

الجمهورية العربية السورية
المعهد العالي للعلوم التطبيقية والتكنولوجيا
قسم النظم المعلوماتية
العام الدراسي 2024/2025

مشروع سنة رابعة
هندسة البرمجيات والذكاء الصناعي

مساعد شخصي لتطوير المهارات

تقديم

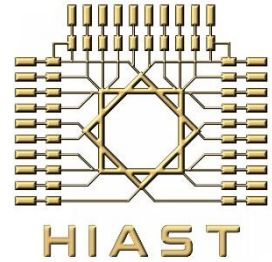
عمّار سمرة

إشراف

د. رياض سنبل

2025/8/9

Syrian Arab Republic
Higher Institute for Applied Sciences and Technology
Department of Information Systems
Academic Year 2024/2025



Fourth Year Project
Software Engineering and Artificial Intelligence

A Personal Assistant for Skill Development

Submitted by
Ammar Samrah

Supervised by
Dr. Riad Sonbol

August 9, 2025

الإهداء

إلى من غرسوا فيّ حب العلم، وسقوا ذلك الغرس بالصبر والدعاء حتى أثمر...
إلى أبي، سندي الأول، وأمي، نبع الحنان...
إلى من كانت جزءاً لا يتجزأ من رحلتي، تشاركني الأمل وتخفف عني عناء الطريق...
إلى أختي الغالية...
وإلى من جعلوا سنوات الدراسة أجمل بذكرياتهم ودعمهم الذي لا ينقطع...
إلى أصدقائي الأوفياء...
إليكم جميعاً، أهدي ثمرة هذا الجهد.

كلمة شكر

أبتدئ بحمد الله وشكره على نعمه التي لا تُحصى، فما كان لهذا العمل أن يتم لولا توفيقه وفضله. أتقدم بجزيل الشكر والتقدير والعرفان إلى أستاذي ومشرفي الدكتور رياض سنبل، الذي لم يبخل عليّ بعلمه وتوجيهاته القيمة ونصائحه السديدة طوال فترة إنجاز هذا المشروع. فلقد كان لإشرافه ومتابعته الدؤوبة الأثر الأكبر في إخراج هذا العمل إلى النور.

كما أتوجه بالشكر الجزيل إلى أساتذتي الأكارم في قسم النظم المعلوماتية في المعهد العالي للعلوم التطبيقية والتكنولوجيا، على ما قدموه لنا من علم ومعرفة خلال سنوات دراستنا.

ولا يفوتني أن أتقدم بالشكر إلى الأهل والأصدقاء الذين كانوا خير داعم ومحفز، والذين أحاطوني بالتشجيع والدعاء، فلهم مني كل الحب والتقدير.

الخلاصة

يقدم هذا المشروع منصة "SkillCraft"، وهي تطبيق ويب تعليمي مبتكر، تم تصميمه لمعالجة التحدي الشائع الذي يواجهه طلاب الهندسة والمهندسين: غياب خارطة طريق واضحة ومنظمة لاكتساب المهارات التقنية والمهنية.

توفر المنصة مسارات تعليمية تفاعلية مصممة بعناية، حيث تقود المستخدم خطوة بخطوة عبر مراحل تعلم أي مهارة. كل مسار يوضح المفاهيم الأساسية، ويقدر الوقت اللازم لإنجاز كل مرحلة، مما يمنح المتعلم رؤية شاملة للرحلة التعليمية. ولتعزيز عملية التعلم، تتيح المنصة للمستخدمين تتبع إنجازاتهم بشكل مرئي، وتقييم مدى استيعابهم للمحتوى عبر اختبارات قصيرة مدمجة.

بذلك، لا يقتصر دور "SkillCraft" على كونه مجرد مستودع للمعلومات، بل يعمل كمرشد شخصي يمكن المستخدمين من بناء مهاراتهم بثقة وكفاءة، ويزيل عنهم حيرة البدايات وتشتت المصادر.

Abstract

This project introduces "SkillCraft," an innovative educational web application designed to address a common challenge faced by engineering students and professionals: the lack of a clear, structured roadmap for acquiring new technical and professional skills.

The platform provides carefully designed, interactive learning paths (Roadmaps) that guide users step-by-step through the phases of mastering a skill. Each roadmap clarifies core concepts, estimates the time commitment for each stage, and gives the learner a comprehensive view of their educational journey. To reinforce learning, the platform enables users to visually track their progress and evaluate their understanding of the material through integrated quizzes.

Therefore, SkillCraft transcends being a simple repository of information to act as a personal guide, empowering users to build their skills with confidence and efficiency. It eliminates the initial confusion of where to begin and the problem of scattered learning resources.

المحتويات

10	قائمة الأشكال
11	قائمة الجداول
12	الرموز والاختصارات
13	مقدمة عامة
14	الفصل الأول
14	1.1- المشكلة وأهميتها
14	2.1- الحل المقترح: منصة SkillCraft
15	3.1- رؤية المشروع وأهدافه
15	4.1- الجمهور المستهدف
16	الفصل الثاني
16	1.2- تحليل المنصات التعليمية الحالية
16	1.1.2- موقع Roadmap.sh
17	2.1.2- منصات الكورسات (FreeCodeCamp, Codecademy)
18	3.2- خلاصة الدراسة
19	الفصل الثالث
19	1.3- تحليل النظام
19	2.3- المتطلبات
19	1.2.3- المتطلبات الوظيفية
21	2.2.3- المتطلبات غير الوظيفية
22	3.3- مخطط حالات الاستخدام
24	4.3- الوصف النصي لحالات الاستخدام ومخططات التسلسل
24	1.3.3- حالة الاستخدام: تسجيل دخول
27	2.3.3- حالة الاستخدام: إنشاء حساب جديد
29	3.3.3- حالة الاستخدام: إضافة Roadmap إلى الملف الشخصي
32	4.3.3- حالة الاستخدام: توليد roadmap عبر الذكاء الاصطناعي
35	الفصل الرابع
35	1.4- البنية التصميمية (Software Architecture)
35	1.1.4- معمارية العميل-الخادم (Client-Server Architecture)
36	2.1.4- المعمارية المتجانسة المقسمة (Modular Monolith)

38	2.4- بيئة العمل وأدوات التطوير.....
38	1.2.4- بيئات التطوير المتكاملة (IDEs) :
39	2.2.4- نظام إدارة الإصدارات (Version Control System)
39	3.2.4- خدمات الذكاء الاصطناعي (Artificial Intelligence Services)
40	3.4- أطر العمل والمكتبات المستخدمة
40	1.3.4- الواجهة الخلفية (Backend)
41	2.3.4- الواجهة الأمامية (Frontend)
41	4.4- تقنيات قواعد البيانات
43	5.4- الأنماط التصميمية المتبعة (Design Patterns)
43	1.5.4- نمط حقن التبعية (Dependency Injection - DI)
43	2.5.4- نمط المستودع (Repository Pattern)
43	3.5.4- نمط وحدة العمل (Unit of Work Pattern)
44	4.5.4- نمط كائن نقل البيانات (Data Transfer Object - DTO)
44	5.5.4- نمط الاستراتيجية (Strategy Pattern)
45	6.5.4- نمط المصنع (Factory Pattern)
45	7.5.4- نمط المحول (Adapter Pattern)
46	الفصل الخامس
46	1.5- تنفيذ الواجهة الخلفية (Backend Implementation)
46	1.1.5- طبقة واجهة برمجة التطبيقات الموحدة (SkillCraft.Api)
47	2.1.5- تنفيذ الوحدات (Modules)
50	2.5- تنفيذ الواجهة الأمامية (Frontend Implementation)
50	1.2.5- التواصل مع الواجهة الخلفية.....
51	3.5- تنفيذ قواعد البيانات
54	الفصل السادس
54	1.6- الاختبارات الوظيفية (Functional Testing)
54	1.1.6- استراتيجية الاختبار الوظيفي.....
55	2.1.6- نتائج الاختبارات الوظيفية.....
55	2.6- الاختبارات غير الوظيفية (Non-Functional Testing)
56	1.2.6- اختبارات الأداء (Performance Testing)
56	2.2.6- نتائج اختبارات الأداء
61	الفصل السابع
61	1.7- الخاتمة

62..... 2.7- الآفاق المستقبلية

63..... المراجع

قائمة الأشكال

22	رسم توضيحي 1 حالات الاستخدام للمستخدم المتلقي
23	رسم توضيحي 2 حالات الاستخدام لمحرر المحتوى
24	رسم توضيحي 3 مخطط حالات الاستخدام لمدير النظام
26	رسم توضيحي 4 مخطط التالي لعملية تسجيل الدخول
29	رسم توضيحي 5 مخطط التالي لعملية إنشاء حساب جديد
31	رسم توضيحي 6 مخطط التالي لعملية إضافة roadmap إلى الملف الشخصي
34	رسم توضيحي 7 مخطط التالي لحالة الاستخدام لتوليد roadmap بالذكاء الاصطناعي
37	رسم توضيحي 8 هيكلية المشروع
38	رسم توضيحي 9
47	رسم توضيحي 10 User repository
49	رسم توضيحي 11 تنفيذ RoadmapDbContext
52	رسم توضيحي 12 حقن UsersDbContext
55	رسم توضيحي 13 نتائج بعض الاختبارات على وحدة إدارة المستخدمين
57	رسم توضيحي 14 soak test on get all roadmaps
58	رسم توضيحي 15 spike test on get all roadmaps
59	رسم توضيحي 16 stress test on get roadmap by id
60	رسم توضيحي 17 load test on get all roadmaps

قائمة الجداول

18	جدول 1 مقارنة بين موقعين ومشروعنا.....
25	جدول 2 السيناريو الرئيسي لحالة تسجيل دخول.....
27	جدول 3 المسار الرئيسي لحالة إنشاء حساب جديد.....
30	جدول 4 المسار الرئيسي لحالة إضافة roadmap إلى الملف الشخصي.....
42	جدول 5 مقارنة بين SQL و NoSQL.....
56	جدول 6 soak test.....
57	جدول 7 spike test.....
58	جدول 8 stress test.....
59	جدول 9 load test.....

الرموز والاختصارات

عربي	English	الرمز
طبقة تجريدية	Abstraction Layer	
فرع	Branch	
البرمجة النصية عبر المواقع	Cross-Site Scripting	XSS
نموذج بيانات	Entity	
ترميز كائنات جافا سكريبت	JavaScript Object Notation	JSON
نموذج لغوي كبير	Large language model	LLM
مرحلة	Milestone	
أداة الربط الكائني-العلائقي	Object-Relational Mapper	ORM
خارطة طريق	Roadmap	
بنية قائمة على الأدوار	Role-Based Access Control	RBAC
خطوة	Step	
أوراق الأنماط الرائعة نحويًا	Syntactically Awesome Style Sheets	SASS

مقدمة عامة

في عصر يتسم بالتطور التكنولوجي المتسارع والمنافسة الشديدة في سوق العمل، أصبح التعلم المستمر وتطوير المهارات ضرورة حتمية للمهندسين والتقنيين الطامحين للتميز. ومع ذلك، يواجه الكثيرون تحديًا كبيرًا يتمثل في غياب التوجيه الواضح والمصادر المنظمة، مما يؤدي إلى تشتت الجهود وإهدار الوقت دون تحقيق نتائج ملموسة. غالبًا ما يجد المتعلم الذاتي نفسه غارقًا في بحر من المعلومات، يتساءل: "من أين أبدأ؟" و "ما هي الخطوة التالية؟".

من هذا المنطلق، يأتي مشروع "SkillCraft" ليقدم حلاً مبتكرًا ومنظمًا لهذه المشكلة. "SkillCraft" هو منصة ويب تعليمية تفاعلية تهدف إلى تمكين طلاب الهندسة والمهنيين من خلال خارطة طريق تعليمية واضحة ومصممة بعناية. تعمل هذه الخرائط كمرشد شخصي، حيث تقود المستخدم في رحلة تعليمية متكاملة، بدءًا من المفاهيم الأساسية وصولًا إلى المستويات المتقدمة في أي مهارة مستهدفة.

تتميز المنصة بتقديم هيكل تعليمي مقسم إلى مراحل رئيسية وخطوات فردية، مع تقدير للوقت اللازم لإنجاز كل جزء، مما يمنح المستخدم القدرة على التخطيط وإدارة وقته بفعالية. بالإضافة إلى ذلك، تتيح المنصة للمستخدمين تتبع تقدمهم بشكل مرئي، وتعزيز تعلمهم عبر اختبارات تقييمية، مما يحول عملية اكتساب المهارات من رحلة عشوائية إلى مسار منظم ومحفز وقابل للقياس.

الفصل الأول

التعريف بالمشروع

يستعرض هذا الفصل الإطار العام لمشروع "SkillCraft"، حيث يبدأ بتحديد المشكلة الأساسية التي يسعى المشروع لحلها، وهي صعوبة التعلم الذاتي المنظم في المجالات الهندسية. بعد ذلك، يقدم الفصل رؤية المشروع وأهدافه الرئيسية، ويحدد الجمهور المستهدف من الطلاب والمهنيين. وأخيراً، يلقي الفصل نظرة شاملة على الحل المقترح، موضحاً الميزات الأساسية للمنصة وكيف تساهم في تحقيق تجربة تعليمية فريدة ومتكاملة.

1.1- المشكلة وأهميتها

في ظل التطورات المتسارعة التي يشهدها قطاع الهندسة والتكنولوجيا، بات اكتساب المهارات الجديدة وتحديث المعارف القائمة أمراً لا غنى عنه للحفاظ على القدرة التنافسية في سوق العمل. ومع ذلك، يواجه العديد من الطلاب والمهندسين عقبة أساسية تتمثل في غياب مسار تعليمي واضح ومنظم. إن وفرة المصادر التعليمية على الإنترنت، على الرغم من كونها ميزة، قد تحولت إلى عبء يخلق حالة من التششت والضباب. فالتعلم الذاتي غالباً ما يقضي وقتاً طويلاً في البحث عن "ماذا يجب أن يتعلم" و"بأي ترتيب"، بدلاً من التركيز على عملية التعلم الفعلية، مما يؤدي إلى الإحباط وتأخير التطور المهني.

2.1- الحل المقترح: منصة SkillCraft

لمعالجة هذه التحديات، يأتي مشروع "SkillCraft" كحل تقني وتعليمي متكامل. المنصة هي عبارة عن تطبيق ويب يهدف إلى تنظيم وهيكلية عملية التعلم الذاتي من خلال تقديم roadmaps تفاعلية. كل roadmap مصممة بعناية من قبل خبراء في المجال لتغطي مهارة معينة بشكل شامل، بدءاً من الأساسيات وانتهاءً بالمواضيع المتقدمة. بذلك، تحول المنصة عملية التعلم من تجربة عشوائية وغير منظمة إلى رحلة واضحة المعالم، وموجهة نحو تحقيق أهداف محددة بكفاءة وفعالية.

3.1- رؤية المشروع وأهدافه

تتمثل رؤية "SkillCraft" في أن يصبح المرشد الرقمي الأول للمهندسين والتقنيين في رحلتهم لتطوير المهارات. ولتحقيق هذه الرؤية، يسعى المشروع إلى تحقيق الأهداف التالية:

- **توفير الوضوح والتوجيه:** إزالة الغموض الذي يحيط بعملية تعلم المهارات الجديدة من خلال تقديم مسارات تعليمية مجربة ومنظمة.
- **تعزيز التحفيز والمتابعة:** تمكين المستخدمين من تتبع تقدمهم بشكل مرئي، مما يخلق شعوراً بالإنجاز ويحفزهم على الاستمرار.
- **تقدير الجهد والوقت:** تزويد المستخدمين بتقديرات زمنية واقعية لإنجاز كل خطوة ومرحلة، لمساعدتهم على إدارة وقتهم وتوقعاتهم.
- **تقييم المعرفة المكتسبة:** إتاحة الفرصة للمستخدمين لاختبار مدى فهمهم للمواد التعليمية من خلال اختبارات قصيرة، مما يعزز من ترسيخ المعلومات.

4.1- الجمهور المستهدف

- تم تصميم منصة "SkillCraft" لتخدم شريحة واسعة من المستخدمين في المجال الهندسي والتقني، وتشمل بشكل أساسي:
- **طلاب الهندسة:** الذين يسعون لبناء أساس متين واكتساب مهارات عملية تتجاوز المناهج الأكاديمية التقليدية لتلبية متطلبات سوق العمل.
 - **المهندسون حديثو التخرج:** الذين يتطلعون إلى سد الفجوة بين المعرفة النظرية والتطبيق العملي، وتحديد المهارات الأكثر طلباً في تخصصاتهم.
 - **المهنيون وأصحاب الخبرة:** الذين يرغبون في مواكبة التطورات التكنولوجية، أو الانتقال إلى مسار مهني جديد يتطلب اكتساب مهارات مختلفة.

الفصل الثاني

الدراسة المرجعية

يهدف هذا الفصل إلى دراسة وتحليل المنصات التعليمية الحالية التي تقدم خدمات مشابهة لمشروع "SkillCraft". من خلال هذه الدراسة، نسعى إلى فهم نقاط القوة والضعف في الحلول المتاحة، وتحديد الفجوات في السوق التي يمكن لمشروعنا أن يسدّها. ستساعدنا هذه الدراسة المرجعية في إبراز الميزات التنافسية لمنصة "SkillCraft" وتأكيد القيمة المضافة التي ستقدمها للجماهير المستهدف.

1.2- تحليل المنصات التعليمية الحالية

لتحقيق فهم عميق للسوق، قمنا بتحليل عدد من المنصات الرائدة التي تعمل في مجال التعليم التقني وتطوير المهارات. ويركز التحليل التالي على وظائفها الأساسية، ومميزاتها، وجوانب القصور فيها.

1.1.2- موقع Roadmap.sh

يعتبر roadmap.sh من أبرز المنصات التي تقدم roadmaps بصرية للمطورين، وهو المصدر الأساسي الذي استلهم منه مشروعنا فكرته.

نقاط القوة:

- **Roadmaps شاملة:** يوفر مخططات تفصيلية وواضحة لمختلف التخصصات التقنية (Frontend, Backend, DevOps, etc.).
- **محتوى مجتمعي:** يعتمد على مساهمات المجتمع، مما يجعله مواكباً لأحدث التقنيات.
- **محتوى إضافي:** يحتوي على أدلة وأسئلة للمقابلات ومقالات تدعم المسارات التعليمية.
- **تتبع التقدم:** يسمح للمستخدمين بتسجيل الدخول وتتبع تقدمهم عبر تحديد المواضيع المنجزة.

نقاط الضعف والفرص المتاحة:

- غياب التقدير الزمني: لا يقدم تقديراً للوقت اللازم لإنجاز كل جزء من roadmap، وهو ما قد يصعب التخطيط على المتعلم.

2.1.2- منصات الكورسات (FreeCodeCamp, Codecademy)

تعتبر هذه المنصات من عمالقة التعليم البرمجي، حيث تقدم محتوى تعليمياً ضخماً ومنظماً على شكل دورات متكاملة.

نقاط القوة:

- محتوى منظم: تقدم مسارات تعليمية مرتبة على شكل دورات، مع شروحات تفصيلية ومشاريع عملية.
- بيئة تفاعلية: توفر بيئة برمجة تفاعلية داخل المتصفح، مما يسهل التطبيق المباشر.
- متابعة التقدم: تتابع تقدم المستخدم بدقة، وتوفر شهادات عند إكمال المسارات.
- ميزات متقدمة: بعضها يقدم ميزات متقدمة مثل محاكاة المقابلات باستخدام الذكاء الاصطناعي وربط الحساب بـ LinkedIn لعرض فرص العمل.

نقاط الضعف والفرص المتاحة:

- التركيز على الدورات لا الخرائط: هدفها الأساسي هو بيع أو تقديم دوراتها الخاصة، وليست مصممة لتقديم نظرة شاملة وعامة للمهارات المطلوبة في مجال معين.
- نموذج الاشتراك: العديد من الميزات المتقدمة والمحتوى المتعمق يكون مدفوعاً، مما قد يشكل عائقاً للبعض.

2.2- تحليل الفجوات والميزات التنافسية

بناءً على تحليل المنصات الحالية، يتضح وجود فجوة في السوق يمكن لمشروع "SkillCraft" أن يملأها بفعالية. تلخص المقارنة التالية كيف يتميز مشروعنا عن المنافسين:

جدول 1 مقارنة بين موقعين ومشروعنا

الميزة	Roadmap.sh	Codecademy	SkillCraft(مشروعنا)
roadmaps شاملة	✓	✗	✓
تقدير الوقت اللازم للتعلم	✗	✓	✓
تتبع التقدم الشخصي	✓	✓	✓
اختبارات	✓	✓	✓
مجاني ومفتوح	✓	جزئياً	✓

3.2- خلاصة الدراسة

تُظهر الدراسة المرجعية أن المنصات الحالية إما أن تركز على تقديم roadmaps بصرية دون تفاعل كافٍ (مثل roadmap.sh)، أو تركز على تقديم دورات تفاعلية لكنها تفتقر إلى النظرة الشاملة والمرونة (مثل FreeCodeCamp).

يأتي مشروع "SkillCraft" ليجمع أفضل ما في العالمين، حيث يقدم خرائط طريق واضحة ومجانية مستوحاة من roadmap.sh، ويدمج معها الميزات التفاعلية الأساسية مثل تتبع التقدم، تقدير الوقت، والاختبارات القصيرة. هذا المزيج يمنح "SkillCraft" ميزة تنافسية واضحة ويجعله حلاً شاملاً ومتكاملاً للمتعلم الذاتي في المجال الهندسي.

الفصل الثالث

الدراسة التحليلية وتحليل المتطلبات

يعتبر هذا الفصل حجر الزاوية في عملية تطوير المشروع، حيث يتم فيه تحليل النظام المقترح وتحديد جميع المتطلبات التي يجب أن يلبيها. سنبدأ بتحليل عام لهيكلية النظام، ثم نحدد بالتفصيل المتطلبات الوظيفية (Functional Requirements) التي تصف ما يجب على النظام أن يفعله، والمتطلبات غير الوظيفية (Non-Functional Requirements) التي تحدد كيف يجب على النظام أن يعمل، وأخيراً سنوضح هذه الوظائف من خلال مخطط حالات الاستخدام.

1.3- تحليل النظام

يعتمد نظام "SkillCraft" على بنية RBAC، حيث يتم منح صلاحيات مختلفة للمستخدمين بناءً على أدوارهم. هذا التصميم يضمن أمان البيانات ويسهل إدارة المحتوى والمستخدمين. ينقسم المستخدمون في النظام إلى ثلاثة أدوار رئيسية:

1. **المستخدم (user/ learner):** هو المستهلك الأساسي للمحتوى. يمكنه تصفح roadmaps، ومتابعة تقدمه، واستخدام الميزات التعليمية المتاحة.
2. **محرر المحتوى (editor):** هو المسؤول عن إنشاء وإدارة المحتوى التعليمي. يمتلك صلاحيات لإضافة وتعديل وحذف (roadmaps, milestones, steps and quizzes).
3. **مدير النظام (admin):** هو المستخدم الأعلى صلاحية في النظام. يمتلك جميع صلاحيات "محرر المحتوى"، بالإضافة إلى القدرة على إدارة حسابات جميع المستخدمين وأدوارهم.

2.3- المتطلبات

1.2.3- المتطلبات الوظيفية

هي الوظائف المحددة التي يجب على النظام تنفيذها. بناءً على القائمة المقدمة، تم ترتيبها حسب الدور كالتالي:

متطلبات المستخدم:

- **FR-U1**: يجب أن يسمح النظام للمستخدم بإنشاء حساب جديد وتأكيده.
- **FR-U2**: يجب أن يسمح النظام للمستخدم بتسجيل الدخول إلى حسابه.
- **FR-U3**: يجب أن يتمكن المستخدم من استعراض قائمة بجميع roadmaps المتاحة.
- **FR-U4**: يجب أن يتمكن المستخدم من البحث عن roadmap معينة.
- **FR-U5**: يجب أن يتمكن المستخدم من فتح ومشاهدة تفاصيل أي roadmap.
- **FR-U6**: يجب أن يتمكن المستخدم من حفظ roadmap في ملفه الشخصي وتتبع حالتها.
- **FR-U7**: يجب أن يتمكن المستخدم من تعديل حالة Steps ضمن roadmap (من غير منجز إلى منجز والعكس).
- **FR-U8**: يجب أن يتمكن المستخدم من استعراض قائمة بالأسئلة المتاحة.
- **FR-U9**: يجب أن يتمكن المستخدم من اختيار سؤال والإجابة عليه ورؤية النتيجة.
- **FR-U10**: يجب أن يتمكن المستخدم من طلب توليد roadmap عبر الذكاء الاصطناعي عن طريق تحديد موضوع.

- **FR-U11**: يجب أن يتمكن المستخدم من طلب توليد أسئلة اختبار عبر الذكاء الاصطناعي حول موضوع معين.

متطلبات محرر المحتوى:

- **FR-E1**: يجب أن يتمكن محرر المحتوى من تسجيل الدخول بحسابه.
- **FR-E2**: يجب أن يمتلك المحرر واجهة خاصة لإدارة المحتوى.
- **FR-E3**: يجب أن يتمكن المحرر من إدارة roadmaps (إضافة، تعديل، حذف).
- **FR-E4**: يجب أن يتمكن المحرر من إدارة steps and milestones وربطها بroadmaps.
- **FR-E5**: يجب أن يتمكن المحرر من إدارة بنك الأسئلة والاختبارات.

متطلبات مدير النظام:

- **FR-A1**: يجب أن يتمكن مدير النظام من تسجيل الدخول بحسابه.
- **FR-A2**: يجب أن يمتلك المدير واجهة إدارة شاملة.

- **FR-A3**: يجب أن يمتلك المدير جميع صلاحيات محرر المحتوى.
- **FR-A4**: يجب أن يتمكن المدير من استعراض قائمة بجميع المستخدمين.
- **FR-A5**: يجب أن يتمكن المدير من حذف أي حساب مستخدم.
- **FR-A6**: يجب أن يتمكن المدير من إنشاء وتعديل وحذف حسابات "محرر المحتوى".

2.2.3- المتطلبات غير الوظيفية

هي المعايير والقيود التي تحكم جودة عمل النظام.

الأداء (Performance):

- **NFR-1**: يجب ألا يتجاوز زمن تحميل الصفحات الرئيسية 3 ثوانٍ.
- **NFR-2**: يجب أن تكون استجابة النظام لعمليات المستخدم (مثل تحديد step كمنجزة) شبه فورية.

الأمان (Security):

- **NFR-3**: يجب تشفير كلمات مرور المستخدمين في قاعدة البيانات.
- **NFR-4**: يجب أن يتم تأكيد البريد الإلكتروني المستخدم في عملية Sign up.
- **NFR-5**: يجب أن يكون النظام محمياً ضد هجمات الويب الشائعة (مثل XSS و SQL Injection).
- **NFR-6**: يجب أن يتم التحقق من صلاحيات المستخدم عند كل طلب لضمان عدم وصوله إلى بيانات لا يملك الحق في رؤيتها.

قابلية الاستخدام (Usability):

- **NFR-7**: يجب أن تكون واجهة المستخدم سهلة وبديهية.
- **NFR-8**: يجب أن يكون تصميم الموقع متجاوباً (Responsive) ليعمل بشكل جيد على مختلف أحجام الشاشات (الحاسوب، الجهاز اللوحي، الهاتف المحمول).

الموثوقية (Reliability):

- **NFR-9**: يجب أن يكون النظام متاحاً للعمل بنسبة 99% من الوقت.

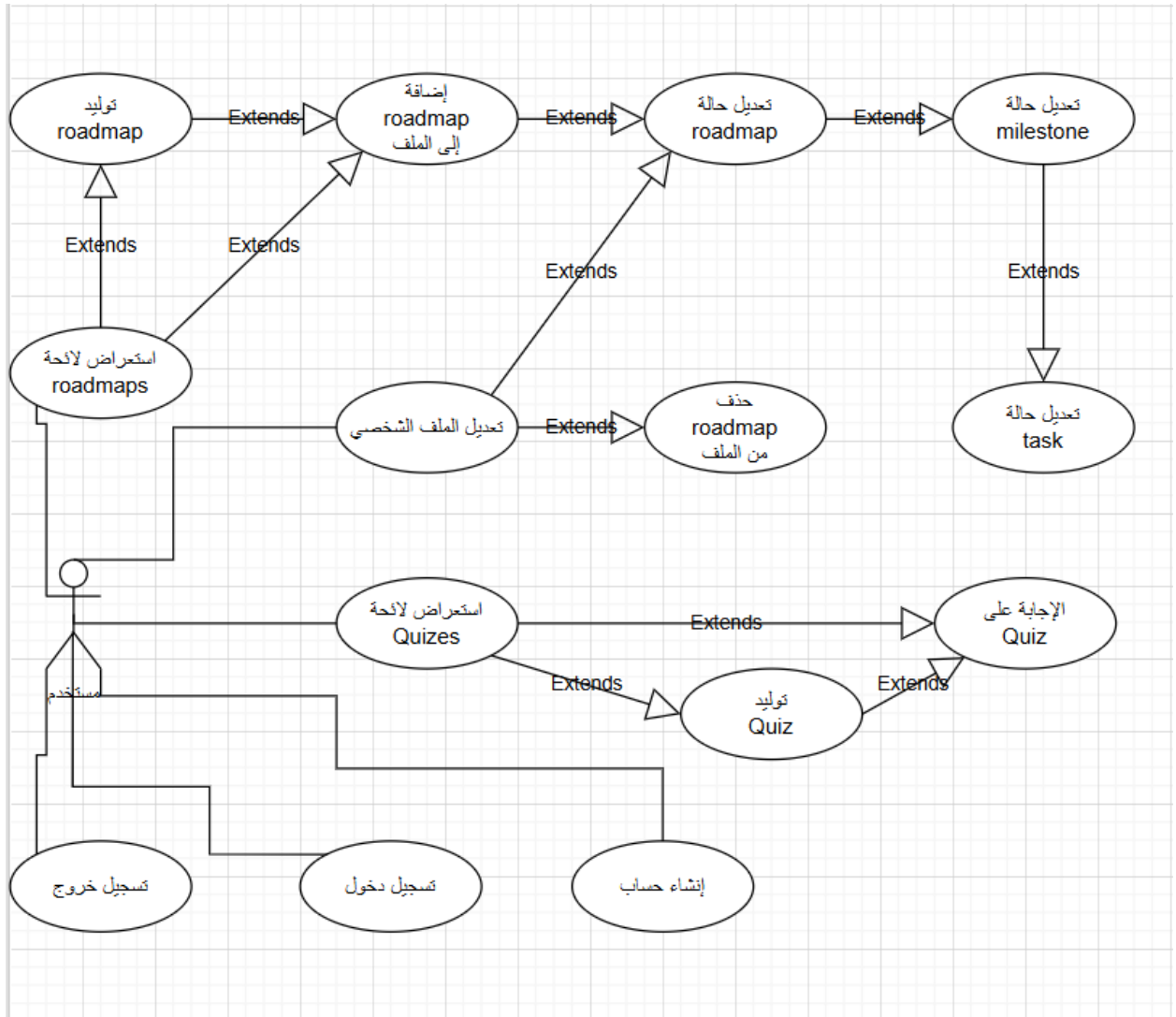
قابلية التوسع (Scalability):

- **NFR-10**: يجب أن تكون بنية النظام قادرة على التعامل مع زيادة عدد المستخدمين والمحتوى في المستقبل دون تدهور كبير في الأداء.

3.3- مخطط حالات الاستخدام

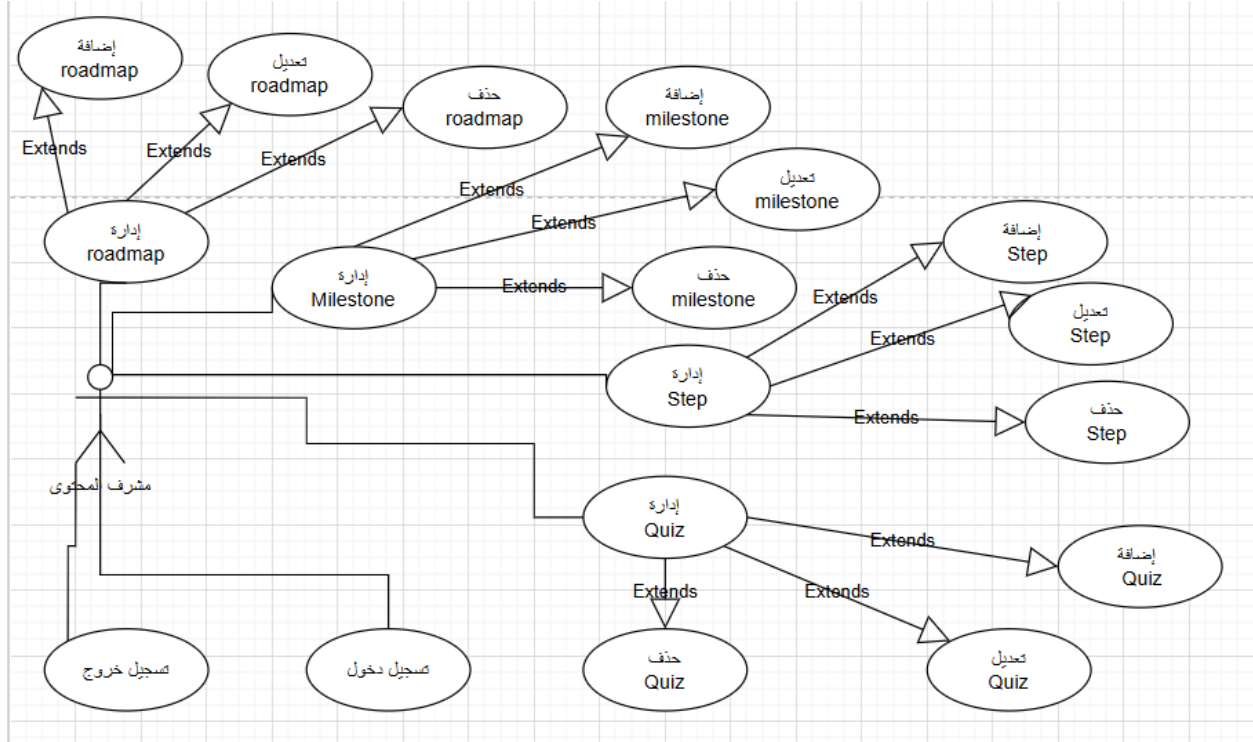
لتوضيح وظائف النظام بشكل مرئي، يمكن تمثيل التفاعلات الرئيسية بين المستخدمين (Actors) والنظام على النحو التالي:

- مخطط حالات الاستخدام للمستخدم المتلقي كما هو موضح في الرسم التوضيحي 1



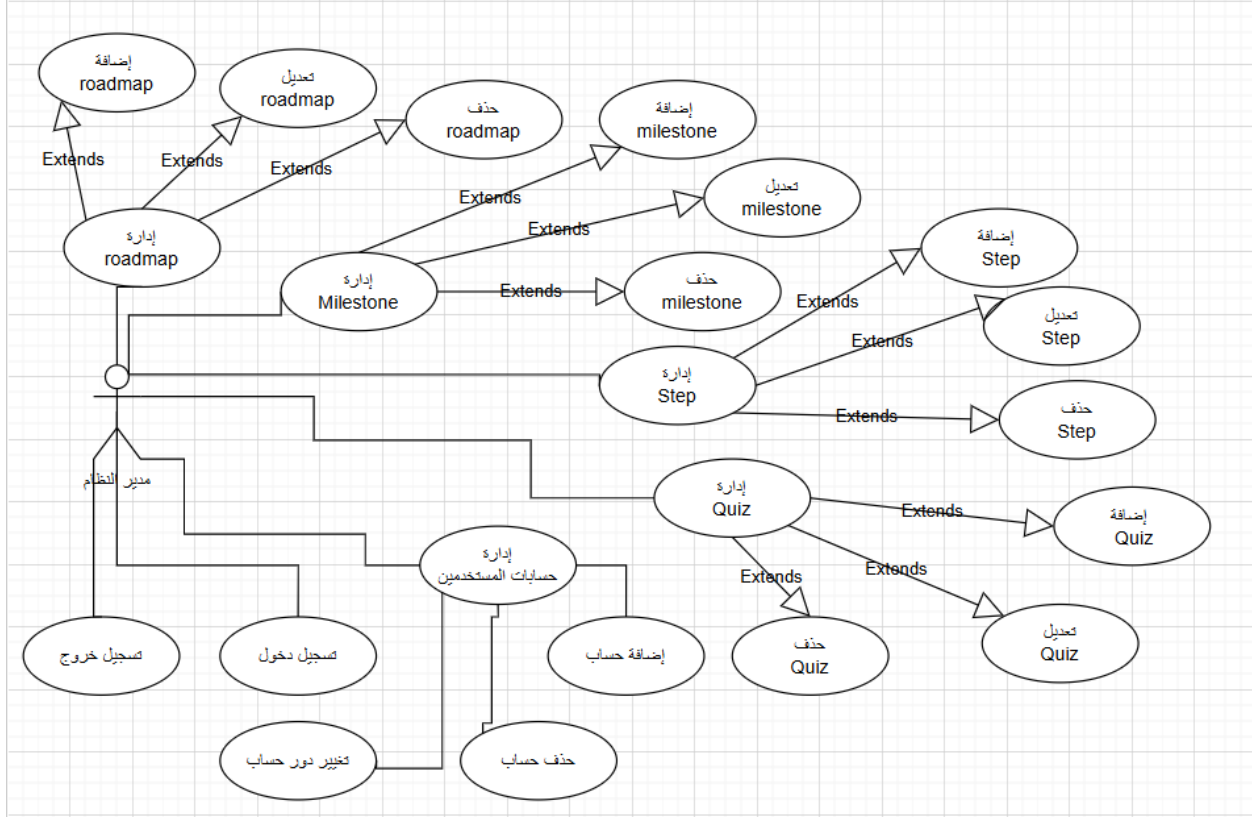
رسم توضيحي 1 حالات الاستخدام للمستخدم المتلقي

- مخطط حالات الاستخدام لمحرر المحتوى كما هو موضح في الرسم التوضيحي 2



رسم توضيحي 2 حالات الاستخدام لمحرر المحتوى

- مخطط حالات الاستخدام لمدير النظام كما هو موضح في الرسم التوضيحي 3



رسم توضيحي 3 مخطط حالات الاستخدام لمدير النظام

4.3- الوصف النصي لحالات الاستخدام ومخططات التسلسل

نستعرض الوصف النصي لجزء من حالات الاستخدام المبينة في الرسوم التوضيحية أعلاه.

1.3.3- حالة الاستخدام: تسجيل دخول

- النمط: أساسية.
- الفاعلون: مستخدم (أي دور).

- الظروف السابقة:

- يجب أن يكون المستخدم مُسجّل في النظام.

- السيناريو الرئيسي الناجح:

جدول 2 السيناريو الرئيسي لحالة تسجيل دخول

المستخدم	النظام
1- تبدأ هذه الحالة عندما يطلب المستخدم تسجيل الدخول إلى النظام	
	2- يعيد النظام استمارة لإدخال البريد الإلكتروني وكلمة السر
3- يقوم المستخدم بإدخال البريد الإلكتروني وكلمة السر	
	4- يتحقق النظام من أن البريد الإلكتروني موجود فعلاً.
	5- بعد التأكد من وجود البريد الإلكتروني يتم التأكد من صحة كلمة السر المدخلة.
	6- يقوم النظام بإعادة Token للمستخدم فيها بعض المعلومات الشخصية.
	7- يسمح النظام للمستخدم بالدخول.

- الظروف اللاحقة:

- يتعرف النظام على المستخدم ويتعامل معه عن طريق Token.

- يسمح للمستخدم بإجراء العمليات المناسبة لدوره.

- المسارات البديلة:

لا يوجد.

• مسارات الأخطاء:

E1 إدخال بريد الكتروني غير موجود.

يبدأ هذا المسار من المرحلة 4 من المسار الرئيسي.

5- يظهر النظام رسالة خطأ بفشل عملية تسجيل الدخول.

6- يطلب النظام من المستخدم إدخال بريد الكتروني موجود ويعود للمرحلة 3.

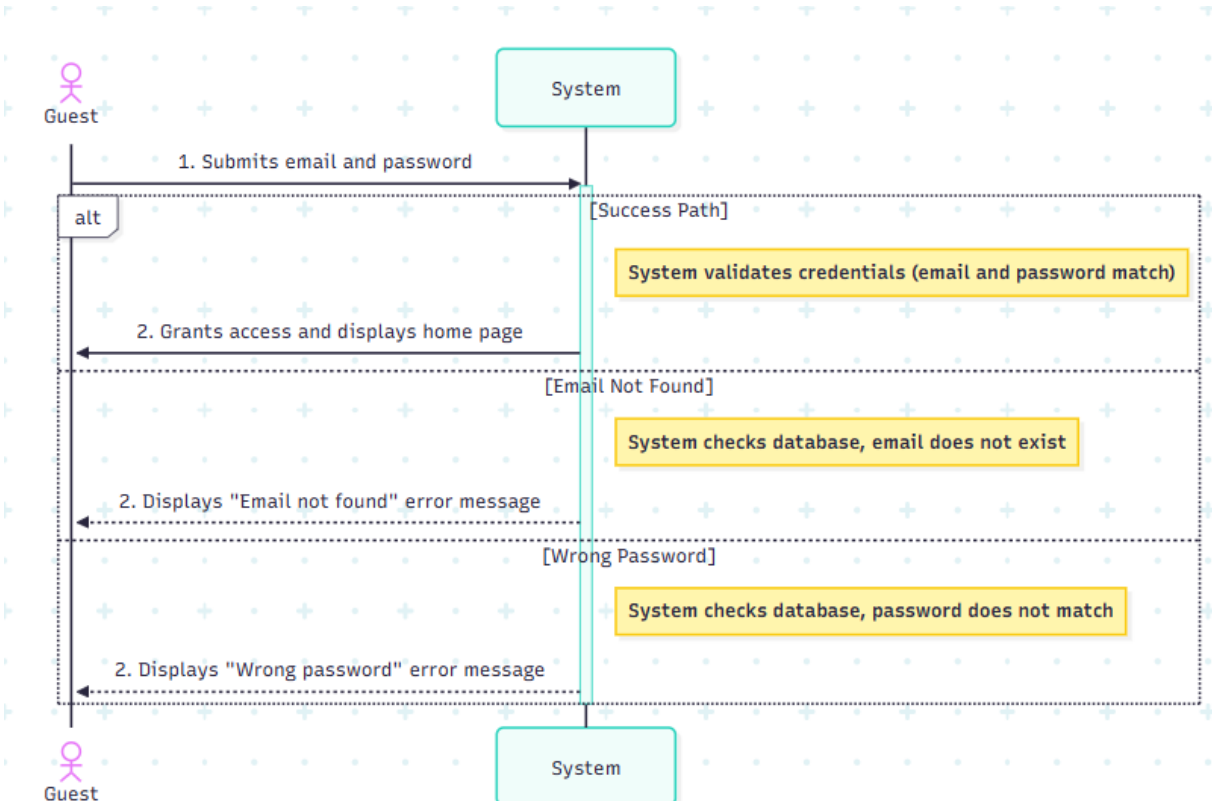
E2 إدخال كلمة سر غير صحيحة.

يبدأ هذا المسار من المرحلة 5 من المسار الرئيسي.

6- يظهر النظام رسالة خطأ بفشل عملية تسجيل الدخول.

7- يطلب النظام من المستخدم إدخال كلمة السر الصحيحة ويعود للمرحلة 3.

• مخطط التالي لحالة الاستخدام:



رسم توضيحي 4 مخطط التالي لعملية تسجيل الدخول

2.3.3- حالة الاستخدام: إنشاء حساب جديد

- النمط: أساسية.
- الفاعلون: مستخدم متلقي.
- الظروف السابقة:
لا يوجد.
- السيناريو الرئيسي الناجح:

جدول 3 المسار الرئيسي لحالة إنشاء حساب جديد

المستخدم	النظام
1- يطلب المستخدم إنشاء حساب جديد.	
	2- يعيد النظام استمارة لإدخال المعلومات الشخصية بما فيها البريد الالكتروني وكلمة السر.
3- يقوم المستخدم بإدخال المعلومات الشخصية.	
	4- يتحقق النظام أن البريد الالكتروني غير مستخدم.
	5- يتأكد من أن البريد الالكتروني غير مخزن في جدول حسابات بانتظار التأكيد.
	6- يرسل النظام رسالة تأكيد إلى البريد الالكتروني المدخل.
	7- يقوم النظام بتشفير كلمة السر.
	8- يقوم النظام بإضافة المستخدم إلى قاعدة البيانات في جدول خاص بمستخدمين لم يتم تأكيد بريدهم الالكتروني.

9- يقوم المستخدم بتأكيد بريده الالكتروني من خلال البريد الذي وصله.	
	10- يقوم النظام بنقل الحساب من جدول حسابات لم يتم تأكيدهم إلى جدول الحسابات.
	11- يقوم النظام بإعادة Token للمستخدم فيها بعض المعلومات الشخصية.
	12- يسمح النظام للمستخدم بالدخول.

• الظروف اللاحقة:

- يصبح المستخدم مُسجَّل في النظام.
- يتعرف النظام على المستخدم ويتعامل معه عن طريق Token.
- يسمح للمستخدم بإجراء العمليات المناسبة لدوره.

• المسارات البديلة:

A1 إدخال بريد الكتروني في جدول حسابات لم يتم تأكيدها.

يبدأ هذا المسار من المرحلة 5 من المسار الرئيسي، يحذف الحساب الموجود في هذا الجدول ويستمر المسار من المرحلة 6 من المسار الرئيسي.

• مسارات الأخطاء:

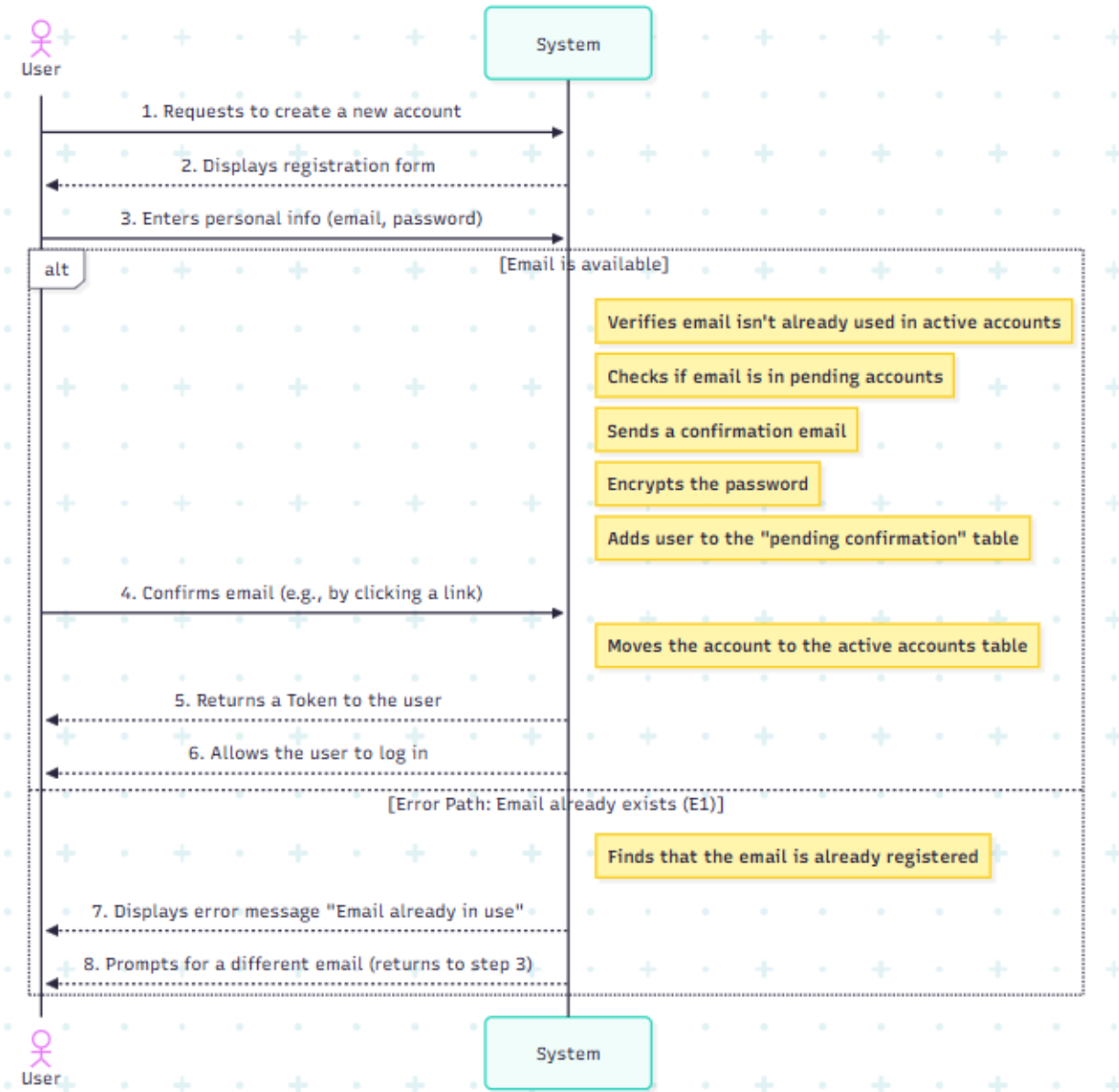
E1 إدخال بريد الكتروني موجود.

يبدأ هذا المسار من المرحلة 4 من المسار الرئيسي.

5- يظهر النظام رسالة خطأ بفشل عملية إنشاء الحساب.

6- يطلب النظام من المستخدم إدخال بريد الكتروني جديد ويعود للمرحلة 3.

• مخطط التالي لحالة الاستخدام:



رسم توضيحي 5 مخطط التالي لعملية إنشاء حساب جديد

3.3.3- حالة الاستخدام: إضافة Roadmap إلى الملف الشخصي

- النمط: أساسية.
- الفاعلون: مستخدم متلقي.

- الظروف السابقة:

- يجب أن يكون المستخدم قد قام بتسجيل الدخول إلى النظام بنجاح.

- السيناريو الرئيسي الناجح:

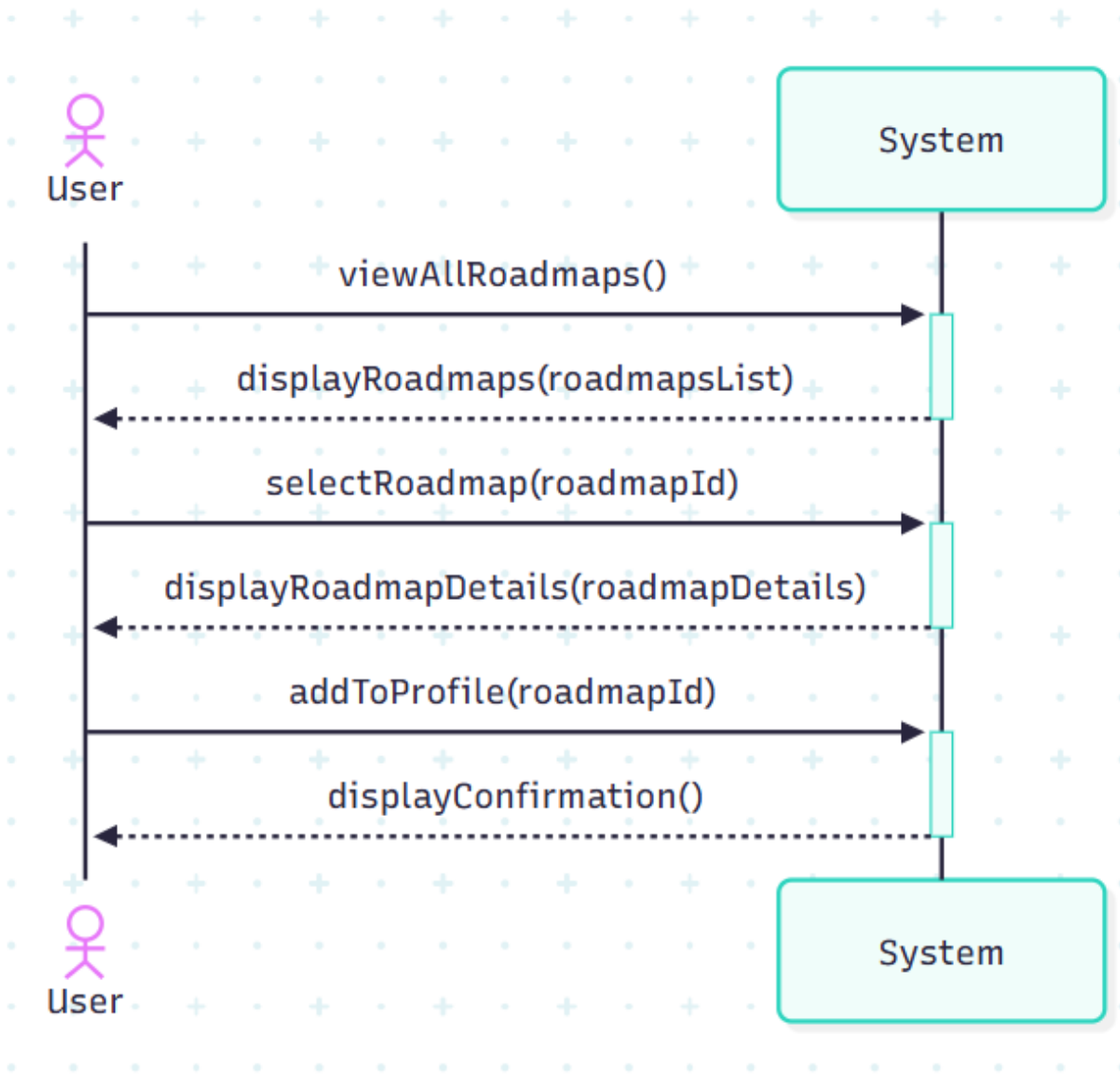
جدول 4 المسار الرئيسي لحالة إضافة roadmap إلى الملف الشخصي

المستخدم	النظام
1- يطلب المستخدم استعراض جميع roadmaps.	
	2- يعرض النظام صفحة تحتوي على قائمة بجميع roadmaps المتاحة.
3- يختار المستخدم roadmap معينة وينقر عليها.	
	4- يعرض النظام صفحة التفاصيل الكاملة لل roadmap المختارة، مع زر "Add to My Profile".
5- ينقر المستخدم على زر "Add to My Profile".	
	6- يقوم النظام بإنشاء سجل جديد في قاعدة بيانات الملفات الشخصية (MongoDB) يربط هوية المستخدم بهوية roadmap.
	7- يحدّث النظام الواجهة ليعرض تأكيداً للمستخدم .

- الظروف اللاحقة:

- تتم إضافة roadmap بنجاح إلى قائمة المتابعة في الملف الشخصي للمستخدم.

- المسارات البديلة:
لا يوجد.
- مسارات الأخطاء:
لا يوجد.
- مخطط التالي لحالة الاستخدام:



رسم توضيحي 6 مخطط التالي لعملية إضافة roadmap إلى الملف الشخصي

4.3.3- حالة الاستخدام: توليد roadmap عبر الذكاء الاصطناعي

- النمط: أساسية.
- الفاعلون: مستخدم متلقي.
- الظروف السابقة:
- يجب أن يكون المستخدم قد قام بتسجيل الدخول إلى النظام بنجاح.
- السيناريو الرئيسي الناجح:

المستخدم	النظام
1- يقوم المستخدم بكتابة اسم المهارة أو الموضوع الذي يرغب في تعلمه (مثال: "Frontend") في شريط الإدخال المخصص للذكاء الاصطناعي.	
	2- يستقبل النظام الطلب ويتحقق من أن المستخدم مصادق عليه.
	3- يقوم النظام بجلب قائمة بالخطوات التي أكملها المستخدم مسبقاً من ملفه الشخصي (Profile).
	4- يختار النظام استراتيجية الإنشاء "AI" (AiRoadmapCreationStrategy).
	5- يقوم النظام بصياغة طلب مفصل (Prompt) بناءً على الموضوع الذي قدمه المستخدم وقائمة الخطوات المكتملة، ويطلب من خدمة الذكاء الاصطناعي هيكلية الاستجابة بتنسيق JSON محدد (يشمل الاسم، الوصف، المراحل، والخطوات).
	6- يرسل النظام الطلب إلى خدمة الذكاء الاصطناعي الخارجية (Gemini).

	7- تقوم خدمة الذكاء الاصطناعي بمعالجة الطلب وتوليد بنية خارطة الطريق المطلوبة..
	8- يستقبل النظام استجابة JSON من خدمة الذكاء الاصطناعي.
	9- يقوم النظام بمعالجة وتحليل (Parse) نص JSON لتحويله إلى كائنات (Roadmap, Milestone,) (Step).
	10- يقوم النظام بحفظ الكائنات الجديدة في قاعدة بيانات MongoDB.
	11- يعيد النظام كائن خارطة الطريق الجديد بالكامل (مع المعرف الفريد Id الخاص به) كاستجابة ناجحة للمستخدم، مما ينهي حالة الاستخدام.

- الظروف اللاحقة:

- يتم إنشاء roadmap جديدة وجميع مكوناتها (milestones and steps) وحفظها في قاعدة البيانات.

- المسارات البديلة:

لا يوجد.

- مسارات الأخطاء:

E1 فشل خدمة الذكاء الاصطناعي.

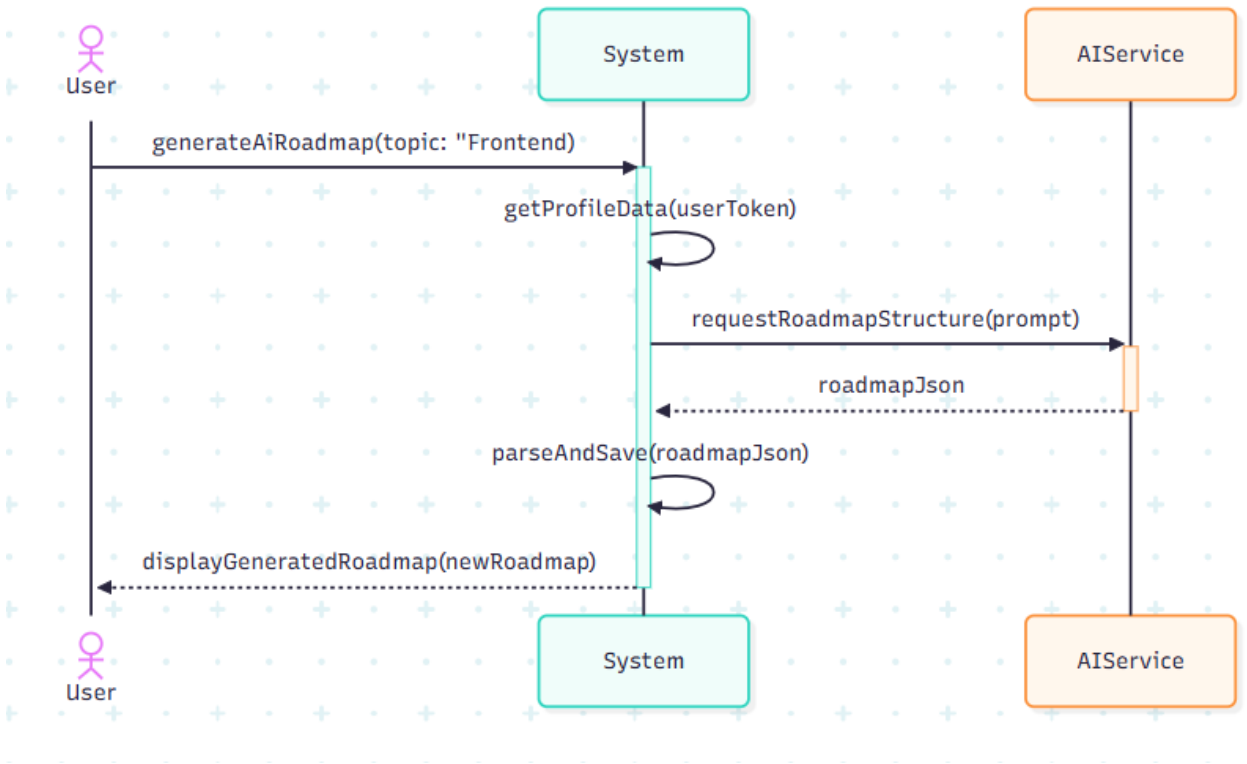
يبدأ هذا المسار من المرحلة 7 من المسار الرئيسي.

8- بعد إرسال الطلب إلى خدمة الذكاء الاصطناعي، تفشل الخدمة في الاستجابة (بسبب انقطاع الخدمة، مفتاح

API غير صالح، أو انتهاء المهلة).

9- يلتقط النظام الخطأ ويعيد خطأ أن الخدمة مشغولة بالوقت الحالي.

- مخطط التالي لحالة الاستخدام توليد roadmap عبر الذكاء الاصطناعي:



رسم توضيحي 7 مخطط التالي لحالة الاستخدام توليد roadmap بالذكاء الاصطناعي

الفصل الرابع

البنية التصميمية وبيئة العمل

يعتبر هذا الفصل الركيزة الأساسية التي توضح القرارات الهندسية والتقنية التي تم اتخاذها لبناء منصة "SkillCraft". يهدف الفصل إلى تقديم شرح مفصل للبنية التصميمية للنظام، والتقنيات وأطر العمل والمكتبات التي تم اختيارها، بالإضافة إلى بيئات التطوير والأدوات التي ساهمت في إنجاز المشروع. سيتم تبرير كل قرار تقني لإظهار كيف يخدم أهداف المشروع ومتطلباته.

1.4 – البنية التصميمية (Software Architecture)

البنية التصميمية أو المعمارية للبرمجيات هي المخطط الأساسي الذي يصف هيكل النظام، وكيفية تفاعل مكوناته المختلفة مع بعضها البعض. اختيار البنية المناسبة يضمن أن يكون النظام قابلاً للصيانة، وموثوقاً، وقابلاً للتطوير في المستقبل. في مشروع "SkillCraft"، تم اعتماد بنية مركبة تدمج بين معماريتين أساسيتين.

1.1.4 – معمارية العميل-الخادم (Client-Server Architecture)

على المستوى العام، يتبع النظام معمارية العميل-الخادم. يتم فصل التطبيق إلى جزأين رئيسيين:

- **العميل (client):** وهو الواجهة الأمامية المبنية بتقنية React، والتي تعمل بالكامل على متصفح المستخدم. هذا الجزء مسؤول عن كل ما يراه المستخدم ويتفاعل معه.
- **الخادم (Server):** وهو الواجهة الخلفية المبنية بتقنية ASP.NET Core، والتي تعمل على خادم مركزي. هذا الجزء مسؤول عن تخزين البيانات، تنفيذ منطق العمل، وتأمين النظام.

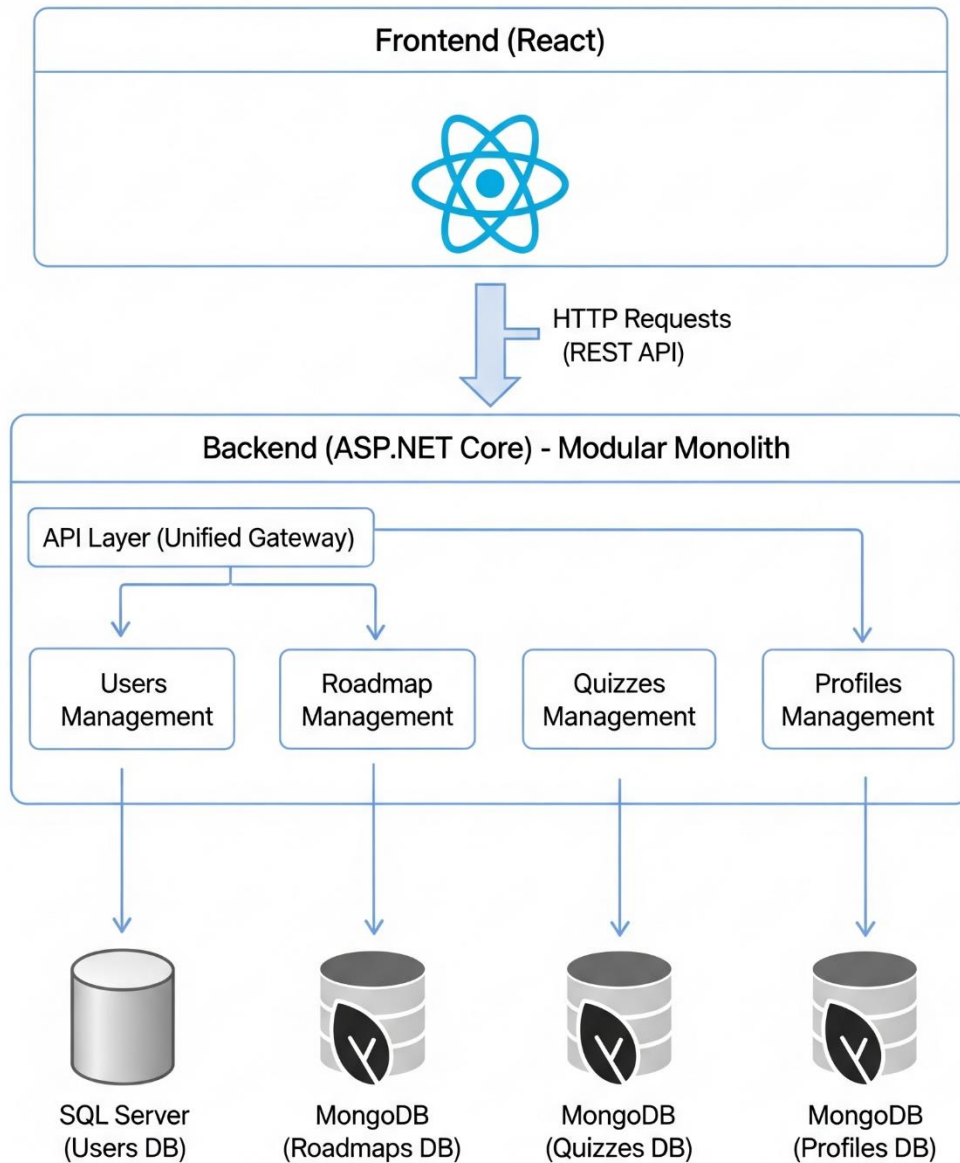
يتواصل العميل مع الخادم عبر شبكة الإنترنت باستخدام بروتوكول HTTP، حيث يرسل العميل طلبات (Requests) ويستقبل الخادم هذه الطلبات ويعيد استجابات (Responses) تحتوي على البيانات المطلوبة.

2.1.4- المعمارية المتجانسة المقسّمة (Modular Monolith)

بالنسبة لتصميم الخادم (الواجهة الخلفية)، تم اتباع معمارية Modular Monolith. هذا النهج هو حل وسيط يجمع بين مزايا البساطة في التطبيقات المتجانسة (Monolithic) ومزايا التنظيم في الخدمات المصغرة (Microservices).

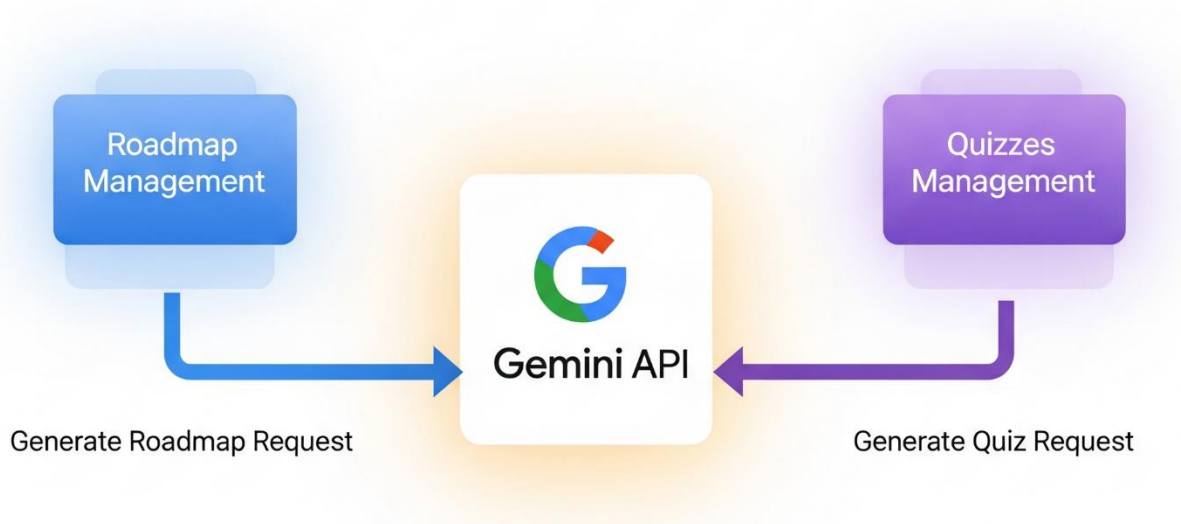
في هذه البنية، يتم بناء التطبيق ككتلة واحدة قابلة للنشر، ولكنه مقسم داخلياً إلى وحدات (Modules) مستقلة ومنفصلة منطقياً. كل وحدة لها مسؤولية واضحة ومحددة. في مشروعنا، تم تحديد الوحدات التالية:

- وحدة إدارة المستخدمين (Users Management): مسؤولة عن التسجيل، تسجيل الدخول، إدارة الأدوار والصلاحيات.
- وحدة إدارة خرائط الطريق (Roadmap Management): مسؤولة عن كل ما يتعلق بإنشاء وتعديل وعرض roadmaps ومكوناتها (milestones and steps).
- وحدة إدارة الاختبارات (Quizzes Management): مسؤولة عن إنشاء وإدارة الاختبارات والأسئلة.
- وحدة الملفات الشخصية (Profiles Management): مسؤولة عن إدارة الملفات الشخصية للمستخدمين. يتم كشف وظائفها للعالم الخارجي عبر واجهة برمجة تطبيقات (API) موحدة، تعمل كبوابة وحيدة للنظام بأكمله.



رسم توضيحي 8 هيكلية المشروع

إن وحدة إدارة roadmaps ووحدة إدارة quizzes تتواصل مع gemini API كما هو موضح بالرسم التوضيحي 9:



رسم توضيحي 9

2.4- بيئة العمل وأدوات التطوير

1.2.4- بيئات التطوير المتكاملة (IDEs) :

بيئة التطوير المتكاملة هي برنامج حاسوبي يوفر للمطورين مجموعة شاملة من الأدوات لتسهيل عملية كتابة الكود واختباره.

- **Microsoft Visual Studio 2022**: تم استخدامه لتطوير الواجهة الخلفية (.NET). يعتبر الخيار الأمثل لتطبيقات .NET. لما يوفره من أدوات قوية لتصحيح الأخطاء (Debugging) ، وإدارة المشاريع، والتكامل السلس مع منصات مايكروسوفت الأخرى.
- **Visual Studio Code (VS Code)**: تم استخدامه لتطوير الواجهة الأمامية (React). هو محرر أكواد خفيف وقوي ومفتوح المصدر، ويحظى بشعبية هائلة بفضل سرعته ودعمه لآلاف الإضافات التي تجعله مناسباً لتطوير أي نوع من التطبيقات تقريباً.

2.2.4- نظام إدارة الإصدارات (Version Control System)

- **Git**: هو نظام موزع لإدارة الإصدارات، يسمح بتتبع كل تغيير يتم على الكود بمرور الوقت. يتيح للمطورين الرجوع إلى إصدارات سابقة، وإنشاء Branches لتجربة ميزات جديدة دون التأثير على النسخة الرئيسية من الكود، ودمج التغييرات من عدة مطورين بسهولة.
- **GitHub**: هي منصة استضافة سحابية لمشاريع Git. تم استخدامها كمستودع مركزي (Central Repository) للأكواد للمشروع (الواجهة الأمامية والخلفية)، مما سهل عملية التعاون وتوزيع العمل ومراجعة الكود.

3.2.4- خدمات الذكاء الاصطناعي (Artificial Intelligence Services)

- **Gemini 2.5 Flash API**: تم الاعتماد على واجهة برمجة التطبيقات الخاصة بنموذج Gemini 2.5 Flash من Google كخدمة خارجية أساسية لتوليد المحتوى الديناميكي. هذا النموذج هو من LLM التي تتميز بكونها خفيفة وسريعة جداً وذات تكلفة منخفضة، مما يجعلها مثالية للمهام التي تتطلب استجابة سريعة وتفاعلاً مستمراً مع المستخدم، مثل توليد roadmaps و quizzes بشكل فوري. تم اختيار هذا النموذج تحديداً لقدرته على الموازنة بين السرعة العالية والجودة المقبولة في توليد المحتوى المنظم (JSON)، وهو ما يتناسب تماماً مع متطلبات الأداء في مشروعنا.

3.4- أطر العمل والمكتبات المستخدمة

1.3.4- الواجهة الخلفية (Backend)

- **ASP.NET Core 8**: هو إطار عمل (Framework) من مايكروسوفت لبناء تطبيقات الويب الحديثة. تم اختياره لأنه مفتوح المصدر، عالي الأداء، ويعمل على مختلف أنظمة التشغيل (Windows, macOS, Linux). يوفر بنية قوية لبناء واجهات برمجة التطبيقات (APIs) الآمنة والقابلة للتوسع.
- **Entity Framework Core**: هي مكتبة من نوع ORM تبسط عملية التعامل مع قواعد البيانات العلائقية. تسمح للمطورين بكتابة استعلامات قاعدة البيانات باستخدام لغة C# بدلاً من SQL، مما يزيد من الإنتاجية ويقلل من الأخطاء.
- **MongoDB.Driver**: هي المكتبة الرسمية للتواصل مع قاعدة بيانات MongoDB. تم استخدامها في جميع الوحدات التي تتعامل مع بيانات غير علائقية (مثل roadmaps, quizzes and profiles) لتنفيذ عمليات القراءة والكتابة والاستعلام.
- **AutoMapper**: هي مكتبة لتبسيط عملية تحويل الكائنات من نوع إلى آخر. تم استخدامها بكثافة في طبقة منطق العمل (BLL) لتحويل نماذج قاعدة البيانات (Entities) إلى كائنات نقل البيانات (DTOs) التي يتم إرسالها للواجهة الأمامية، مما ساهم في فصل الاهتمامات والحفاظ على نظافة الكود.
- **Microsoft.AspNetCore.Authentication.JwtBearer**: هي المكتبة المسؤولة عن تفعيل وتكوين نظام المصادقة باستخدام JSON Web Tokens (JWT). تم استخدامها لتأمين نقاط النهاية (Endpoints) في الـ API، حيث تقوم بالتحقق من صحة التوكن المرفق مع كل طلب وتحديد هوية المستخدم وصلاحياته.
- **Swashbuckle.AspNetCore (Swagger)**: تم استخدام هذه المكتبة لتوليد واجهة مستخدم تفاعلية (Swagger UI) لتوثيق واختبار نقاط النهاية الخاصة بالـ API مباشرة من المتصفح، مما سرّع من عملية التطوير والاختبار.

- **Moq & FluentAssertions (للاختبار):** تم استخدام مكتبة Moq لإنشاء كائنات وهمية (Mock Objects) في اختبارات الوحدات (Unit Tests)، مما سمح باختبار طبقة منطق العمل (BLL) بمعزل عن طبقة الوصول للبيانات. وتم استخدام FluentAssertions لكتابة تأكيدات (Assertions) في الاختبارات بطريقة سلسلة وأكثر قابلية للقراءة.

2.3.4 - الواجهة الأمامية (Frontend)

- **React:** هي مكتبة JavaScript لبناء واجهات المستخدم. تعتمد على مفهوم "المكونات (Components)"، حيث يتم بناء الواجهة كمجموعة من القطع المستقلة والقابلة لإعادة الاستخدام، مما يجعل إدارة الواجهات المعقدة أكثر سهولة.
- **React Router:** هي مكتبة للتوجيه (Routing) داخل تطبيقات React. تسمح بالتنقل بين الصفحات المختلفة دون الحاجة لإعادة تحميل الصفحة بأكملها، مما يوفر تجربة مستخدم سريعة وسلسلة تشبه تطبيقات سطح المكتب.
- **Axios:** هي مكتبة JavaScript لإجراء طلبات HTTP. تم استخدامها لتسهيل التواصل بين الواجهة الأمامية والـ API في الواجهة الخلفية.
- **Bootstrap & SASS:** تم استخدام Bootstrap كنظام شبكي وإطار عمل CSS أساسي لتسريع عملية تصميم واجهات متجاوبة. وتم استخدام SASS لكتابة شيفرة CSS أكثر تنظيماً وقابلية للصيانة.

4.4 - تقنيات قواعد البيانات

قواعد البيانات هي أنظمة لتخزين وإدارة البيانات بكفاءة. يوجد نوعان رئيسيان تم استخدامهما في هذا المشروع.

جدول 5 مقارنة بين SQL و NoSQL

الميزة	قواعد البيانات العلائقية (Relational - SQL)	قواعد البيانات غير العلائقية (NoSQL)
هيكلية البيانات	منظمة في جداول ذات أعمدة وصفوف محددة مسبقاً.	مرنة، تعتمد على نماذج مختلفة (مستندات، Key-Value).
المرونة	أقل مرونة، تتطلب تحديد المخطط (Schema) مسبقاً.	عالية المرونة، يمكن تغيير بنية البيانات بسهولة.
مثال	SQL Server, MySQL	MongoDB, Redis

في مشروع "SkillCraft"، تم اتباع نهج هجين (Hybrid Approach) للاستفادة من مزايا كلا النوعين:

- **Microsoft SQL Server**: تم اختياره لتخزين البيانات التي تتطلب هيكلية صارمة وعلاقات واضحة، مثل بيانات المستخدمين (Users) والأدوار (Roles). الطبيعة العلائقية لـ SQL Server تضمن تكامل هذه البيانات الحساسة واتساقها.
 - **MongoDB**: تم اختياره لتخزين المحتوى التعليمي (roadmaps, milestones, steps and quizzes). طبيعة هذه البيانات هرمية ومتغيرة باستمرار، ونموذج المستندات (Documents) في MongoDB يوفر المرونة اللازمة لتخزينها وتعديلها بكفاءة عالية دون قيود المخطط الصارم.
- هذا النهج الهجين سمح لنا باختيار الأداة الأنسب لكل نوع من البيانات، مما أدى إلى نظام أكثر كفاءة وقابلية للتطوير.

5.4- الأنماط التصميمية المتبعة (Design Patterns)

لضمان بناء كود نظيف، قابلة للصيانة، ومنظمة، تم اتباع مجموعة من الأنماط التصميمية القياسية في الواجهة الخلفية.

1.5.4- نمط حقن التبعية (Dependency Injection - DI)

هذا النمط هو جزء أساسي من إطار عمل ASP.NET Core. المبدأ الأساسي هو أن الكائن لا يجب أن يقوم بإنشاء تبعياته بنفسه، بل يجب أن يتم "حقنها" من الخارج. تم تطبيق هذا النمط بكثافة في ملف Program.cs عند تسجيل الخدمات.

2.5.4- نمط المستودع (Repository Pattern)

يهدف هذا النمط إلى إنشاء Abstraction Layer بين منطق العمل وطبقة الوصول إلى البيانات. لكل Entity مثل User أو Roadmap، تم إنشاء مستودع خاص به (مثل UserRepository و RoadmapRepository). هذا المستودع يحتوي على جميع العمليات المتعلقة بقاعدة البيانات لهذا النموذج.

الفائدة: يعزل هذا النمط منطق العمل عن تفاصيل كيفية تخزين البيانات (سواء كانت في SQL Server أو MongoDB)، مما يسمح بتغيير مصدر البيانات في المستقبل دون الحاجة لتعديل منطق العمل.

3.5.4- نمط وحدة العمل (Unit of Work Pattern)

يعمل هذا النمط جنباً إلى جنب مع نمط المستودع. يقوم بتجميع مجموعة من العمليات (التي قد تشمل عدة مستودعات) في معاملة (Transaction) واحدة. تم تنفيذ هذا النمط في UnitOfWork.cs.

مثال:

عندما يقوم محرر بحفظ roadmap جديدة مع مكوناتها، قد يتضمن ذلك عمليات كتابة متعددة على قاعدة البيانات. يقوم نمط Unit of Work بضمان أن جميع هذه العمليات إما أن تنجح معاً أو تفشل معاً، وذلك من خلال استدعاء دالة CompleteAsync() مرة واحدة في النهاية.

الفائدة: يضمن هذا النمط اتساق وتكامل البيانات (Data Integrity).

4.5.4- نمط كائن نقل البيانات (Data Transfer Object - DTO)

يستخدم هذا النمط لإنشاء كائنات بسيطة مهمتها الوحيدة هي نقل البيانات بين طبقات النظام المختلفة، وخاصة بين الواجهة الخلفية (API) والواجهة الأمامية (Client). في المشروع، تم إنشاء DTOs مثل UserDto و RoadmapDto. الفائدة:

1. الأمان: يمنع كشف بنية قاعدة البيانات الداخلية للعميل.
2. التخصيص: يسمح بتشكيل البيانات بالشكل الذي تحتاجه الواجهة الأمامية بالضبط، مما يقلل من حجم البيانات المرسل.
3. فصل الاهتمامات: يفصل نماذج العرض عن نماذج قاعدة البيانات.

5.5.4- نمط الاستراتيجية (Strategy Pattern)

الهدف: يسمح بتعريف عائلة من الخوارزميات، ووضع كل منها في صف منفصل، وجعلها قابلة للتبديل. التطبيق في المشروع: تم استخدامه للفصل بين طريقتي إنشاء المحتوى: اليدوية وعبر الذكاء الاصطناعي. تم إنشاء واجهة IRoadmapCreationStrategy تحتوي على دالة CreateRoadmapAsync. تم إنشاء صفتين يطبقان هذه الواجهة:

ManualRoadmapCreationStrategy: يحتوي على منطق إنشاء roadmap بالطريقة اليدوية.

AiRoadmapCreationStrategy: يحتوي على منطق استدعاء GeminiAiAdapter لتوليد roadmap.

الفائدة: هذا الفصل سمح بإضافة طريقة التوليد عبر الذكاء الاصطناعي دون تعديل الكود الموجودة مسبقاً، وجعل النظام مرناً لإضافة استراتيجيات أخرى في المستقبل (مثلاً، التوليد من ملف).

6.5.4- نمط المصنع (Factory Pattern)

الهدف: يوفر واجهة لإنشاء كائنات في صف رئيسي، لكنه يسمح للصفوف الفرعية بتغيير نوع الكائنات التي سيتم إنشاؤها. **التطبيق في المشروع:** تم استخدامه لإنشاء كائن الاستراتيجية المناسب دون أن يعرف العميل (مثل RoadmapsService) التفاصيل الداخلية لعملية الإنشاء.

- تم إنشاء StrategyFactory الذي يحتوي على دالة CreateStrategy(string key).
- تقوم هذه الدالة بإرجاع كائن من نوع AiRoadmapCreationStrategy إذا كانت قيمة المتغير key هي ai، أو كائن من نوع ManualRoadmapCreationStrategy إذا كانت manual.

الفائدة: يركز هذا النمط المسؤولية عن إنشاء الاستراتيجيات في مكان واحد، ويفصل منطق الاختيار عن منطق التنفيذ.

7.5.4- نمط المحول (Adapter Pattern)

الهدف: يسمح للكائنات ذات الواجهات غير المتوافقة بالعمل معًا. **التطبيق في المشروع:** تم استخدامه لعزل نظامنا عن التفاصيل المعقدة لواجهة برمجة التطبيقات الخارجية الخاصة بـ Gemini AI.

- تم إنشاء واجهة داخلية IAiGenerator تحدد الوظائف التي يحتاجها نظامنا (مثل GenerateJsonAsync).
- تم إنشاء صف GeminiAiAdapter الذي يطبق هذه الواجهة. داخلياً، يقوم هذا الصف بترجمة استدعاء دالة GenerateRoadmapAsync إلى طلب HTTP متوافق مع Gemini AI، ثم يقوم بتحويل استجابة Gemini إلى كائن AiGeneratedRoadmap الذي يفهمه نظامنا.

الفائدة: إذا قررنا في المستقبل تغيير مزود خدمة الذكاء الاصطناعي (من Gemini إلى OpenAI مثلاً)، كل ما نحتاجه هو كتابة Adapter جديد (OpenAiAdapter) دون الحاجة لتغيير أي شيفرة في باقي أجزاء النظام.

الفصل الخامس

التنفيذ

بعد اكتمال مرحلة التنفيذ، تأتي مرحلة الاختبار كخطوة حيوية لضمان جودة المنتج البرمجي والتأكد من أنه يلبي جميع المتطلبات الوظيفية وغير الوظيفية التي تم تحديدها في الفصل الثالث. يهدف هذا الفصل إلى استعراض استراتيجية الاختبار التي تم اتباعها، وتقديم مجموعة من حالات الاختبار الرئيسية التي تم تنفيذها على نظام "SkillCraft"، بالإضافة إلى عرض النتائج التي تم التوصل إليها وتقييم لواجهات المستخدم النهائية.

1.5- تنفيذ الواجهة الخلفية (Backend Implementation)

تم تنفيذ الواجهة الخلفية بالاعتماد على معمارية Modular Monolith، حيث تم تقسيم النظام إلى وحدات منطقية منفصلة، وكل وحدة تتبع بنية ثلاثية الطبقات (DAL, BLL, API).

1.1.5- طبقة واجهة برمجة التطبيقات الموحدة (SkillCraft.Api)

تعمل هذه الطبقة كبوابة دخول وحيدة (Unified Gateway) للنظام بأكمله، حيث تستقبل جميع طلبات HTTP من الواجهة الأمامية وتوجهها إلى الوحدة والخدمة المناسبة.

- **Program.cs**: هو الملف المركزي لإعداد وتشغيل الواجهة الخلفية. تم فيه تسجيل جميع الخدمات والمستودعات من مختلف الوحدات باستخدام آلية حقن التبعية (Dependency Injection). كما تم فيه تكوين إعدادات المصادقة باستخدام JWT Bearer Tokens، وتحديد سياسات التفويض القائمة على الأدوار، وإعداد سياسة CORS للسماح للواجهة الأمامية بالتواصل مع الـ API.
- **Controllers**: يحتوي هذا المجلد على وحدات التحكم الخاصة بكل وحدة، مثل UsersController و RoadmapsController. تعمل هذه المتحكمات كنقاط نهاية (Endpoints) للـ API، حيث تستقبل الطلبات وتستدعي الخدمات المناسبة من طبقة منطق العمل.

2.1.5- تنفيذ الوحدات (Modules)

كل وحدة من الوحدات الأربع تمثل جزءاً متخصصاً من النظام، وتحتوي على طبقتي BLL و DAL الخاصتين بها.

1. وحدة إدارة المستخدمين (UsersManagement Module)

- طبقة منطق العمل (BLL):

- **AuthService.cs**: هي الخدمة المحورية في هذه الوحدة. تحتوي على منطق العمل لعمليتي التسجيل وتسجيل الدخول. في دالة `SignUpAsync`، يتم التحقق من عدم وجود البريد الإلكتروني مسبقاً، ومن ثم تشفير كلمة المرور باستخدام SHA512 لحفظها.
- **TokenService.cs**: مسؤولة عن إنشاء توكن JWT بعد نجاح عملية تسجيل الدخول، حيث يتم تضمين معلومات المستخدم الأساسية (مثل المعرف والدور) داخل التوكن.

- طبقة الوصول للبيانات (DAL):

- **UserRepository.cs**: هذا المستودع مسؤول عن التفاعل المباشر مع قاعدة بيانات SQL Server باستخدام Entity Framework Core. يحتوي على دوال مثل `AddAsync` و `GetByEmailAsync` لتنفيذ عمليات القراءة والكتابة على جدول المستخدمين.

```
4 references | AbSamrah, 3 days ago | 1 author, 7 changes
public class UserRepository : IUserRepository
{
    private readonly UsersDbContext _usersDbContext;

    1 reference | AbSamrah, 3 days ago | 1 author, 3 changes
    public UserRepository(UsersDbContext usersDbContext) {
        _usersDbContext = usersDbContext;
    }

    5 references | AbSamrah, 3 days ago | 1 author, 5 changes
    public async Task AddAsync(User user)
    {
        user.Id = Guid.NewGuid();
        await _usersDbContext.AddAsync(user);
        await _usersDbContext.SaveChangesAsync();
    }

    2 references | AbSamrah, 134 days ago | 1 author, 1 change
    public async Task<User> DeleteAsync(Guid id) { }

    2 references | AbSamrah, 14 days ago | 1 author, 3 changes
    public async Task<List<User>> GetAllAsync(string? email = null, string? firstName = null, string? lastName = null, int pageNumber = 0, int pageSize = 10) { }

    3 references | AbSamrah, 111 days ago | 1 author, 2 changes
    public async Task<User> GetAsync(Guid id) { }

    8 references | AbSamrah, 114 days ago | 1 author, 1 change
    public async Task<User> GetByEmailAsync(string email) { }

    3 references | AbSamrah, 111 days ago | 1 author, 2 changes
    public async Task<User> UpdateAsync(User user) { }

    9 references | AbSamrah, 114 days ago | 1 author, 1 change
    public async Task<bool> UserExistsAsync(string email) { }
}
```

رسم توضيحي 10 User repository

2. وحدة إدارة خرائط الطريق (RoadmapManagement Module)

- طبقة منطق العمل (BLL):

- **RoadmapsService.cs**: تحتوي على منطق إنشاء وعرض roadmaps.

- تنفيذ الذكاء الاصطناعي:

- عند طلب توليد خريطة طريق بالذكاء الاصطناعي، تقوم StrategyFactory بإنشاء

- كائن من نوع AiRoadmapCreationStrategy.

- تقوم استراتيجية الذكاء الاصطناعي (AiRoadmapCreationStrategy) بصياغة

- الطلب (Prompt) وإرساله إلى GeminiAiAdapter. هذا ال Adapter هو

- المسؤول عن التواصل الفعلي مع خدمة Gemini الخارجية.

- بعد استلام استجابة JSON من ال AI، تقوم الاستراتيجية بمعالجتها وتحويلها إلى

- Roadmap, Milestone, Step قبل حفظها.

- طبقة الوصول للبيانات (DAL):

- **RoadmapRepository.cs**: يتفاعل هذا المستودع مع قاعدة بيانات MongoDB

- باستخدام مكتبة MongoDB.Driver لحفظ واسترجاع المستندات الخاصة ب roadmap

- ومكوناتها.


```

2 references | AbSamrah, 7 days ago | 1 author, 4 changes
public class RoadmapDbContext : IRoadmapDbContext
{
    2 references | AbSamrah, 92 days ago | 1 author, 1 change
    private IMongoDatabase Database { get; set; }
    5 references | AbSamrah, 92 days ago | 1 author, 1 change
    public IClientSessionHandle Session { get; private set; }
    4 references | AbSamrah, 92 days ago | 1 author, 1 change
    public MongoClient MongoClient { get; private set; }
    private readonly List<Func<Task>> _commands;
    private readonly IConfiguration _configuration;
    private readonly bool _supportsTransactions;

    0 references | AbSamrah, 7 days ago | 1 author, 3 changes
    public RoadmapDbContext(IConfiguration configuration)
    {
        _configuration = configuration;
        _commands = new List<Func<Task>>();

        var connectionString = _configuration["MongoSettings:Connection"];
        var dbName = _configuration["MongoSettings:Databases:RoadmapDB"];

        MongoClient = new MongoClient(connectionString);
        Database = MongoClient.GetDatabase(dbName);
        _supportsTransactions = MongoClient.Cluster.Description.Type == ClusterType.ReplicaSet;
    }

```

رسم توضيحي 11 تنفيذ RoadmapDbContext

3. وحدة إدارة الاختبارات (QuizzesManagement Module)

- طبقة منطق العمل (BLL):
- QuizService.cs: تحتوي على منطق إنشاء وعرض quizzes.
- تنفيذ الذكاء الاصطناعي: عند طلب توليد اختبار بالذكاء الاصطناعي، يتم اتباع نفس الآلية: تقوم StrategyFactory بإنشاء AiQuizCreationStrategy، والتي تستخدم GeminiAiAdapter للتواصل مع Gemini AI وطلب أسئلة من أنواع محددة (MCQ, True/False).
- طبقة الوصول للبيانات (DAL):
- QuizRepository.cs: مسؤول عن حفظ واسترجاع مستندات الاختبارات من قاعدة بيانات MongoDB.

4. وحدة إدارة الملفات الشخصية (ProfilesManagement Module)

- طبقة منطق العمل (BLL):

- **ProfileService.cs**: تحتوي على منطق العمليات المتعلقة بالملف الشخصي للمستخدم، مثل إضافة roadmap للمتابعة، أو تحديث حالة step معينة (إنجازها)، أو جلب جميع roadmaps التي يتابعها المستخدم.
- طبقة الوصول للبيانات (DAL):
- **ProfileRepository.cs**: يتفاعل مع قاعدة بيانات MongoDB لحفظ واسترجاع مستندات UserProfile التي تحتوي على تقدم كل مستخدم.

2.5- تنفيذ الواجهة الأمامية (Frontend Implementation)

تم بناء الواجهة الأمامية باستخدام مكتبة React، مع تنظيم المشروع في مجلدات واضحة (pages, components, api, context).

1.2.5- التواصل مع الواجهة الخلفية

تم بناء الواجهة الأمامية باستخدام مكتبة React، مع هيكلية المشروع بشكل منظم لتسهيل الصيانة والتطوير.

- **src/pages**: يحتوي هذا المجلد على المكونات الرئيسية التي تمثل صفحات كاملة في التطبيق، مثل HomePage.jsx, LoginPage.jsx, RoadmapPage.jsx, ContentDashboardPage.jsx.
- **src/components**: يحتوي على المكونات الصغيرة القابلة لإعادة الاستخدام في مختلف أنحاء التطبيق، مثل NavBar.jsx, Button.jsx.
- **src/api**: هذا المجلد هو المسؤول عن كل ما يتعلق بالتواصل مع الواجهة الخلفية.
 - **apiClient.js**: أهم ملف في هذا المجلد. يتم فيه إعداد نسخة من مكتبة Axios مع استخدام Interceptors. هذا الـ Interceptor يعترض كل طلب قبل إرساله ليقوم تلقائياً بإرفاق توكن المصادقة (JWT) في ترويسة الطلب إذا كان موجوداً، مما يغنينا عن إضافته يدوياً في كل مرة.
- **src/context**: يستخدم لإدارة الحالات العامة (Global State) في التطبيق.
 - **AuthContext.js**: أهم ملف في هذا المجلد. يوفر Context يقوم بتخزين معلومات المستخدم المسجل دخوله (التوكن، الاسم، الدور) ويجعلها متاحة لجميع المكونات دون الحاجة لتمريرها بشكل يدوي عبر شجرة المكونات.

- **src/App.js**: هو الملف الرئيسي الذي يتم فيه تعريف مسارات التطبيق باستخدام مكتبة React Router. يتم فيه استخدام مكون PrivateRoute.jsx لحماية المسارات التي تتطلب صلاحيات معينة، حيث يتم التحقق من دور المستخدم قبل السماح له بالوصول.

3.5- تنفيذ قواعد البيانات

لتحقيق أقصى استفادة من نقاط القوة لكل تقنية، اعتمد المشروع على نهج هجين (Hybrid Approach) في إدارة البيانات، حيث تم استخدام نوعين مختلفين من قواعد البيانات: SQL Server كقاعدة بيانات علائقية، و MongoDB كقاعدة بيانات NoSQL موجهة للمستندات. تم تكوين وإدارة الاتصال بكل منهما بشكل مركزي في الواجهة الخلفية.

• Entity Framework Core و SQL Server (لبيانات المستخدمين)

تم اختيار قاعدة بيانات SQL Server لتخزين البيانات المنظمة والحساسة مثل معلومات المستخدمين والأدوار والصلاحيات، وذلك لما توفره من موثوقية وأمان وتكامل للبيانات (Data Integrity). تم استخدام Entity Framework Core (EF Core) كـ Mapper (ORM) لتسهيل التفاعل مع قاعدة البيانات.

1. تعريف سلسلة الاتصال (Connection String):

في ملف appsettings.json، تم تعريف سلسلة الاتصال التي تحتوي على جميع المعلومات اللازمة للوصول إلى قاعدة بيانات SQL Server، مثل اسم الخادم، اسم قاعدة البيانات، ومعلومات المصادقة.

2. إعداد سياق قاعدة البيانات (DbContext):

تم إنشاء صف UsersDbContext.cs الذي يرث من DbContext الخاص بـ EF Core. يعمل هذا الصف "كسياق" أو "جلسة" للتواصل مع قاعدة البيانات. بداخله، يتم تعريف الكيانات التي ستحول إلى جداول في قاعدة البيانات باستخدام <DbSet>T.

3. تسجيل الخدمة في Program.cs:

لتمكين حقن التبعية (Dependency Injection) واستخدام UsersDbContext في جميع أنحاء التطبيق، تم

تسجيله كخدمة في ملف Program.cs. هنا يتم ربط UsersDbContext بسلسلة الاتصال التي تم تعريفها في الخطوة الأولى.

```
builder.Services.AddDbContext<UsersDbContext>(options =>
{
    options.UseSqlServer(connectionString: builder.Configuration.GetConnectionString("UsersMangement"));
});
```

رسم توضيحي 12 حقن UsersDbContext

4. استخدام المستودع (Repository):

يقوم UserRepository.cs بطلب نسخة من UsersDbContext عبر المنشئ (Constructor). هذا يسمح للمستودع بتنفيذ عمليات CRUD (إنشاء، قراءة، تحديث، حذف) على قاعدة البيانات باستخدام دوال EF Core البسيطة والواضحة.

• MongoDB (لبيانات المحتوى التعليمي)

تم اختيار MongoDB لتخزين المحتوى الديناميكي والمتغير بطبيعته، مثل خرائط الطريق ومراحلها وخطواتها، بالإضافة إلى الاختبارات. مرونة MongoDB في التعامل مع البيانات غير المهيكلة (Schema-less) تجعلها مثالية لهذا النوع من المحتوى.

1. تعريف إعدادات الاتصال:

في ملف appsettings.json، تم تعريف الإعدادات الخاصة بـ MongoDB، والتي تشمل سلسلة الاتصال واسم قاعدة البيانات.

2. تسجيل الإعدادات والعميل (Client):

في Program.cs، تم تسجيل إعدادات MongoDB كـ Singleton، مما يضمن وجود نسخة واحدة منها طوال عمر التطبيق. كما تم تسجيل عميل (IMongoClient) MongoDB كـ Singleton أيضاً لضمان إدارة فعالة للاتصالات مع قاعدة البيانات.

3. إنشاء سياق مخصص (Custom DbContext):

على الرغم من أن MongoDB لا تحتوي على مفهوم DbContext مثل EF Core، فقد تم إنشاء صفوف سياق مخصصة لكل وحدة مثل RoadmapDbContext.cs لمحاكاة هذا النمط وتوفير نقطة وصول مركزية للمجموعات

(Collections). يقوم هذا السياق باستقبال IMongoClient وإعدادات قاعدة البيانات للحصول على المجموعات المطلوبة.

4. استخدام المستودع (Repository):

يقوم RoadmapRepository.cs بطلب نسخة من RoadmapDbContext للوصول إلى مجموعة Roadmaps وتنفيذ العمليات عليها باستخدام الدوال التي يوفرها MongoDB Driver الرسمي.

الفصل السادس

الاختبار والنتائج

بعد اكتمال مرحلة التنفيذ، تأتي مرحلة الاختبار كخطوة حيوية لضمان جودة المنتج البرمجي والتأكد من أنه يلبي جميع المتطلبات الوظيفية وغير الوظيفية التي تم تحديدها في الفصل الثالث. يهدف هذا الفصل إلى استعراض استراتيجية الاختبار التي تم اتباعها، وتقديم النتائج الملموسة التي تم الحصول عليها من خلال تنفيذ حالات الاختبار على الواجهة الخلفية لنظام "SkillCraft".

1.6- الاختبارات الوظيفية (Functional Testing)

تهدف الاختبارات الوظيفية إلى التحقق من أن كل جزء من النظام يعمل كما هو متوقع وفقاً للمتطلبات التي تم تحديدها في الفصل الثالث.

1.1.6- استراتيجية الاختبار الوظيفي

نظراً للبنية التصميمية متعددة الطبقات (DAL, BLL, API) التي تم اعتمادها في الواجهة الخلفية، تم اتباع استراتيجية اختبار شاملة ومتدرجة لضمان تغطية كافة جوانب النظام. تنقسم هذه الاستراتيجية إلى ثلاثة أنواع رئيسية من الاختبارات:

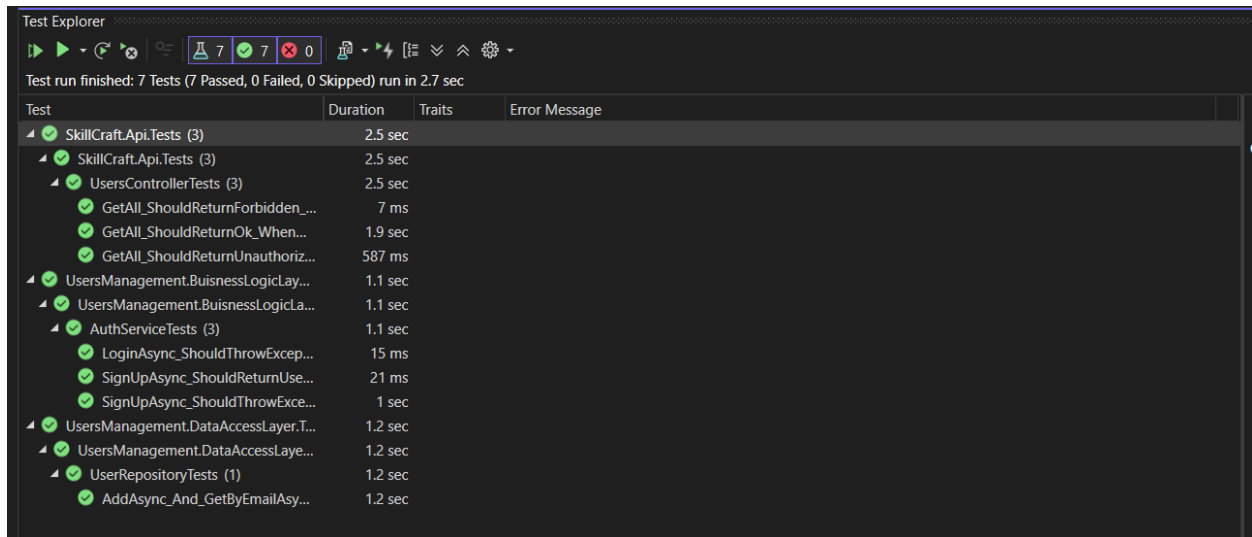
- **اختبار الوحدة (Unit Test):** تركز هذه الاختبارات على أصغر جزء من الكود، وهي طبقة منطق العمل (BLL). الهدف منها هو عزل الخدمات (Services) مثل AuthService والتحقق من أن منطق العمل الخاص بها يعمل بشكل صحيح بمعزل عن أي اعتماديات خارجية مثل قاعدة البيانات.
- **اختبار التكامل (Integration Test):** تتحقق هذه الاختبارات من أن طبقة الوصول للبيانات (DAL) تتفاعل مع قاعدة البيانات بشكل صحيح. تم اختبار المستودعات (Repositories) مثل UserRepository للتأكد من أن عمليات الإضافة والقراءة والتعديل والحذف (CRUD) تعمل كما هو متوقع.
- **اختبارات واجهة برمجة التطبيقات (API Tests):** تُعد هذه الاختبارات بمثابة اختبارات شاملة (End-to-End) للواجهة الخلفية. تم من خلالها محاكاة طلبات HTTP حقيقية إلى نقاط النهاية (Endpoints) في المتحكمات (Controllers) مثل UsersController.

2.1.6- نتائج الاختبارات الوظيفية

تم تنفيذ مجموعة من الاختبارات الآلية التي تغطي الطبقات الثلاث لوحدة إدارة المستخدمين. كانت نتيجة تنفيذ جميع الاختبارات ناجحة بنسبة 100%، مما يؤكد على أن الوحدة تعمل بشكل صحيح وموثوق.

توزعت الاختبارات الناجحة على النحو التالي:

- **SkillCraft.Api.Tests (3 اختبارات ناجحة):** تم اختبار UsersController للتحقق من أن آلية التحقق من الصلاحيات (Authorization) تعمل بشكل صحيح، حيث أثبتت أن نقطة النهاية GetAll لا يمكن الوصول إليها إلا من قبل المستخدمين الذين يملكون دور "Admin".
- **UsersManagement.BusinessLogicLayer.Tests (3 اختبارات ناجحة):** تم اختبار AuthService للتأكد من أن منطق المصادقة يعمل كما هو متوقع، بما في ذلك سيناريوهات تسجيل الدخول والاشتراك الناجحة والفاشلة.
- **UsersManagement.DataAccessLayer.Test (اختبار واحد ناجح):** تم إجراء اختبار تكاملي على UserRepository للتأكد من أن عملية إضافة مستخدم جديد إلى قاعدة البيانات واسترجاعه تتم بنجاح.



Test	Duration	Traits	Error Message
✓ SkillCraft.Api.Tests (3)	2.5 sec		
✓ SkillCraft.Api.Tests (3)	2.5 sec		
✓ UsersControllerTests (3)	2.5 sec		
✓ GetAll_ShouldReturnForbidden...	7 ms		
✓ GetAll_ShouldReturnOk_When...	1.9 sec		
✓ GetAll_ShouldReturnUnauthoriz...	587 ms		
✓ UsersManagement.BuisnessLogicLay...	1.1 sec		
✓ UsersManagement.BuisnessLogicLay...	1.1 sec		
✓ AuthServiceTests (3)	1.1 sec		
✓ LoginAsync_ShouldThrowExcep...	15 ms		
✓ SignUpAsync_ShouldReturnUse...	21 ms		
✓ SignUpAsync_ShouldThrowExce...	1 sec		
✓ UsersManagement.DataAccessLayer.T...	1.2 sec		
✓ UsersManagement.DataAccessLaye...	1.2 sec		
✓ UserRepositoryTests (1)	1.2 sec		
✓ AddAsync_And_GetByEmailAsy...	1.2 sec		

رسم توضيحي 13 نتائج بعض الاختبارات على وحدة إدارة المستخدمين

2.6- الاختبارات غير الوظيفية (Non-Functional Testing)

تهدف هذه الاختبارات إلى قياس جودة أداء النظام تحت ظروف مختلفة، للتحقق من تلبيته للمتطلبات غير الوظيفية مثل الأداء وقابلية التوسع.

1.2.6- اختبارات الأداء (Performance Testing)

تم التركيز على اختبارات الأداء لتقييم استجابة النظام وقدرته على تحمل الضغط. تم استخدام أداة k6، وهي أداة حديثة ومفتوحة المصدر لاختبارات التحميل، لمحاكاة سيناريوهات استخدام حقيقية بأعداد مختلفة من المستخدمين الافتراضيين (VUs).

تم تصميم وتنفيذ اختبارات تحميل وإجهاد (Load and Stress Tests) على نقاط النهاية الحيوية في النظام، وأهمها:

- عملية تسجيل الدخول (/api/auth/login)
- جلب جميع خرائط الطريق (/api/roadmaps)
- توليد خريطة طريق باستخدام الذكاء الاصطناعي (/api/roadmaps/ ai)

2.2.6- نتائج اختبارات الأداء

في هذا القسم، سيتم عرض بعض النتائج الرئيسية التي تم الحصول عليها من اختبارات الأداء.

- نتائج soak test على API get all roadmaps لمدة 4 ساعات.

soak test 6 جدول

المقياس	القيمة
متوسط زمن الاستجابة (avg)	3.91ms
أدنى زمن استجابة (min)	0s
أعلى زمن استجابة (max)	422.22ms
معدل الطلبات في الثانية (reqs/s)	41.23req/s
معدل الأخطاء (http_req_failed)	0.00%

كما هو موضح في الرسم التوضيحي 14:


```

Administrator: Windows PowerShell
/ _____ \
WARN[0000] There were unknown fields in the options exported in the script error="json: unknown field '\executor'"
execution: local
script: .\soak-test.js
output: -

scenarios: (100.00%) 1 scenario, 50 max VUs, 4h0m30s max duration (incl. graceful stop):
  * default: 50 looping VUs for 4h0m30s (gracefulStop: 30s)

THRESHOLDS
http_req_duration
  [ 'p(95)<500' p(95)=0.05ms
http_req_failed
  [ 'rate<0.01' rate=0.00%

TOTAL RESULTS
checks_total.....: 718968 41.227267/s
checks_succeeded.....: 100.00% 718968 out of 718968
checks_failed.....: 0.00% 0 out of 718968
[ get all successful (status 200)
[ response body contains roadmaps

CUSTOM
get_all_roadmaps_duration.....: avg=1.910000 min=0 med=1.0021 max=22.2200 p(90)=1.3451 p(95)=1.051955
HTTP
http_req_duration.....: avg=.93ms min=0 med=1.00ms max=22.22ms p(90)=.54ms p(95)=.05ms
[ expected_response:true .....: avg=.93ms min=0 med=1.00ms max=22.22ms p(90)=.54ms p(95)=.05ms
http_req_failed.....: 0.00% 0 out of 359435
http_reqs.....: 359435 20.613624/s

EXECUTION
iteration_duration.....: avg=.96s min=1s med=1s max=0.913635s p(90)=1s p(95)=1s
iterations.....: 359435 20.610814/s
vus.....: 1 min=1 max=50
vus_max.....: 50 min=50 max=50

NETWORK
data_received.....: 3.7 GB 211.88/s
data_sent.....: 41 MB 2.3 kb/s

running (4h50m39.1s), 00/50 VUs, 359435 complete and 50 interrupted iterations
default [-----] 50 VUs 4h0m30s
PS C:\Users\lab-Samrah\Desktop\Amman\MyRoadmap\SkillCraft\Backend\SkillCraft.Api\PerformanceTests\GetAllRoadmapsTests>

```

رسم توضيحي 14 soak test on get all roadmaps

- نتائج spike test على get all roadmaps API.

spike test 7 جدول

المقياس	القيمة
متوسط زمن الاستجابة (avg)	5.29ms
أدنى زمن استجابة (min)	0s
أعلى زمن استجابة (max)	396.62ms
معدل الطلبات في الثانية (reqs/s)	1092.51req/s
معدل الأخطاء (http_req_failed)	0.00%

كما هو موضح في الرسم التوضيحي 15:

```

Administration Windows PowerShell
WARN[0000] There were unknown fields in the options exported in the script  error="json: unknown field \\"executor\\"""
  execution: local
    script: ./spike-test.js
    output: -

  scenarios: (100.00%) 1 scenario, 1000 max VUs, 2m50s max duration (incl. graceful stop):
    * default: Up to 1000 looping VUs for 2m20s over 4 stages (gracefulRampDown: 30s, gracefulStop: 30s)

THRESHOLDS
get_all_roadmaps_duration
  * "p(95)<800" p(95)=5.7377
http_req_duration
  * "p(95)<500" p(95)=5.73ms
http_req_failed
  * "rate<0.01" rate=0.00%

TOTAL RESULTS
checks_total.....: 153970 100% 510159/s
checks_succeeded.....: 100.00% 153970 out of 153970
checks_failed.....: 0.00% 0 out of 153970
  * got all successful (status 200)
  * response body contains roadmaps

CUSTOM
get_all_roadmaps_duration.....: avg=,287691 min=0 med=,9948 max=,96,6181 p(90)=,0511 p(95)=5.7377
HTTP
http_req_duration.....: avg=,28ms min=0 med=,1,29ms max=,96,61ms p(90)=,05ms p(95)=5.73ms
{ expected_response:true }.....: avg=,28ms min=0 med=,1,29ms max=,96,61ms p(90)=,05ms p(95)=5.73ms
http_req_failed.....: 0.00% 0 out of 76985
http_reqs.....: 76985 540,25518/s

EXECUTION
iteration_duration.....: avg=0 min=0 med=0 max=,1,4s p(90)=,02s p(95)=,1,02s
iterations.....: 76985 540,25518/s
vus.....: 1 min=1 max=1000
vus_max.....: 1000 min=1000 max=1000

NETWORK
data_received.....: 710 MB 5.0 MB/s
data_sent.....: 0.0 MB 0.0 MB/s

running (2m20.9s), 80000/1000 VUs, 76985 complete and 0 interrupted iterations
default [=====] 0000/1000 VUs 2m20s

```

رسم توضيحي 15 spike test on get all roadmaps

- نتائج stress test على API get roadmap by id.

جدول 8 stress test

المقياس	القيمة
متوسط زمن الاستجابة (avg)	22.97ms
أدنى زمن استجابة (min)	2.39ms
أعلى زمن استجابة (max)	1.03s
معدل الطلبات في الثانية (reqs/s)	441.47req/s
معدل الأخطاء (http_req_failed)	0.00%

كما هو موضح في الرسم التوضيحي 16:

```
Administrator: Windows PowerShell
WARN[0000] There were unknown fields in the options exported in the script error="json: unknown field \"executor\""
execution: local
script: ./stress-test.js
output:

scenarios: (100.000) 1 scenario, 400 vus, 10m10s max duration (1m1s graceful stop):
  * default: Up to 400 looping VUs for 38m0s over 9 stages (gracefulRampDown: 30s, gracefulStop: 30s)

THRESHOLDS
get_roadmap_duration
  [ "p(95)<800" p(95)=114.46682
http_req_duration
  [ "p(95)<500" p(95)=114.46ms
http_req_failed
  [ "rate<0.01" rate=0.00%

TOTAL RESULTS
checks_total.....: 1007350/441,742053/s
checks_succeeded.....: 100.00% 1007350 out of 1007350
checks_failed.....: 0.00% 0 out of 1007350
  [ get_roadmap_successful (status: 200)
  [ response body is roadmap

CUSTOM
get_roadmap_duration.....: avg=22.973s min=2.395s med=1.364s max=1809.3782 p(90)=53.7462 p(95)=114.46682

HTTP
http_req_duration.....: avg=22.97ms min=0.39ms med=1.36ms max=1.03s p(90)=53.74ms p(95)=114.46ms
  { expected_response_time }.....: avg=2.57ms min=0.39ms med=1.36ms max=1.03s p(90)=53.74ms p(95)=114.46ms
http_req_failed.....: 0.00% 0 out of 503675
http_reqs.....: 503675 220.871026/s

EXECUTION
iteration_duration.....: avg=1.02s min=0s med=0s max=1.06s p(90)=1.06s p(95)=1.12s
iterations.....: 503675 220.871026/s
vus.....: 3 400 1 max=400
vus_max.....: 400 400 1 max=400

NETWORK
data_received.....: 9.9 GB 4.3 MB/s
data_sent.....: 70 MB 31.40/s

running (38m0.4s), 000/400 VUs, 503675 complete and 0 interrupted iterations
```

رسم توضيحي 16 stress test on get roadmap by id

- نتائج load test على API get all roadmaps.

جدول 9 load test

المقياس	القيمة
متوسط زمن الاستجابة (avg)	6.72ms
أدنى زمن استجابة (min)	0.62ms
أعلى زمن استجابة (max)	578.34ms
معدل الطلبات في الثانية (reqs/s)	170.23req/s
معدل الأخطاء (http_req_failed)	0.00%

كما هو موضح في الرسم التوضيحي 17:

```
Command Prompt
output: -

scenarios: (100.00%) 1 scenario, 100 max VUs, 7m30s max duration (incl. graceful stop):
* default: Up to 100 looping VUs for 7m0s over 3 stages (gracefulRampDown: 30s, gracefulStop: 30s)

THRESHOLDS

http_req_duration
  [p(95)<500] p(95)=16.97ms

http_req_failed
  [rate<0.01] rate=0.00%

TOTAL RESULTS

checks_total.....: 71526 170.229572/s
checks_succeeded.....: 100.00% 71526 out of 71526
checks_failed.....: 0.00% 0 out of 71526

  [get all successful (status 200)]
  [response body contains roadmaps]

CUSTOM
get_all_roadmaps_duration.....: avg=6.72422s min=0.6239 med=4.1478 max=578.347 p(90)=9.67466 p(95)=16.97698

HTTP
http_req_duration.....: avg=6.72ms min=623.9µs med=4.14ms max=578.34ms p(90)=9.67ms p(95)=16.97ms
{ expected response:true }.....: avg=6.72ms min=623.9µs med=4.14ms max=578.34ms p(90)=9.67ms p(95)=16.97ms
http_req_failed.....: 0.00% 0 out of 35763
http_reqs.....: 35763 85.114786/s

EXECUTION
iteration_duration.....: avg=1s min=1s med=1s max=1.58s p(90)=1.01s p(95)=1.02s
iterations.....: 35763 85.114786/s
vus.....: 2 min=2 max=100
vus_max.....: 100 min=100 max=100

NETWORK
data_received.....: 92 MB 219 kB/s
data_sent.....: 4.1 MB 9.7 kB/s

running (7m00.2s), 000/100 VUs, 35763 complete and 0 interrupted iterations
default [*****] 000/100 VUs 7m0s
```

رسم توضيحي 17 load test on get all roadmaps

الفصل السابع

الخاتمة والآفاق المستقبلية

يقدم هذا الفصل خلاصة شاملة للمشروع، حيث نستعرض الإنجازات الرئيسية التي تم تحقيقها وكيف نجحت المنصة في تلبية الأهداف الموضوعية. كما يلقي الفصل نظرة على المستقبل، ويقترح مجموعة من الميزات والتحسينات المحتملة التي يمكن إضافتها لتعزيز قيمة "SkillCraft" وتحويله إلى بيئة تعليمية متكاملة.

1.7- الخاتمة

في ختام رحلة هذا المشروع، تم بنجاح تصميم وتطوير منصة "SkillCraft"، وهي تطبيق ويب تعليمي متكامل يهدف إلى حل مشكلة التشتت وغياب التوجيه التي تواجه المعلمين في المجالات التقنية. لقد تم تحقيق الأهداف الأساسية التي تم وضعها في بداية المشروع، حيث تم بناء نظام يوفر خرائط طريق تعليمية واضحة، ويتيح للمستخدمين تتبع تقدمهم، ويقدم لهم اختبارات لتقييم معرفتهم.

من خلال اعتماد بنية Modular Monolith، تم بناء واجهة خلفية قوية ومنظمة، قابلة للصيانة والتطوير. كما أثبت النهج الهجين في استخدام قواعد البيانات (SQL Server و MongoDB) فعاليته في التعامل مع مختلف أنواع البيانات بكفاءة. ومن أبرز ما تم إنجازه هو الدمج الناجح لخدمات الذكاء الاصطناعي (Gemini 2.5 Flash API)، مما أضاف ميزة تنافسية قوية للمنصة عبر تمكينها من توليد محتوى تعليمي مخصص بشكل فوري.

أكدت نتائج الاختبارات الوظيفية وغير الوظيفية (اختبارات الأداء) أن النظام يعمل بشكل مستقر، ويلبي المتطلبات المحددة، وقادر على التعامل مع الأحمال المتوقعة، مما يثبت جودة الحل الهندسي المتبع. بذلك، لا يمثل "SkillCraft" حلاً للمشكلة المطروحة فحسب، بل يشكل أساساً متيناً لمنصة تعليمية شاملة ومبتكرة.

2.7- الآفاق المستقبلية

على الرغم من أن النسخة الحالية من المشروع تلبي الأهداف الأساسية، إلا أن البنية المرنة التي تم تصميمها تفتح الباب أمام العديد من التحسينات والتوسعات المستقبلية التي يمكن أن تحول "SkillCraft" إلى منظومة تعليمية متكاملة. تشمل أهم هذه الآفاق:

• تطوير نظام الاختبارات:

- إضافة أنواع جديدة من الأسئلة: يمكن تعزيز نظام الاختبارات ليشمل أنواعاً أكثر تفاعلية وعملية، مثل أسئلة كتابة الشيفرة البرمجية (Code Snippets) التي يتم تصحيحها تلقائياً، لتقييم المهارات التطبيقية للمستخدم بشكل أفضل.

• إثراء المحتوى التعليمي:

- إضافة مقالات ومشاريع: يمكن إثراء المنصة بمكتبة من المقالات التعليمية وأفكار المشاريع العملية المرتبطة بكل خطوة في خرائط الطريق. هذا سيساعد المستخدمين على تعميق فهمهم وتطبيق ما تعلموه في سياقات حقيقية.

• تعزيز الملف الشخصي للمستخدم:

- توليد السيرة الذاتية بالذكاء الاصطناعي: يمكن تطوير ميزة متقدمة تستخدم الذكاء الاصطناعي لتحليل المهارات التي اكتسبها المستخدم والخرائط التي أكملها، ومن ثم توليد سيرة ذاتية (CV) احترافية ومخصصة بشكل تلقائي، جاهزة للتقديم إلى فرص العمل.

• زيادة التفاعل والمشاركة:

- (Gamification): يمكن دمج عناصر الألعاب مثل النقاط، الشارات (Badges)، ولوحات الصدارة (Leaderboards) لتحفيز المستخدمين على الاستمرار في التعلم وزيادة تفاعلهم مع المنصة.

- بناء مجتمع تعليمي: يمكن إضافة ميزات اجتماعية مثل المنتديات أو مجموعات الدراسة لكل خريطة طريق، مما يسمح للمستخدمين بتبادل الخبرات والمعارف وحل المشاكل بشكل جماعي.

إن هذه التحسينات المقترحة ستساهم في جعل "SkillCraft" ليس فقط أداة لتنظيم التعلم، بل بيئة تعليمية شاملة ومحفزة تساعد المهندسين والتقنيين على تحقيق أهدافهم المهنية بكفاءة وفعالية.

المراجع

1. **ASP.NET Core Documentation.** Microsoft. <https://docs.microsoft.com/en-us/aspnet/core/>
2. **Axios Documentation.** <https://axios-http.com/docs/intro>
3. **Entity Framework Core Documentation.** Microsoft. <https://docs.microsoft.com/en-us/ef/core/>
4. Fowler, M. (2014). **Patterns of Enterprise Application Architecture.** Addison-Wesley Professional.
5. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). **Design Patterns: Elements of Reusable Object-Oriented Software.** Addison-Wesley Professional.
6. **Gemini API Documentation.** Google. <https://ai.google.dev/docs>
7. **MongoDB Documentation.** MongoDB Inc. <https://docs.mongodb.com/>
8. **React Documentation.** Meta. <https://reactjs.org/docs/getting-started.html>
9. **React Router Documentation.** <https://reactrouter.com/en/main>