الجمهـوريـة العربيــة الســـورية

المعهد العالي للعلوم التطبيقية والتكنولوجيا

قسـم النظم المعلوماتية

العام الدراسي 2024/2023

**الدراسة التصميمية**

تصميم النظام

إدارة حالة مشاريع المعهد العالي

تقديم

حسن بهجت خضور

إشراف

د.مصطفى دقاق م. محمود الياس

6/8/2024

الفصل الأول

الانماط التصميمية المستخدمة

نبيّن في هذا الفصل شرحا نظريا للانماط التصميمية والبنية المعمارية وبعض المنهجيات المتبعة في تنجيز النظام.

1.1 البنية المعمارية والانماط التصميمية

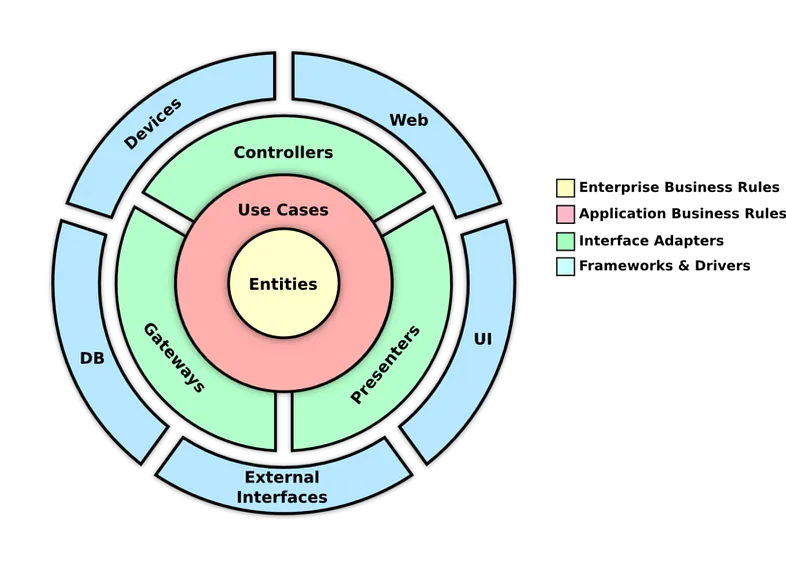
2.1 البنية المعمارية النظيفة – Clean Architecture

1.2.1 مفهوم البنية المعارية النظيفة

البنية المعمارية النظيفة هي فلسفة تصميم برمجيات تهدف إلى إنشاء أنظمة قابلة للصيانة والتوسعة، وتكون مستقلة عن تفاصيل التنفيذ. تعتمد هذه البنية على مبدأ فصل مكونات النظام إلى طبقات مستقلة،مركزة على قواعد العمل, مما يسهم في تقليل التبعيات بين هذه المكونات. يتيح هذا التصميم لكل مكون أن يتطور أو يتغير دون التأثير على باقي النظام، مما يؤدي إلى نظام أكثر مرونة وقابلية للتكيف مع المتغيرات.

وينتج عن استخدا مالبنية المعمارية النظيفة نظام له الخواص التالية :

* مستقل عن إطار العمل
* مستقل عن آلية تخزين البيانات
* نظام قابل للاختبار



الشكل 1 : مكونات البنية المعمارية النظيفة

2.2.1 الاعتمادية في البنية المعمارية النظيفة

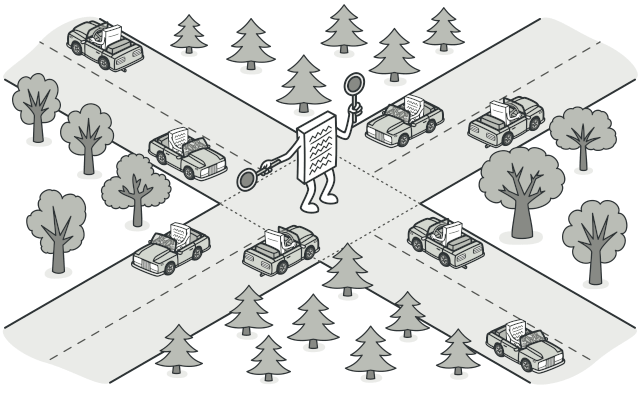
3.2.1 مكونات البنية المعمارية النظيفة

* الكيانات Entities : تعد الكيانات الأساس الذي يقوم عليه منطق الأعمال داخل النظام. الكيانات هي النماذج التي تعبر عن قواعد العمل الأساسية، وهي مصممة لتكون مستقلة عن التقنيات أو الأطر الخارجية.
* حالات الاستخدام Use Cases :

3.1 النمط التصميمي الوسيط Mediator Design Pattern

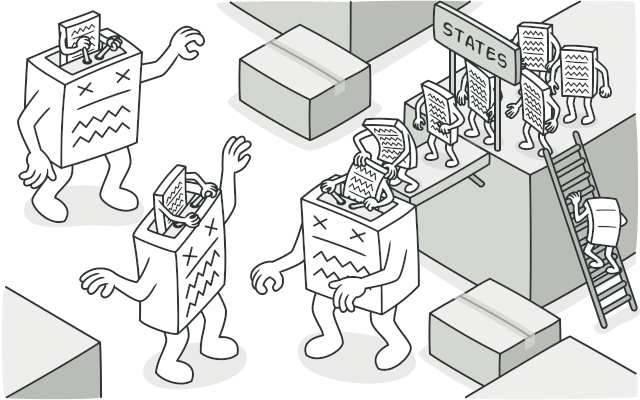
النمط التصميمي **الوسيط (Mediator)** هو نمط تصميمي سلوكي يهدف إلى تنظيم التفاعلات المعقدة بين الكائنات في نظام برمجي. في كثير من الأحيان، تكون العلاقات بين الكائنات في نظام برمجي معقدة، حيث يتواصل كل كائن مع العديد من الكائنات الأخرى. هذا يؤدي إلى وجود تشابك كبير بين الكائنات ويجعل من الصعب تعديل أو صيانة النظام.

ويأتي دور النمط الوسيط لتبسيط هذا التعقيد عن طريق إدخال كائن وسيط واحد يتحكم في جميع التفاعلات بين الكائنات. بدلاً من أن تتواصل الكائنات مباشرة مع بعضها البعض، يتم توجيه جميع التفاعلات من خلال الوسيط. هذا يقلل من التبعيات بين الكائنات ويجعل النظام أكثر تنظيماً وأسهل في التعديل والتوسيع.

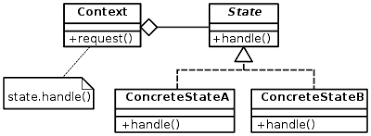


الشكل 2 : مثال توضيحي عن النمط الوسيط

4.1 النمط التصميمي الحالة State Design Pattern



الشكل 3 : مثال توضيحي عن نمط الحالة



الشكل 4 : مخطط صفوف تجريدي يصف نمط الحالة

5.1 فصل مسؤوليات الأوامر والاستعلامات Command Query Responsibility Segregation

الفصل الثاني

الأدوات و أطر العمل

نبيّن في هذا الفصل اأدوات وأطر العمل التي استخدمناها لتنجيز النظام.

1.1 إطار العمل .Net Core

2.2 إطار العمل Angular

3.2 قاعدة المعطيات SQL Server

4.2 نمط الخيارات Option Pattern

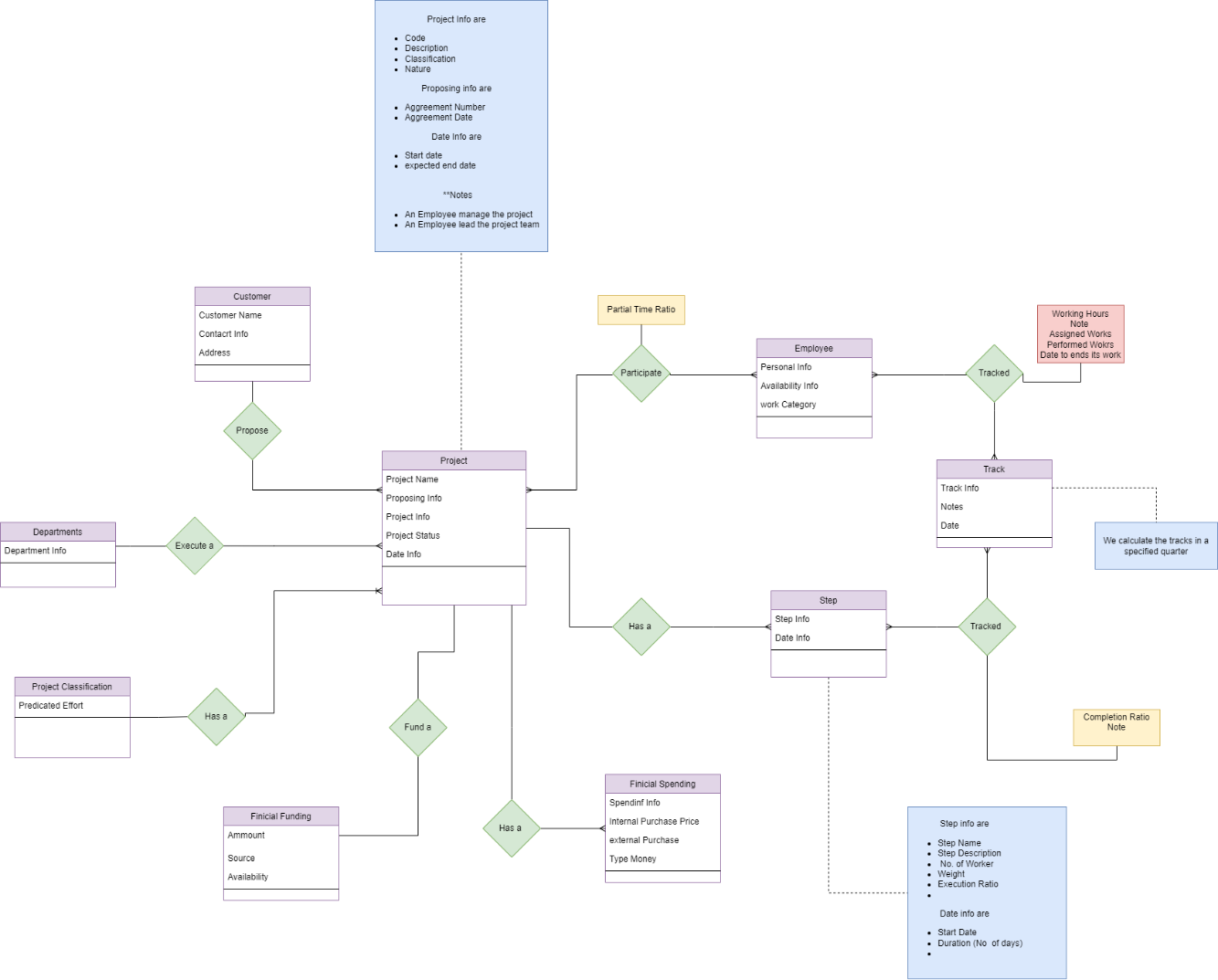
5.2 JWT JSON Web Token

الفصل الثالث

التصميم

نبيّن في هذا الفصل اأدوات وأطر العمل التي استخدمناها لتنجيز النظام.

* 1. مخطط العلاقات بين الكائنات Entity Relationship Diagram



2.1 قاعدة البيانات

