمستهلكين في هذا المجال، ويوفر فرصة للحصول على القطع المستعملة بأسعار تنافسية.

**2- وظائف المشروع :**

**تطبيق المستخدم :**

- صفحة homeتحوي على منشورات للقطع المستعملة منشورة من قبل متسخدمين

يمكن فلترة المنشورات حسب نوع القطع، المدينة، مجال سعر

نوعية المنشورات التي سوف تظهر للمستخدم تعتمد على اكثر انواع القطع المستعملة التي يبحث عنها او يضغط عليها خلال التصفح (سوف يتم توظيف الذكاء هنا).

المنشور عبارة عن عنوان ووصف للقطعة وصور (على الأقل صورة واحدة) وسعر القطعة.

يمكن التصويت على منشور ما بشكل ايجابي او سلبي ( أي هل المستخدم يرى ان هذه القطعة جيدة ام لا)

مع إمكانية التعليق أو فتح دردشة خاصة مع صاحب القطعة للاستفسار او لطلب حسم خاص من البائع على هذه القطعة.

عند تمام عملية البيع يجب للبائع تسليم القطعة لموظف أقرب مستودع وذلك عن طريق مسح رمز QR على جهاز الموظف.

- **صفحة محادثات :**   
تحوي الصفحة على قسمين:

- قسم للرد على استفسارات المستخدمين عن قطعة ما:

تجمع فيه المحادثات حسب القطعة المراد بيعها

- قسم للاستفسار عن قطعة ما:

يظهر للمستخدم لائحة محادثات تحوي صورة القطعة وعنوانها مع اسم البائع

- **صفحة معاملاتي** تحوي جميع المشتريات والمبيعات.

**تطبيق موظف مستودع :**

يستخدم التطبيق للقيام بعمليات تسليم واستلام قطع للمستودع مع المستخدمين عن طريق مسح رمز QR

**تطبيق أدمن :**

- ٳحصائيات عن أداء المنتج وأرباحه  
- يتم التحكم بالاصناف الممكنة للقطع

- معاينة التبليغات الواردة عن منشورات معينة وازالتها ان كانت تنافي الشروط والسياسات.

**3- التقنيات البرمجية المستخدمة :**

**HTML 1-3**

Hyper Text Markup Language ويشار اليها اختصارا HTML تم وصفها في اواخر 1990 من قبل Tim Berners-Lee هي لغة تصميم صفحات ومواقع ويب, وتعتبر من أقدم اللغات و اوسعها استخداماً وهي ليست لغة برمجة ولا تحتاج الى مترجم خاص بها و وغير مرتبطة بنظام تشغيل معين لأنه يتم تفسيرها وتنفيذ تعليماتها مباشرة من قبل المتصفح ، يمكننا القول بأنها عبارة عن الهيكل الرئيسي لصفحات الويب والبنية التحتية لها، إذ تقدم وصفاً مفصل حول الكيفية التي ستكون عليها آلية عرض محتويات الموقع الإلكتروني ويكون ذلك بتقسيمه إلى عنوان وفقرات ويتم ذلك كله بالاعتماد على ما يُعرف بالوسوم Tags ويقوم متصفح الويب بتحويل السطور المكتوبة بهذه اللغة إلى صفحات مفهومة وسهلة القراءة لزوار الموقع

تتألف صفحات HTML من عناصر, والتي تتألف عادة من وسم بداية ووسم نهاية, ويكون بينهما محتوى نصي عادة, ويمكن أن نضع في وسم البداية بعض الخاصيات التي تغير من سلوك العنصر او تؤدي الى ضبطه. لاحظ أن بعض العناصر يمكن ان تتشعب داخل بعضها بعضا. الأقسام الأساسية للعناصر هي:  
  
**وسم البداية:** وهو يحتوي على اسم العنصر, موضوعا ضمن قوسين على شكل زاوية, وقد يلي الاسم الخاصيات التي تؤثر عليه. مثلا <p>  
  
**وسم النهاية:** وهو يحتوي على اسم العنصر ايضا مسبوق بخط مائل قبله للاشارة الى نهاية العنصر, ولاحظ أن نسيان وسم النهاية قد يسبب أخطاء في بعض الأحيان, لذا خذ حذرك وتذكره. مثلا <\p>

**المحتوى:** وهو موجود بين وسمي البداية والنهاية, ويمثل في معضم الاحيان محتوى العنصر.  
  
**العنصر:** هو وسم البداية ووسم النهاية اضافة الى المحتوى.[5]

**مثال:**

<!DOCTYPE html>  
<html>  
<body>  
<h1>My First Heading</h1>  
<p>My first paragraph.</p>  
</body>  
</html>

الشكل 3-1 مثال عن HTML

**3-2HTML5**

تم في هذا الإصدار إضافة العديد من العناصر والميزات الجديدة حيث تم بناء هذه المميزات

اعتماداً على HTML DOSS, Java Script وتم إضافة ميزات أفضل في عملية معالجة الأخطاء.

المميزات المضافة في HTML5 :

* العنصر canvas لتنفيذ التعليمات الرسومية.
* العنصرين video audio لتشغيل ملفات الصوت والصورة.
* دعم أفضل لتخزين الصفحات في نمط الاتصال Offline Storage.
* عناصر جديدة لدعم أنواع معينة من المحتوى مثل article, footer, header .
* عناصر إدخال جديدة مثل calendar, date, time, email .

**CSS 3-3**

تعتبر CSS اختصار ل (Cascading Style Sheet) - لغة توصيفية تقوم بمنح موقع الويب شكله الجميل وتصميمه الفريد الذي سيميزه عن غيره من المواقع. تعد لغة ال CSS صديقة للغة ال HTML دائماً . فهي مرافقة لها وبجانبها في تصميم وإنشاء صفحات الويب. تشكل كل من هاتان اللغتان الأساس لكل موقع فال HTML هي أساس كل عناصر صفحة الويب أما ال CSS فهي أساس تصميم وشكل الموقع فبدونها ستظل مواقع الويب نصا عاديًا على خلفيات بيضاء, عليك تعلم CSS ان اردت ان تكون مطور واجهات متخصص, كما عليك بالتأكيد تعلم HTML لبناء اي موقع متميز. قبل البدء ببناء و تطوير لغة ال CSS في عام 1996 من قبل اتحاد شبكة الويب العالمية (W3C) ، كانت صفحات الويب محدودة و بسيطة للغاية من حيث الشكل والوظيفة. فقد كانت المتصفحات القديمة تعرض صفحات الويب كصفحات تتكون من نصوص سوداء بخلفية بيضاء فقط (نص - صور - روابط .. إلخ) بدون أي تصميم يذكر . ولم يكن هناك تخطيط لموعد إطلاق لغة CSS JI

سمحت CSS بتصميم وتشكيل صفحات الويب بطرق متعددة و مختلفة لتتمكن من استيعاب مدى إبداع الشخص مثل القدرة على :

* تحديد الخطوط والهوامش والمسافات لعناصر الموقع
* تحديد لون وحجم العناصر
* تطبيق الألوان على الخلفيات
* تحريك العناصر بحرية والحصول على العديد من الحركات في الموقع . والعديد من المزايا الأخرى. [1]

مثال : دعنا نفترض أن المطور يريد جعل النص داخل الرأس أرجوانيا ، بخط من

نوع Arial حجمه 15px يمكننا عمل ذلك باستخدام هذا السطر البسيط

.PageHead{  
Font-family: Arial, sans-sref;  
Font-size: 15px;  
Color: purple;  
}

الشكل 3-2 مثال عن CSS

**4-3 لغة JavaScript :**

جافا سكريبت أو في الغالب يطلق عليها اختصارًا JS بالإنجليزية JavaScript :

هي لغة برمجة عالية المستوى تستخدم أساساً في متصفحات الويب لإنشاء صفحات أكثر تفاعلية يتم تطويرها حالياً من طرف شركة موزيلا.

كانت لغة الجافاسكريبت موجهة للمبرمجين الهواة وغير المحترفين، إلا أنه تزايد الاهتمام بها وجذبت اهتمام مبرمجين محترفين بعد إضافتها لتقنيات جديدة كانتشار تقنية أجاكس التي أدت إلى سرعة في التفاعل بين الخادم والعميل.

تُستخدم لغة الجافا سكربت لإنشاء صفحات ويب تفاعلية، ولتوفير تطبيقات ويب بما في ذلك الألعاب؛ وهي مُستعملة من أغلبية المواقع، وتدعمها جميع المتصفحات تقريبا دون الحاجة

إلى إضافات خارجية.

في الماضى كانت لغة JavaScript محدودةً إلى متصفحات الويب فيما قد سلف، لكن توسع مجال JavaScript في الفترة الأخيرة لتضم إنشاء تطبيقات ويب من جهة الخادم (server-side) مع التعامل مع قواعد البيانات، إضافة إلى إمكانية استخدامها في بيئات ليست متعلقة بالويب مثل معالجات النصوص وبرمجيات PDF ، ومن الممكن استعمالها أيضًا لكتابة تطبيقات سطح المكتب أو تطبيقات الهواتف، وحتى تطوير الألعاب .

يتألف أساس لغة JavaScript من الميزات الشائعة للغات البرمجة، التي تسمح لنا:

* بتخزين القيم داخل متغيرات
* إجراء عمليات معالجة على السلاسل النصية
* تنفيذ إجراءات معينة اعتمادًا على وقوع الأحداث.

ما يجب أن يثير اهتمامنا هو الواجهات البرمجية التي تُضيف وظائف على لغة JavaScript، وتسمى هذه الواجهات البرمجية بالمصطلح APIs

(Application Programming Interfaces )، والتي تسمح للمطورين باستخدام لغة

JavaScript للتعامل مع كثير من الأشياء، في مجال متصفحات الويب يمكننا استخدام JavaScript للتعامل مع شجرة DOM، ولاستخدام الواجهة البرمجية للمواقع الجغرافية (Geolocation API) ، ولإنشاء رسوميات ثنائية وثلاثية الأبعاد عبر Canvas و WebGL ، وللتحكم بالوسائط مثل الفيديو والصوت، وغير ذلك.

يمكن تضمين شيفرات JavaScript في الصفحات بطريقتين :

1. تضمين الشيفرات داخل مستند HTML.

2. استخدام ملف خارجي .

يمكن تضمين شيفرة JavaScript في أي مكان داخل المستند, ولكن يُنصح بوضعها قبل وسم الاغلاق للعنصر <body> تُضاف الشيفرة داخل العنصر <script> كما يلي:

<script>  
//شيفرات JavaScript  
</script>

الشكل 3-3 تضمين JS داخل المستند

أما اذا كانت شيفرة JavaScript في ملف خارجي وليكن اسمه script.js فيمكن تضمينه باستخدام العنصر <script> أيضا, لكن مع استخدام الخاصية src كما يلي:

<script src="script.js"> </script>

الشكل 3-4 تضمين JS من ملف خارجي

**3-5 لغة Type Script :**

هو لغة برمجة تم إنشاؤها من قبل مايكروسوفت وهي تعتبر إضافة إلى الـ JavaScript. تم تصميم TypeScript ليكون عبارة عن نظام كتابة الأكواد يساعد على تطوير تطبيقات ويب أكثر قوة وصحة.

تتميز TypeScript بتوفير نظام قوي للأنواع يتيح للمطورين تحديد أنواع البيانات الخاصة بكل عنصر من عناصر البرنامج، مما يساعد على تحسين جودة الأكواد وتجنب الأخطاء الناتجة عن تحويل الأنواع. كما أن TypeScript يدعم العديد من الميزات الحديثة في JavaScript مثل الوعود (Promises) والتعبيرات الشرطية (Conditional Expressions) والتعبيرات العامة (Arrow Functions).

تساعد TypeScript على تحسين إنتاجية المطورين وتسهل عليهم القيام بالأعمال الخاصة بالتعامل مع الأكواد، وتجعل عملية التطوير أكثر سلاسة وسهولة وأمان.

**3-6 ٳطار العمل React JS :**

هو إطار عمل (Framework) لتطوير واجهات المستخدم (UI)، وهو يستخدم لبناء تطبيقات الويب الديناميكية. تم تطوير ReactJS بواسطة فريق عمل في Facebook، وتم إصداره لأول مرة في عام 2013. ومنذ ذلك الحين، أصبح ReactJS شائعًا جدًا ومشهورًا في مجتمع تطوير الويب.

ما يميز ReactJS هو أسلوبه في إنشاء واجهات المستخدم الديناميكية. يعتمد ReactJS على مفهوم المكونات (Components)، حيث يتم تجزئة واجهة المستخدم إلى عناصر صغيرة وقابلة لإعادة الاستخدام تسمى المكونات. تتفاعل المكونات مع بعضها البعض لإنشاء واجهة مستخدم متجاوبة وديناميكية.

يستخدم ReactJS أيضًا مفهوم "التركيز على الحالة" (State)، حيث يسمح للمطورين بتعريف حالة التطبيق وتتبع تغييراتها عبر الوقت. وعندما يتغير الحالة، يعيد ReactJS بناء الجزء المتأثر من واجهة المستخدم بدلاً من إعادة بناء الصفحة بأكملها، وهذا يؤدي إلى تحسين أداء التطبيق.

يمكن استخدام ReactJS لبناء تطبيقات الويب الكبيرة والمعقدة، وهو متوافق مع العديد من التقنيات الأخرى وإطارات العمل (Frameworks)، مثل Redux وReact Router. وبفضل مجتمعه النشط والدعم المستمر من قبل Facebook والمطورين الآخرين، أصبح ReactJS واحدًا من الأدوات الأكثر شعبية واستخدامًا في مجال تطوير الويب.

**مثال عن React Js :**

import React, { useState } from 'react';

function Example() {

  // Declare a new state variable, which we'll call "count"

  const [count, setCount] = useState(0);

  return (

    <div>

      <p>You clicked {count} times</p>

      <button onClick={() => setCount(count + 1)}>

        Click me

      </button>

    </div>

  );

}

الشكل 3-5 مثال عن React Js

**3-7 مكتبة Material UI :**

Material-UI هي مكتبة واجهة مستخدم (UI) مفتوحة المصدر تعتمد على تصميم المواد (Material Design) الذي تم تطويره بواسطة Google. تهدف Material-UI إلى توفير مجموعة شاملة من المكونات والأدوات المصممة بشكل جميل ومتكامل لبناء واجهات مستخدم حديثة وجذابة في تطبيقات الويب.

تتميز Material-UI بالنقاط التالية:

* تصميم مفتوح المصدر: يتيح Material-UI الوصول إلى مصدر الكود والمساهمة في تطويره وتحسينه. يمكن للمطورين استخدام المكونات المتاحة أو تخصيصها وفقًا لاحتياجاتهم الخاصة.
* تصميم المواد (Material Design): يتبع Material-UI تصميم المواد الذي تم تطويره بواسطة Google. يتميز بتصميم نظيف وبسيط وحديث يوفر تجربة مستخدم موحدة ومرئية جذابة.
* مكونات جاهزة للاستخدام: يوفر Material-UI مجموعة كبيرة من المكونات الجاهزة للاستخدام مثل الأزرار والحقول والقوائم والجداول والبطاقات وغيرها. تتميز هذه المكونات بتصميم متجاوب وقابلية للتخصيص وسهولة الاستخدام.
* دعم للتصميم الاستجابي: تدعم Material-UI التصميم الاستجابي، مما يعني أن المكونات والتخطيطات يمكن أن تتكيف بشكل تلقائي مع مختلف أحجام الشاشة وأجهزة الجهاز المختلفة.
* توافق مع React: تعتمد Material-UI على إطار العمل React لبناء واجهات المستخدم، مما يسهل عملية الاندماج والتفاعل مع تطبيقات React الموجودة بالفعل.[6]

**3-8 بيئة التشغيل Node Js :**

هو إطار عمل تم تطويره من JavaScript والتي يمكن تشغيلها من طرف الخادم بعكس جافاسكريبت.

Node.js عبارة عن نظام أساسي مبني على وقت تشغيل JavaScript في Chrome لبناء تطبيقات شبكة سريعة وقابلة للتطوير بسهولة.

يستخدم Node.js نموذج إدخال / إخراج يحركه الحدث ولا يحظره مما يجعله سريع وفعال، ومثالي لتطبيقات الوقت الفعلي كثيفة البيانات التي تعمل عبر الأجهزة الموزعة.

Node.js هي بيئة تشغيل مفتوحة المصدر تعمل عبر الأنظمة الأساسية لتطوير تطبيقات الخادم والشبكات.

تتم كتابة تطبيقات Node.js بلغة JavaScript ، ويمكن تشغيلها خلال وقت تشغيل Node.js على أنظمة التشغيل OS X و Microsoft Windows و Linux يوفر Node.js أيضًا مكتبة غنية من وحدات JavaScript المتنوعة التي تبسط تطوير تطبيقات الويب باستخدام Node.js إلى حد كبير.[8]

**3-8-1 لماذا Node Js:**

سرعة Node js الكبيرة تعد أساسا لعاملين اساسين هما: محرك chrome v8 المتطور وآلية استقبال وإرسال المدخلات والمخرجات 1/0 والتي يطلق عليها اسم non blocking

وبالمقابل توجد آلية أخرى تدعى blocking التي تنتهجها اللغات الأخرى.

**3-9 ٳطار العمل Express JS :**

Express.js هو إطار عمل لتطوير تطبيقات ويب قائمة على Node.js. يهدف Express.js إلى تبسيط عملية إنشاء تطبيقات الويب من خلال توفير طرق ووظائف مجربة ومعتمدة لإدارة الطلبات والاستجابات.

يعتبر Express.js خفيف الوزن ومرنًا وسهل الاستخدام، مما يجعله شائعًا بين المطورين لبناء تطبيقات الويب السريعة والقابلة للتطوير بسهولة. يقدم Express.js مجموعة من الميزات التالية:

* إدارة المسارات (Routing): يوفر Express.js طرقًا بسيطة لتعريف المسارات (routes) ومعالجتها. يمكن تعريف المسارات بناءً على طريقة الطلب (GET، POST، PUT، DELETE) والعنوان (URL)، مما يسمح بإنشاء تطبيقات ويب متعددة الصفحات وعرض محتوى متنوع.
* الوسيطات (Middleware): يدعم Express.js نظام الوسيطات، والذي يتيح تنفيذ وظائف مختلفة في وسط سير المعالجة. يمكن استخدام الوسيطات لإجراء عمليات مثل التحقق من الهوية، والتحقق من الصلاحيات، وتسجيل الوقت، وإدارة الأخطاء، وغيرها.
* القوالب (Templates): يدعم Express.js استخدام قوالب لإنشاء صفحات الويب بطريقة ديناميكية. يمكن استخدام محركات القوالب المختلفة مثل Pug وEJS وHandlebars لتوليد HTML يحتوي على بيانات متغيرة.
* التعامل مع قاعدة البيانات: يوفر Express.js مرونة كبيرة في التعامل مع قواعد البيانات المختلفة. يمكن استخدام مكتبات مثل Mongoose للتعامل مع قواعد البيانات غير العلائقية مثل MongoDB، أو استخدام مكتبات SQL مثل Sequelize للتعامل مع قواعد البيانات .[3]

const express = require('express')

const app = express()

const port = 3000

app.get('/', (req, res) => {

  res.send('Hello World!')

})

app.listen(port, () => {

  console.log(`Example app listening on port ${port}`)

})

الشكل 3-6 مثال عن Express Js

**3-10 ٳطار العمل Flutter :**

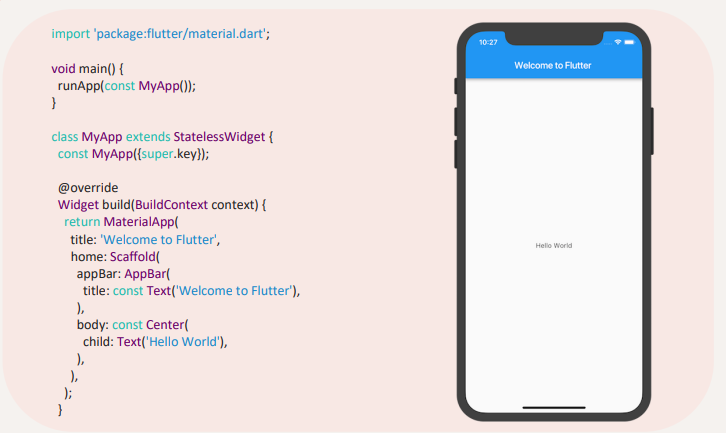
هو إطار عمل من شركة جوجل مخصص لبرمجة تطبيقات الهواتف الذكية العاملة بنظام الاندرويد وال IOS و فوشيا (نظام تشغيل جديد من جوجل) حيث يركز على تجربة المستخدم و المطور في نفس الوقت و هذا لأنه يوفر إطار عمل شامل بلغة دارت مخصص لرسم الوجهات بجودة عالية و تطبيقات أصلية بأداء ممتاز و في نفس الوقت يعطي للمطور الأدوات التي تـجـعـلـه يـبـنـي تـطـبـيـقـات كـامـلـة فـي أقل وقت.

من مميزات Flutter التي جعلته أكثر سهولة لتطوير تطبيقات الهواتف هو إعتماده على نظام (Widgets) فكل عنصر من Flutter يعتبر Widget وقد تم إستنباط هذا النظام من React JSX، وهذه تعتبر ميزة للمطورين اللذين سبق لهم التعامل مع React، فقد يجدون أن صناعة التطبيقات بإستخدام Flutter أكثر سهولة. نظام Widgets في إطار Flutter جعل من السهل إنشاء مكونات (Components جديدة وقوة لغة Dart ساهمت أيضا بكونها لغة كائنية التوجه في هذا الأمر بحيث يمكن أخذ Widget رئيسي و إضافة عليه خصائص جديدة من تأثيرات و حركات (Animation) بشكل بسيط وبدون تعقيد. هذا الأمر مَكَّنَ من إنشاء مكونات جديدة داخل الإطار و إدراج Material Design في المكونات التي تعمل على الأندرويد و Cupertino على IOS واحدة من مميزات إطار Flutter هي دعمه بشكل كامل لواجهات ال Material Design فعلى نظام الأندرويد تأتي المكونات من Text و Button و Switch إفتراضيا على تنسيق الماتريال ديزاين هذا ما سيوفر عليك

عناء التعديل على الديزاين كما الحال على الأندرويد بالجافا. [4]

**3-10-1 ماهو الاختلاف في Flutter:**

هنالك عدة افكار جديدة جاء بها Flutter, حيث أن فلسفة فريق فلاتر هي امكانية رسم كل بكسل على الشاشة بحرية وعدم الارتباط بالنظام أي مثل الألعاب تماما و لكن العامل الذي جعل فلاتر يصل إلى ما وصل إليه الآن هو دعم مباشر من فريق آخر في جوجل و هو فريق الماتريال ديزاين لان فلاتر يوفر كل مكونات الماتريال ديزاين باصدارتها الاخيرة حتى قبل ان يحصل عليها الاندرويد بنفسه و هذا ما تم التصريح به في موؤتمر جوجل و من الاشياء التي يوجد فيها اختلاف كبير هو طريقة فلاتر في رسم الواجهات لان فلاتر يستخدم لغة البرمجة دارت في كل شيء من أجل الكتابة و التصميم ايضا.



الشكل 3-7 مثال عن Flutter

**3-10-2 لغة Dart :**

هي لغة برمجة كائنية التوجه ظهرت في سنة 2011 و تم عملها من طرف شركة جوجل كمحاولة لاستبدال لغة جافا سكربت لكن لم تنجح وهذا واضح جدا لكن مع ذلك فهو توفر الخصائص المناسبة التي يحتاجها فلاتر لكي ينمو بشكل صحيح حيث ان لغة Dart هي:

* لغة برمجة كائنية التوجه و تنفذ كل مفاهيم البرمجة الكائنية منها تعدد الوراثة تحت اسم mixins.
* يتحكم فريق Flutter بشكل كامل في لغة Dart لكي تناسب Flutter في الاونة الاخيرة حصل تغير كبير على اللغة لكي تناسب Flutter فمثلا تم جعل كلمة new "التي سيتعرف عليها مبرمج الجافا و السي شارب " كلمة اختيارية فقط لكي يتم تحسين طريقة كتابة الواجهات .
* اثناء البرمجة تستخدم لغة Dart مترجم JIT : Just in time و التي بفضلها ظهرت خاصية الهوت ريلود و هي انه عند تغير الكود يتم ارسال فقط البايتات التي تم تغييرها بالتالي حصل التغير في برنامجك في اقل من ثانية و يبقى محافظا على حالته .
* أثناء نشر التطبيق تستخدم Dart مترجم AOT : Ahead of time و الذي ينتج لنا تطبيق اصلي و بالتالي اداء رائع .
* يمكن ترجمة لغة Dart الى جافا سكربت محسنة الاداء و هذا ما يجعل عمل تطبيقات ويب بفلاتر امرا ممكنا .
* لغة Dart تم عملها لكي تكون سهلة لمبرمج الجافا سكربت و مبرمجي الجافا و السي شارب أيضا. [2]

**أمثلة عن Dart :**

**المتغيرات:**

var name = "majed";  
var year = 2000;  
var flybyObjects = ['Jupiter', 'Saturn', 'Uranus', 'Neptune'];  
var image = {  
 'tags': ['saturn'],  
 'url': '//path/to/saturn.jpg'  
};

الشكل 3-8 المتغييرات في Dart

**التوابع:**

int Fibonacci (int n) {  
 if (n == 0 || n == 1) return n;  
 return fibonacci (n - 1) + fibonacci(n - 2);  
}  
var result = fibonacci(20);

الشكل 3-9 التوابع في Dart

**الشروط والحلقات:**

if (*year* >= 2001) {  
print('21st century');  
} else if (*year* >= 1901) {  
print('20th century');  
}  
for (final *object* in flybyObjects) {  
print(*object*);  
}  
for (int *month* = 1; month <= 12; month++) {  
print(*month*);  
}  
while (*year* < 2016) {  
*year* += 1;  
}

الشكل 3-10 الشروط والحلقات في Dart

**3-11 قاعدة البيانات Mongo DB :**

MongoDB هو نظام إدارة قواعد البيانات غير العلائقية (NoSQL) يستخدم لتخزين واستعراض البيانات. يتميز MongoDB بتوفير أداء عالي وقابلية للتوسع وسهولة الاستخدام والتشغيل.

تتميز قاعدة بيانات MongoDB بالنقاط التالية:

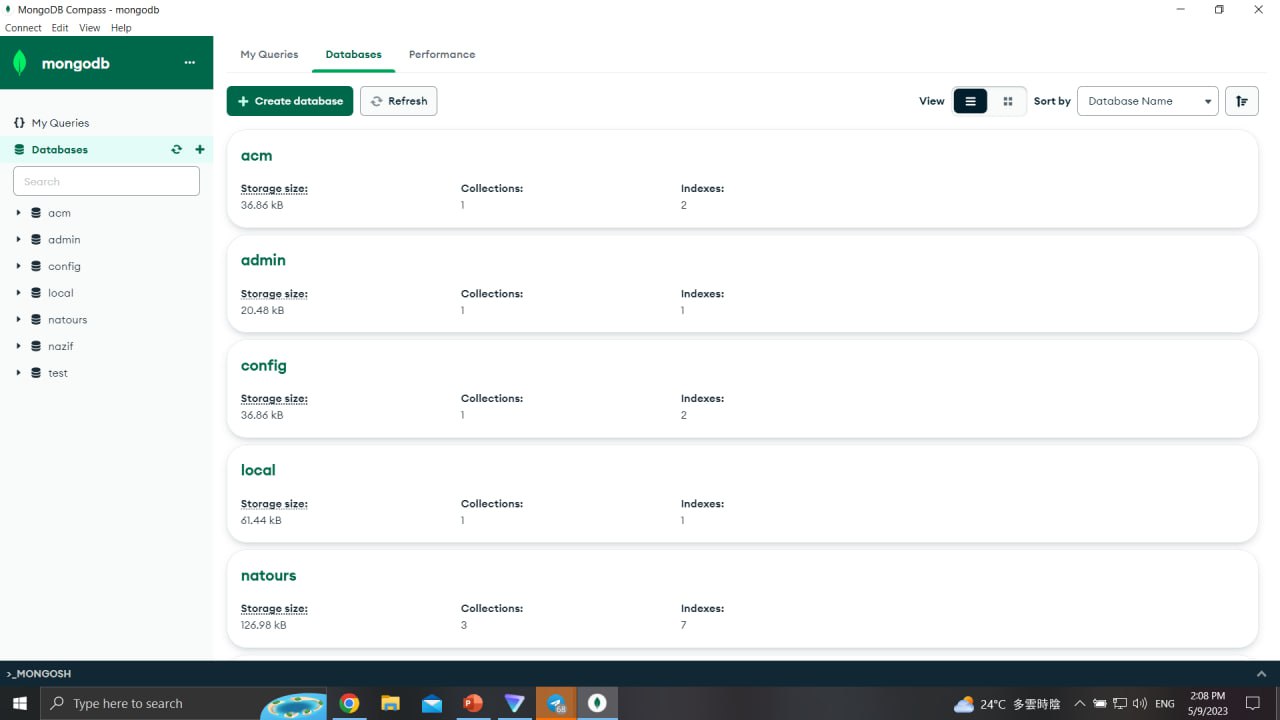
* هيكل بيانات غير علائقي: يستخدم MongoDB نموذجًا مرنًا لتخزين البيانات يسمى BSON (Binary JSON)، والذي يتيح تخزين البيانات في شكل وثائق مشابهة لتنسيق JSON. وهذا يعني أنه لا يتعين على المستخدمين تحديد هيكل البيانات مقدمًا، وبالتالي يوفر مرونة أكبر في تطوير التطبيقات وإدارة التغييرات.
* توفر القابلية للتوسع: يتيح MongoDB توزيع البيانات على عدة خوادم، مما يتيح زيادة القدرة على التحمل والأداء باستخدام عمليات التوازن التلقائي للحمل. يمكن للمستخدمين إضافة خوادم جديدة وتوزيع البيانات بسهولة للتعامل مع أحمال العمل المتزايدة.
* دعم استعلامات قوية: يوفر MongoDB مجموعة واسعة من العمليات والاستعلامات للبحث والتحديث والتجميع والتجزئة للبيانات. يستخدم لغة الاستعلام MongoDB (MongoDB Query Language) لتنفيذ استعلامات متقدمة ومرونة في استعلامات البحث.
* توافق مع التطبيقات الحديثة: يدعم MongoDB العديد من لغات البرمجة والأطر الشهيرة مثل Python و JavaScript و Node.js وJava وغيرها. كما يوفر MongoDB مكتبات وأدوات قوية للتكامل مع بيئات التطوير المختلفة.

بشكل عام، MongoDB يعتبر خيارًا شائعًا لتطبيقات الويب والتطبيقات .[7]

يمكنك إنشاء قاعدة بيانات باستخدام MongoDB Compass عن طريق اتباع الخطوات التالية:

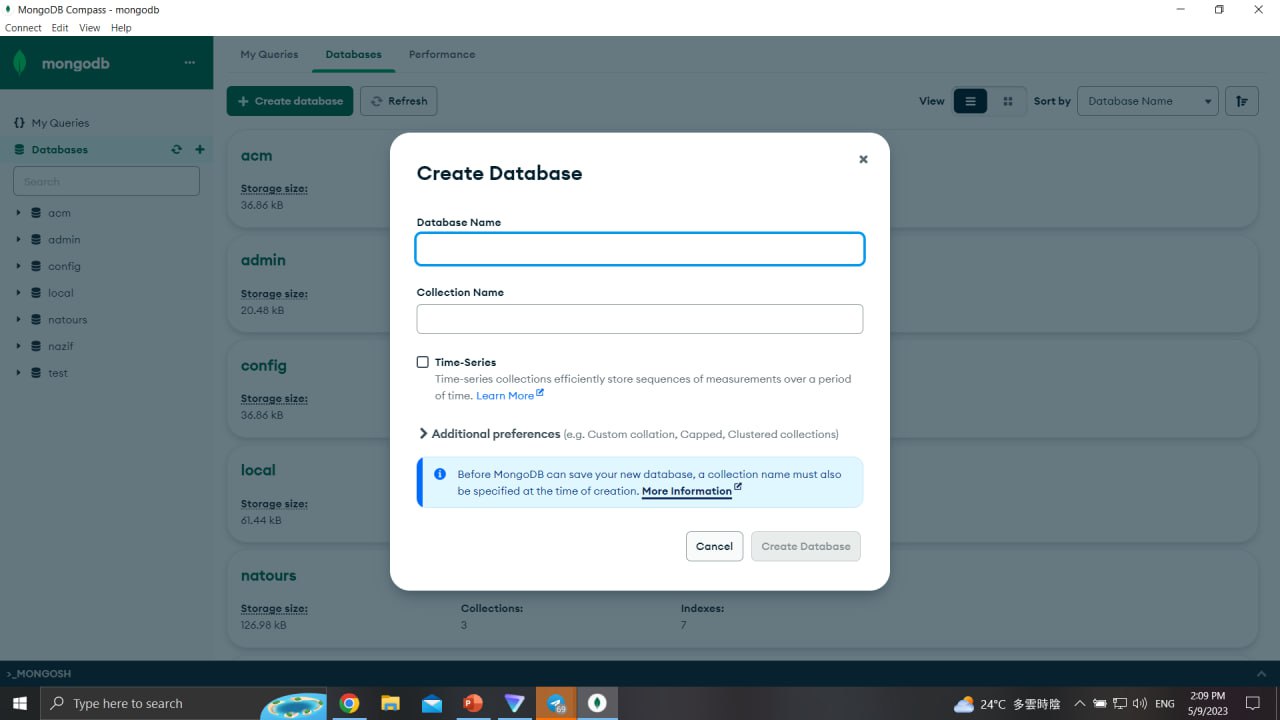
تنزيل وتثبيت MongoDB Compass على جهازك من الموقع الرسمي لشركة MongoDB.

فتح MongoDB Compass والنقر على زر "Create Database".



الشكل 3-11 انشاء قاعدة بيانات Mongo DB

ستظهر لك نافذة جديدة لإنشاء قاعدة بيانات جديدة. يمكنك تحديد اسم قاعدة البيانات واسم المجموعة الرئيسية (Primary Key) والمجموعات الفرعية (Collections) التي تريد إنشاؤها.



الشكل 3-12 انشاء قاعدة بيانات Mongo DB

بعد الانتهاء من تحديد تفاصيل قاعدة البيانات، انقر على زر "Create Database" في الأسفل لإنشاء القاعدة والمجموعات الفرعية.

بعد إنشاء قاعدة البيانات، يمكنك الاتصال بها من MongoDB Compass والبدء في إدخال البيانات والتعامل معها. يمكنك أيضًا تنفيذ العديد من الإجراءات الأخرى على قاعدة البيانات باستخدام MongoDB Compass، مثل إنشاء مؤشرات (indexes) وتحليل الأداء والبحث عن البيانات.

**3-11-1 مكتبة mongoose :**

Mongoose هو حزمة (package) للتعامل مع قاعدة بيانات MongoDB باستخدام لغة JavaScript. يعمل Mongoose كطبقة تجريد (abstraction layer) فوق MongoDB ويوفر واجهة برمجة التطبيقات (API) لتسهيل التفاعل مع قاعدة البيانات وتنفيذ العمليات المختلفة مثل الاستعلامات والإدخال والتحديث والحذف.

يقدم Mongoose العديد من الميزات والوظائف التي تسهل عملية التطوير مع قاعدة بيانات MongoDB، بما في ذلك:

* تعريف نماذج البيانات: يسمح Mongoose للمطورين بتعريف نماذج هيكل البيانات (schemas) التي تمثل الهيكل والخصائص المتوقعة للبيانات. يتيح هذا تحديد أنواع البيانات والقيود والتحقق من الصحة في البيانات.
* العلاقات والصلات: يوفر Mongoose آليات لإنشاء وإدارة العلاقات بين المستندات في قاعدة البيانات. يمكن تعريف الصلات والمفاتيح الخارجية والاستعلام عن البيانات المرتبطة بشكل فعال.
* الاستعلام والتحديث: يوفر Mongoose واجهة بسيطة وقوية لتنفيذ استعلامات البحث والتحديث على البيانات. يمكن استخدام توابع البحث والتصفية والفرز لاستعلام البيانات بشكل مرن.
* المراقبة والتشغيل: يوفر Mongoose وظائف لمراقبة وتتبع التغييرات في قاعدة البيانات، بما في ذلك رصد الحدث والإشعارات والتلاعب بالبيانات.

باستخدام Mongoose، يمكن للمطورين بسهولة إنشاء تطبيقات قوية ومتقدمة باستخدام قاعدة بيانات MongoDB، وتنظيم البيانات وإدارة العلاقات وتنفيذ الاستعلامات بشكل فعال.

**3-12 ٳطار العمل Swagger:**

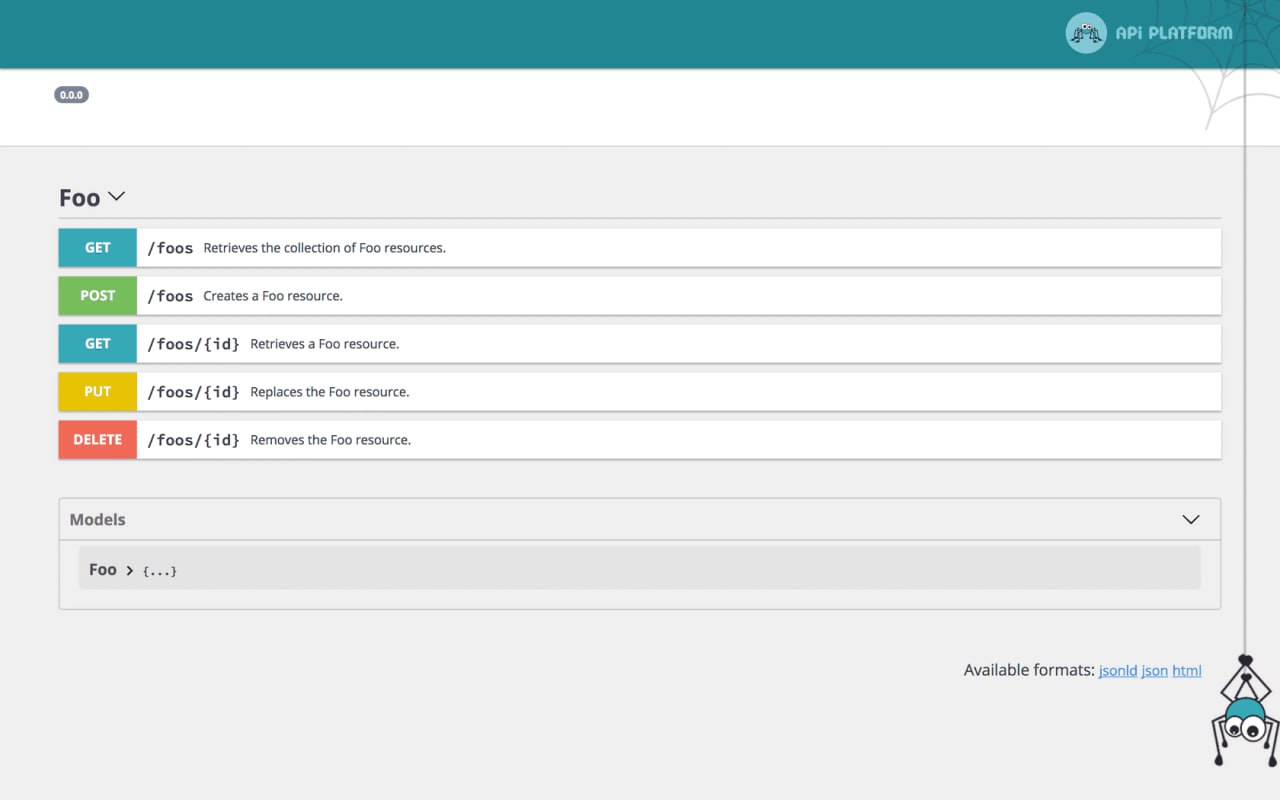
هو إطار عمل (Framework) يستخدم لوصف وتوثيق واجهات برمجة التطبيقات (APIs) بشكل مستند وقابل للقراءة بواسطة البشر والآلات. يهدف Swagger إلى تسهيل فهم واستخدام الAPIs من خلال توفير وثائق دقيقة ومفصلة للمطورين والمستخدمين.

يعتمد Swagger على لغة وصف الواجهة القياسية المعروفة باسم OpenAPI Specification (OAS)، التي توفر تنسيقًا مستقلًا عن لغة البرمجة يمكن استخدامه لوصف تفاصيل الواجهة البرمجية. تشمل هذه التفاصيل معلومات حول المسارات (endpoints) المتاحة والمعلمات المطلوبة والاستجابات المتوقعة والتوثيق الإضافي مثل الوصف الوظيفي وأمثلة الاستخدام والمصطلحات المستخدمة.

توفر أدوات Swagger مجموعة من المميزات مثل واجهة تفاعلية لاستكشاف واختبار الواجهات البرمجية، وتوليد توثيق تلقائي، ودعم للتوثيق الإضافي مثل العمليات الأمنية وإدارة الاستثناءات. بفضل Swagger، يمكن لفرق التطوير والمستخدمين فهم واستخدام الAPIs بشكل أفضل وتسهيل التعاون وتقليل الاحتياج إلى الاستفسارات الإضافية والتواصل غير الفعال.

**3-11-1 لماذا Swagger :**

* وصف الواجهة القياسي: Swagger يستخدم تنسيق OpenAPI Specification (OAS) القياسي لوصف وثائق الواجهة البرمجية. يعني ذلك أنه يتم توفير تنسيق موحد وقابل للقراءة من قبل المطورين والمستخدمين، مما يسهل فهم وتبادل المعلومات بين الأطراف المعنية.
* توليد التوثيق التلقائي: Swagger يوفر إمكانية توليد توثيق تلقائي لواجهات البرمجة. بمجرد وصف الواجهة البرمجية بشكل صحيح في Swagger، يمكنك توليد توثيق مفصل ومنسق تلقائيًا، بما في ذلك معلومات الوصف وأمثلة الاستخدام والمعلمات المتوقعة والاستجابات.
* واجهة تفاعلية: Swagger يوفر واجهة تفاعلية تسمح للمطورين والمستخدمين باستكشاف واختبار الواجهات البرمجية بشكل مرئي وتفاعلي. يمكنك تجربة واختبار المسارات (endpoints) المختلفة وعرض الاستجابات وتحليل البيانات بسهولة.
* التوثيق الإضافي: بالإضافة إلى وصف الواجهة البرمجية الأساسية، يمكنك إضافة التوثيق الإضافي في Swagger مثل العمليات الأمنية وإدارة الاستثناءات والمعلومات الإضافية الأخرى التي ترغب في توثيقها.[9]

****

الشكل 3-13 Swagger

**تحليل النظام**

**4-1 مخطط UseCase UML**

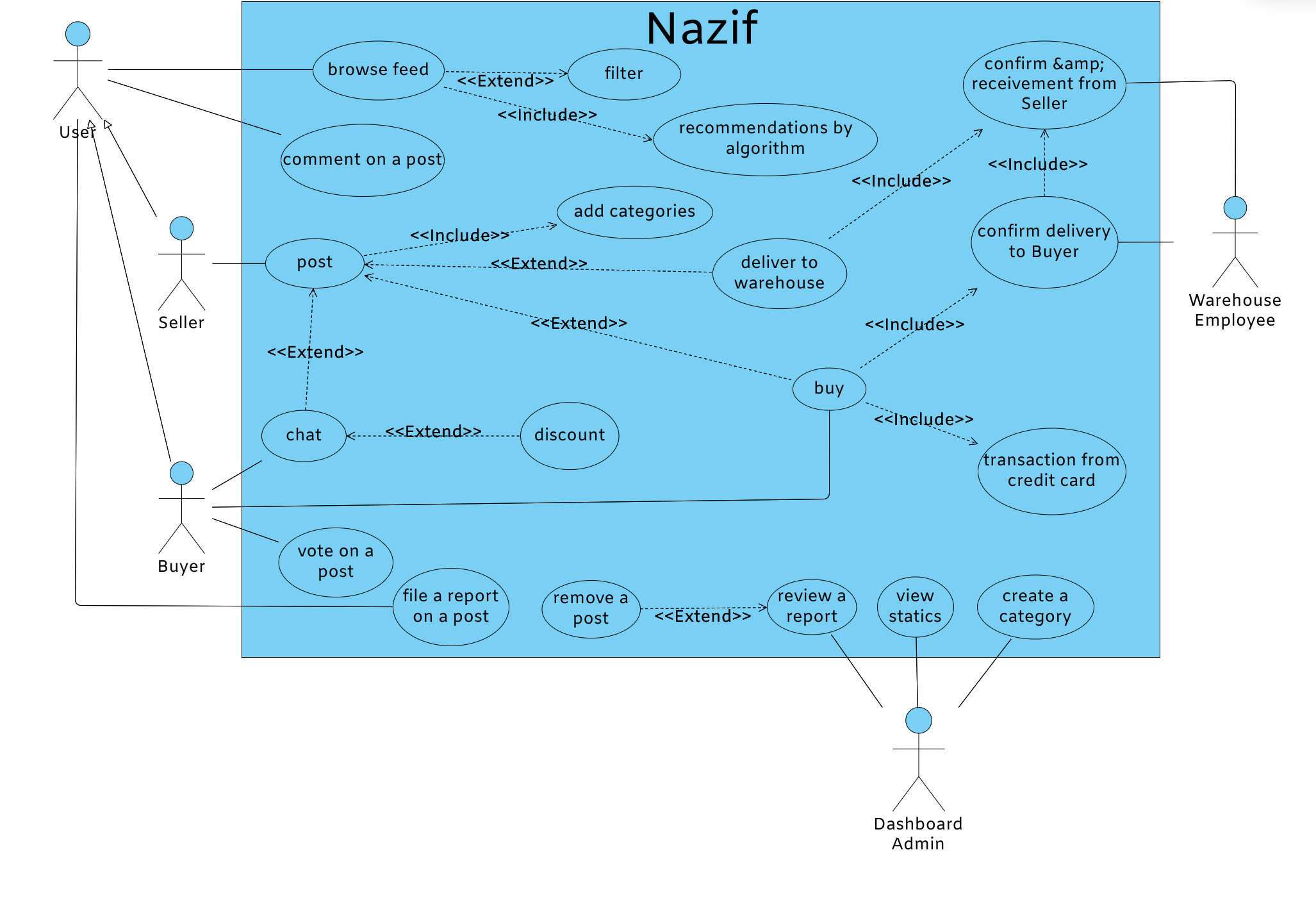
Use Case Diagram (مخطط حالات الاستخدام) هو أحد مخططات

UML (Unified Modeling Language) التي تستخدم لتوضيح التفاعلات بين المستخدمين (Actors) ونظام البرمجيات المحدد (System)، وذلك من خلال تمثيل سيناريوهات استخدام مختلفة. يهدف مخطط حالات الاستخدام إلى فهم وتوثيق متطلبات النظام وتحليل وظيفته من خلال تحديد الأنشطة التي يمكن للمستخدمين تنفيذها.

يتكون مخطط حالات الاستخدام من مجموعة من المكونات الرئيسية:

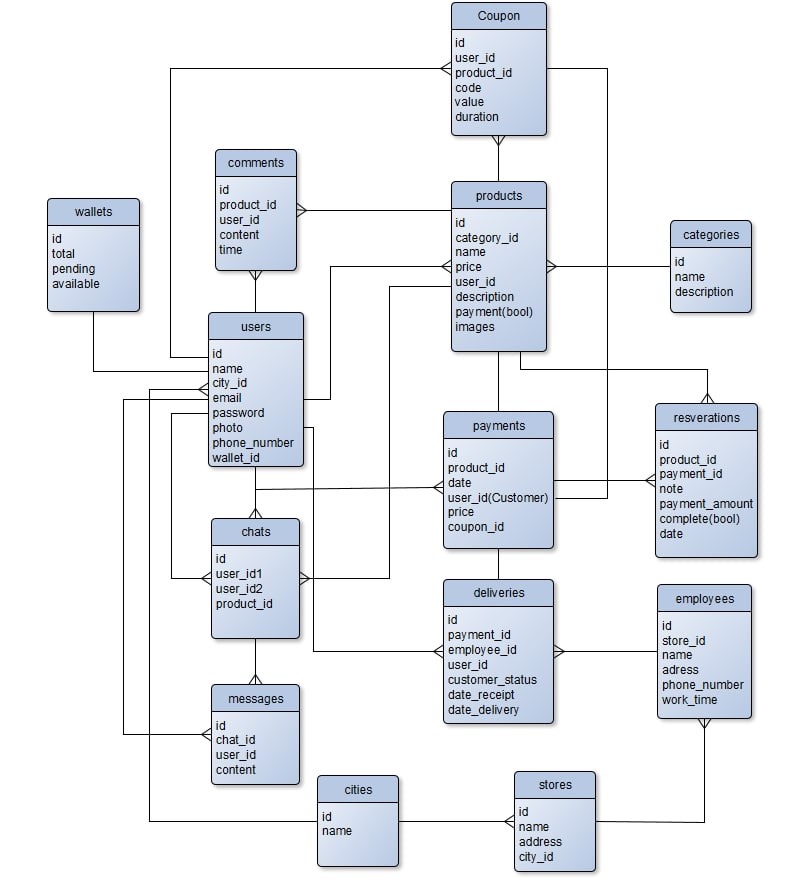
* المستخدمون (Actors): هم الكيانات الخارجية التي تتفاعل مع النظام. يمكن أن تكون المستخدمين أشخاصًا حقيقيين أو أنظمة أخرى أو حتى أجهزة.
* حالات الاستخدام (Use Cases): هي الأنشطة التي يقوم بها المستخدمون في النظام. تمثل كيفية استخدام المستخدم للنظام لتحقيق هدف محدد. توجد حالات استخدام رئيسية وحالات استخدام فرعية (مشتقة).
* العلاقات: توجد عدة أنواع من العلاقات بين المستخدمين وحالات الاستخدام، مثل العلاقة الرئيسية (Main) والعلاقة الشاملة (Include) والعلاقة الاختيارية (Extend)، والتي توضح الترابط بين حالات الاستخدام.

يعتبر مخطط حالات الاستخدام أداة هامة لتوضيح وتوثيق متطلبات النظام وتحليل الأنشطة المختلفة التي يمكن للمستخدمين القيام بها. يمكن استخدامه للتواصل بين المهندسين والمحللين وفرق التطوير والمستخدمين النهائيين لفهم وتوثيق وتوضيح تفاصيل عمل النظام.



الشكل 4-1 مخطط UseCase

**4-2 مخطط العلاقات ERD :**

****

الشكل 4-2 مخطط ERD

**3-4 جدول المستخدم :**

جدول يحتوي المعلومات الخاصة بالمستخدمين في النظام

الجدول 4-1 جدول المستخدم

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| name | اسم المستخدم | string | - |
| City\_id | معرف المدينة | objectId | مفتاح ثانوي |
| email | البريد الالكتروني | string | - |
| password | كلمة المرور | string | - |
| photo | صورة المستخدم | string | - |
| Phone\_number | رقم هاتف المستحدم | string | - |
| Wallet\_id | معرف الفاتورة | objectId | مفتاح ثانوي |

**4-4 جدول المنتج :**

جدول يحتوي تفاصيل المنتج.

الجدول 4-2 جدول المنتج

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| Category\_id | معرف نوع المنتج | objectId | مفتاح ثانوي |
| name | اسم المنتج | string | - |
| price | سعر المنتج | double | - |
| User\_id | معرف البائع | objectId | مفتاح ثانوي |
| Description | وصف المنتج | string | - |
| Payment | حالة المنتج | boolean | تم الشراء ام لا |
| images | صور للمنتج | string | - |

**4-5 جدول التعليقات :**

جدول يحتوي تعليقات على كل منتج.

الجدول 4-3 جدول التعليقات

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| product\_id | معرف المنتج | objectId | مفتاح ثانوي |
| user\_id | معرف المستخدم | objectId | مفتاح ثانوي |
| content | محتوى التعليق | string | - |
| time | تاريخ نشر التعليق | date | - |

**4-6 جدول المحادثات :**

جدول يحتوي على محادثات بين البائع والمشتري على منتج ما.

الجدول 4-4 جدول المحادثات

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| user\_id1 | معرف البائع | objectId | مفتاح ثانوي |
| user\_id2 | معرف المشتري | objectId | مفتاح ثانوي |
| product\_id | معرف المنتج | objectId | مفتاح ثانوي |

**4-7 جدول الرسائل :**

جدول يحتوي على الرسائل في المحادثات.

الجدول 4-5 جدول الرسائل

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| user\_id | معرف المستخدم | objectId | مفتاح ثانوي |
| chat\_id | معرف المحادثة | objectId | مفتاح ثانوي |
| content | محتوى الرسالة | string | - |

**4-8 جدول التوصيل :**

جدول يحتوي على تفاصيل عملية التوصيل.

الجدول 4-6 جدول التوصيل

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| payment\_id | معرف الدفع | objectId | مفتاح ثانوي |
| employee\_id | معرف الموظف | objectId | مفتاح ثانوي |
| user\_id | معرف المشتري | objectId | مفتاح ثانوي |
| customer\_status | حالة استلام المشتري | boolean | - |
| date\_receipt | تاريخ استلام المشتري | date | - |
| date\_delivery | تاريخ توصيل البائع | date | - |

**4-9 جدول المدفوعات :**

جدول يحتوي على تفاصيل عملية الدفع مثل (المنتج-المشتري-السعر-الخصم)

الجدول 4-7 جدول المدفوعات

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| product\_id | معرف الدفع | objectId | مفتاح ثانوي |
| date | تاريخ الدفع | date | - |
| user\_id | معرف المشتري | objectId | مفتاح ثانوي |
| price | السعر | double | - |
| coupon\_id | معرف الخصم | objectId | مفتاح ثانوي |

**4-10 جدول الموظف :**

جدول يحتوي على المعلومات الخاصة بالموظف (اسم-عنوان-مكان العمل-رقم الهاتف)

الجدول 4-8 جدول الموظف

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| store\_id | معرف المستودع | objectId | مفتاح ثانوي |
| name | اسم الموظف | string | - |
| address | عنوان الموظف | string | - |
| phone\_number | رقم هاتف الموظف | string | - |
| work\_time | وقت الدوام الرسمي | string | - |

**4-11 جدول المستودع :**

جدول يحتوي على التفاصيل الخاصة بالمستودع (اسم-موقع-المدينة)

الجدول 4-9 جدول المستودع

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| name | اسم المستودع | string | - |
| address | عنوان المستودع | string | - |
| city\_id | المدينة المتواجد فيها المستودع | objectId | مفتاح ثانوي |

**4-12 جدول الخصومات :**

جدول يحتوي تفاصيل الخصم المقدم للزبون.

الجدول 4-10 جدول الخصومات

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| user\_id | معرف المشتري | objectId | مفتاح ثنوي |
| product\_id | معرف المنتج | objectId | مفتاح ثانوي |
| code | رمز الخصم | string | - |
| value | قيمة الخصم | int | - |
| duration | صلاحية الخصم | date | - |

**4-13 جدول المدينة :**

جدول يحتوي على المدن التي تتواجد فيها المستودعات.

الجدول 4-11 جدول المدينة

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| name | اسم المدينة | string | - |

**4-14 جدول الفاتورة :**

جدول يحتوي على تفاصيل الفاتورة.

الجدول 4-12 جدول الفاتورة

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| total | القيمة الاجمالية | double | **-** |
| pending | أموال المعلقة | double | **-** |
| available | أموال المتاحة | double | **-** |

**4-15 جدول حالة الدفع :**

جدول يحتوي على معلومات عملية الدفع

الجدول 4-13 جدول حالة الدفع

|  |  |  |  |
| --- | --- | --- | --- |
| الحقل | الوصف | النوع | ملاحظات |
| id | معرف | objectId | مفتاح أساسي |
| product\_id | معرف المنتج | objectId | مفتاح ثانوي |
| payment\_id | معرف عملية الدفع | objectId | مفتاح ثانوي |
| note | ملاحظات | string | - |
| payment\_amount | كمية الدفع | double | - |
| complete | مكتمل | boolean | اكتملت أم لا |
| date | تاريخ الدفع | date | - |

**المراجع :**

1. CSS Docs

[CSS: Cascading Style Sheets | MDN (mozilla.org)](https://developer.mozilla.org/en-US/docs/Web/CSS)

1. Dart Documentation

[Dart programming language | Dart](https://dart.dev/)

1. Express Js

[Express 5.x - API Reference (expressjs.com)](https://expressjs.com/en/5x/api.html)

1. Flutter Documentation

[Flutter - Build apps for any screen](https://flutter.dev/)

1. HTML Docs

[HTML: HyperText Markup Language | MDN (mozilla.org)](https://developer.mozilla.org/en-US/docs/Web/HTML)

1. Material UI

[React Components - Material UI (mui.com)](https://mui.com/material-ui/)

1. Mongo DB

[MongoDB Documentation](https://www.mongodb.com/docs/)

1. Node Js

[Documentation | Node.js (nodejs.org)](https://nodejs.org/en/docs)

1. Swagger

[API Documentation Made Easy - Get Started | Swagger](https://swagger.io/solutions/api-documentation/)