

# نظام إلكتروني لإدارة تذاكر الصيانة في المعهد العالي

مشروع يهدف إلى تطوير نظام ويب متكامل لإدارة تذاكر الصيانة في المعهد العالي للعلوم التطبيقية والتكنولوجيا.



# ملخص المشروع

يُعد تبسيط عمليات الصيانة في المؤسسات التعليمية وتحويلها من إجراءات ورقية معقدة إلى نظام إلكتروني متكامل أمراً بالغ الأهمية لضمان سرعة الاستجابة، وتتبع الطلبات، ورفع كفاءة فريق الدعم الفني. انطلاقاً من هذه الحاجة، قمنا بتصميم نظام ويب متكامل لإدارة تذاكر الصيانة في المعهد العالي.

## التكامل السلس

يعمل النظام بتكامل سلس مع نظام المصادقة الخاص بالمعهد، مما يضمن تجربة مستخدم مخصصة وآمنة.

## إشعارات ذكية

يضمن آلية إشعارات ذكية تُعلم مسؤولي الصيانة فوراً بالطلبات الجديدة، وتُبلغ المستخدمين تلقائياً بأي تحديث لحالة تذاكرهم.

## تجربة مستخدم محسّنة

بُني النظام ليكون سريعاً وآمناً وسهل الاستخدام، مع حماية بيانات المستخدمين عبر تشفير متقدم، وتوافقه مع جميع الأجهزة.

## تحول شامل

يحقق هذا الحل تحولاً شاملاً في إجراءات الصيانة؛ بتقليل الوقت الضائع، وتعزيز الشفافية بين المستخدمين وفريق الدعم، مما ينعكس إيجاباً على جودة الخدمات.



## أهداف المشروع

يهدف مشروع "تذاكر صيانة" إلى أتمتة عمليات إدارة طلبات الصيانة في المعهد العالي عبر نظام مركزي وفعال لتسجيل ومتابعة التذاكر، بهدف تبسيط العملية من الإنشاء حتى الحل مع تقليل التدخل البشري والحد من الأخطاء.



### تحسين التواصل

إشعارات فورية للموظفين وفرق الصيانة لضمان معالجة سريعة ودقيقة للمشكلات.



### أتمتة العمليات

تبسيط إدارة طلبات الصيانة من الإنشاء حتى الحل وتقليل التدخل البشري والأخطاء.



### التحول الرقمي

تطبيق مفاهيم هندسة البرمجيات وإدارة قواعد البيانات والاستفادة من أنظمة المصادقة مثل LDAP.



### تعزيز الشفافية

لوحات تحكم تتيح تتبع حالة الطلبات بسهولة.

# المتطلبات الوظيفية

تم تصميم النظام لتلبية مجموعة من المتطلبات الوظيفية الأساسية لضمان كفاءة وسهولة الاستخدام.

## 3. معالجة التذاكر

- إشعار فوري لمسؤولي الصيانة عند إنشاء تذكرة جديدة.
- السماح لمسؤولي الصيانة بتغيير حالة التذاكر.
- إمكانية إضافة ملاحظات وتحديثات على التذكرة.
- إرسال إشعارات للموظفين عند تحديث حالة تذاكرهم.

## 4. لوحات التحكم

- لوحة تحكم شخصية للموظفين (تذاكرهم فقط).
- لوحة تحكم شاملة لمسؤولي الصيانة (جميع التذاكر، إحصاءات، تصفية).

## 1. تسجيل الدخول والصلاحيات




- السماح بتسجيل الدخول باستخدام بيانات LDAP.
- تعبئة بيانات المستخدم تلقائياً من LDAP.
- توفير مستويين للصلاحيات: موظف عادي ومسؤول صيانة.

## 2. إدارة تذاكر الصيانة

- واجهة لإنشاء تذاكر جديدة (وصف، نوع الجهاز، معرّف اختياري، مرفقات).
- توليد أرقام تذاكر فريدة تلقائياً.
- السماح بالبحث عن التذاكر السابقة (رقم، تاريخ، حالة).

# المتطلبات غير الوظيفية

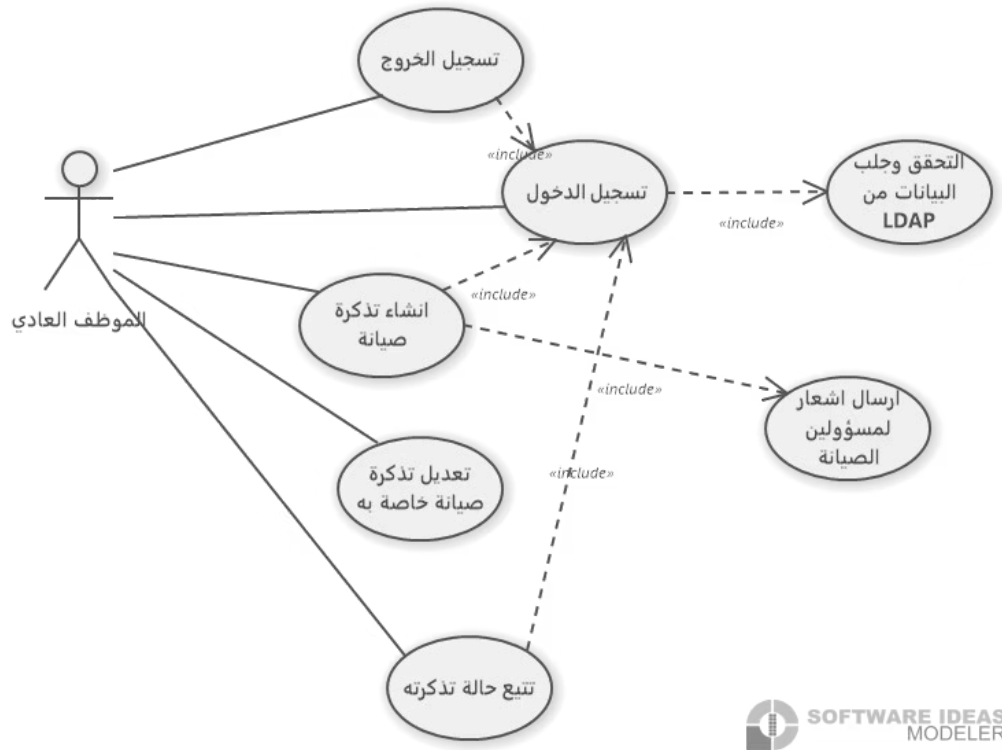
بالإضافة إلى الوظائف الأساسية، يلتزم النظام بمعايير أداء وأمان وتوافقية عالية لضمان تجربة مستخدم موثوقة.

الأداء 	الأمان 	التوافقية 
<ul style="list-style-type: none"><li>استجابة خلال أقل من ثانيتين.</li><li>دعم 100 مستخدم متزامن على الأقل.</li><li>وقت تشغيل 99.9%.</li><li>توفر دائم على مدار الساعة.</li></ul>	<ul style="list-style-type: none"><li>تشفير TLS 1.2+ لجميع الاتصالات.</li><li>التحقق من الصلاحيات قبل كل عملية حساسة.</li><li>تخزين كلمات المرور بتقنيات التجزئة الآمنة.</li></ul>	<ul style="list-style-type: none"><li>يعمل على أحدث إصدارين من Google Chrome، Mozilla Firefox، Microsoft Edge.</li><li>متجاوب ويعمل بكفاءة على الأجهزة المحمولة.</li><li>استخدام NET و React لتسهيل التطوير المستقبلي.</li></ul>

# الدراسة التحليلية: مخططات حالات الاستخدام

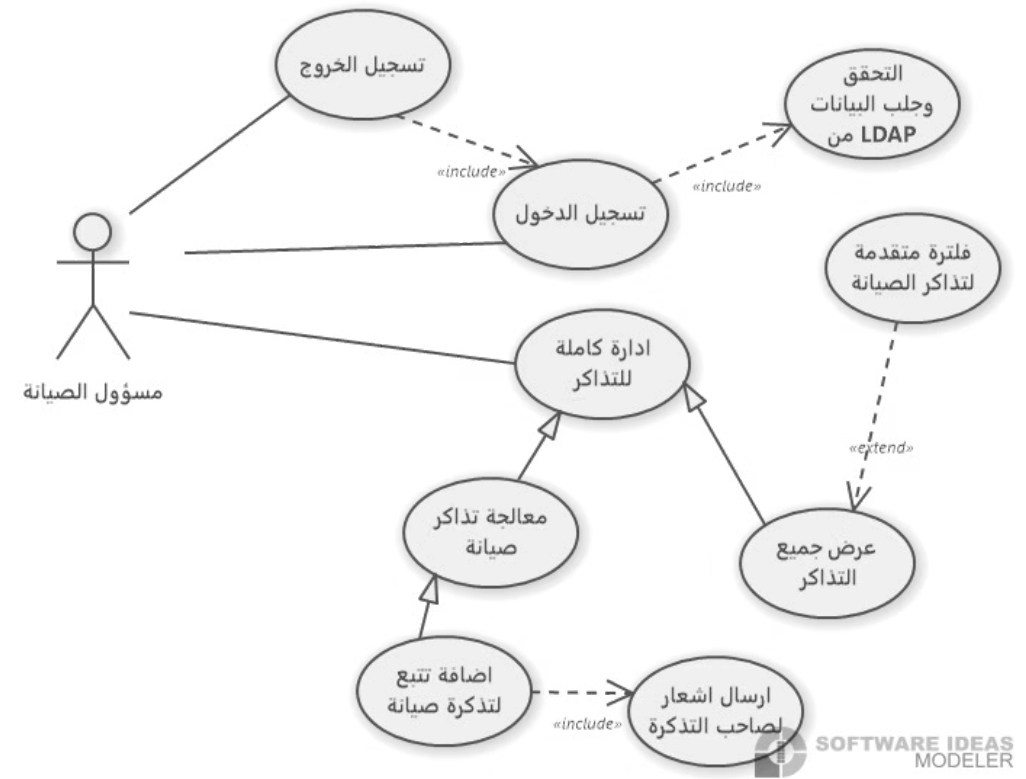
توضح مخططات حالات الاستخدام التفاعلات الرئيسية بين المستخدمين والنظام، مقسمة حسب أدوارهم المختلفة.

مخطط حالات الاستخدام للموظف



يوضح هذا المخطط المهام التي يمكن للموظف العادي القيام بها، مثل تسجيل الدخول، إنشاء تذكير جديدة، وتتبع حالة تذكيره.

مخطط حالات الاستخدام لمسؤول الصيانة



يبين هذا المخطط الصلاحيات الموسعة لمسؤول الصيانة، بما في ذلك معالجة التذاكر، وتغيير حالتها، وعرض جميع التذاكر.

مخطط حالات الاستخدام لمدير النظام

يصور هذا المخطط المهام الإدارية لمدير النظام، مثل استعراض الإحصاءات وتصفية التذاكر.

# الدراسة التحليلية: وصف حالات الاستخدام

نستعرض هنا وصفاً نصياً مفصلاً لبعض حالات الاستخدام الرئيسية في النظام، مع توضيح السيناريوهات الناجحة والبديلة.



## استعراض الإحصاءات

يريد المدير رؤية إحصاءات عن التذاكر. يعرض النظام لألحة الإحصاءات التي تحوي مخططات ورسومات. في حال عدم وجود تذاكر، يرجع النظام رسالة "Tickets Not Found".



## تسجيل دخول الموظف

يبدأ الموظف بإدخال بياناته (الاسم، كلمة المرور، القسم). يتحقق النظام من المعلومات عبر LDAP، وفي حال النجاح، يدخل المستخدم ويعطيه Token خاصة به. إذا كانت البيانات غير صحيحة، يرسل النظام تنبيهاً بفشل تسجيل الدخول مع توضيح السبب.



## معالجة تذكرة

يريد موظف الصيانة تعديل حالة تذكرة. يعرض النظام جميع التذاكر، يختار الموظف التذكرة، ثم يعرض النظام واجهة لإضافة تتبع. بعد إدخال البيانات، يتحقق النظام ويرسل إشعاراً للموظف صاحب التذكرة.



## إنشاء تذكرة جديدة

يرغب الموظف بإنشاء تذكرة جديدة، فيدخل البيانات المطلوبة (ملاحظات، نوع الجهاز، مرفقات). يتحقق النظام من صحة البيانات ويرسل رسالة بنجاح العملية وإشعاراً لموظفي الصيانة.

# بيئة العمل والدراسة التصميمية: البنية المعمارية

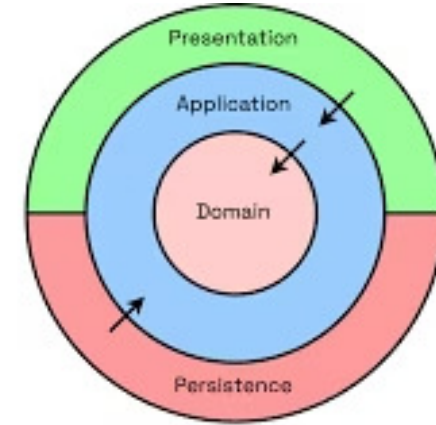
تعتبر البنية المعمارية للنظام أساساً لضمان المرونة، قابلية الصيانة، والتكيف مع التغييرات المستقبلية.

## مفهوم البنية المعمارية النظيفة

هي فلسفة لتصميم البرمجيات تهدف إلى إنشاء نظام برمجي قابل للصيانة ومستقل عن تفاصيل التنفيذ. تعزز استقلالية المكونات (واجهات المستخدم، منطق الأعمال، تخزين البيانات) لتتطور بشكل مستقل دون التأثير على النظام بأكمله.

- **استقلالية واجهات المستخدم:** يمكن تغيير واجهة المستخدم دون التأثير على منطق الأعمال الأساسي.
- **استقلالية تخزين البيانات:** إمكانية تغيير آلية تخزين المعطيات دون التأثير على بقية أجزاء النظام.
- **نظام قابل للاختبار:** إمكانية اختبار قواعد العمل بدون واجهة المستخدم أو قاعدة البيانات.

## مكونات البنية النظيفة



- **الكيانات (Entities):** تمثل عناصر الأعمال الأساسية وقواعد التطبيق.
- **حالات الاستخدام (Use Cases):** تحدد منطق عمل التطبيق وتنظم التفاعلات بين الكيانات.
- **محولات الواجهة (Interface Adapters):** تسد الفجوة بين حالات الاستخدام والعالم الخارجي.
- **الأطر والمحركات (Frameworks and Drivers):** الطبقة الخارجية التي توفر الأدوات والبنية التحتية.



# الأدوات المستخدمة في المشروع

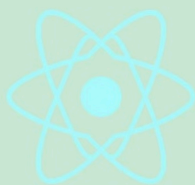
تم اختيار مجموعة من الأدوات والتقنيات الحديثة لتطوير النظام، مما يضمن الكفاءة والأداء العالي.



## ASP.NET

ASP.NET Web API

إطار عمل من Microsoft لبناء خدمات HTTP يمكن استهلاكها من قبل العديد من العملاء، يدعم RESTful API.



React


مكتبة JavaScript مفتوحة المصدر لبناء واجهات المستخدم الديناميكية والتفاعلية، تعتمد على المكونات و Virtual DOM.



## SQL Server

SQL Server

نظام إدارة قواعد بيانات علائقية من Microsoft، يوفر أداءً ممتازاً وسعة تخزين عالية.



## JSON Web Token

Json Web Token (JWT)

يستخدم لنقل المعلومات بين طرفين بشكل آمن عبر الويب، كآلية مصادقة عديمة الحالة.



## SignalR

SignalR

مكتبة مفتوحة المصدر من Microsoft لإنشاء تطبيقات ويب تفاعلية في الوقت الحقيقي (Real-time) باستخدام WebSockets.



## MailKit

MailKit

مكتبة مفتوحة المصدر لمعالجة البريد الإلكتروني في تطبيقات .NET، تدعم بروتوكولات البريد الحديثة.

# القسم العملي: منهجية التنفيذ

تم اتباع منهجية واضحة في بناء النظام لضمان قابلية الاختبار والتعديل، مع التركيز على البنية المعمارية النظيفة والتصميم المقاد بالمجال.

## البنية المقترحة

الاعتماد على البنية المعمارية النظيفة لفصل قواعد العمل وحالات الاستخدام عن الخيارات التشغيلية، مما يتيح استبدالها عند الحاجة. كما تم الاعتماد على التصميم المقاد بالمجال لاحترام قواعد العمل بشكل صحيح.

## تنفيذ التطبيق الخلفي (Back-end)

- **هيكلية المجلدات العامة:** تتضمن عناصر الحل، الرماز المصدري، والاختبارات لضمان بيئة عمل مشتركة ومنظمة.
- **هيكلية الرماز المصدري:** تحترم منهجية البنية المعمارية النظيفة بفصل الطبقات (طبقة التطبيق، طبقة البنية التحتية، طبقة المجال، طبقة العرض).
- **إدارة الأدوار:** ثلاثة أدوار أساسية (موظف عادي، مسؤول صيانة، مدير نظام) مع صلاحيات محددة.
- **المصادقة والتفويض:** باستخدام رموز JWT مع أسلوب تحميل الصلاحيات ضمن الرمز لتقليل العبء على الشبكة.
- **تخزين المعطيات:** استخدام Entity Framework مع SQL Server، وتحقيق التزامن بمنهجية القفل المتفائل.
- **الإشعارات:** استخدام SignalR لتسهيل الاتصال في الوقت الحقيقي بين المستخدمين والمخدم.

## تنفيذ الواجهات الأمامية (Front-end)

- **منهجية العمل:** استخدام مكتبة React لتنفيذ الواجهات الأمامية بمنهجية تطبيق الصفحة الواحدة (Single Page Application) وتقسيم التطبيق لمكونات.
- **معالجة الاتصال مع التطبيق الخلفي:** استخدام مكتبة Axios لتحقيق الاتصال وإرسال طلبات HTTP، مع فصل المكونات عن المكتبة عبر الصف (ClientContext).
- **معالجة الأخطاء:** إنشاء معترض للطلبات (interceptor) لمعالجة الأخطاء القادمة من التطبيق الخلفي قبل وصولها للواجهة.

## معايير تصميمية متبعة

- النظام مطابق للمتطلبات الوظيفية وغير الوظيفية.
- قابل للتوسع بدرجة كبيرة ويتمتع بالعمومية.
- غني بالوظائف والتقنيات الحديثة لجذب الموظفين لاستخدامه.

