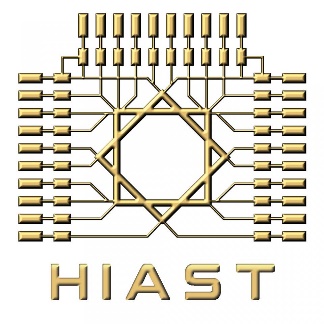
**الجمهورية العربية السورية**

**المعهد العالي للعلوم التطبيقية والتكنولوجيا**

**قسم النظم المعلوماتيّة**

**العام الدراسي 2025/2024**

**مشروع سنة رابعة**

هندسة البرمجيات والذكاء الصنعي

مساعد شخصي لتطوير المهارات

Assistant for growing skills

تقديم

**عمّار سمرة**

إشراف

**د. رياض سنبل**

23/7/2025

الإهداء

أُهدي هذا العمل إلى أهلي وأصدقائي.

كلمة شكر

أتقدم بخالص الشكر للدكتور رياض سنبل للمساعدة على إنجاز هذا العمل، ولتوجيهاته القيّمة ونصائحه المستمرّة.

الخلاصة

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

توفر المنصة مسارات تعليمية تفاعلية (Roadmaps) 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 (Quizzes) مدمجة.

بذلك، لا يقتصر دور "SkillCraft" على كونه مجرد مستودع للمعلومات، بل يعمل كمرشد شخصي يمكّن 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

[قائمة الأشكال 8](#_Toc205036500)

[قائمة الجداول 9](#_Toc205036501)

[الرموز والاختصارات 10](#_Toc205036502)

[مقدمة عامّة 11](#_Toc205036503)

[الفصل الأول 12](#_Toc205036504)

[1.1- المشكلة وأهميّتها 12](#_Toc205036505)

[2.1- الحل المقترح: منصة SkillCraft 12](#_Toc205036506)

[3.1- رؤية المشروع وأهدافه 13](#_Toc205036507)

[4.1- الجمهور المستهدف 13](#_Toc205036508)

[الفصل الثاني 14](#_Toc205036509)

[1.2- تحليل المنصات التعليمية الحالية 14](#_Toc205036510)

[1.1.2- موقع Roadmap.sh 14](#_Toc205036511)

[2.1.2- منصات الكورسات (FreeCodeCamp, Codecademy) 15](#_Toc205036512)

[3.2- خلاصة الدراسة 16](#_Toc205036513)

[الفصل الثالث 17](#_Toc205036514)

[1.3- تحليل النظام 17](#_Toc205036515)

[2.3- المتطلّبات 18](#_Toc205036516)

[1.2.3- المتطلّبات الوظيفية 18](#_Toc205036517)

[2.2.3- المتطلّبات غير الوظيفيّة 19](#_Toc205036518)

[3.3- مخطط حالات الاستخدام 20](#_Toc205036519)

[4.3- الوصف النصيّ لحالات الاستخدام ومخطّطات التسلسل 23](#_Toc205036520)

[1.3.3- حالة الاستخدام: تسجيل دخول 23](#_Toc205036521)

[2.3.3- حالة الاستخدام: إنشاء حساب جديد 26](#_Toc205036522)

[الفصل الرابع 29](#_Toc205036523)

[1.4- البنية التصميمية (Software Architecture) 29](#_Toc205036524)

[1.1.4- معمارية العميل-الخادم (Client-Server Architecture) 29](#_Toc205036525)

[2.1.4- المعمارية المتجانسة المقسّمة (Modular Monolith) 30](#_Toc205036526)

[2.4- بيئة العمل وأدوات التطوير 30](#_Toc205036527)

[1.2.4- بيئات التطوير المتكاملة (IDEs): 30](#_Toc205036528)

[2.2.4- نظام إدارة الإصدارات (Version Control System) 31](#_Toc205036529)

[3.4- أطر العمل والمكتبات المستخدمة 31](#_Toc205036530)

[1.3.4- الواجهة الخلفية (Backend) 31](#_Toc205036531)

[2.3.4- الواجهة الأمامية (Frontend) 32](#_Toc205036532)

[4.4- تقنيات قواعد البيانات 32](#_Toc205036533)

[5.4- الأنماط التصميمية المتبعة (Design Patterns) 33](#_Toc205036534)

[1.5.4- نمط حقن التبعية (Dependency Injection - DI) 33](#_Toc205036535)

[2.5.4- نمط المستودع (Repository Pattern) 34](#_Toc205036536)

[3.5.4- نمط وحدة العمل (Unit of Work Pattern) 34](#_Toc205036537)

[4.5.4- نمط كائن نقل البيانات (Data Transfer Object - DTO) 34](#_Toc205036538)

[5.5.4- نمط الاستراتيجية (Strategy Pattern) 35](#_Toc205036539)

[6.5.4- نمط المصنع (Factory Pattern) 35](#_Toc205036540)

[7.5.4- نمط المحول (Adapter Pattern) 36](#_Toc205036541)

[الفصل الخامس 37](#_Toc205036542)

[1.5- تنفيذ الواجهة الخلفية (Backend Implementation) 37](#_Toc205036543)

[1.1.5- إعداد المشروع الرئيسي (SkillCraft.Api) 37](#_Toc205036544)

[2.1.5- تنفيذ وحدة إدارة المستخدمين (Users Management Module) 38](#_Toc205036545)

[3.1.5- تنفيذ وحدة إدارة خرائط الطريق (Roadmap Management Module) 38](#_Toc205036546)

[2.5- تنفيذ الواجهة الأمامية (Frontend Implementation) 38](#_Toc205036547)

[1.2.5- التواصل مع الواجهة الخلفية 39](#_Toc205036548)

[2.2.5- إدارة الحالة والتوجيه (State Management & Routing) 39](#_Toc205036549)

[3.2.5- تنفيذ الواجهات الرئيسية 40](#_Toc205036550)

[3.5- تنفيذ قواعد البيانات 40](#_Toc205036551)

[الفصل السادس 42](#_Toc205036552)

**قائمة الأشكال**

[رسم توضيحي 1 حالات الاستخدام للمسخدم المتلقي 21](#_Toc205048859)

[رسم توضيحي 2 حالات الاستخدام لمحرر المحتوى 22](#_Toc205048860)

[رسم توضيحي 3 مخطط حالات الاستخدام لمدير النظام 23](#_Toc205048861)

[رسم توضيحي 4 مخطط التتالي لعملية تسجيل الدخول 25](#_Toc205048862)

[رسم توضيحي 5 مخطط التتالي لعملية إنشاء حساب جديد 28](#_Toc205048863)

[رسم توضيحي 6 مخطط التتالي لعملية إضافة roadmap إلى الملف الشخصي 30](#_Toc205048864)

[رسم توضيحي 7 مخطط التتالي لحالة الاستخدام توليد roadmap بالذكاء الاصطناعي 33](#_Toc205048865)

[رسم توضيحي 8 هيكلية المشروع 36](#_Toc205048866)

[رسم توضيحي 9 استخدام Interceptors 45](#_Toc205048867)

[رسم توضيحي 10 لحماية المسارات 46](#_Toc205048868)

**قائمة الجداول**

[جدول 1 مقارنة بين موقعين ومشروعنا 16](#_Toc204439992)

[جدول 2 السيناريو الرئيسي لحالة تسجيل دخول 23](#_Toc204439993)

[جدول 3 المسار الرئيسي لحالة إنشاء حساب جديد 24](#_Toc204439994)

[جدول 4 مقارنة بين SQL و NoSQL 29](#_Toc204439995)

الرموز والاختصارات

|  |  |  |
| --- | --- | --- |
| الرمز | English | عربي |
|  | Roadmap | خارطة طريق |
| RBAC | Role-Based Access Control | بنية قائمة على الأدوار |

مقدمة عامّة

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 وأهميّتها

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 التشتت والضياع. فالمتعلم 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 وهيكلة عملية التعلم الذاتي من خلال تقديم خرائط طريق تفاعلية. كل خريطة طريق 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 المرجعيّة

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 يسدها. ستساعدنا هذه الدراسة المرجعية في إبراز الميزات التنافسية لمنصة "SkillCraft" وتأكيد القيمة المضافة التي ستقدمها للجمهور 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 خرائط طريق بصرية للمطورين، وهو المصدر الأساسي الذي استلهم منه مشروعنا فكرته.

**نقاط القوة:**

* **خرائط طريق 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:**

* **نقص التفاعل:** على الرغم من فائدته، إلا أنه يفتقر إلى عناصر تفاعلية مثل الاختبارات المدمجة أو المشاريع التطبيقية لتقييم الفهم.
* **غياب التقدير الزمني:**  لا يقدم تقديراً للوقت اللازم لإنجاز كل جزء من الخريطة، وهو ما قد يصعّب 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 "Skill Craft" أن يملأها بفعالية. تلخص المقارنة التالية كيف يتميز مشروعنا عن المنافسين:

جدول 1 مقارنة بين موقعين ومشروعنا

|  |  |  |  |
| --- | --- | --- | --- |
| الميزة | Roadmap.sh | Codecademy | SkillCraft (مشروعنا) |
| خرائط طريق 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 خرائط طريق 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 التحليليّة وتحليل المتطلّبات

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 قائمة على الأدوار (Role-Based Access Control - 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 خرائط الطريق، ومتابعة تقدمه، واستخدام الميزات التعليمية المتاحة.
2. **محرر المحتوى (editor):** هو المسؤول عن إنشاء وإدارة المحتوى التعليمي. يمتلك صلاحيات لإضافة وتعديل وحذف خرائط الطريق، والمراحل، والخطوات، والأسئلة.
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- المتطلّبات

1.2.3- المتطلّبات الوظيفية

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

**متطلّبات المستخدم:**

* **FR-U1:** يجب أن يسمح النظام للمستخدم بإنشاء حساب جديد.
* **FR-U2:** يجب أن يسمح النظام للمستخدم بتسجيل الدخول إلى حسابه.
* **FR-U3:** يجب أن يتمكن المستخدم من استعراض قائمة بجميع خرائط الطريق المتاحة.
* **FR-U4:** يجب أن يتمكن المستخدم من البحث عن خريطة طريق معينة.
* **FR-U5:** يجب أن يتمكن المستخدم من فتح ومشاهدة تفاصيل أي خريطة طريق.
* **FR-U6:** يجب أن يتمكن المستخدم من حفظ خريطة طريق في ملفه الشخصي وتتبع حالتها.
* **FR-U7:** يجب أن يتمكن المستخدم من تعديل حالة المهام (Steps) ضمن خريطة الطريق التي يتابعها (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 خريطة طريق عبر الذكاء الاصطناعي عن طريق تحديد موضوع.
* **FR-11:** يجب أن يتمكن المستخدم من طلب توليد أسئلة اختبار عبر الذكاء الاصطناعي حول موضوع معين.

**متطلّبات 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 خرائط الطريق (إضافة، تعديل، حذف).
* **FR-E4:** يجب أن يتمكن المحرر من إدارة المراحل والخطوات وربطها بخرائط الطريق.
* **FR-E5:** يجب أن يتمكن المحرر من إدارة بنك الأسئلة والاختبارات.

**متطلّبات 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- المتطلّبات غير الوظيفيّة

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 مهمة كمنجزة) شبه فورية.

**الأمان (Security):**

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

**قابليّة الاستخدام (Usability):**

* **NFR-6:** يجب أن تكون واجهة المستخدم سهلة وبديهية.
* **NFR-7:** يجب أن يكون تصميم الموقع متجاوباً (Responsive) ليعمل بشكل جيد على مختلف أحجام الشاشات (الحاسوب، الجهاز اللوحي، الهاتف المحمول).

**الموثوقيّة (Reliability):**

* **NFR-8:** يجب أن يكون النظام متاحاً للعمل بنسبة 99% من الوقت.

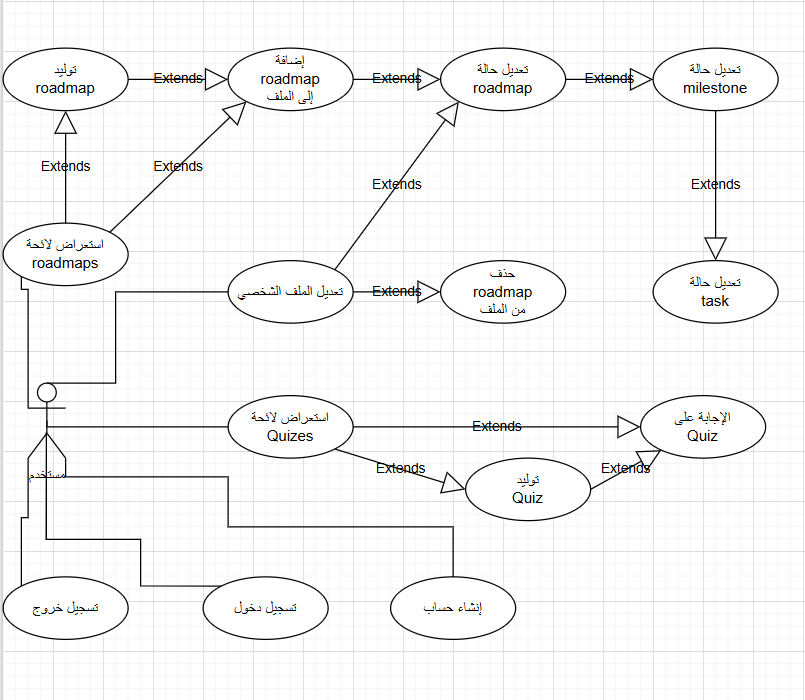
**قابليّة التوسّع (Scalability):**

* **NFR-9:** 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

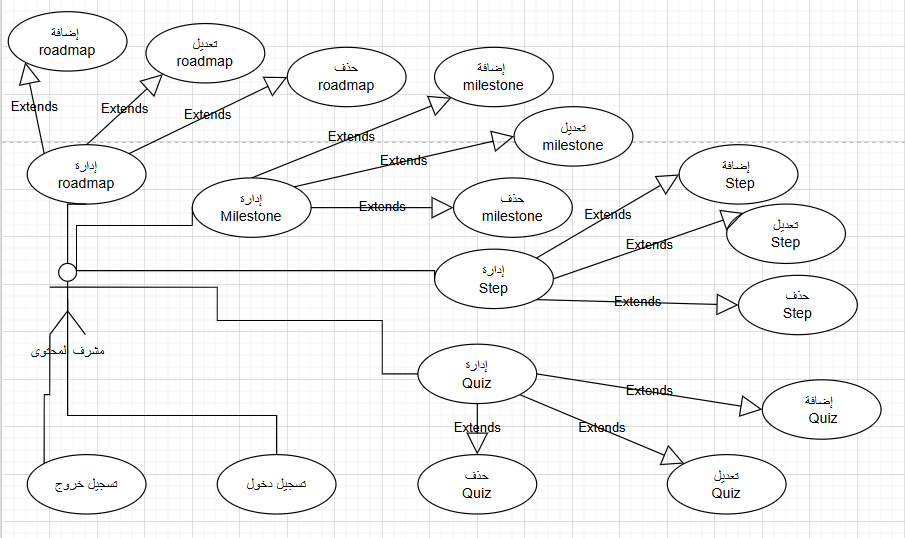
لتوضيح وظائف النظام بشكل مرئي، يمكن تمثيل التفاعلات الرئيسية بين المستخدمين (Actors) والنظام على النحو التالي:

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



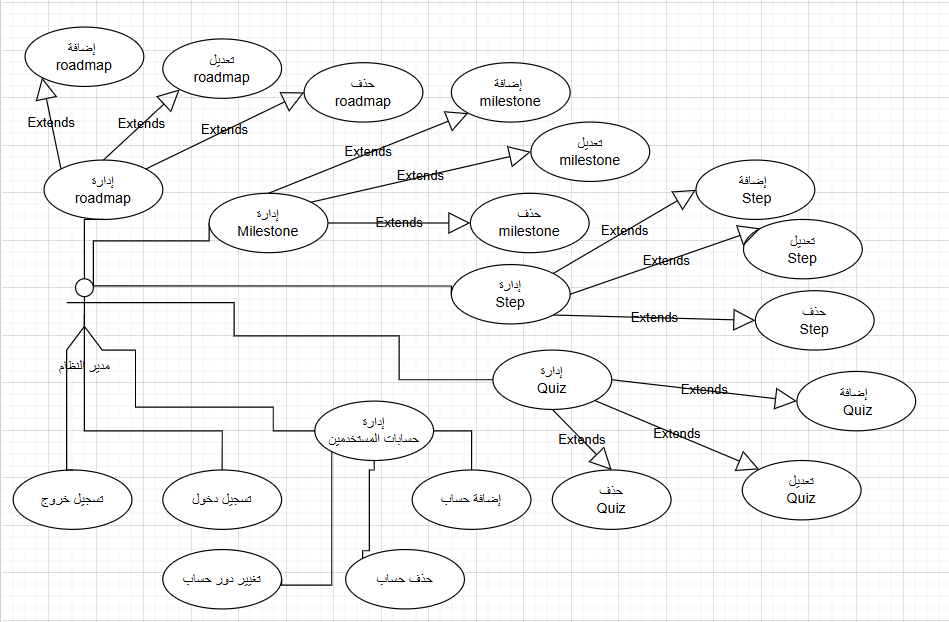
رسم توضيحي 1 حالات الاستخدام للمسخدم المتلقي

* 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 النصيّ لحالات الاستخدام ومخطّطات التسلسل

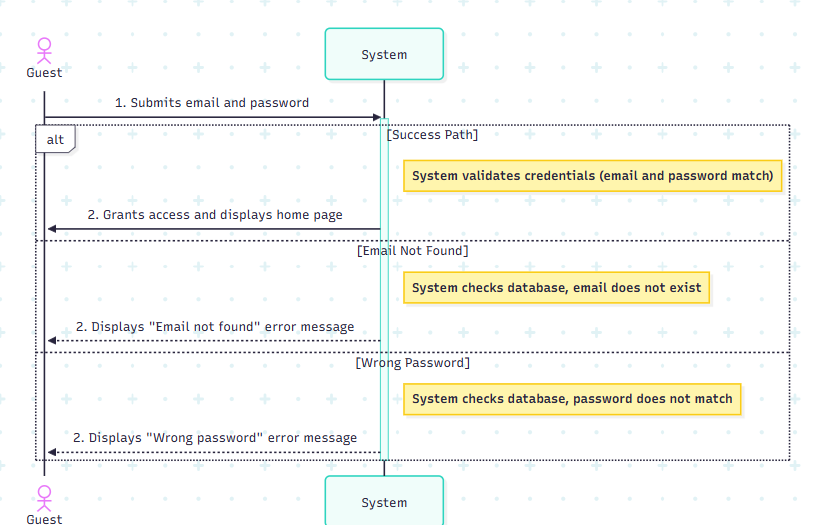
نستعرض الوصف النصي لجزء من حالات الاستخدام المبينة في الرسوم التوضيحية أعلاه.

1.3.3- حالة الاستخدام: تسجيل دخول

* **النمط:** أساسيّة.
* **الفاعلون:** مستخدم (أي دور).
* **الظروف السابقة:   
  -** يجب أن يكون المستخدم مُسجّل في النظام.
* **السيناريو الرئيسي الناجح:**

جدول 2 السيناريو الرئيسي لحالة تسجيل دخول

|  |  |
| --- | --- |
| **النظام** | **المستخدم** |
|  | 1- تبدأ هذه الحالة عندما يطلب المستخدم تسجيل الدخول إلى النظام |
| 2- يعيد النظام استمارة لإدخال البريد الالكتروني وكلمة السر |  |
|  | 3- يقوم المستخدم بإدخال البريد الالكتروني وكلمة السر |
| 4- يتحقق النظام من أن البريد الالكتروني موجود فعلاً. |  |
| 5- بعد التأكد من وجود البريد الالكتروني 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 التتالي لحالة الاستخدام:**

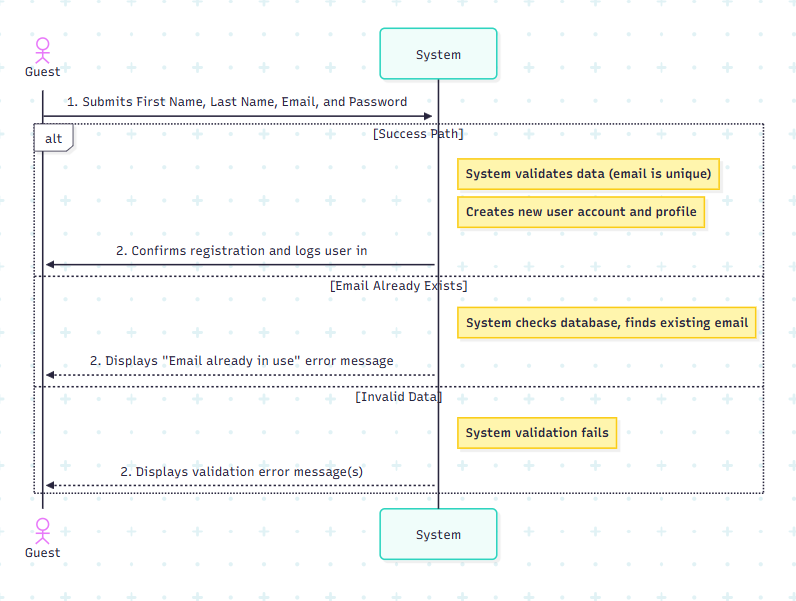
رسم توضيحي 4 مخطط التتالي لعملية تسجيل الدخول

2.3.3- حالة الاستخدام: إنشاء حساب جديد

* **النمط:** أساسيّة.
* **الفاعلون:** مستخدم متلقّي.
* **الظروف السابقة:**لا يوجد.
* **السيناريو الرئيسي الناجح:**

جدول 3 المسار الرئيسي لحالة إنشاء حساب جديد

|  |  |
| --- | --- |
| **النظام** | **المستخدم** |
|  | 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- يقوم النظام بتشفير كلمة السر. |  |
| 6- يقوم النظام بإضافة المستخدم إلى قاعدة البيانات. |  |
| 7- يقوم النظام بإعادة Token للمستخدم فيها بعض المعلومات الشخصية. |  |
| 8- 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:**لا يوجد.
* **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 التتالي لحالة الاستخدام:  
  **

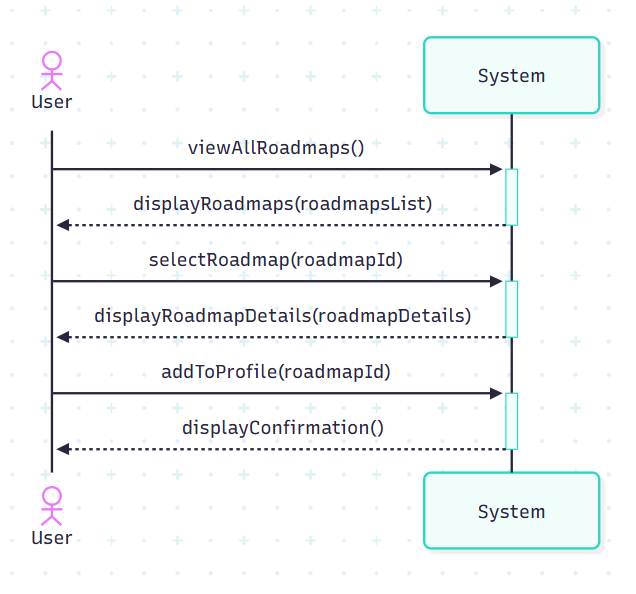
رسم توضيحي 5 مخطط التتالي لعملية إنشاء حساب جديد

3.3.3- حالة الاستخدام: إضافة Roadmap إلى الملف الشخصي

* **النمط:** أساسيّة.
* **الفاعلون:** مستخدم متلقّي.
* **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

|  |  |
| --- | --- |
| **النظام** | **المستخدم** |
|  | 1- يطلب المستخدم استعراض جميع خرائط الطريق. |
| 2- يعرض النظام صفحة تحتوي على قائمة بجميع خرائط الطريق المتاحة. |  |
|  | 3- يختار المستخدم خريطة طريق معينة وينقر عليها. |
| 4- يعرض النظام صفحة التفاصيل الكاملة لخريطة الطريق 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 . |  |

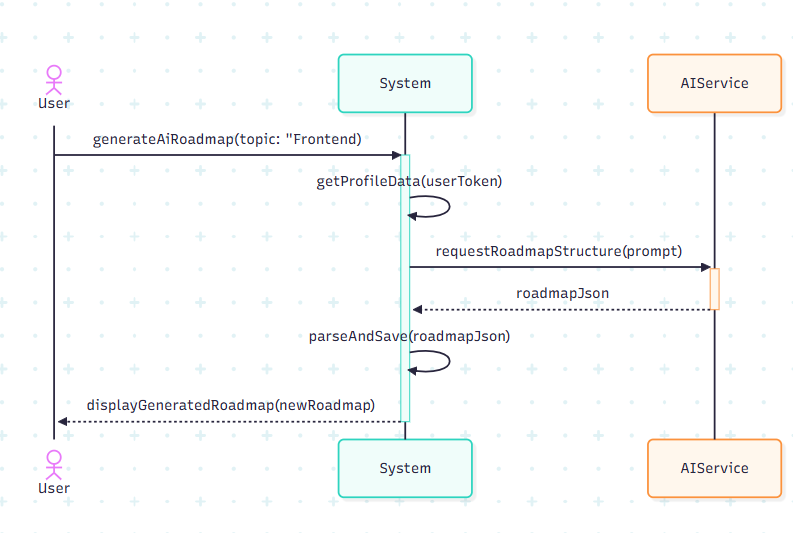
* **الظروف اللاحقة:**- تتم إضافة خريطة الطريق بنجاح إلى قائمة المتابعة في الملف الشخصي للمستخدم.
* **المسارات البديلة:**لا يوجد.
* **مسارات الأخطاء:**لا يوجد.
* **مخطط التتالي لحالة الاستخدام:**  
  

رسم توضيحي 6 مخطط التتالي لعملية إضافة roadmap إلى الملف الشخصي

4.3.3- حالة الاستخدام: توليد roadmap عبر الذكاء الاصطناعي

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

|  |  |
| --- | --- |
| **النظام** | **المستخدم** |
|  | 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 هيكلة 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 المعرّف الفريد Id الخاص به) كاستجابة ناجحة للمستخدم، مما ينهي حالة الاستخدام. |  |

* **الظروف اللاحقة:**- يتم إنشاء خارطة طريق جديدة وجميع مكوناتها (المراحل والخطوات) 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 استجابة خطأ "Internal Server Error (500)" تشير إلى وجود مشكلة في الاتصال بالخدمة الخارجية.  
    
  **E2 استجابة غير صالحة من الذكاء الاصطناعي.**يبدأ هذا المسار من المرحلة 7 من المسار الرئيسي.  
  8- تُرجع خدمة الذكاء الاصطناعي بيانات بتنسيق غير صالح (JSON غير مكتمل أو بنية غير متوقعة).  
  9- يفشل النظام في معالجة وتحليل الاستجابة.  
  10- يلتقط النظام الخطأ ويعيد استجابة خطأ "Internal Server Error (500)" تشير إلى وجود مشكلة في معالجة البيانات المستلمة.
* **مخطط التتالي لحالة الاستخدام توليد roadmap عبر الذكاء الاصطناعي:**  
  

رسم توضيحي 7 مخطط التتالي 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 بتقنيةReact ، والتي تعمل بالكامل على متصفح المستخدم. هذا الجزء مسؤول عن كل ما يراه المستخدم ويتفاعل معه.
* **الخادم (Server):** وهو الواجهة الخلفية المبنية بتقنيةASP.NET Core، والتي تعمل على خادم مركزي. هذا الجزء مسؤول عن تخزين البيانات، تنفيذ منطق العمل، وتأمين النظام .

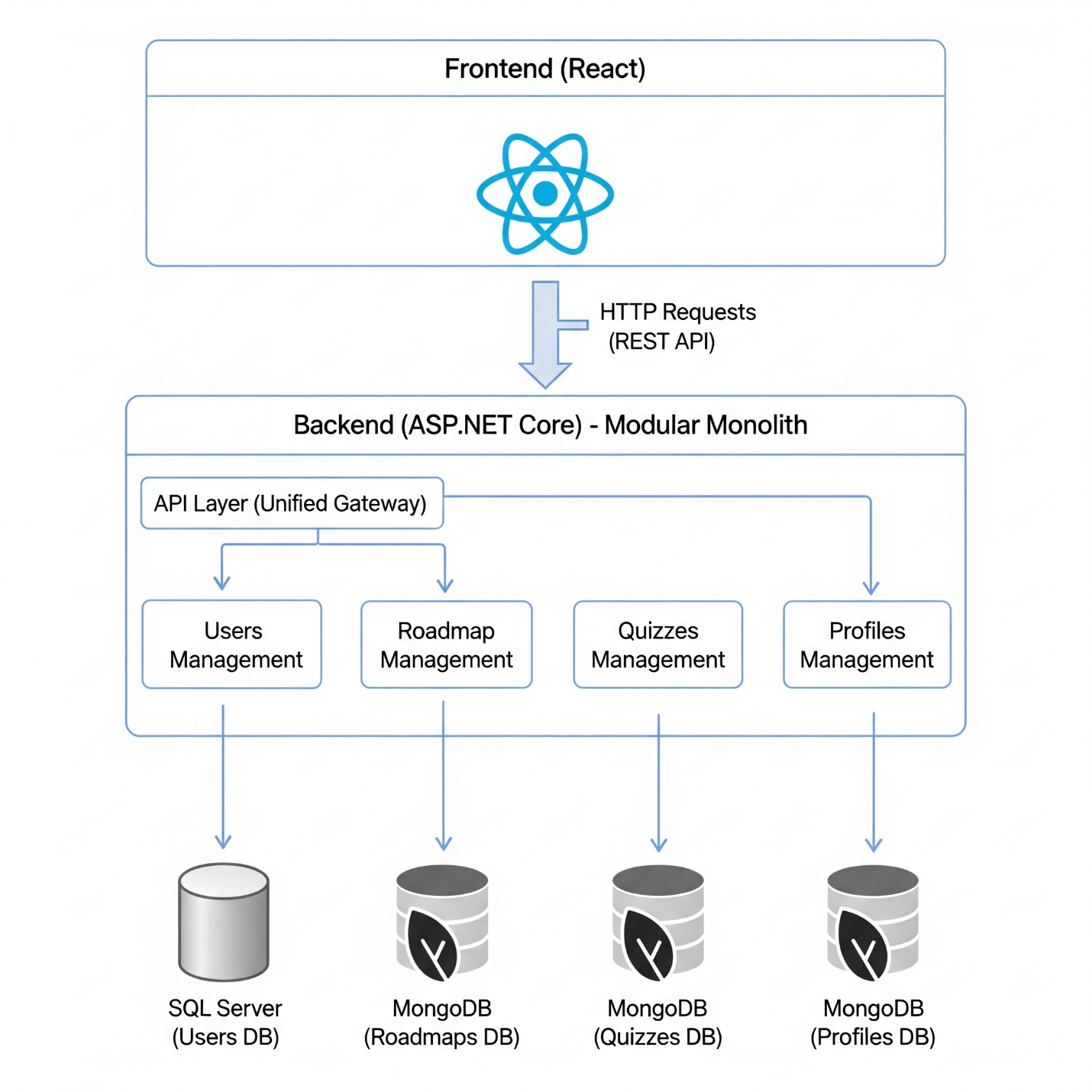
يتواصل العميل مع الخادم عبر شبكة الإنترنت باستخدام بروتوكول HTTP، حيث يرسل العميل طلبات (Requests) ويستقبل الخادم هذه الطلبات ويعيد استجابات (Responses) تحتوي على البيانات المطلوبة.

**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 الخرائط ومكوناتها (المراحل والخطوات).
* **وحدة إدارة الاختبارات (Quizzes Management):** مسؤولة عن إنشاء وإدارة الاختبارات والأسئلة.
* **وحدة الملفات الشخصية (Profiles Management):** مسؤولة عن إدارة الملفات الشخصية للمستخدمين.

تتشارك هذه الوحدات في بعض المكونات الأساسية (مثل قواعد البيانات) ولكنها تبقى منفصلة في الكود ، مما يسهل فهمها وتطويرها. ويتم كشف وظائفها للعالم الخارجي عبر واجهة برمجة تطبيقات (API) موحدة، تعمل كبوابة وحيدة للنظام بأكمله.  
  


رسم توضيحي 8 هيكلية المشروع

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 فروع (Branches) لتجربة ميزات جديدة دون التأثير على النسخة الرئيسية من الكود، ودمج التغييرات من عدة مطورين بسهولة.
* **GitHub:** هي منصة استضافة سحابية لمشاريعGit . تم استخدامها كمستودع مركزي (Central Repository) للأكواد للمشروع (الواجهة الأمامية والخلفية)، مما سهل عملية التعاون وتوزيع العمل ومراجعة الكود.

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 (Object-Relational Mapper - 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.

2.3.4- الواجهة الأمامية (Frontend)

* **React**: هي مكتبة JavaScript لبناء واجهات المستخدم. تعتمد على مفهوم "المكونات" (Components)، حيث يتم بناء الواجهة كـمجموعة من القطع المستقلة والقابلة لإعادة الاستخدام، مما يجعل إدارة الواجهات المعقدة أكثر سهولة**.**
* **React Router:** هي مكتبة للتوجيه (Routing) داخل تطبيقاتReact . تسمح بالتنقل بين الصفحات المختلفة دون الحاجة لإعادة تحميل الصفحة بأكملها، مما يوفر تجربة مستخدم سريعة وسلسة تشبه تطبيقات سطح المكتب.
* **Axios:** هي مكتبة JavaScript لإجراء طلباتHTTP . 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 (معالج CSS) 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 6 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. |
| قابلية التوسع | تتوسع عمودياً ( بزيادة موارد الخادم). | تتوسع أفقياً (بإضافة المزيد من الخوادم). |
| 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 (خرائط الطريق، المراحل، الخطوات، والاختبارات). 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 طبقة تجريدية (Abstraction Layer) بين منطق العمل وطبقة الوصول إلى البيانات. لكل نموذج بيانات (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 خريطة طريق جديدة مع مراحلها، 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 المرسلة.
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 كلاس 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 صفين يطبّقان هذه الواجهة:

ManualRoadmapCreationStrategy: يحتوي على منطق إنشاء خريطة الطريق بالطريقة التقليدية.

AiRoadmapCreationStrategy: يحتوي على منطق استدعاء GeminiAiAdapter لتوليد خريطة الطريق.

الفائدة: هذا الفصل سمح بإضافة طريقة التوليد عبر الذكاء الاصطناعي دون تعديل الشيفرة 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 كلاس رئيسي، لكنه يسمح للكلاسات 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 كلاس GeminiAiAdapter الذي يطبق هذه الواجهة. داخلياً، يقوم هذا الكلاس 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- إعداد المشروع الرئيسي (SkillCraft.Api)

يمثل هذا المشروع نقطة الدخول الوحيدة للنظام بأكمله. في ملف Program.cs، تم إعداد وإعداد جميع الخدمات والوسائط اللازمة (Middleware):

* **تسجيل الخدمات (Services Registration):** تم تسجيل جميع الخدمات من طبقات منطق العمل (BLL) والمستودعات (Repositories) من طبقات الوصول إلى البيانات (DAL) باستخدام آلية حقن التبعية (Dependency Injection).
* **إعداد قواعد البيانات:** تم تكوين الاتصال مع قاعدتي البيانات SQL Server و MongoDB.
* **إعداد المصادقة والتفويض:** تم تفعيل المصادقة باستخدام JWT Bearer Tokens وتحديد سياسات التفويض (Authorization) القائمة على الأدوار.
* **إعداد CORS (Cross-Origin Resource Sharing):** للسماح للواجهة الأمامية (React)بالتواصل مع الـ API.

2.1.5- تنفيذ وحدة إدارة المستخدمين (Users Management Module)

هذه الوحدة مسؤولة عن كل ما يتعلق بالمستخدمين والأدوار.

* **المصادقة (AuthController):** يحتوي هذا المتحكم على نقاط النهاية (Endpoints) الخاصة بعمليات التسجيل (Register) وتسجيل الدخول (Login). عند تسجيل الدخول بنجاح، تقوم خدمة AuthService بإنشاء توكن JWT وإعادته للمستخدم كما هو موضح في الرسم التوضيحي 4.
* **إدارة المستخدمين (UsersController):** يوفر هذا المتحكم عمليات CRUD (Create, Read, Update, Delete) لإدارة المستخدمين، وهي محمية بحيث لا يمكن الوصول إليها إلا من قبل مدير النظام (Admin).

3.1.5- تنفيذ وحدة إدارة خرائط الطريق (Roadmap Management Module)

هذه هي الوحدة الأساسية في النظام.

* **المتحكمات (RoadmapsController, MilestonesController, StepsController):** توفر هذه المتحكمات نقاط النهاية اللازمة لإدارة المحتوى التعليمي.
* **طبقة الوصول للبيانات (DAL):** تم إنشاء مستودعات (Repositories) مثل RoadmapRepository للتفاعل مع قاعدة بيانات MongoDB.

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

* **إعداد Axios:** في ملف api/apiClient.js، تم إنشاء نسخة من Axios مع استخدام Interceptors لإرفاق توكن المصادقة تلقائياً مع كل طلب.

2.2.5- إدارة الحالة والتوجيه (State Management & Routing)

* إدارة حالة المصادقة **(AuthContext)**: تم استخدام Context API من React لتخزين معلومات المستخدم المسجل دخوله ومشاركتها مع جميع المكونات.
* المسارات المحمية **(PrivateRoute.jsx)**: تم إنشاء مكون مخصص لحماية المسارات التي تتطلب صلاحيات معينة.

3.2.5- تنفيذ الواجهات الرئيسية

* عرض خريطة الطريق **(RoadmapPage.jsx):** تقوم هذه الصفحة بجلب تفاصيل خريطة طريق محددة وعرضها على شكل خط زمني عمودي.
* لوحات التحكم **(Dashboards):** تم إنشاء لوحات تحكم خاصة لكل دور (Admin, Editor) لإدارة المحتوى والمستخدمين.

3.5- تنفيذ قواعد البيانات

* **SQL Server:** تم استخدام EF Core Migrations لإنشاء وتحديث جداول قاعدة البيانات (Users,Roles) بشكل برمجي.
* **MongoDB:** تم تكوين سلاسل الاتصال في ملف appsettings.json، وتقوم المكتبة الرسمية بإنشاء قاعدة البيانات والمجموعات (Collections) تلقائيا.

الفصل السادس

الاختبار والنتائج