

Faculty of Engineering

Department of Informatics Engineering

Software Engineering and Information Systems



Integrated Social Media Platform (NOONIFY)

A junior project report - submitted to complete the requirements for obtaining a bachelor's degree in informatics engineering – Software Engineering and Information Systems

Prepared by

Sara Eyad Attah

Aya Al-salamat

Supervised by

Dr. Eng. Akram Massouh

January 2026

منصة تواصل اجتماعي متكامل

تقرير مشروع فصلي قدّم استكمالاً لمتطلبات الحصول على درجة
البكالوريوس في هندسة المعلوماتية - هندسة البرمجيات ونظم المعلومات

إعداد

ساره اياد عطا

آيه هيثم السلامة

اشراف

الدكتور. أكرم مسوح

SUPERVISION CERTIFICATION

I Certify that the preparation of this project entitled

[**Integrated Social Media Platform (NOONIFY)**]

Prepared by

[Sara Eyad Atta - Aya Alsalamat]

was made under my supervision at Faculty of Informatics Engineering in partial Fulfillment of the
Requirements for the Degree of Bachelor of Software Engineering and Information System

Name:

Signature:

Date:

الملخص

يهدف هذا المشروع إلى تصميم منصة تواصل اجتماعي متكاملة تتيح للمستخدمين إنشاء حساباتهم، مشاركة المحتوى النصي والمرئي، والتفاعل مع الآخرين بطريقة آمنة وسلسة، يعتمد النظام على MERN Stack (MongoDB, Express.js, React.js, Node.js) لتغطية الجوانب الأمامية والخلفية، إضافة إلى أدوات داعمة مثل Clerk لإدارة الحسابات وتوثيق المستخدمين، وImageKit لمعالجة وتحسين الصور، وInngest لإدارة المهام الخلفية المجدولة.

توفّر المنصة مجموعة واسعة من الوظائف تشمل التسجيل والدخول (Sign Up / Sign In)، إدارة الملف الشخصي (Profile Management)، إنشاء المنشورات (Posts) والقصص (Stories)، التفاعل الاجتماعي عبر الإعجابات والتعليقات والمتابعة (Follow / Unfollow)، بالإضافة إلى نظام مراسلة فورية (Real-time Chat).

كما تتيح المنصة البحث والاكتشاف (Search & Discover)، الإشعارات اللحظية (Real-time Notifications)، وإدارة الوسائط باستخدام حلول تخزين سحابية فعّالة.

من الناحية التقنية، يتميز النظام بالمرونة العالية (Scalability) بفضل بنية Node.js وMongoDB، مع واجهة مستخدم تفاعلية وسهلة الاستخدام مبنية بـ React.js. كما تم الاهتمام بعوامل الأمان (Security) باستخدام HTTPS، وضمان الاعتمادية (Reliability) وقابلية الصيانة (Maintainability) من خلال تصميم مكونات واضحة ومنفصلة.

بشكل عام، يقدم المشروع نموذجًا عمليًا متكاملًا لتطبيق تواصل اجتماعي حديث، يربط بين تجربة المستخدم الممتعة والأداء القوي المدعوم بأحدث تقنيات Full Stack Development.

Abstract

This project aims to develop a comprehensive social media platform that enables users to create accounts, share text and media content, and interact with others in a secure and seamless environment. The system is built using the MERN Stack (MongoDB, Express.js, React.js, Node.js), along with supporting tools such as Clerk for authentication and account management, ImageKit for media optimization, and Inngest for background task scheduling.

The platform provides a wide range of functionalities including Sign Up / Sign In, Profile Management, creating and managing Posts and Stories, engaging with others through Likes, Comments, and Follow/Unfollow, and real-time messaging (Real-time Chat). It also includes Search & Discover, Real-time Notifications, and efficient Media Management using cloud storage services.

Technically, the system ensures Scalability with Node.js and MongoDB, offers a highly interactive user interface built with React.js, and maintains strong Security standards via HTTPS. It emphasizes Reliability and Maintainability through modular code structure and clear component separation.

Overall, the project presents a fully functional and modern social networking application that combines an enjoyable user experience with robust technical performance, showcasing practical Full Stack Development skills

Table of Contents

٤ الملخص	
٥ Abstract	
١ ١. الفصل الأول: المقدمة	
٢ أهمية المشروع	١,١
٣ الهدف من المشروع	٢,١
٤ الدراسات المرجعية لمنصة التواصل الاجتماعي	٣,١
٤ الدراسة الأولى	١,٣,١
٥ الدراسة الثانية	٢,٣,١
٦ جدول مقارنة بين الدراسات المرجعية	3.3.1
٧ الجدول الزمني Gantt Chart	٤,١
٧ المنهجية المتبعة في تنفيذ المشروع (منهجية SCRUM)	٥,١
٨ تطبيق منهجية SCRUM في مشروع منصة التواصل الاجتماعي	1.5.1
٩ المعمارية (System Architecture)	٦,١
٩ أولاً: معمارية Client–Server	١,٦,١
١٠ مزايا معمارية Client–Server	١,١,٦,١
١٠ سلبيات معمارية Client–Server	2.1.6.1
١١ ثانياً: معمارية MVC (Model – View – Controller)	٢,٦,١
١٢ إيجابيات معمارية MVC	1.2.6.1
١٢ سلبيات معمارية MVC	2.2.6.1
١٣ الربط بين Client–Server و MVC في المشروع	3.2.6.1
١٤ ٢. الفصل الثاني : الدراسة التحليلية	
١٥ المتطلبات العامة (General Requirements)	١,٢
١٦ المتطلبات الوظيفية (Functional Requirements)	٢,٢
١٦ نظام التسجيل والدخول (Authentication System)	١,٢,٢
١٧ إدارة الملف الشخصي (User Profile Management)	٢,٢,٢
١٧ إدارة المنشورات (Posts Management)	3.2.2
١٨ إدارة القصص (Stories / Status System)	4.2.2
١٨ نظام المراسلة (Real-time Chat & Messages)	5.2.2

١٨	نظام العلاقات الاجتماعية (Connections System)	6.2.2
١٩	نظام البحث والاكتشاف (Search & Discover System)	7.2.2
١٩	نظام الإشعارات (Notifications System)	8.2.2
١٩	إدارة الوسائط (Media Management)	9.2.2
٢٠	النظام الخلفي (Backend Operations)	10.2.2
٢٠	المتطلبات غير الوظيفية (Non-Functional Requirements)	٣,٢
٢٠	المتطلبات التي سيتم تنفيذها في المشروع الفصلي	٤,٢
٢٣	المخطط المعماري (Architecture Diagram)	٥,٢
٢٥	مخطط ال Use Case Diagram	6.2
٢٦	توصيف حالات الاستخدام Use Case Specifications	٧,٢
٢٦	مخططات النشاط Activity Diagrams	٨,٢
٢٦	مخططات التسلسل Sequence Diagrams	9.2
٦١	٣. الفصل الثالث : الدراسة التصميمية	
٦٢	مخطط قاعدة البيانات ERD	١,٣
٦٣	مخطط الصفوف Class Diagram	٢,٣
٦٣	جدول Test Cases	٣,٣
٦٥	جدول RTM (Requirements Traceability Matrix)	4.3
٦٦	مخطط ال Site Map	٥,٣
٦٨	٤. الفصل الرابع : تنفيذ المشروع	
٦٩	الأدوات المستخدمة	١,٤
٦٩	أدوات الواجهة الأمامية (Front-End Tools)	1.1.4
٧٠	أدوات الواجهة الخلفية (Back-End Tools)	2.1.4
٧١	أدوات قاعدة البيانات (Database Tools)	3.1.4
٧٢	ملخص الأدوات المستخدمة	4.1.4
٧٢	الواجهات	٢,٤
٧٨	الخاتمة	3.4
٧٩	الآفاق المستقبلية (Future Work)	4.4

List of Tables

٦	جدول ١ مقارنة بين الدراسات المرجعية
٢٧	جدول 2 توصيف حالة تسجيل الحساب
٢٩	جدول 3 توصيف حالة تسجيل الدخول
٣١	جدول 4 توصيف حالة الدخول الآمن
٣٢	جدول 5 توصيف حالة تسجيل الخروج
٣٣	جدول 6 توصيف حالة عرض الواجهة الأساسية
٣٥	جدول 7 توصيف حالة التعديل على الحساب
٣٧	جدول 8 توصيف حالة عرض معلومات الحسابات
٣٩	جدول 9 توصيف حالة البحث عن الأصدقاء
٤٢	جدول 10 توصيف حالة نشر منشور جديد
٤٤	جدول ١١ توصيف حالة عرض المنشورات
٤٦	جدول ١٢ توصيف حالة الإعجاب بالمنشورات
٤٨	جدول ١٣ توصيف حالة التعليق على المنشورات
٤٩	جدول ١٤ توصيف حالة مشاركة المنشورات
٥١	جدول ١٥ توصيف حالة إضافة صديق
٥٣	جدول 16 توصيف معرفة حالة طلب الصداقة
٥٥	جدول 17 توصيف حالة إضافة قصة جديدة
٥٧	جدول 18 توصيف عرض القصص المتاحة للمتابعين
٥٩	جدول 19 توصيف عرض قائمة المشاهدين
٦٥	جدول ٢٠ ال Test Cases
٦٦	جدول 21 ال RTM

List of Figuers

٢٣	١ مخطط دمج المعماريين
٢٥	٢ مخطط ال Class Diagram
٢٧	3 مخطط النشاط حالة تسجيل الحساب
٢٨	4 مخطط التسلسل حالة تسجيل الحساب
٢٩	5 مخطط النشاط حالة تسجيل الدخول
٣٠	6 مخطط التسلسل حالة تسجيل الدخول
٣٢	7 مخطط التسلسل حالة تسجيل الخروج
٣٣	8 مخطط النشاط حالة تسجيل الخروج
٣٤	9 مخطط التسلسل حالة عرض الواجهة الأساسية
٣٤	10 مخطط النشاط حالة عرض الواجهة الأساسية
٣٦	11 مخطط التسلسل حالة التعديل على الحساب

٣٦	12 مخطط النشاط حالة التعديل على الحساب
٣٨	13 مخطط النشاط حالة عرض معلومات الحسابات
٣٨	14 مخطط النشاط حالة عرض معلومات الحسابات
٤٠	١٥ مخطط النشاط حالة البحث عن الأصدقاء
٤٠	١٦ مخطط التسلسل حالة البحث عن الأصدقاء
٤٢	17 مخطط التسلسل حالة نشر منشور جديد
٤٣	١٨ مخطط النشاط حالة نشر منشور جديد
٤٤	١٩ مخطط النشاط حالة عرض المنشورات
٤٥	٢٠ مخطط التسلسل حالة عرض المنشورات
٤٦	٢١ مخطط التسلسل حالة الإعجاب بالمنشورات
٤٧	٢٢ مخطط النشاط حالة الإعجاب بالمنشورات
٤٨	٢٣ مخطط النشاط حالة التعليق على المنشورات
٤٨	٢٤ مخطط التسلسل حالة التعليق على المنشورات
٤٩	٢٥ مخطط النشاط حالة مشاركة المنشورات
٥٠	٢٦ مخطط التسلسل حالة مشاركة المنشورات
٥١	٢٧ مخطط التسلسل حالة إضافة صديق
٥٢	٢٨ مخطط النشاط حالة إضافة صديق
٥٣	٢٩ مخطط التسلسل معرفة حالة طلب الصداقة
٥٤	٣٠ مخطط النشاط معرفة حالة طلب الصداقة
٥٥	٣١ مخطط التسلسل حالة إضافة قصة جديدة
٥٦	٣٢ مخطط النشاط حالة إضافة قصة جديدة
٥٧	٣٣ مخطط النشاط حالة عرض القصص المتاحة للمتابعين
٥٨	٣٤ مخطط التسلسل حالة عرض القصص المتاحة للمتابعين
٦٠	٣٥ مخطط النشاط عرض قائمة المشاهدين
٦٠	٣٦ مخطط النشاط عرض قائمة المشاهدين
٦٢	٣٧ مخطط ال ERD
٦٣	٣٨ مخطط ال Class Diagram
٦٧	٣٩ مخطط ال Site Map
٧٢	٤٠ الشكل (١,١) واجهة تسجيل حساب جديد
٧٣	٤١ الشكل (١,٢) واجهة تسجيل الدخول
٧٤	42 الشكل (١,٣) زر تسجيل الخروج
٧٤	٤٣ الشكل (١,٤) عرض المعلومات الأساسية للمستخدم
٧٥	٤٤ الشكل (١,٥) رفع قصة جديدة
٧٥	٤٥ الشكل (١,٦) إنشاء منشورات جديد
٧٦	46 الشكل (١,٧) عرض ملفات المستخدمين الآخرين
٧٦	٤٧ الشكل (١,٨) تعديل الملف الشخصي
٧٧	٤٨ الشكل (١,٩) عرض القصص
٧٨	٤٩ الشكل (١,١٠) عرض منشورات الآخرين

١ . الفصل الأول : المقدمة

١.١ أهمية المشروع

تتبع أهمية هذا المشروع من الحاجة المتزايدة إلى منصات تواصل اجتماعي حديثة وآمنة تتيح للمستخدمين التفاعل ومشاركة المحتوى بسهولة وكفاءة، ومع التطور السريع في تقنيات الويب وازدياد الاعتماد على التطبيقات الرقمية في التواصل اليومي، أصبح من الضروري تطوير نظام تواصل اجتماعي متكامل يجمع بين الأداء العالي وتجربة المستخدم المميزة.

يساهم المشروع في تعزيز مهارات المطورين في مجال Full Stack Development من خلال تطبيق عملي يعتمد على تقنيات (MongoDB, Express.js, React.js, Node.js) MERN Stack، والتي تمكّن من بناء نظام متكامل يغطي جميع الجوانب: Front-End لتصميم واجهة تفاعلية، و Back-End لمعالجة الطلبات وإدارة البيانات، و Database لتخزين المعلومات بطريقة فعّالة وآمنة.

تكمّن أهمية المشروع أيضًا في كونه يدمج مفاهيم وتقنيات حديثة مثل Real-time Communication، وإدارة الوسائط باستخدام ImageKit، ونظام توثيق المستخدمين الآمن بواسطة Clerk، بالإضافة إلى إدارة المهام الخلفية باستخدام Inngest.

وبذلك، لا يقتصر المشروع على كونه تجربة تعليمية فحسب، بل يشكّل نموذجًا متكاملًا يمكن تطويره مستقبلاً ليصبح منصة تواصل اجتماعي حقيقية تتميز بالأمان، والاعتمادية، وسهولة الاستخدام، وتواكب التطورات التقنية الحديثة في عالم الويب.

٢.١ الهدف من المشروع

يهدف هذا المشروع إلى تطوير منصة تواصل اجتماعي متكاملة تتيح للمستخدمين إنشاء حساباتهم، إدارة ملفاتهم الشخصية، ومشاركة المحتوى النصي والمرئي بطريقة آمنة وسلسة. كما يسعى إلى توفير بيئة تفاعلية تجمع بين التصميم الجذاب، الأداء العالي، وسهولة الاستخدام عبر مختلف الأجهزة والمنصات.

يتمثل الهدف الأساسي في بناء تطبيق ويب يعتمد على MERN Stack (MongoDB, Express.js, React.js, Node.js) لتغطية جميع طبقات النظام من الواجهة الأمامية (Front-End) وحتى النظام الخلفي (Back-End) وقاعدة البيانات (Database). كما يهدف المشروع إلى تعزيز الجوانب التقنية من خلال دمج أدوات وخدمات حديثة مثل Clerk لإدارة المستخدمين وتوثيق الحسابات، وImageKit لمعالجة الصور وتحسين أدائها، وInngest لتنفيذ المهام الخلفية المجدولة مثل الإشعارات أو حذف البيانات المؤقتة. ومن الأهداف المهمة كذلك، تطبيق مفاهيم Real-time Features مثل المراسلة الفورية (Real-time Chat) والإشعارات اللحظية (Real-time Notifications) لتوفير تجربة استخدام ديناميكية وتفاعلية. إضافة إلى ذلك، يسعى المشروع إلى بناء واجهة استخدام متجاوبة (Responsive UI) باستخدام React.js و CSS لتتيح سهولة التنقل وتفاعل المستخدمين بسلاسة.

باختصار، يهدف المشروع إلى تقديم نموذج تطبيقي متكامل يربط بين الجانب الأكاديمي والعملي، ويُمكن المطورين من تطبيق مهاراتهم في بناء Social Media Platform احترافية تُبرز المفاهيم الأساسية لتقنيات الويب الحديثة وأفضل ممارسات التطوير المتكامل.

٣.١ الدراسات المرجعية لمنصة التواصل الاجتماعي

١.٣.١ الدراسة الأولى

بنية بيانات وتخزين الشبكة الاجتماعية

"TAO: Facebook's Distributed Data Store for the Social Graph"

تتناول هذه الدراسة النظام TAO الذي طوّره شركة Facebook لإدارة العلاقات بين المستخدمين (Social Graph) بكفاءة عالية، يهدف النظام إلى تحسين سرعة الاستعلام والوصول إلى البيانات من خلال Data Store مخصص ومُحسّن للقراءة، مع واجهة API موحّدة تسهّل عمليات الاستعلام دون الحاجة لفهم البنية الداخلية للبيانات ويُعتبر TAO نموذجًا لتصميم قاعدة بيانات مخصّصة للتطبيقات الاجتماعية ذات الحجم الكبير، حيث يجمع بين MySQL و Caching Layer لتقليل زمن الاستجابة وتحسين الأداء.

أبرز الدروس المستفادة:

- تصميم طبقة بيانات متخصصة لتتناسب طبيعة التطبيقات الاجتماعية.
- استخدام Caching و API Layer يقلّل من الحمل على قواعد البيانات.
- التوزيع الجغرافي للبيانات يُحسّن من سرعة الوصول للمستخدمين حول العالم.

المصدر: Nathan Bronson et al., USENIX ATC, 2013 – Facebook Research.

بنية الأنظمة وتحجيم منصات التراسل/التواصل

"Scaling big data mining infrastructure: the Twitter experience"

تستعرض هذه الدراسة تجربة Twitter في بناء بنية تحتية قادرة على التعامل مع مليارات الرسائل يوميًا مع الحفاظ على الأداء اللحظي (Real-time). استخدمت Twitter تقنيات مثل Kafka, HBase, Distributed Storage Systems لمعالجة التدفق المستمر من البيانات وتحليلها بسرعة. تركز التجربة على أهمية بناء نظام مرن (Scalable) يسمح بتحديث المكونات دون توقف الخدمة، وتحقيق توازن بين Latency و Throughput لتقديم تجربة تفاعلية مستقرة. أبرز الدروس المستفادة:

- اعتماد تقنيات Streaming و Message Queue مثل Kafka ضروري للميزات اللحظية.
- بناء بنية مرنة قابلة للتوسع والصيانة دون انقطاع.
- أهمية أدوات المراقبة (Monitoring) والـ Logging في تحليل الأداء.

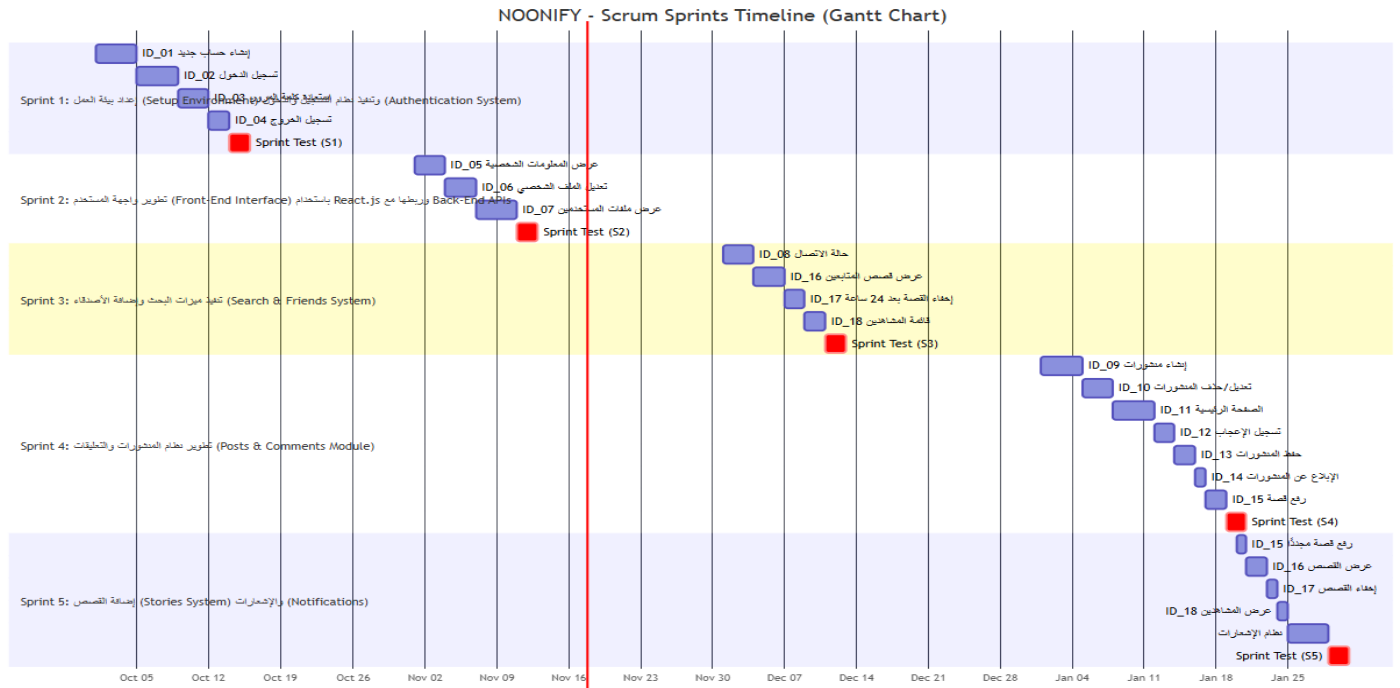
المصدر: Twitter Engineering Reports, Scaling Big Data Systems.

٣.٣.١ جدول مقارنة بين الدراسات المرجعية.

الدراسة	TAO–Facebook	Twitter	مشروعنا NOONIFY
التقنيات المستخدمة	Data Store Caching Layer,API	Kafka,HBase,Distributed Storage.	MERN Stack
الإيجابيات	أداء عالٍ واستجابة سريعة للبيانات	معالجة بيانات ضخمة في الزمن الحقيقي	واجهات حديثة , أداء فعال تكاملاً مكونات جاهزة.
السلبيات	تعقيد البنية وصعوبة الصيانة	تكلفة تشغيل عالية وتعقيد إداري	اعتماد جزئي على خدمات خارجية.
التحديات	الحفاظ على اتساق البيانات الموزعة	التوازن بين السرعة والتوافر	ضمان الأمان والخصوصية مع الأداء العالي.
أوجه التميز	تخصيص النظام لمعالجة العلاقات الاجتماعية	بنية مرنة تدعم التحليل اللحظي	دمج تقنيات حديثة بمرونة عالية وسرعة تطوير.

جدول ١ مقارنة بين الدراسات المرجعية

٤.١ الجدول الزمني Gantt Chart



مخطط ال Gantt Chart

٥.١ المنهجية المتبعة في تنفيذ المشروع (منهجية SCRUM)

تم اعتماد منهجية SCRUM كإطار عمل (Agile Framework) لتنفيذ هذا المشروع، نظرًا لمرونتها العالية وقدرتها على تنظيم عملية التطوير بطريقة تفاعلية وتدرجية (Iterative and Incremental Process)، تُعد منهجية SCRUM واحدة من أكثر منهجيات Agile Software Development استخدامًا في المشاريع البرمجية الحديثة، حيث تركز على تقسيم العمل إلى مراحل قصيرة تُعرف باسم Sprints، يتم خلالها تنفيذ مجموعة محددة من المهام ذات الأولوية العالية، ومراجعتها بشكل مستمر لتحسين جودة المنتج النهائي.

١.٥.١ تطبيق منهجية SCRUM في مشروع منصة التواصل الاجتماعي

تم تقسيم عملية التطوير إلى عدة Sprints، وكل Sprint يركّز على مجموعة محددة من الوظائف الأساسية ضمن المنصة.

والوظائف هي:

Sprint 1: إعداد بيئة العمل (Setup Environment) وتنفيذ نظام التسجيل والدخول (Authentication System).

Sprint 2: تطوير واجهة المستخدم (Front-End Interface) باستخدام React.js وربطها مع Back-End APIs.

Sprint 3: تنفيذ ميزات البحث وإضافة الأصدقاء (Search & Friends System).

Sprint 4: تطوير نظام المنشورات والتعليقات (Posts & Comments Module).

Sprint 5: إضافة القصص (Stories System) والإشعارات (Notifications).

بعد كل Sprint، يتم مراجعة (Sprint Review Meeting) لعرض التقدم وتقييم ما تم إنجازه، يعقبه

اجتماع تخطيطي (Sprint Planning Meeting) لتحديد مهام الدورة التالية.

٦.١ المعمارية (System Architecture)

١.٦.١ أولاً: معمارية Client-Server

تم اعتماد معمارية Client-Server كأساس لبناء المنصة، وهي من أكثر المعماريات استخدامًا في تطبيقات الويب الحديثة، خصوصًا تلك التي تعتمد على الاتصال بين واجهة المستخدم (Client) والخادم

(Server) عبر بروتوكول HTTP/HTTPS.

في هذا المشروع، يمثل الـ Client واجهة المستخدم المبنية باستخدام React.js، بينما يمثل الـ Server

الخادم الخلفي المطور باستخدام Node.js و Express.js والمتصل بقاعدة البيانات MongoDB.

يتم تبادل البيانات بين الطرفين بصيغة JSON عبر واجهات RESTful APIs.

آلية العمل:

١. يقوم المستخدم (Client) بإرسال طلب (Request) عبر المتصفح مثل تسجيل الدخول أو عرض

المنشورات.

٢. يستقبل الخادم (Server) الطلب من خلال Express.js Controllers ويعالجه وفقًا للمنطق البرمجي.

٣. يتصل الخادم بقاعدة البيانات MongoDB لاسترجاع أو تعديل البيانات المطلوبة.

٤. يعيد الخادم استجابة (Response) إلى Client، تُعرض النتيجة عبر الواجهة باستخدام React.js.

١.١.٦.١ مزايا معمارية Client-Server:

- فصل واضح بين الواجهة والتطبيق الخلفي (Separation of Concerns) مما يسهل التطوير والتحديث.
- أمان أعلى لأن معالجة البيانات الأساسية تتم في الخادم وليس في واجهة المستخدم.
- قابلية التوسع (Scalability) بإمكانية توزيع الخوادم أو تحديث الواجهة بشكل مستقل.
- إعادة الاستخدام (Reusability) للخدمات الخلفية عبر واجهات API يمكن استدعاؤها من تطبيقات أخرى مستقبلاً (مثل تطبيق موبايل).

٢.١.٦.١ سلبات معمارية Client-Server:

- اعتماد الواجهة على توفر الخادم بشكل دائم، مما يجعل انقطاعه يوقف النظام.
- زيادة الحمل على الخادم مع زيادة عدد المستخدمين، مما يتطلب تحسين الأداء أو توزيع الأحمال.
- الحاجة إلى إدارة أمان الاتصال بشكل دقيق لتجنب الثغرات.

٢.٦.١ ثانيًا: معمارية (MVC (Model – View – Controller

بالإضافة إلى معمارية Client-Server، سيتم تطبيق نمط MVC داخل الخادم (Server-Side) لتنظيم الكود وتحسين قابلية الصيانة والتوسع.

تقوم معمارية MVC على تقسيم النظام إلى ثلاث طبقات رئيسية:

١. Model (النموذج): تمثل البيانات والهياكل المرتبطة بها، وتشمل جميع العمليات الخاصة بالتخزين والاسترجاع من قاعدة البيانات.

٢. View (العرض): تمثل واجهات المستخدم التي تُعرض عبر React.js في الواجهة الأمامية.

٣. Controller (المتحكم): تدير التفاعل بين Model و View، وتتولى استقبال الطلبات من المستخدم وتنفيذ المنطق اللازم ثم إرسال النتائج إلى الواجهة.

تطبيق MVC في هذا المشروع

Controllers المستخدمة:

- MessageControl: مسؤول عن إدارة الرسائل والمحادثات.
- PostControl: مسؤول عن إنشاء، تعديل، وحذف المنشورات.
- StoryControl: مسؤول عن رفع القصص (Stories) وإدارتها.
- UserControl: مسؤول عن تسجيل المستخدمين وتحديث معلوماتهم.

Models المستخدمة:

- Email, Message, Post, Story, User, Connection

- تمثل كل منها الجداول أو المجموعات (Collections) في قاعدة البيانات MongoDB، وتُعرّف الخصائص (Attributes) والعلاقات فيما بينها.

:View

- واجهات المستخدم المبنية باستخدام React.js و CSS، والتي تتلقى البيانات من الخادم عبر RESTful APIs وتعرضها للمستخدم بطريقة تفاعلية وسهلة الاستخدام.

١.٢.٦.١ إيجابيات معمارية MVC:

١. تنظيم الكود (Code Organization): تقسيم الكود إلى مكونات منفصلة يسهل قراءتها وصيانتها.
٢. سهولة التطوير الجماعي: يمكن لفريق الواجهة الأمامية (Front-End) وفريق الخادم (Back-End) العمل بالتوازي دون تعارض.
٣. إعادة الاستخدام (Reusability): يمكن إعادة استخدام الموديلات (Models) في خدمات أو مشاريع أخرى.

٤. قابلية الاختبار (Testability): سهولة اختبار كل طبقة على حدة (Unit Testing).

١. مرونة التطوير: يسهل تحديث أي جزء من النظام دون التأثير على الأجزاء الأخرى.

٢.٢.٦.١ سلبيات معمارية MVC:

١. زيادة التعقيد (Complexity): في المشاريع الصغيرة قد تبدو المعمارية معقدة أكثر من اللازم.
٢. تعدد الطبقات (Multiple Layers): يسبب زيادة في عدد الملفات والاعتمادات البرمجية.

٣. صعوبة تتبع الأخطاء أحيانًا: لأن الطلب ينتقل بين أكثر من طبقة قبل الوصول للنتيجة النهائية.

٣.٢.٦.١ الربط بين Client-Server و MVC في المشروع

يُعتبر النظام في هذا المشروع مزيجًا من المماريتين:

من الخارج، النظام قائم على Client-Server Architecture، حيث يتفاعل المستخدم عبر الواجهة (Client) مع الخادم (Server) عبر الإنترنت. ومن الداخل، يُدار الخادم وفق MVC Pattern الذي ينظم سير البيانات داخل النظام بين Model – Controller – View، مما يجعل النظام قابلاً للتوسع وسهل الصيانة.

بذلك تحقق المنصة توازنًا مثاليًا بين التوزيع الشبكي (Networked Architecture) وتنظيم المكونات الداخلية (Software Structure)، ما يؤدي إلى نظام قوي، آمن، وسهل التطوير المستقبلي.

٢ . الفصل الثاني : الدراسة التحليلية

١.٢ المتطلبات العامة (General Requirements)

تمثل المتطلبات العامة الإطار العام للمشروع، وتشمل الأهداف والقيود والعناصر الأساسية الواجب تحقيقها لضمان بناء منصة تواصل اجتماعي متكاملة.

وتتمثل فيما يلي:

١. تطوير نظام تواصل اجتماعي متكامل (Integrated Social Media Platform) يسمح للمستخدمين بالتفاعل عبر المنشورات، الرسائل، والقصص.

٢. استخدام تقنيات (MongoDB, Express.js, React.js, Node.js) لبناء واجهات أمامية وخلفية مترابطة.

٣. ضمان تجربة مستخدم سلسة (User Experience – UX) وواجهة تفاعلية جذابة (User Interface – UI) باستخدام React.js و CSS.

٤. تحقيق الأمان العالي (Security) في توثيق المستخدمين وإدارة الجلسات باستخدام Clerk, و HTTPS.
٥. دعم الميزات اللحظية (Real-time Features) مثل الدردشة الفورية (Chat) والإشعارات (Notifications) باستخدام Socket.io.
٦. تخزين الوسائط (Media Storage) ومعالجتها عبر ImageKit لضمان السرعة وجودة الصور.
٧. إدارة المهام الخلفية (Background Jobs) عبر Inngest لتشغيل العمليات المجدولة مثل حذف القصص المنتهية أو إرسال التنبيهات.
٨. نشر النظام عبر الإنترنت (Deployment) ليكون متاحًا للمستخدمين باستخدام خوادم سحابية (Cloud Hosting). والأنظمة.

٢.٢ المتطلبات الوظيفية (Functional Requirements)

١.٢.٢ نظام التسجيل والدخول (Authentication System)

١. إمكانية إنشاء حساب جديد (Sign Up) باستخدام البريد الإلكتروني أو رقم الهاتف.

٢. تسجيل الدخول (Sign In) باستخدام بيانات الاعتماد أو عبر / Social Login (Google / GitHub).

٣. تفعيل خاصية إعادة تعيين كلمة المرور (Reset Password) عند نسيانها.

٤. تسجيل الخروج من الجلسة الحالية (Log Out).

٢.٢.٢ إدارة الملف الشخصي (User Profile Management)

١. عرض المعلومات الأساسية للمستخدم (الاسم، الصورة، Bio، عدد المتابعين).

٢. تعديل الملف الشخصي (Edit Profile) وتحديث البيانات.

٣. عرض ملفات المستخدمين الآخرين (View Other Profiles).

٤. عرض حالة الاتصال (Online / Offline Status).

٥. حماية الدخول وإدارة الجلسات باستخدام Tokens و Clerk Authentication.

٣.٢.٢ إدارة المنشورات (Posts Management)

١. إنشاء منشورات جديدة (Create Post) نصية أو تحتوي على صور.

٢. تعديل أو حذف المنشورات (Edit / Delete Post).

٣. عرض المنشورات في الصفحة الرئيسية (Feed) بترتيب زمني.

٤. التفاعل مع المنشورات عبر الإعجابات (Like) والتعليقات (Comment).

٥. حفظ المنشورات المفضلة (Save Post) أو مشاركتها (Share Post).
٦. الإبلاغ عن المنشورات المخالفة (Report Post).

٤.٢.٢ إدارة القصص (Stories / Status System)

١. رفع قصة جديدة (Upload Story) بصيغة صورة أو فيديو قصير.
٢. عرض القصص الخاصة بالمستخدمين الذين يتابعهم (View Stories).
٣. حذف القصة يدويًا أو اختفاؤها تلقائيًا بعد ٢٤ ساعة.
٤. عرض قائمة المشاهدين (Viewers List).

٥.٢.٢ نظام المراسلة (Real-time Chat & Messages)

١. بدء محادثة جديدة (Start Conversation) بين المستخدمين.
٢. إرسال واستقبال الرسائل النصية والصورية والرموز التعبيرية (Emojis).
٣. دعم المراسلة اللحظية (Real-time Chat) باستخدام Socket.io.
٤. عرض حالة الرسائل (مرسلة - مستلمة - مقروءة).
٥. حذف أو كتم المحادثة (Mute / Delete Chat).

٦.٢.٢ نظام العلاقات الاجتماعية (Connections System)

١. متابعة أو إلغاء متابعة المستخدمين (Follow / Unfollow).
٢. إرسال واستقبال طلبات الصداقة (Friend Requests).
٣. عرض قوائم المتابعين والمتابعين (Followers / Following Lists).

٤. اقتراح أصدقاء بناءً على الاهتمامات المشتركة (Friend Suggestions).

٥. حظر أو إلغاء حظر المستخدمين (Block / Unblock).

٧.٢.٢ نظام البحث والاكتشاف (Search & Discover System)

١. البحث عن المستخدمين بالاسم أو اسم المستخدم (Username Search).

٢. البحث عن المستخدمين من خلال الموقع.

٣. البحث عن المنشورات باستخدام الوسوم (Hashtags).

٤. عرض صفحة اكتشاف (Discover Page) تحتوي على المنشورات الأكثر تفاعلاً والمستخدمين المقترحين.

٨.٢.٢ نظام الإشعارات (Notifications System)

١. إرسال إشعارات فورية (Real-time Notifications) عند التفاعل مع المنشورات أو المتابعة الجديدة.

٢. عرض جميع الإشعارات في صفحة مخصصة (Notifications Page).

٣. إمكانية حذف الإشعارات أو وضع علامة "تمت القراءة" (Mark as Read).

٩.٢.٢ إدارة الوسائط (Media Management)

٢. رفع الصور عبر ImageKit وتحسينها تلقائياً.

٣. دعم صيغ الصور المختلفة (JPEG, PNG, WEBP).

٤. ضغط الصور لتقليل الحجم دون فقدان الجودة.

٥. حماية روابط الوسائط من الوصول غير المصرح به (Access Control).

١٠.٢.٢ النظام الخلفي (Backend Operations)

١. إنشاء واجهات للتفاعل مع المستخدمين والمنشورات والرسائل.
٢. تنفيذ المهام المجدولة عبر Inngest مثل حذف القصص المنتهية.
٣. تسجيل الأخطاء والنشاطات (Error Logging & Monitoring).

٣.٢ المتطلبات غير الوظيفية (Non-Functional Requirements)

١. Security: استخدام Clerk للتحقق، و HTTPS لتشفير البيانات.
٢. Performance: سرعة تحميل عالية، استجابة فورية، Caching، Real-time Chat.
٣. Scalability: دعم التوسعة الأفقية باستخدام Node.js و MongoDB.
٤. Usability: واجهة تفاعلية، تصميم متجاوب (Responsive Design).
٥. Reliability: تشغيل مستقر، مراقبة النظام (Monitoring & Logging).
٦. Maintainability: كود منظم (Modular Code)، توثيق جيد.
٧. Compatibility: دعم جميع المتصفحات والأجهزة (Cross-Platform).
٨. Privacy: حماية بيانات المستخدمين، إعدادات خصوصية مرنة.

٤.٢ المتطلبات التي سيتم تنفيذها في المشروع الفصلي

١. إمكانية إنشاء حساب جديد (Sign Up) باستخدام البريد الإلكتروني أو رقم الهاتف.

٢. تسجيل الدخول (Sign In) باستخدام بيانات الاعتماد أو عبر / Social Login (Google / GitHub).

٣. تفعيل خاصية إعادة تعيين كلمة المرور (Reset Password) عند نسيانها.

٤. تسجيل الخروج من الجلسة الحالية (Log Out).

٥. عرض المعلومات الأساسية للمستخدم (الاسم، الصورة، Bio، عدد المتابعين).

٦. تعديل الملف الشخصي (Edit Profile) وتحديث البيانات.

٧. عرض ملفات المستخدمين الآخرين (View Other Profiles).

٨. عرض حالة الاتصال (Online / Offline Status).

٩. إنشاء منشورات جديدة (Create Post) نصية أو تحتوي على صور.

١٠. تعديل أو حذف المنشورات (Edit / Delete Post).

١١. عرض المنشورات في الصفحة الرئيسية (Feed) بترتيب زمني.

١٢. التفاعل مع المنشورات عبر الإعجابات (Like).

١٣. التفاعل مع المنشورات عبر التعليقات (Comment).

١٤. حفظ المنشورات المفضلة (Save Post) أو مشاركتها (Share Post).

١٥. الإبلاغ عن المنشورات المخالفة (Report Post).

١٦. رفع قصة جديدة (Upload Story) بصيغة صورة أو فيديو قصير.

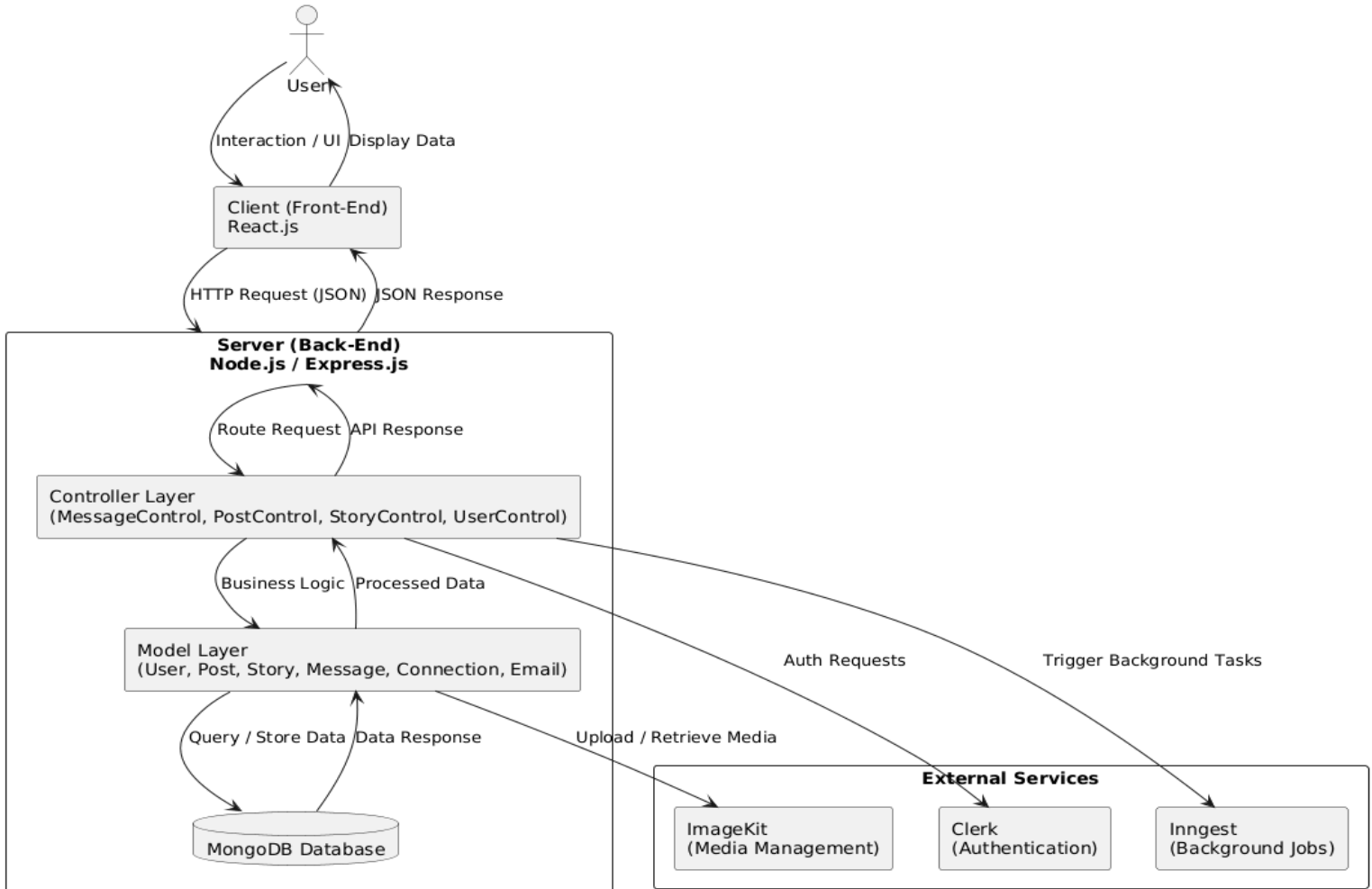
١٧. عرض القصص الخاصة بالمستخدمين الذين يتابعهم (View Stories).

١٨. حذف القصة يدويًا أو اختفاؤها تلقائيًا بعد ٢٤ ساعة.

١٩. عرض قائمة المشاهدين (Viewers List).
٢٠. Security: استخدام Clerk للتحقق، و HTTPS لتشفير البيانات.
٢١. Usability: واجهة تفاعلية، تصميم متجاوب (Responsive Design).
٢٢. Compatibility: دعم جميع المتصفحات والأجهزة (Cross-Platform).

٥.٢ المخطط المعماري (Architecture Diagram)

Architecture Diagram - Client Server + MVC Integration



٢ مخطط دمج المعماريين

يوضح المخطط المعماري كيفية تكامل معماريتي Client-Server و MVC في مشروع منصة التواصل

الاجتماعي.

يتفاعل المستخدم مع واجهة المنصة المبنية بتقنية React.js (العميل – Client)، والتي ترسل الطلبات إلى

الخادم عبر HTTP/HTTPS

يتكوّن الخادم (Server) من ثلاث طبقات رئيسية وفق نمط MVC:

Controller Layer: تستقبل الطلبات وتنفّذ المنطق المطلوب من خلال وحدات مثل MessageControl

و PostControl و StoryControl و UserControl.

Model Layer: تتعامل مع البيانات الممثلة في نماذج مثل User و Post و Story و Message

و Connection و Email، وتتصل بقاعدة البيانات MongoDB.

View Layer: تُرسل البيانات النهائية إلى الواجهة الأمامية ليتم عرضها للمستخدم.

يرتبط النظام أيضًا بخدمات خارجية مثل Clerk للتوثيق، و ImageKit لإدارة الوسائط، و Inngest للمهام

الخلفية.

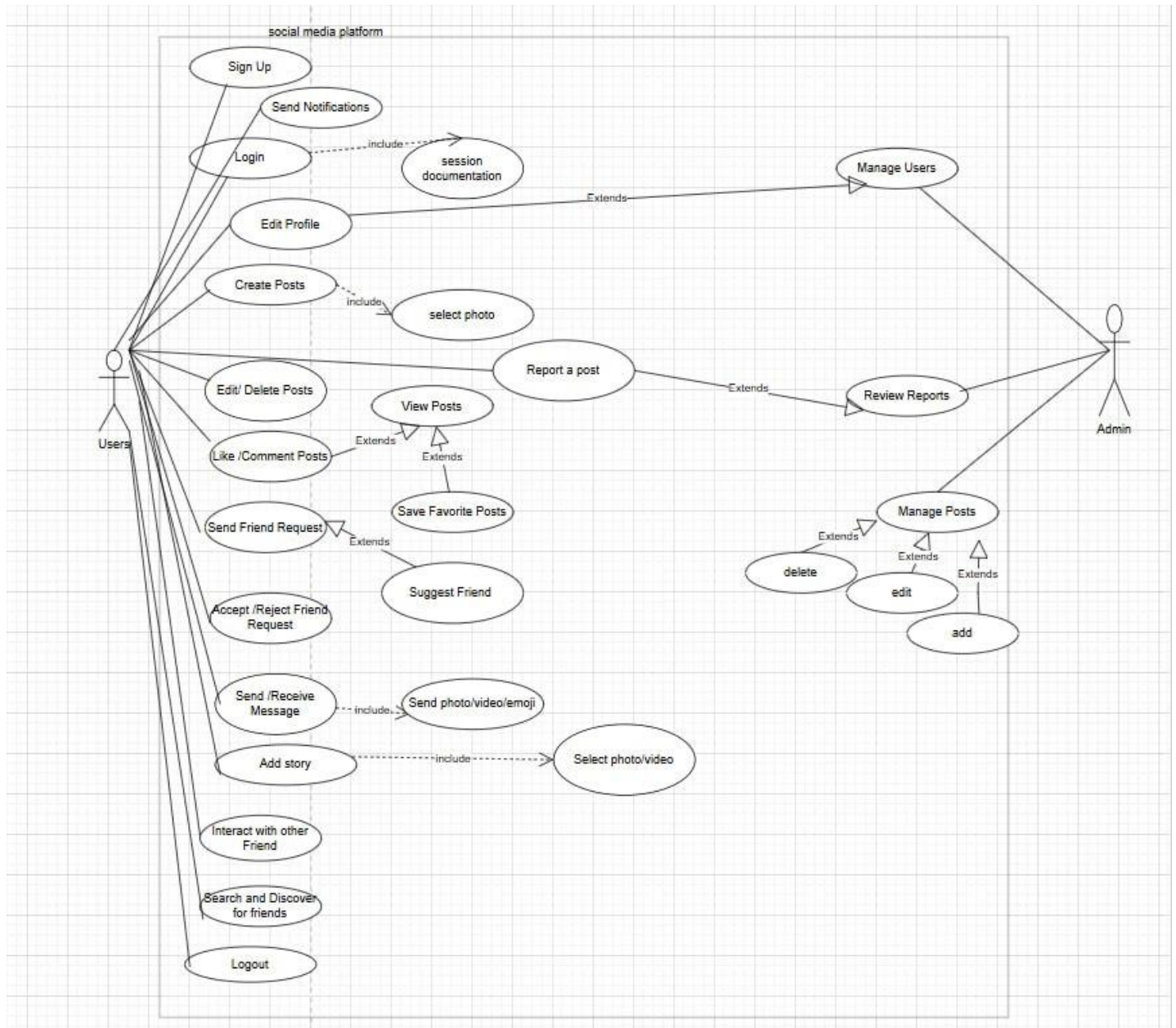
تسير البيانات في تسلسل واضح:

المستخدم → الواجهة → الخادم (Models و Controllers) → قاعدة البيانات → ثم تعود النتائج إلى

المستخدم عبر الواجهة.

يُظهر المخطط بوضوح الترابط بين مكونات النظام وكيفية تدفق المعلومات بطريقة منظمة وفعالة.

٦.٢ مخطط ال Use Case Diagram



٣ مخطط ال Class Diagram

٧.٢ توصيف حالات الاستخدام Use Case Specifications

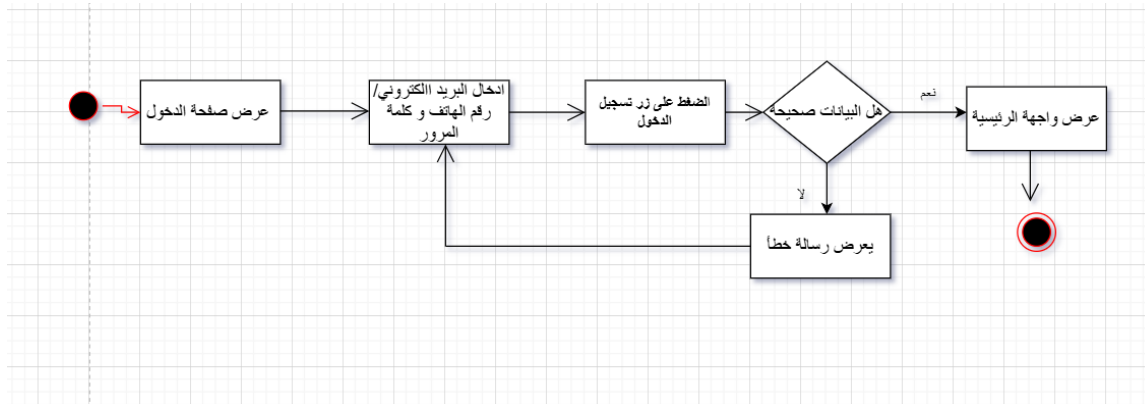
٨.٢ مخططات النشاط Activity Diagrams

٩.٢ مخططات التسلسل Sequence Diagrams

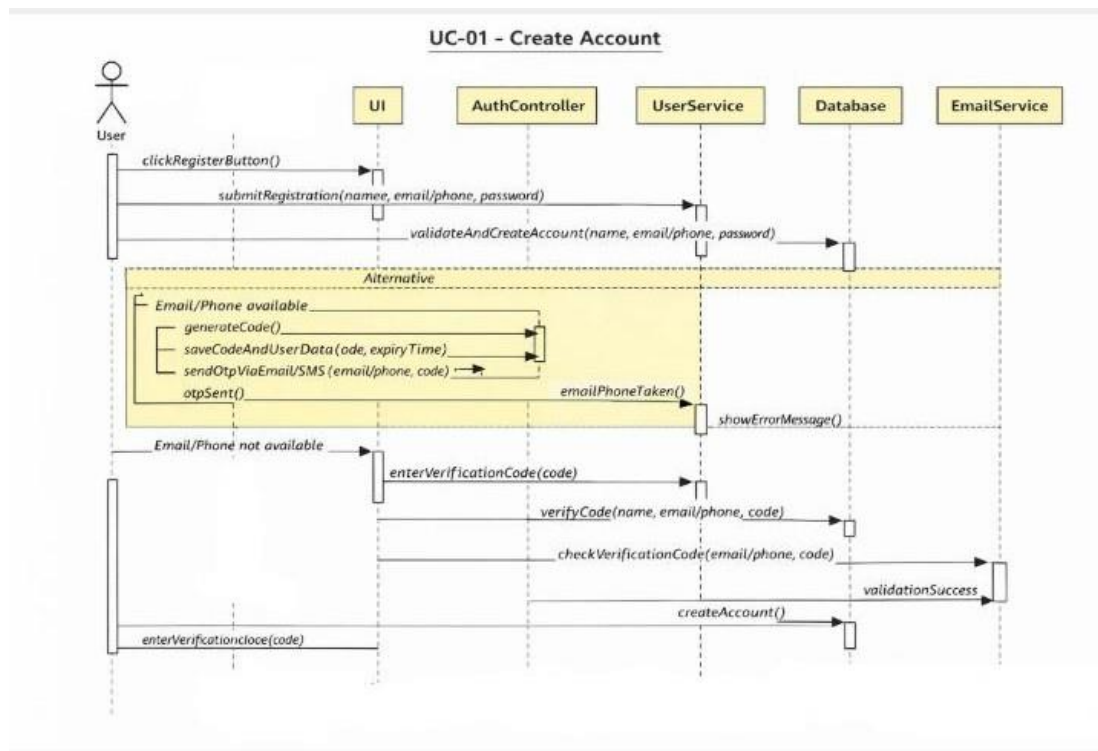
UC-01	Use Case Name
المستخدم	Actors
تمكين المستخدم من انشاء حساب جديد على النظام باستخدام البريد الالكتروني او رقم الهاتف للتحقق من الهوية والوصول الى خدمات النظام.	Description
ليس لديه أي تسجيل دخول.	Precondition
١- يعرض النظام نموذج التسجيل . ٢- يدخل المستخدم البيانات (الاسم و البريد الالكتروني وكلمة المرور او عن طريق GOOGLE) . ٣- يضغط المستخدم على زر "REGISTER" . ٤- يتحقق النظام من صحة البيانات ويرسل رمز تحقق لينشئ الحساب. ٥- يدخل المستخدم رمز التحقق الذي ارسل له عبر بريده الالكتروني. ٦- يرسل رسالة تأكيد ويوجه المستخدم.	Main flow

المستخدم يلغي العملية ويعود للحالة السابقة: A1: المستخدم يدخل البريد الالكتروني مكرر يرسل رسالة البريد الالكتروني: A2: مأخوذ مسبقاً. لا يرسل رمز للتحقق ويقوم المستخدم بإعادة ارسال رسالة: A3:	Alternative flow
تم انشاء الحساب وتسجيل الدخول تلقائي.	Postconditions

توصيف حالة تسجيل الحساب 2 جدول



4 مخطط النشاط حالة تسجيل الحساب

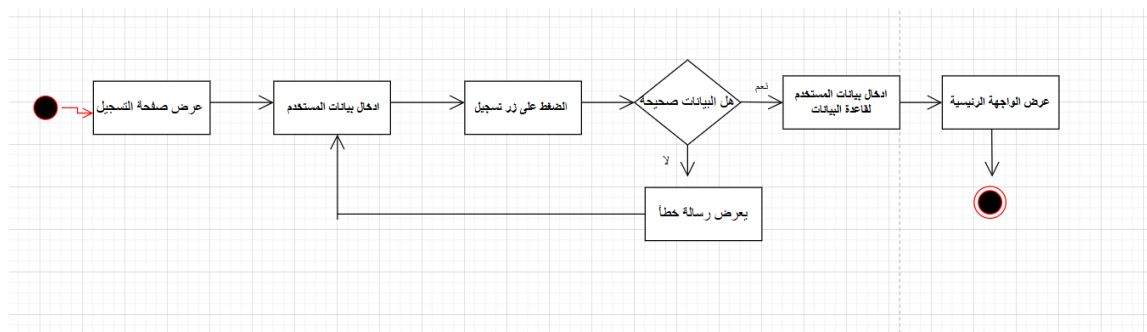


5 مخطط التسلسل حالة تسجيل الحساب

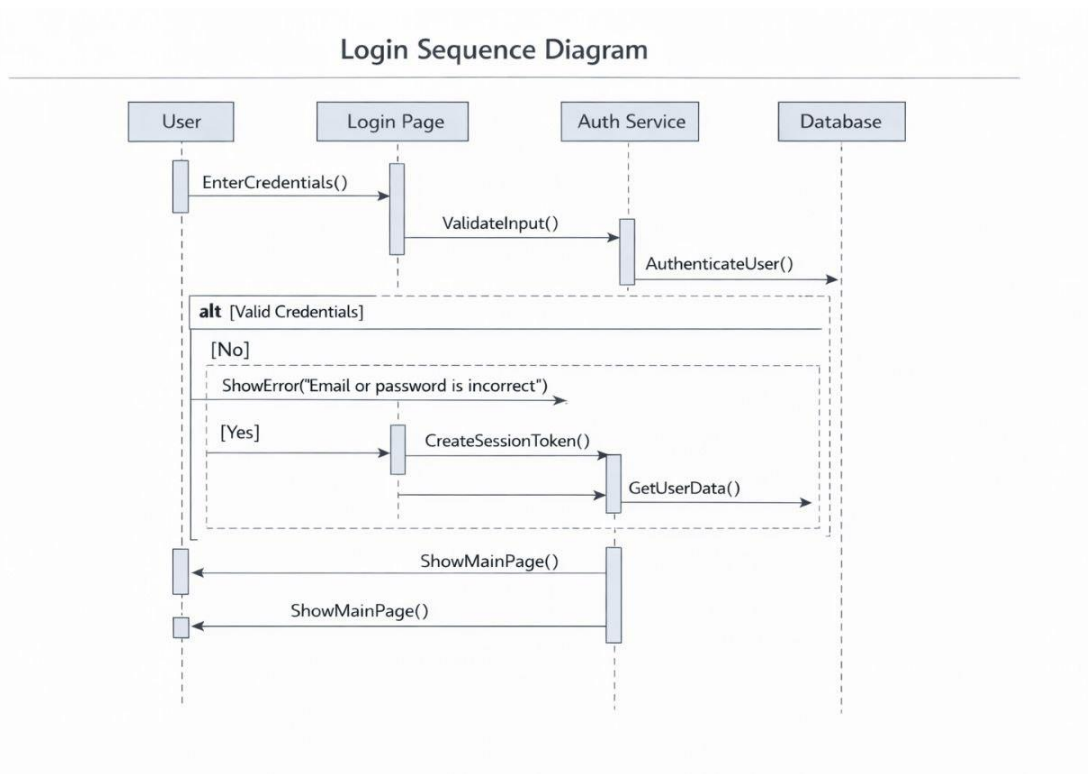
UC-02	Use Case Name
المستخدم	Actors
يُتيح للمستخدم الوصول إلى حسابه المسجل.	Description
ليس لديه أي تسجيل دخول.	Precondition

<p>١- يفتح المستخدم صفحة تسجيل الدخول .</p> <p>٢- يدخل المستخدم البريد الإلكتروني أو رقم الهاتف وكلمة المرور.</p> <p>٣- يتحقق النظام من صحة البيانات مقابل قاعدة البيانات أو خدمة Clerk Authentication.</p> <p>٤- إذا كانت البيانات صحيحة النظام يصدر Token للجلسة ويعطي المستخدم وصولاً كاملاً للنظام.</p> <p>٥- النظام يعرض واجهة المستخدم الرئيسية بعد تسجيل الدخول.</p>	Main flow
<p>A١: facebook ,google يتيح النظام تسجيل دخول عبر حساب</p> <p>A٣: يعرض النظام رسالة خطأ "البريد الإلكتروني او كلمة المرور غير صحيحة"</p>	Alternative flow
<p>يتم التحقق من هوية المستخدم ويسمح للمستخدم بالوصول الى حسابه وميزات النظام المختلفة.</p>	Postconditions

جدول 3 توصيف حالة تسجيل الدخول



6 مخطط النشاط حالة تسجيل الدخول



7 مخطط التسلسل حالة تسجيل الدخول

UC-03	Use Case Name
المستخدم	Actors

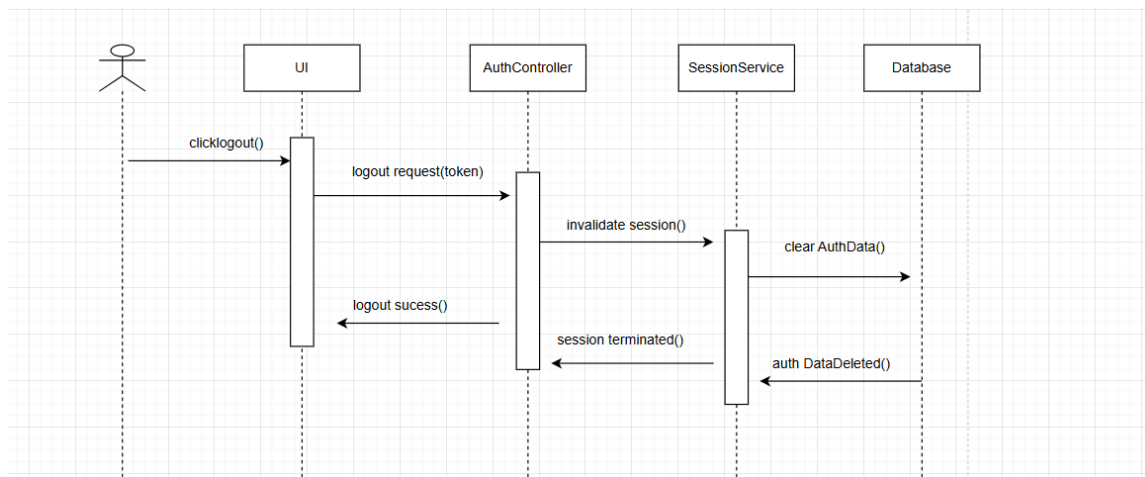
Description	يتيح للمستخدم اكمال عمایة تسجيل الدخول بعد ادخال اسم المستخدم وكلمة المرور بنجاح من خلال رمز التحقق المؤقت, يتم ارساله الى البريد الالكتروني لتعزيز الأمان.
Precondition	المستخدم قام بإدخال اسم المستخدم وكلمة المرور بشكل صحيح.
Main flow	<p>١ - النظام يتحقق من صحة اسم المستخدم وكلمة المرور.</p> <p>٢ - النظام ينشئ رمز تحقق مؤقت صالح لمدة زمنية محددة ليريد الالكتروني للمستخدم.</p> <p>٣ - النظام يعرض شاشة ادخال الرمز ليقوم المستخدم بإدخاله.</p> <p>٤ - النظام يتحقق من صحة الرمز المدخل ومن صلاحيته الزمنية.</p> <p>٥ - النظام يؤكد نجاح العملية وينشئ الجلسة ويقوم بتوجيه المستخدم للصفحة الرئيسية.</p>
Alternative flow	<p>المستخدم يدخل رمز تحقق خاطئ النظام يعرض رسالة خطأ توضح ان : A٢ الرمز غير صحيح..</p> <p>انتهاء صلاحية رمز التحقق , النظام يعرض رسالة بانتهاء صلاحية الرمز : A٤ حيث يوفر خيار إعادة رمز جديد.</p>
Postconditions	تم انشاء الجلسة بأمان.

جدول 4 توصيف حالة الدخول الآمن

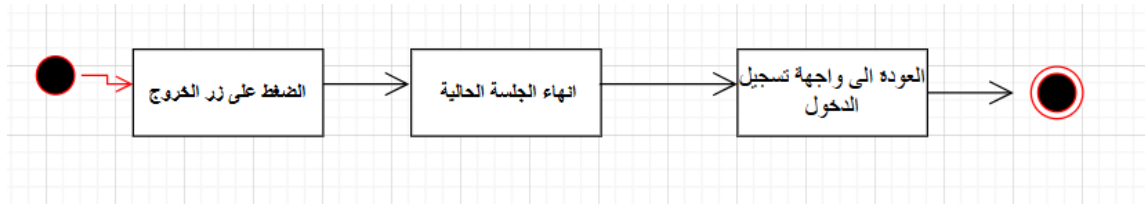
UC-04	Use Case Name
-------	---------------

المستخدم	Actors
انهاء الجلسة الحالية والخروج بأمان من الحساب.	Description
النظام مسجل دخول بالفعل الى النظام.	Precondition
<p>١ - المستخدم يضغط على زر تسجيل الخروج.</p> <p>٢ - النظام يقوم بإلغاء الجلسة الحالية ويحذف بيانات المصادقة من المتصفح.</p> <p>٣ - النظام يعيد توجيه المستخدم الى صفحة تسجيل الدخول ويعرض رسالة "تم تسجيل الخروج بنجاح".</p>	Main flow
<p>المستخدم يختار الغاء عملية الخروج ويبقى غي الصفحة الحالية: A1</p> <p>النظام يكتشف ان الجلسة منتهية ويتم تحويله مباشرة الى تسجيل الدخول: A٤</p>	Alternative flow
يتم تسجيل الخروج بنجاح ويتم تحويل المستخدم الى صفحة الدخول.	Postconditions

جدول 5 توصيف حالة تسجيل الخروج



8 مخطط التسلسل حالة تسجيل الخروج

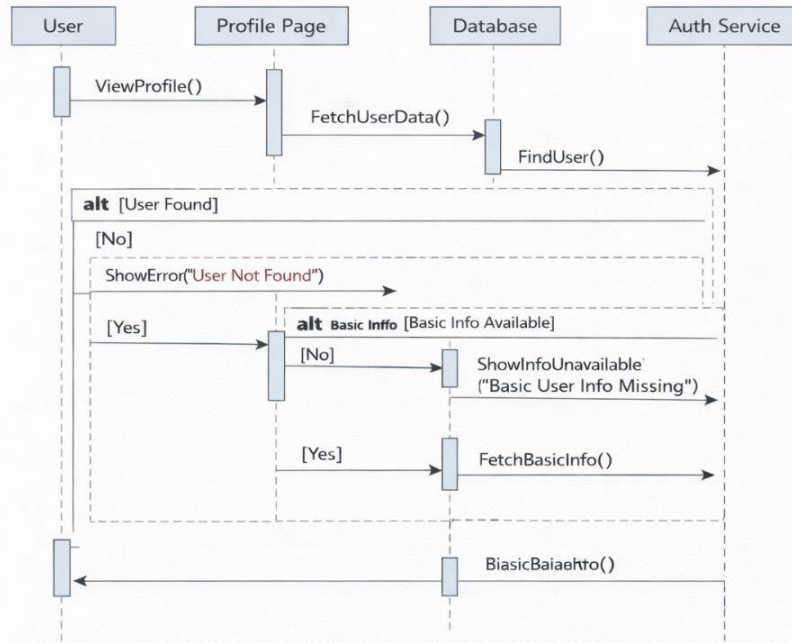


9 مخطط النشاط حالة تسجيل الخروج

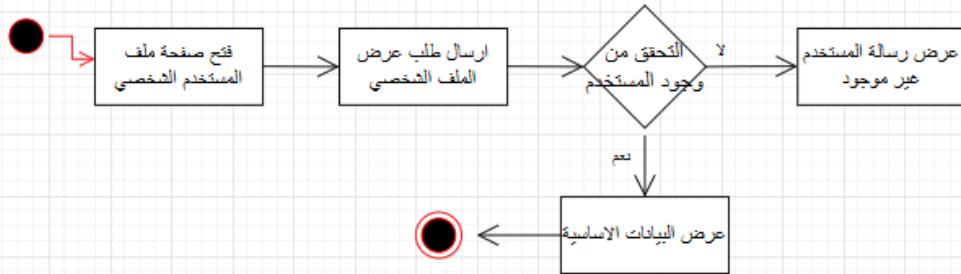
UC-05	Use Case Name
المستخدم	Actors
تمكين المستخدم من رؤية المعلومات الأساسية لحساب مستخدم معين مثل الاسم والصورة الشخصية والنبذة وعدد المتابعين.	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقاً.	Precondition
١- يفتح صفحة ملف المستخدم الشخصي . ٢- يتحقق من وجود المستخدم في قاعدة البيانات . ٣- يجلب البيانات الأساسية (الاسم, الصورة, النبذة, عدد المتابعين) . ٤- يعرض المعلومات على واجهة المستخدم في صفحة ال profile.	Main flow
اذا لم يعثر على المستخدم في قاعدة البيانات يعرض النظام رسالة خطأ: A2: "المستخدم غير موجود". اذا لم تتوفر بعض المعلومات يعرض النظام عدم وجود المعلومات في الحقل: A3: الاتي.	Alternative flow
تظهر المعلومات الأساسية للمستخدم بنجاح على الشاشة .	Postconditions

جدول 6 توصيف حالة عرض الواجهة الأساسية

Show Basic Information Sequence Diagram



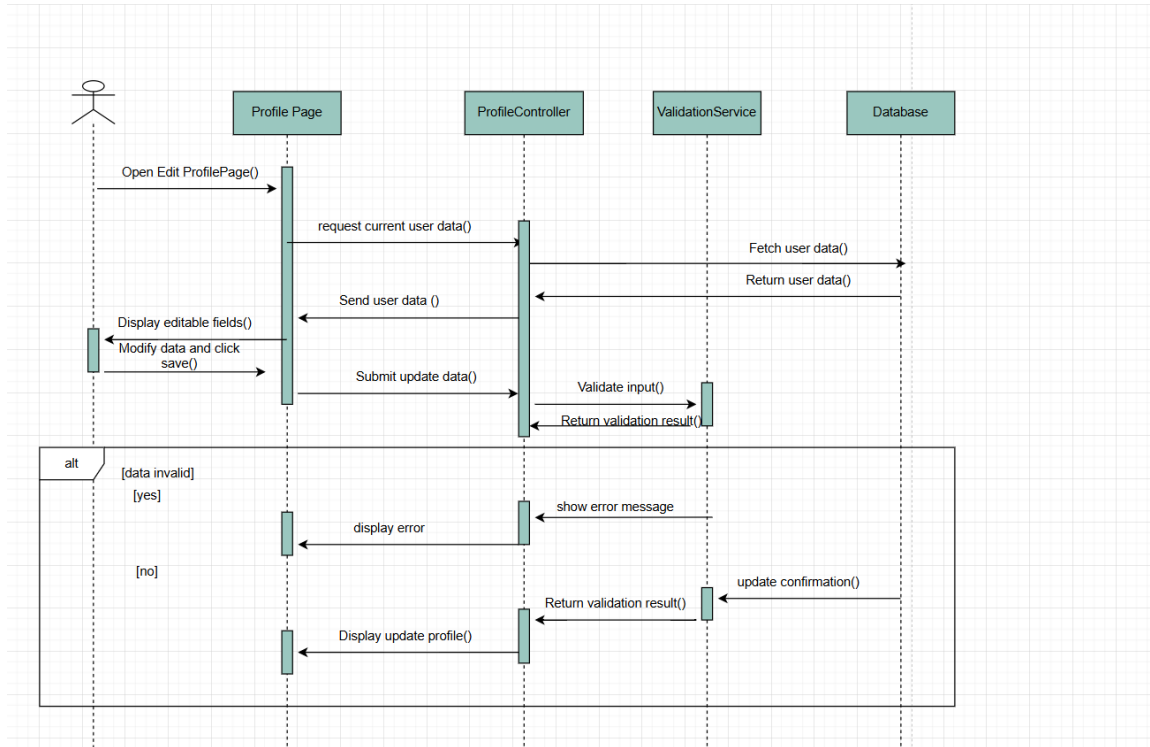
10 مخطط التسلسل حالة عرض الواجهة الأساسية



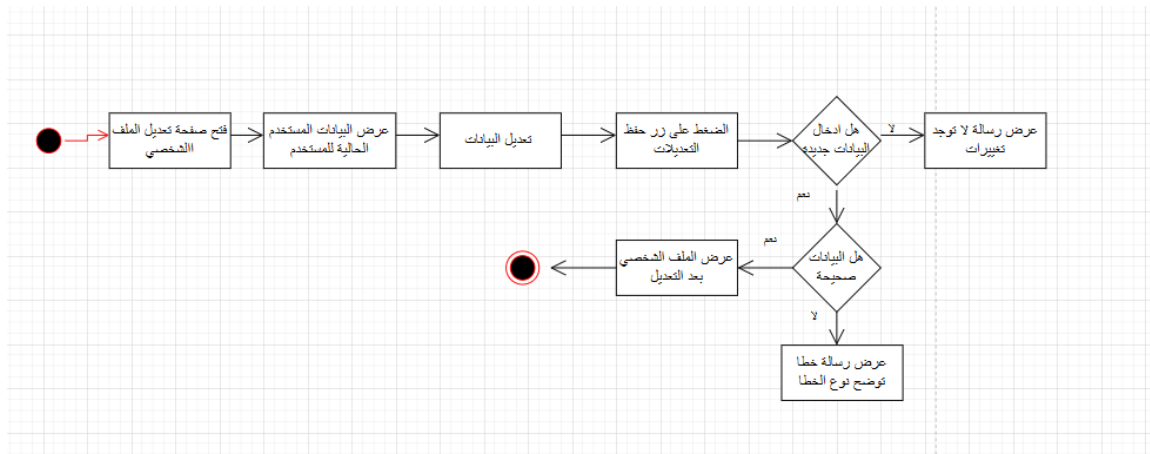
11 مخطط النشاط حالة عرض الواجهة الأساسية

UC-06	Use Case Name
المستخدم	Actors
تمكين المستخدم من تعديل بياناتهم الشخصية المسجلة في النظام مثل الاسم و النبذة والصورة و البريد الالكتروني و كلمة المرور .	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقاً.	Precondition
<p>١ - يفتح صفحة تعديل الملف الشخصي .</p> <p>٢ - يعرض النظام البيانات الحالية للمستخدم داخل حقول قابلة للتعديل .</p> <p>٣ - يقوم المستخدم بتعديل البيانات المطلوبة (الاسم والصورة والبريد ...) .</p> <p>٤ - يضغط المستخدم على زر "حفظ التعديلات" .</p> <p>٥ - يتحقق النظام من صحة المدخلات (تنسيق البريد الالكتروني او حجم الصورة).</p> <p>٦ - يقوم النظام بتحديث البيانات في قاعدة البيانات.</p> <p>٧ - يعرض النظام رسالة "تم تعديل الملف الشخصي بنجاح" .</p> <p>٨ - يعرض النظام الملف الشخصي بالبيانات الجديدة .</p>	Main flow
<p>إذا كانت البيانات المدخلة غير صحيحة يعرض النظام رسالة خطأ توضح: A2 نوع الخطأ ويطلب من المستخدم التصحيح .</p> <p>إذا لم يغير المستخدم أي بيانات وضغط "حفظ" يعرض النظام رسالة "لا: A3 توجد تغييرات لتحديثها" .</p>	Alternative flow
تظهر رسالة تأكيد بنجاح عملية التحديث وتعرض البيانات الجديدة في صفحة الملف الشخصي بعد التعديل .	Postconditions

جدول 7 توصيف حالة التعديل على الحساب



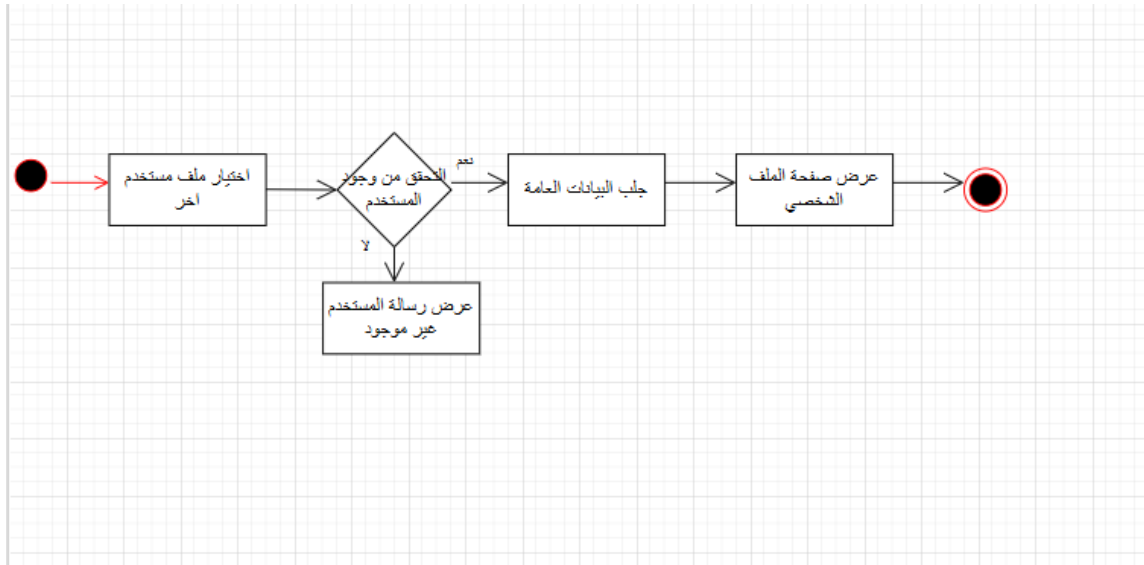
12 مخطط التسلسل حالة التعديل على الحساب



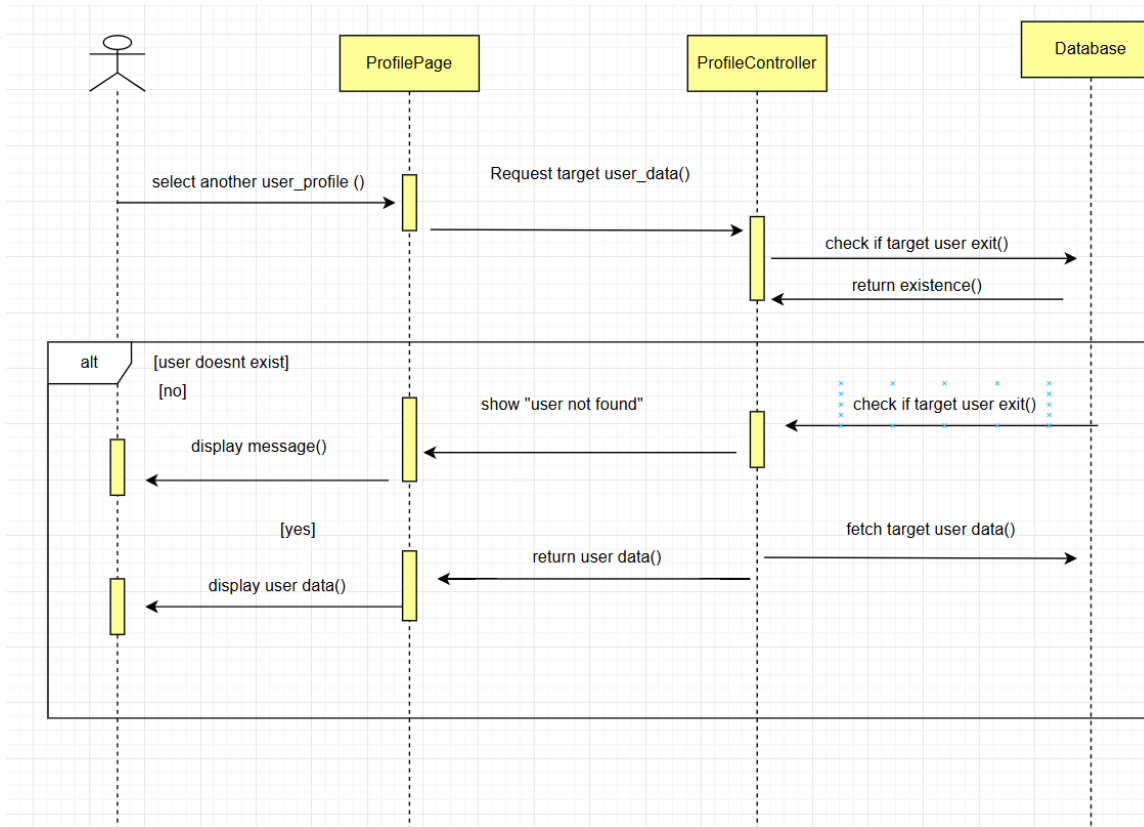
13 مخطط النشاط حالة التعديل على الحساب

UC-07	Use Case Name
المستخدم	Actors
تمكين المستخدم من زيارة صفحات المستخدمين الآخرين في النظام وعرض المعلومات عنه .	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقا.	Precondition
<p>١- يختار المستخدم ملف مستخدم آخر لعرضه اما من نتائج البحث او من قائمة المتابعين .</p> <p>٢- يتحقق النظام من وجود المستخدم الهدف في قاعدة البيانات.</p> <p>٣- يجلب النظام بيانات المستخدم الهدف وفق مستوى الصلاحية (الاسم والصورة والنبذة و عدد المتابعين و المنشورات العامة).</p> <p>٤- يعرض النظام صفحة الملف الشخصي للمستخدم الآخر بالمعلومات المسموح عرضها .</p>	Main flow
<p>اذا لم يجد النظام المستخدم في قاعدة البيانات يعرض النظام رسالة: A2 "المستخدم غير موجود".</p> <p>اذا كان الملف خاصا والمستخدم الحالي ليس من المتابعين المصرح لهم: A٣ يعرض النظام رسالة "هذا الحساب خاص" ويظهر فقط المعلومات المحدودة (الصورة والاسم).</p>	Alternative flow
تعرض صفحة المستخدم ملف المستخدم بالمعلومات العامة المسموح عرضها.	Postconditions

جدول 8 توصيف حالة عرض معلومات الحسابات



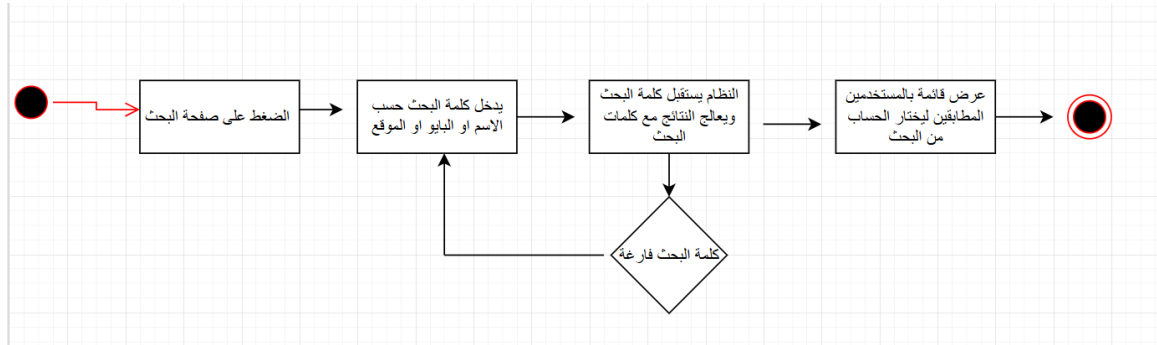
14 مخطط النشاط حالة عرض معلومات الحسابات



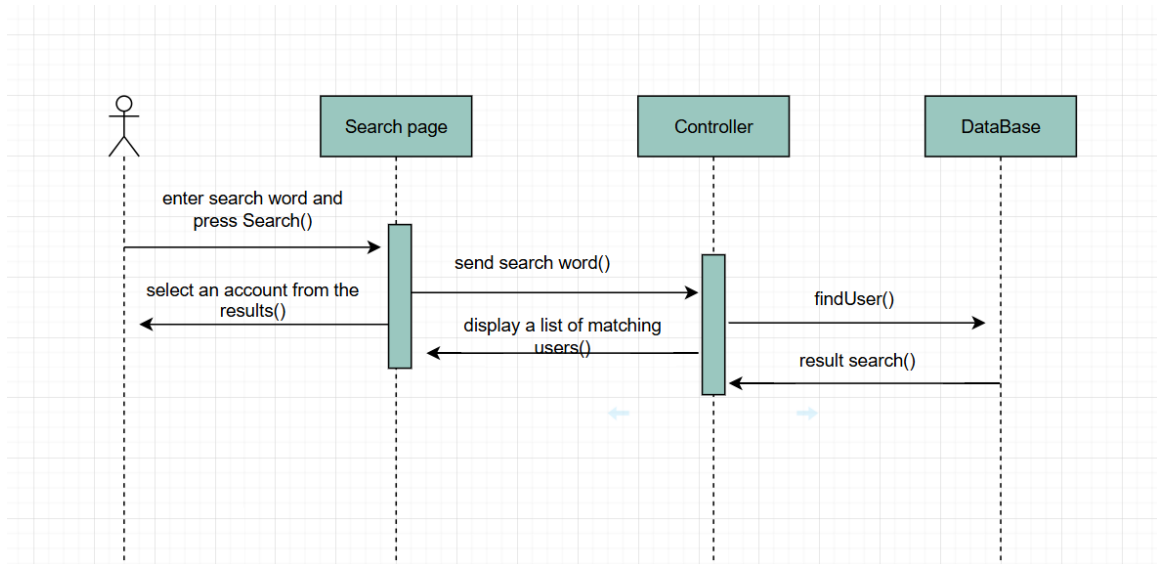
15 مخطط النشاط حالة عرض معلومات الحسابات

UC-08	Use Case Name
المستخدم	Actors
يُتيح للمستخدم البحث عن مستخدمين آخرين داخل النظام باستخدام كلمات مفتاحية .	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقاً.	Precondition
<p>١ - يضغط المستخدم على صفحة البحث ويدخل كلمة البحث .</p> <p>٢ - المستخدم يضغط على زر البحث .</p> <p>٣ - النظام يستقبل كلمة البحث ويعالج النتائج ويطابقها مع كلمات البحث .</p> <p>٤ - النظام يعرض قائمة بالمستخدمين المطابقين ويختار الحساب من نتائج البحث.</p>	Main flow
<p>ادخال كلمة بحث فارغة النظام يعرض رسالة تنبيه تطلب ادخال كلمة : A٣ البحث.</p> <p>لا توجد نتائج النظام يعرض رسالة لا توجد نتائج مطابقة: A٤</p>	Alternative flow
تم عرض نتائج البحث بنجاح.	Postconditions

جدول 9 توصيف حالة البحث عن الأصدقاء



١٦ مخطط النشاط حالة البحث عن الأصدقاء

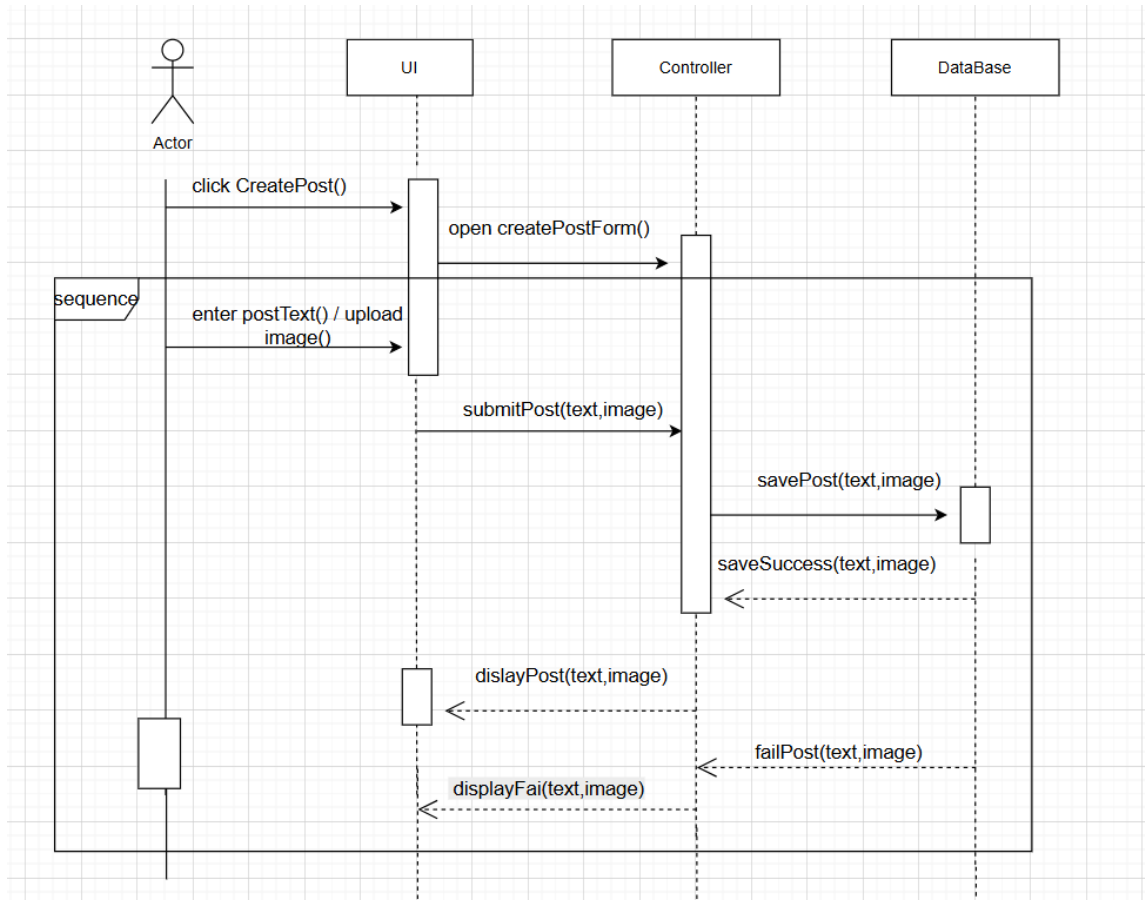


١٧ مخطط التسلسل حالة البحث عن الأصدقاء

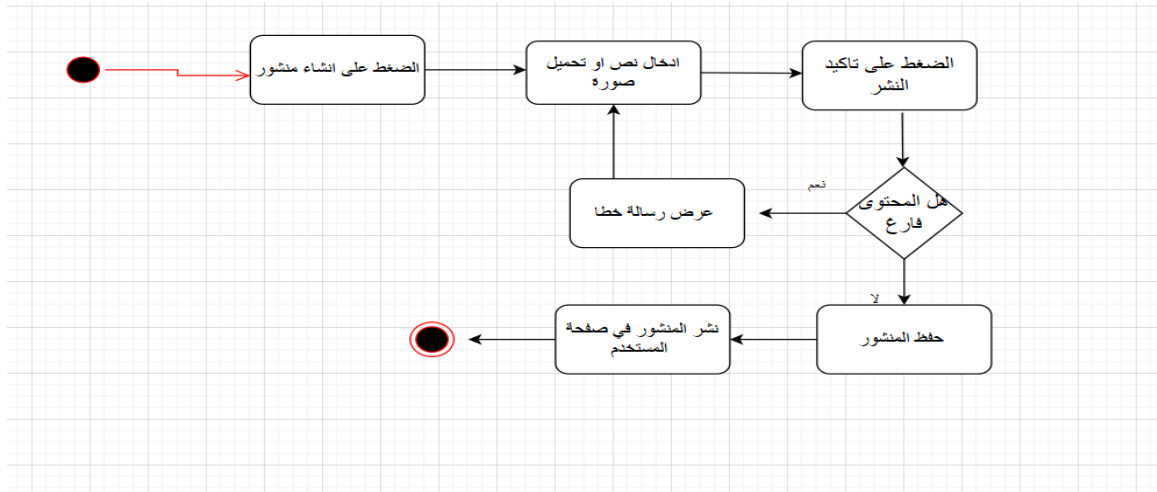
UC-09	Use Case Name
المستخدم	Actors
تمكين المستخدم من نشر منشور جديد يحتوي على صورة او نص او كليهما .	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقا.	Precondition
<p>٥- يضغط المستخدم على زر "إنشاء منشور جديد" .</p> <p>٦- تظهر نافذة لإدخال النص او تحميل صورة .</p> <p>٧- يكتب المستخدم نص المنشور او يختار صورة .</p> <p>٨- يضغط المستخدم على زر "تأكيد نشر" .</p> <p>٩- يتحقق النظام من صحة البيانات (وجود صورة او نص) .</p> <p>١٠- يخزن النظام المنشور في قاعدة البيانات مع بيانات المستخدم والوقت .</p> <p>١١- يعرض النظام المنشور الجديد في صفحة المستخدم .</p>	Main flow
<p>المستخدم لم يدخل أي محتوى (لا نص ولا صورة) يظهر تنبيه "المنشور فارغ"</p> <p>A٥ - فشل رفع الصورة يظهر تنبيه "حدث خطأ أثناء رفع الصورة"</p>	Alternative flow

تم عرض المنشور في صفحة المستخدم وحفظه في قاعدة البيانات	Postconditions
---	----------------

جدول 10 توصيف حالة نشر منشور جديد



18مخطط التسلسل حالة نشر منشور جديد

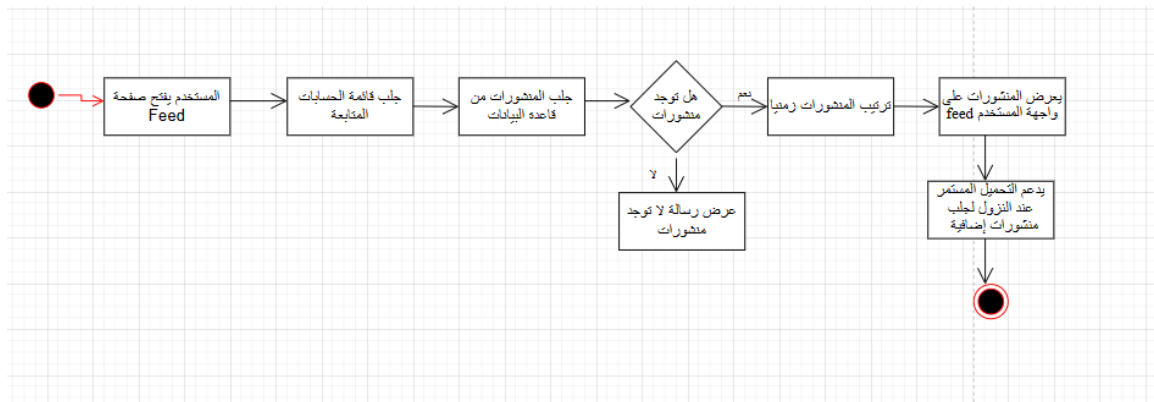


٩١ مخطط النشاط حالة نشر منشور جديد

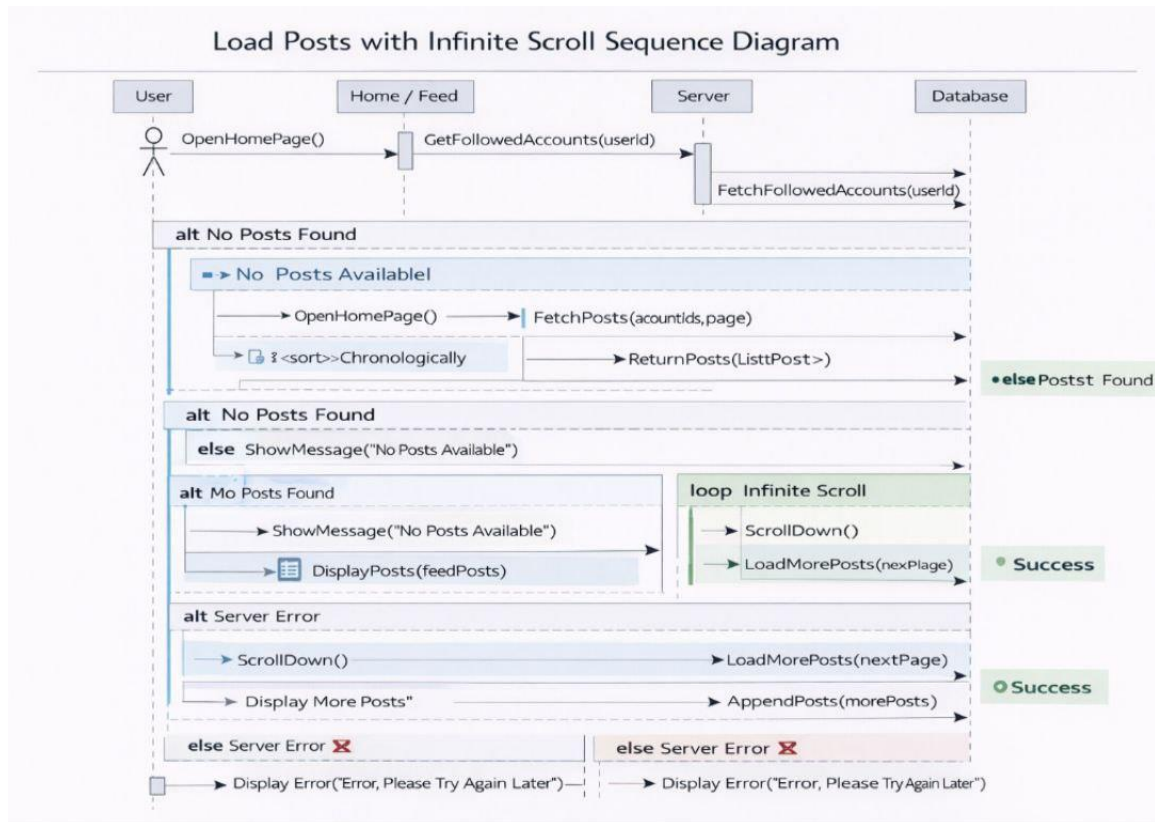
UC-10	Use Case Name
المستخدم	Actors
يجلب النظام المنشورات من قاعدة البيانات في الصفحة ويعرضها وفق ترتيب زمني مع دعم تحميل المستمر عند النزول.	
	Description

<p>1. تسجيل دخول المستخدم إلى حسابه مسبقا.</p> <p>2. المستخدمون لديه متابعون او يتابع حسابات تنشر منشورات.</p>	Precondition
<p>١-يفتح المستخدم الصفحة الرئيسية (Home/Feed).</p> <p>٢-يجلب النظام قائمة الحسابات التي يتابعها المستخدم .</p> <p>٣-يجلب النظام المنشورات من هؤلاء الحسابات من قاعدة البيانات.</p> <p>٤-يرتب المنشورات ترتيبا زمنيا من الاحداث الى الاقدم.</p> <p>٥-يعرض المنشورات على واجهة المستخدم (Feed).</p> <p>٦-يدعم التحميل المستمر عند النزول (infinite scroll) لجلب منشورات إضافية عند الحاجة.</p>	Main flow
<p>لم يجد النظام أي منشورات الحسابات التي يتابعها المستخدم يعرض رسالة: A3 "لا توجد منشورات حاليا".</p> <p>عند نشر منشورات جديدة يقوم النظام بتحديث الصفحة بشكل ديناميكي: A٥</p>	Alternative flow
<p>١-تظهر جميع المنشورات للمستخدم بالترتيب الزمني الصحيح.</p> <p>٢-يتم تحديث الصفحة بشكل مستمر عند وجود منشورات جديدة.</p>	Postconditions

جدول ١١ توصيف حالة عرض المنشورات



٢٠ مخطط النشاط حالة عرض المنشورات

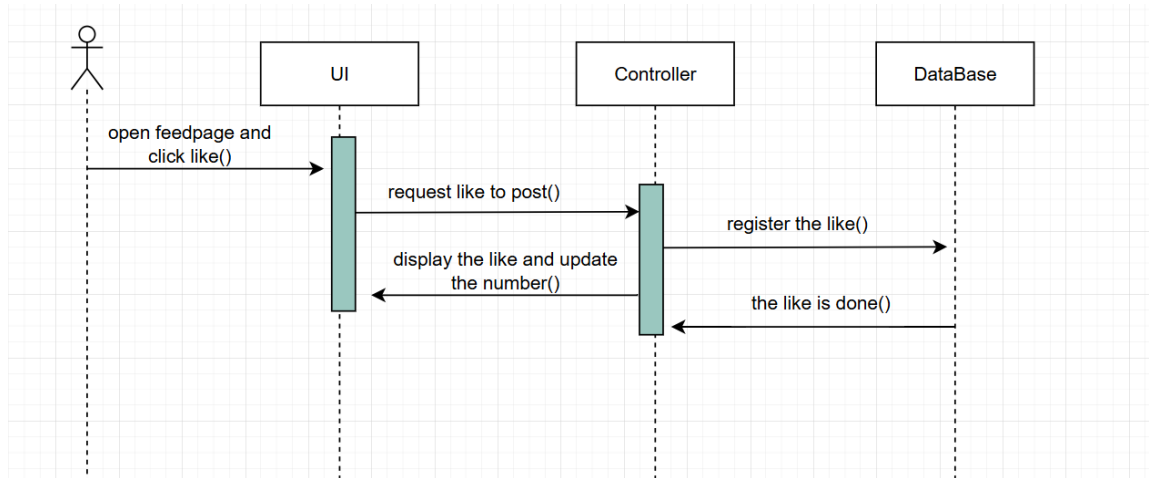


٢١ مخطط التسلسل حالة عرض المنشورات

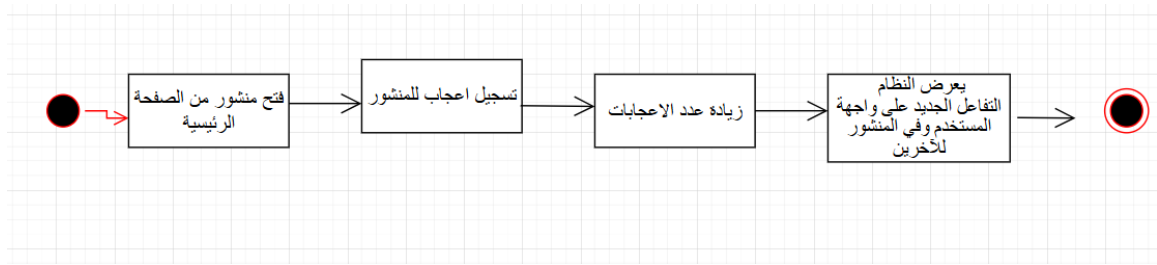
UC-11	Use Case Name
المستخدم	Actors
يسمح للمستخدم بإظهار رأيه تجاه المنشورات عبر إعطاء إعجاب	Description

<p>1. تسجيل دخول المستخدم إلى حسابه مسبقا.</p> <p>2. المنشور موجود ويمكن الوصول إليه للسماح بالتفاعل مع المنشور.</p>	Precondition
<p>١- يفتح المستخدم المنشورة في الصفحة الرئيسية او الملف الشخصي.</p> <p>٢- يضغط المستخدم زر اعجاب.</p> <p>٣- يسجل النظام التفاعل ويزيد عدد الاعجابات.</p> <p>٤- يعرض النظام التفاعل الجديد على واجهة المستخدم وفي المنشور للآخرين.</p>	Main flow
يعرض النظام رسالة: هذا المنشور غير متاح: A3	Alternative flow
يزداد عدد الاعجابات على المنشور ويظهر للمستخدمين الآخرين .	Postconditions

جدول ١٢ توصيف حالة الإعجاب بالمنشورات



٢٢ مخطط التسلسل حالة الإعجاب بالمنشورات



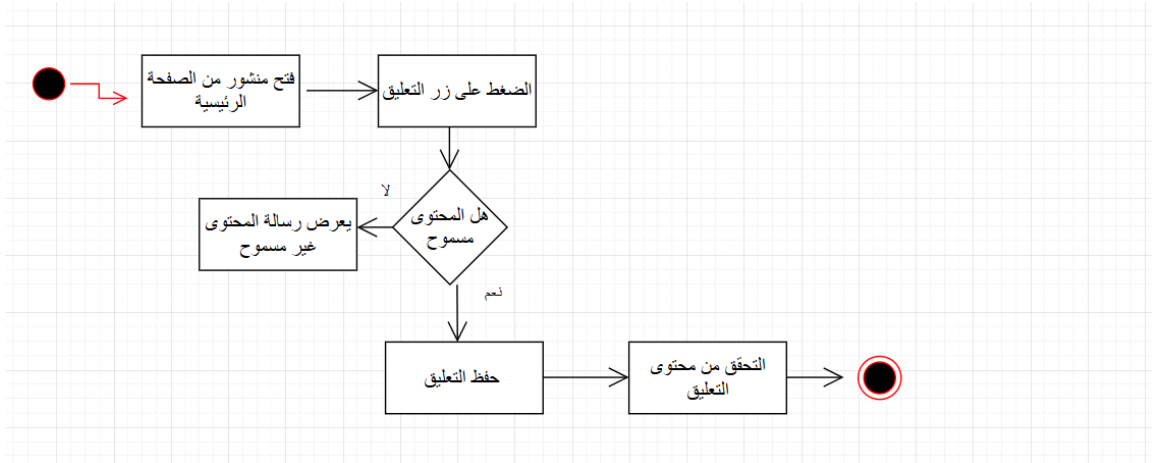
٢٣ مخطط النشاط حالة الإعجاب بالمنشورات

UC-12	Use Case Name
المستخدم	Actors
يسمح للمستخدم بإظهار رأيه تجاه المنشورات عبر إضافة تعليق .	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقا. 2. المنشور موجود ويمكن الوصول إليه للسماح بالتفاعل مع المنشور.	Precondition
١-يفتح المستخدم المنشورة في الصفحة الرئيسية او الملف الشخصي. ٢-يضغط المستخدم زر تعليق . ٣-يتحقق النظام من وجود المنشور وصلاحيه الوصول اليه. ٤-يتحقق النظام من محتوى التعليق ثم يحفظه في قاعدة البيانات. ٥-يعرض النظام التفاعل الجديد على واجهة المستخدم وفي المنشور للآخرين.	Main flow
A3: يعرض النظام رسالة: هذا المنشور غير متاح. A5: النظام يرفض التعليقات ويعرض رسالة "التعليق يحتوي على محتوى غير مسموح به".	Alternative flow

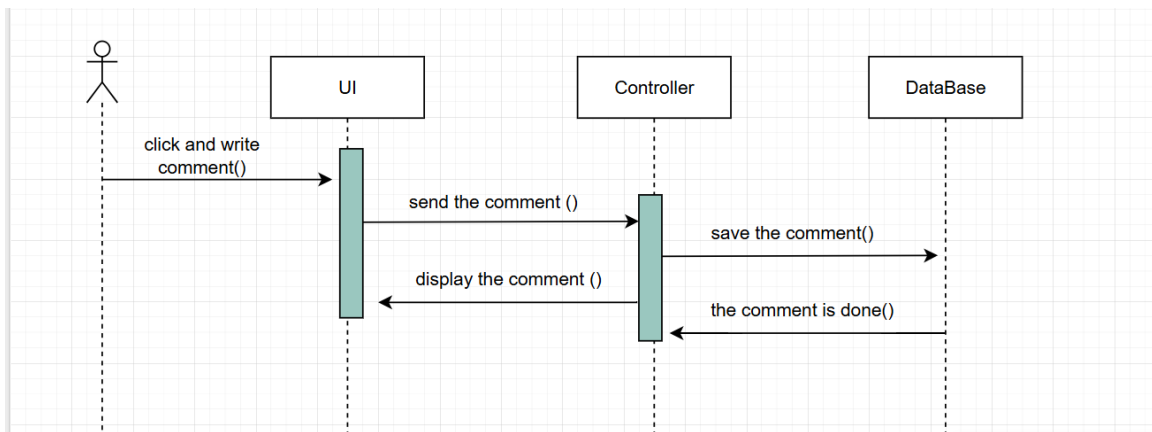
يتم حفظ التعليق في قاعدة البيانات ويظهر علو المنشور بالترتيب الزمني الصحيح.

Postconditions

جدول ١٣ توصيف حالة التعليق على المنشورات



٢٤ مخطط النشاط حالة التعليق على المنشورات



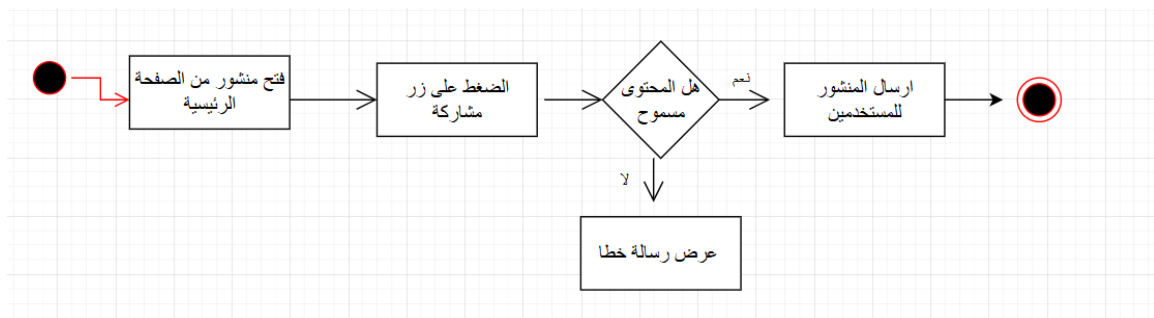
٢٥ مخطط التسلسل حالة التعليق على المنشورات

UC-13

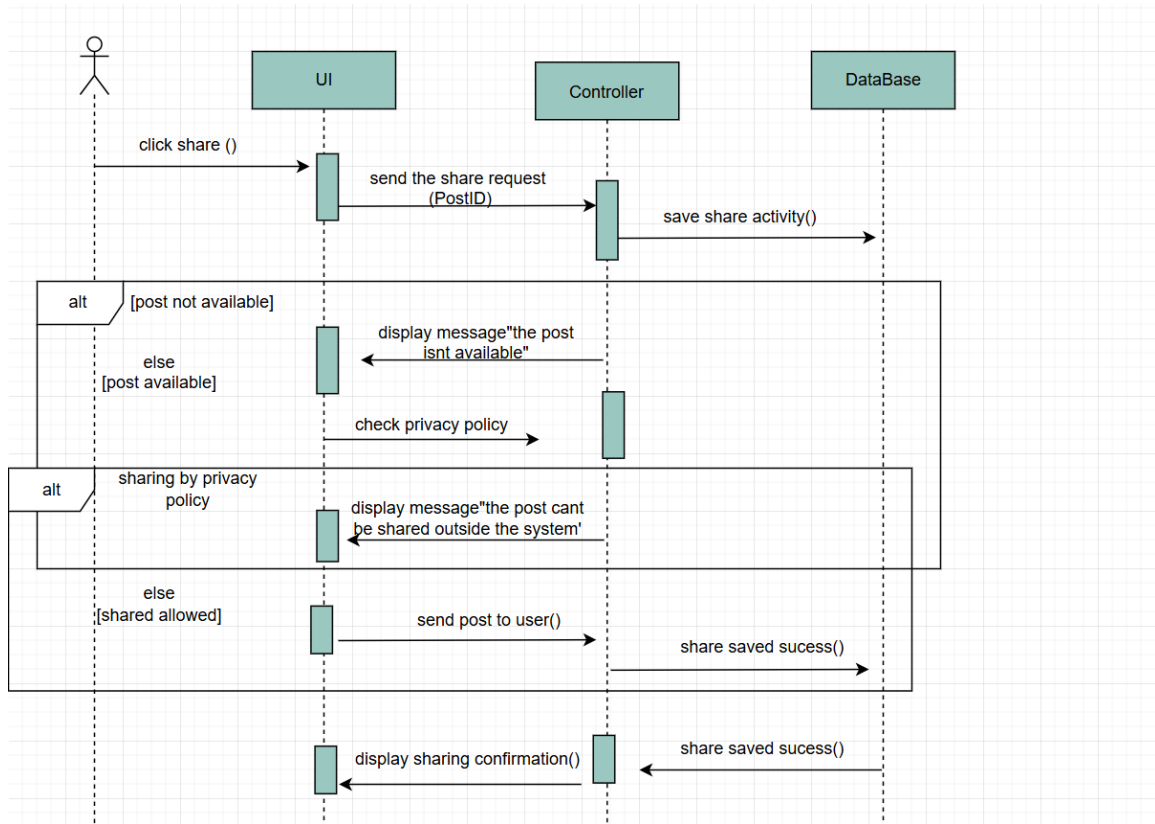
Use Case Name

المستخدم	Actors
يُتيح للمستخدم لتحكم بالمنشورات التي يود مشاركتها.	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقا.	Precondition
١-يفتح المستخدم المنشورة في الصفحة الرئيسية او الملف الشخصي.. ٢-يضغط المستخدم زر للمشاركة . ٣-يتحقق النظام من وجود المنشور وصلاحيه الوصول اليه. ٤-يقوم المستخدم بنشر المنشور للمستخدمين .	Main flow
يعرض النظام رسالة :هذا المنشور غير متاح:A3 إذا كانت سياسة الخصوصية تمنع مشاركة المنشور خارجياً، يظهر النظام :A٥ رسالة: "لا يمكن مشاركة هذا المنشور خارج النظام".	Alternative flow
يتم انشاء رابط للمشاركة او ارسال المنشور للمستخدمين الاخرين وتسجل عملية المشاركة في النظام.	Postconditions

جدول ١٤ توصيف حالة مشاركة المنشورات



٢٦ مخطط النشاط حالة مشاركة المنشورات

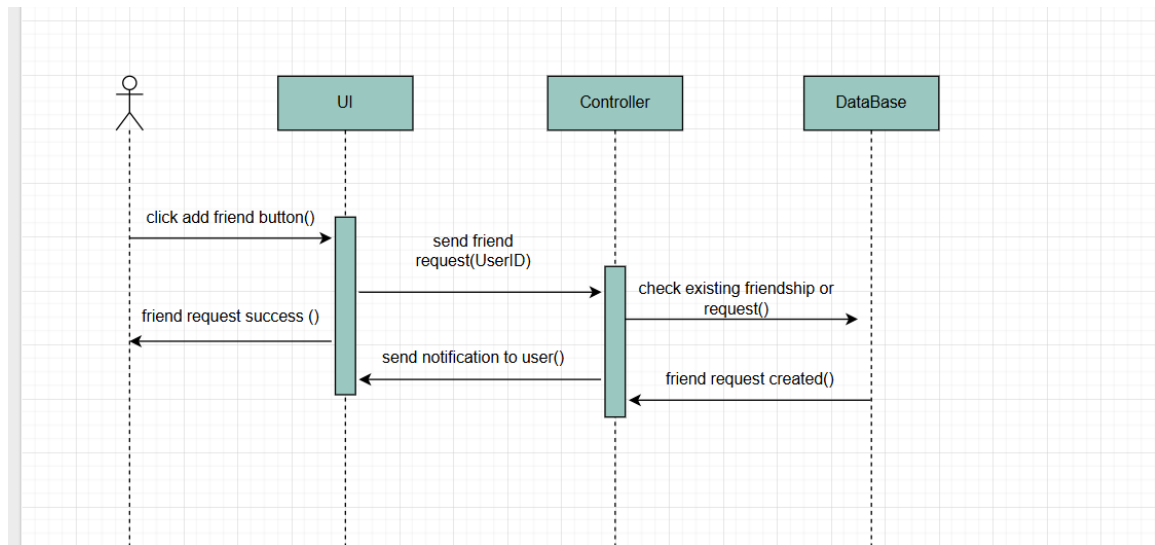


٢٧ مخطط التسلسل حالة مشاركة المنشورات

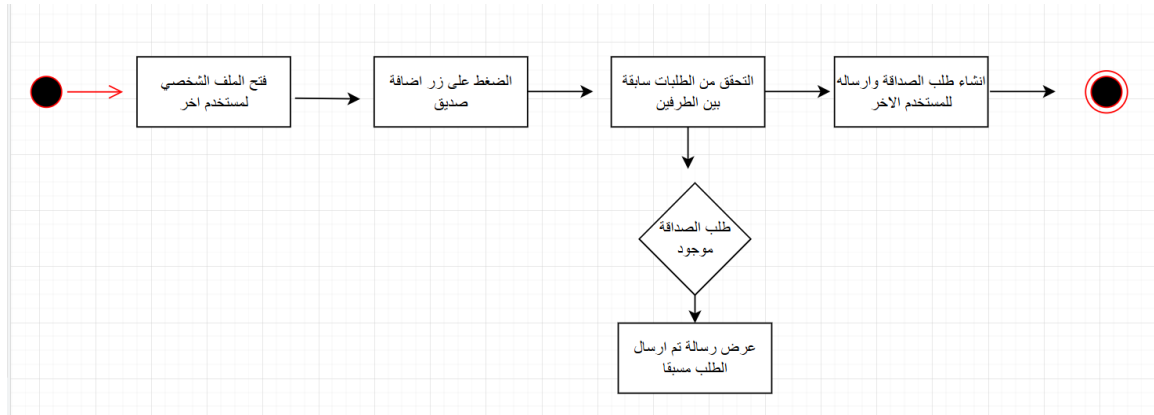
UC-14	Use Case Name
المستخدم	Actors
ارسال طلب صداقة الى المستخدم .	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقا.	Precondition

<p>١-يفتح المستخدم الملف الشخصي لمستخدم اخر.</p> <p>٢-يضغط المستخدم على زر إضافة صديق.</p> <p>٣-النظام يتحقق من عدم وجود صداقة او طلب سابق بين الطرفين.</p> <p>٤-النظام ينشئ طلب صداقة جديد .</p> <p>٥-النظام يرسل اشعارا للمستخدم الاخر بوجود طلب صداقة.</p> <p>٦- النظام يعرض رسالة بان طلب الصداقة قد تم ارساله.</p>	Main flow
طلب الصداقة موجود مسبقا النظام يعرض رسالة تم ارسال طلب مسبقا :A٣	Alternative flow
يتم انشاء طلب صداقة.	Postconditions

جدول ١٥ توصيف حالة إضافة صديق



٢٨ مخطط التسلسل حالة إضافة صديق

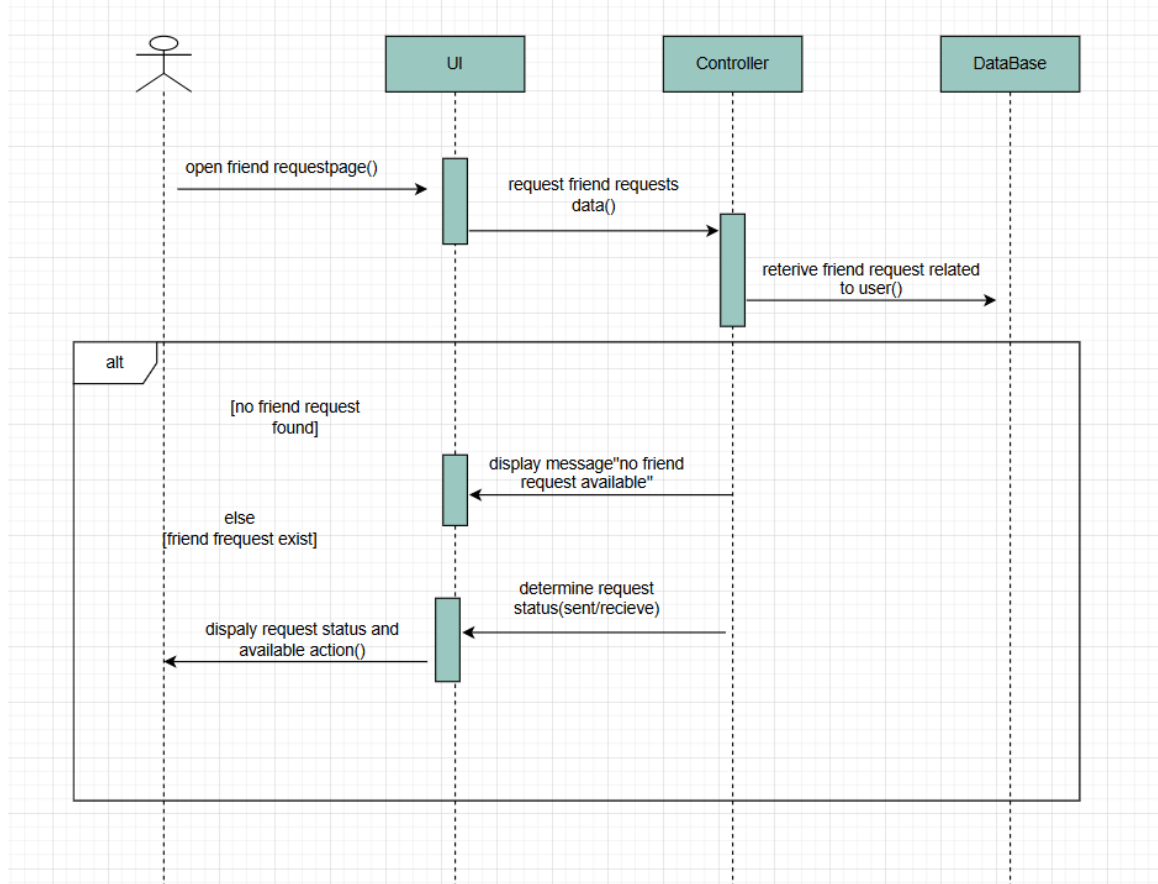


٢٩ مخطط النشاط حالة إضافة صديق

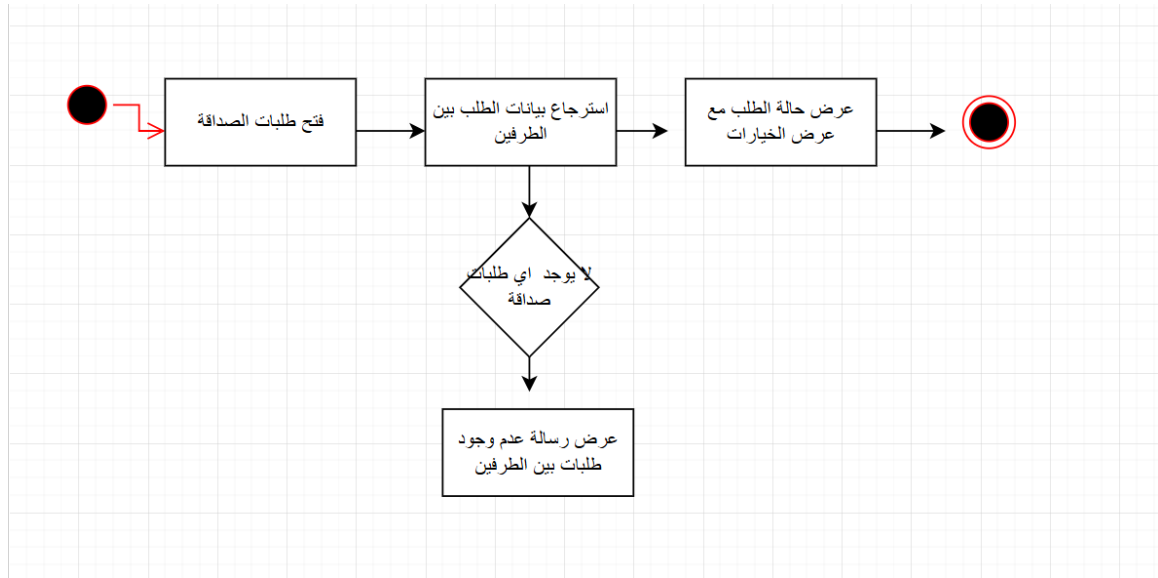
UC-15	Use Case Name
المستخدم	Actors
يُتيح للمستخدم معرفة حالة طلب الصداقة الذي قام بإرساله أو استقبله .	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقا.	Precondition
١. المستخدم يفتح صفحة طلبات الصداقة . ٢. النظام يسترجع بيانات طلب الصداقة المرتبط بين المستخدمين. ٣. النظام يتحقق ويعرض حالة الطلب مع عرض الخيارات المتاحة(الغاء أو قبول أو رفض).	Main flow
لا يوجد أي طلبات صداقة: A٢	Alternative flow

عرض حالة طلب المستخدم.	Postconditions
------------------------	----------------

جدول 16 توصيف معرفة حالة طلب الصداقة



٣٠ مخطط التسلسل معرفة حالة طلب الصداقة

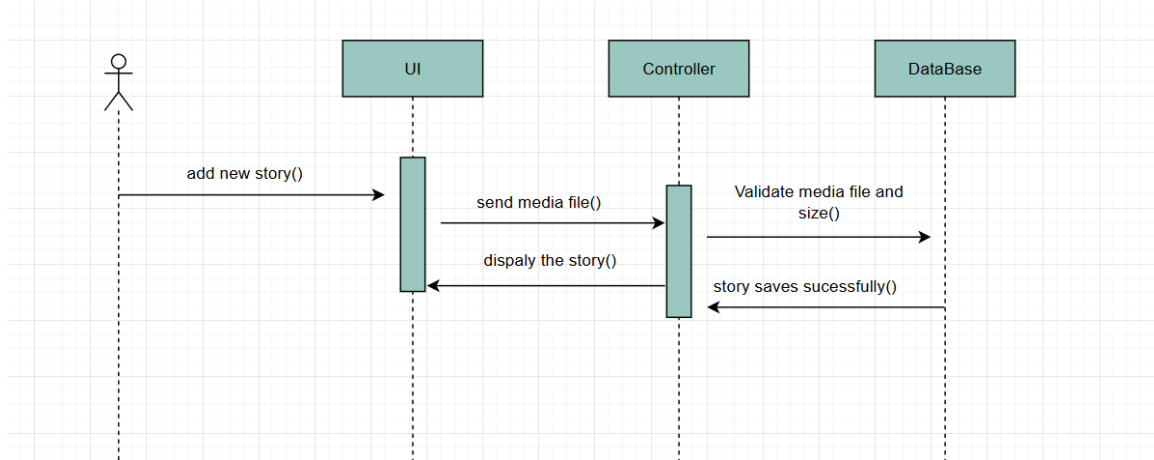


٣١ مخطط النشاط معرفة حالة طلب الصداقة

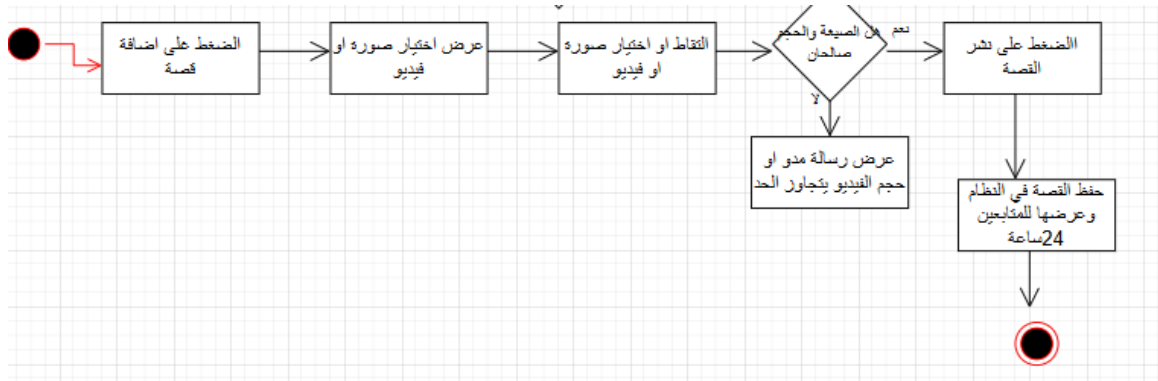
UC-16	Use Case Name
المستخدم	Actors
يُتيح للمستخدم رفع قصة جديدة على حسابه بصيغة صورة أو فيديو قصير، بحيث تكون متاحة للمشاهدة لفترة زمنية محددة (٢٤ ساعة)، مما يعزز التفاعل والمشاركة داخل المنصة.	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقاً.	Precondition

<p>١. المستخدم يضغط على إضافة قصة جديدة.</p> <p>٢. النظام يعرض خيار اختيار صورة أو فيديو قصير.</p> <p>٣. المستخدم يلتقط صورة/فيديو أو يختار من المعرض.</p> <p>٤. النظام يتحقق من صيغة وحجم الملف.</p> <p>٥. المستخدم يضيف (نص، ملصقات، فلاتر – اختياري) ويضغط على نشر القصة.</p> <p>٦. النظام ينشر القصة ويجعلها مرئية لمدة ٢٤ ساعة.</p>	Main flow
<p>حجم الفيديو أكبر من المسموح يعرض رسالة: "مدة أو حجم الفيديو: A٤ يتجاوز الحد المسموح"</p>	Alternative flow
<p>يتم نشر القصة بنجاح وتصبح مرئية للمتابعين أو الأصدقاء حسب إعدادات الخصوصية.</p>	Postconditions

جدول 17 توصيف حالة إضافة قصة جديدة



٣٢ مخطط التسلسل حالة إضافة قصة جديدة

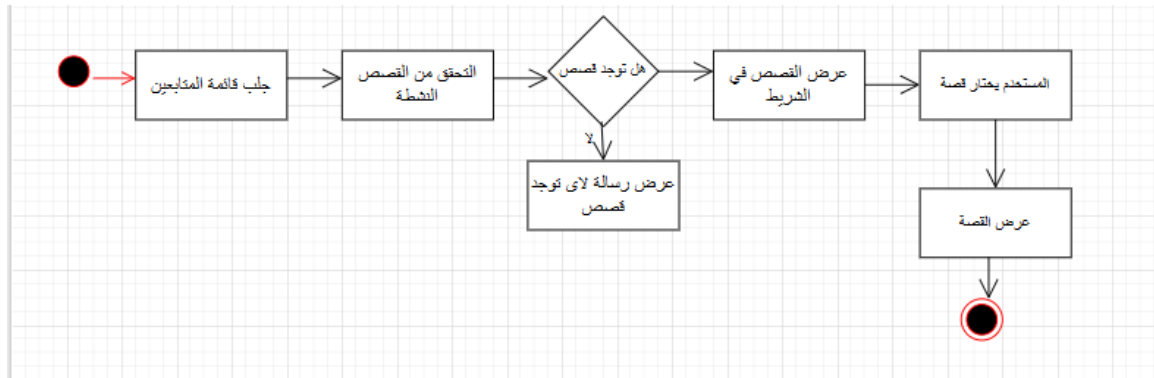


٣٣ مخطط النشاط حالة إضافة قصة جديدة

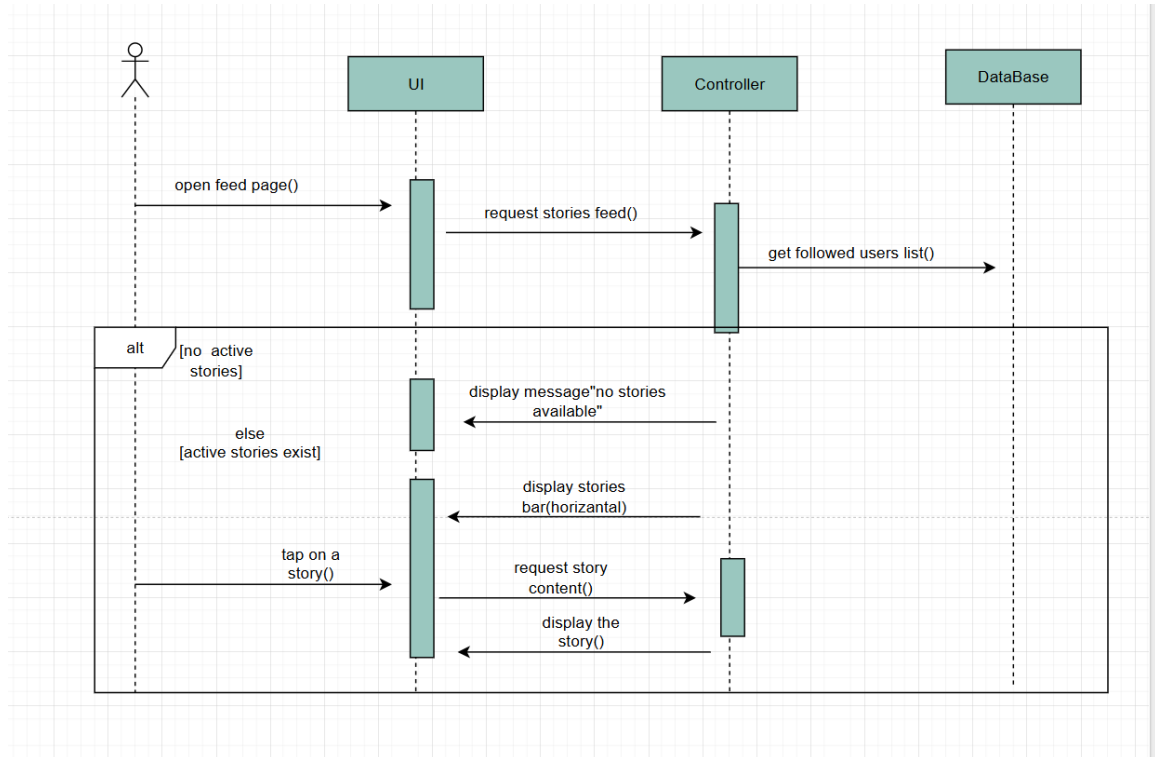
UC-17	Use Case Name
المستخدم	Actors
يتيح للمستخدم عرض القصص المنشورة من قبل المستخدمين الذين يتابعهم فقط، وذلك وفق إعدادات الخصوصية ومدة صلاحية القصة، مما يضمن تجربة مشاهدة مخصصة وأمنة.	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقاً.	Precondition
١. المستخدم يدخل إلى الصفحة الرئيسية. ٢. النظام يجلب قائمة المستخدمين الذين يتابعهم المستخدم و يتحقق من وجود قصص نشطة. ٣. النظام يعرض القصص في شريط علوي بشكل أفقي. ٤. المستخدم يضغط على قصة أحد المتابعين. ٥. النظام يعرض القصة (صورة أو فيديو). ٦. بعد انتهاء القصة، النظام ينتقل تلقائياً إلى القصة التالية.	Main flow

<p>لا توجد قصص نشطة: A2</p> <p>القصة محجوبة بالخصوصية القصة غير مسموح بعرضها لهذا المستخدم: A2</p> <p>انتهاء مدة القصة أثناء المشاهدة وتنتهي مدة صلاحية القصة: A6</p>	Alternative flow
<p>يتم عرض القصص المتاحة للمتابعين بترتيب زمني.</p>	Postconditions

جدول 18 توصيف عرض القصص المتاحة للمتابعين



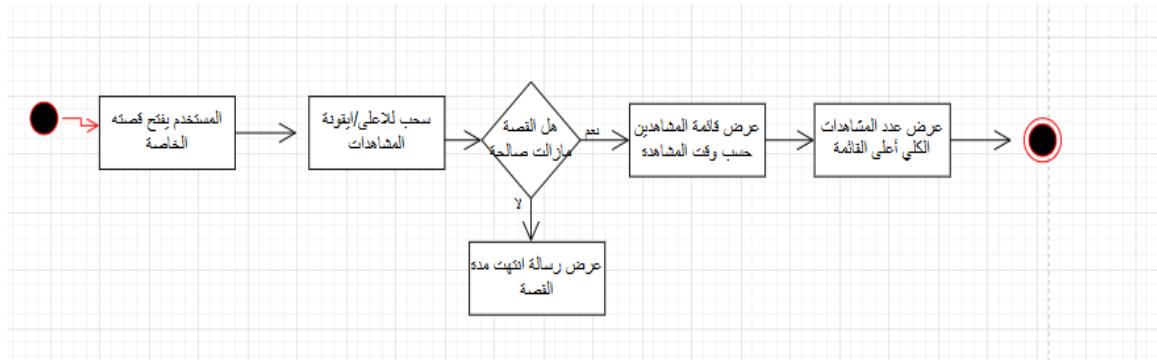
٣٤ مخطط النشاط حالة عرض القصص المتاحة للمتابعين



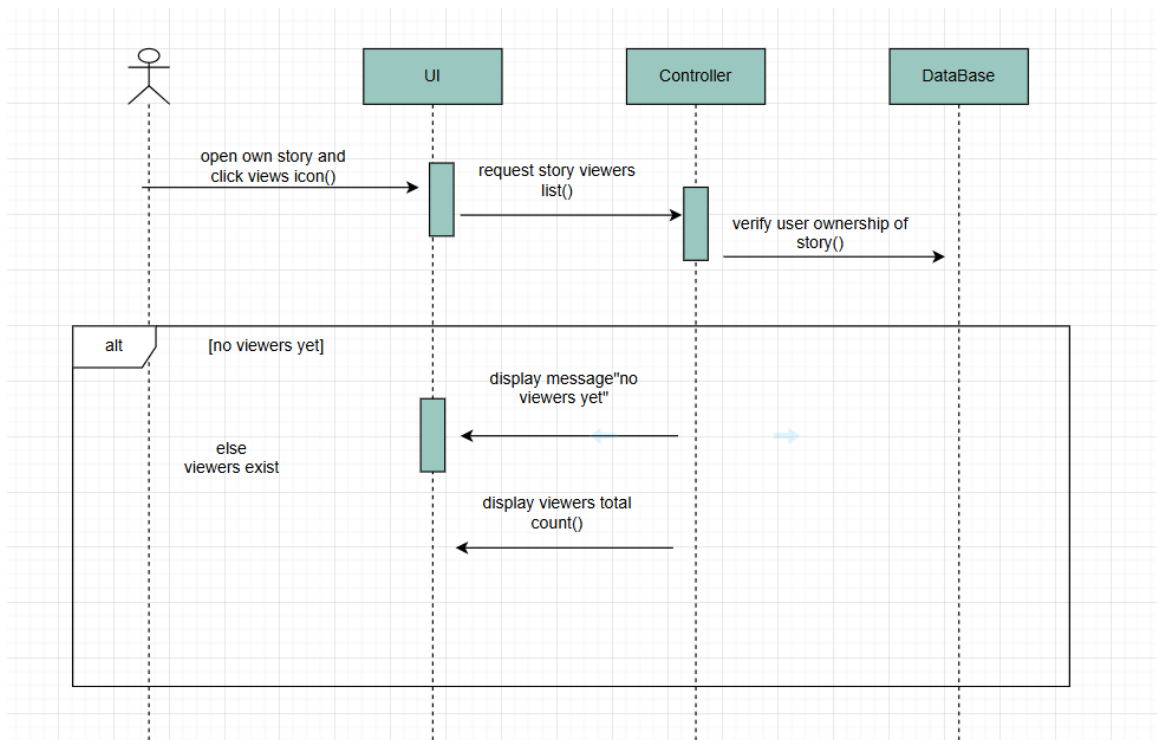
٣٥ مخطط التسلسل حالة عرض القصص المتاحة للمتابعين

UC-18	Use Case Name
المستخدم	Actors
يُتيح لصاحب القصة عرض قائمة المستخدمين الذين قاموا بمشاهدة قصته، مما يوفر شفافية حول التفاعل مع المحتوى المؤقت داخل المنصة.	Description
1. تسجيل دخول المستخدم إلى حسابه مسبقاً.	Precondition
<p>١. المستخدم يفتح قصته الخاصة.</p> <p>٢. المستخدم يسحب للأعلى أو يضغط على أيقونة المشاهدين.</p> <p>٣. النظام يتحقق من صلاحية المستخدم لعرض القائمة.</p> <p>٤. النظام يعرض قائمة المشاهدين مرتبة حسب وقت المشاهدة.</p> <p>٥. النظام يعرض عدد المشاهدات الكلي أعلى القائمة.</p>	Main flow
<p>A1: انتهاء مدة القصة (٢٤ ساعة).</p> <p>A٢: لا يوجد مشاهدون لا توجد أي مشاهدة بعد.</p>	Alternative flow
يتم عرض قائمة المشاهدين بشكل صحيح مع وقت المشاهدة	Postconditions

جدول 19 توصيف عرض قائمة المشاهدين



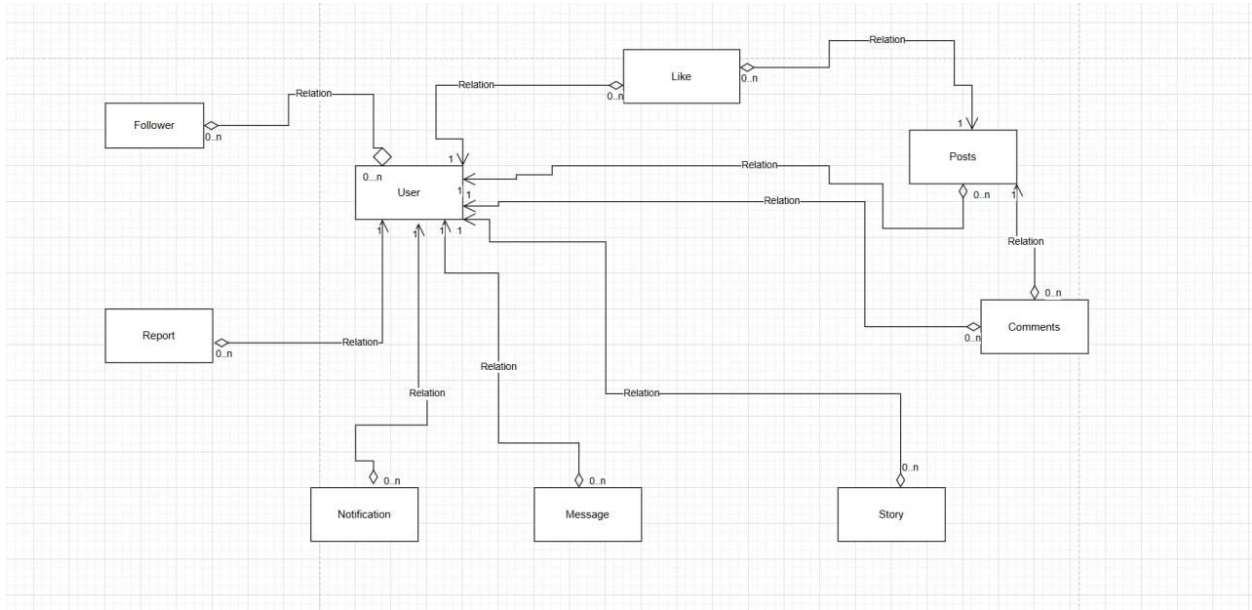
٣٦ مخطط النشاط عرض قائمة المشاهدين



٣٧ مخطط النشاط عرض قائمة المشاهدين

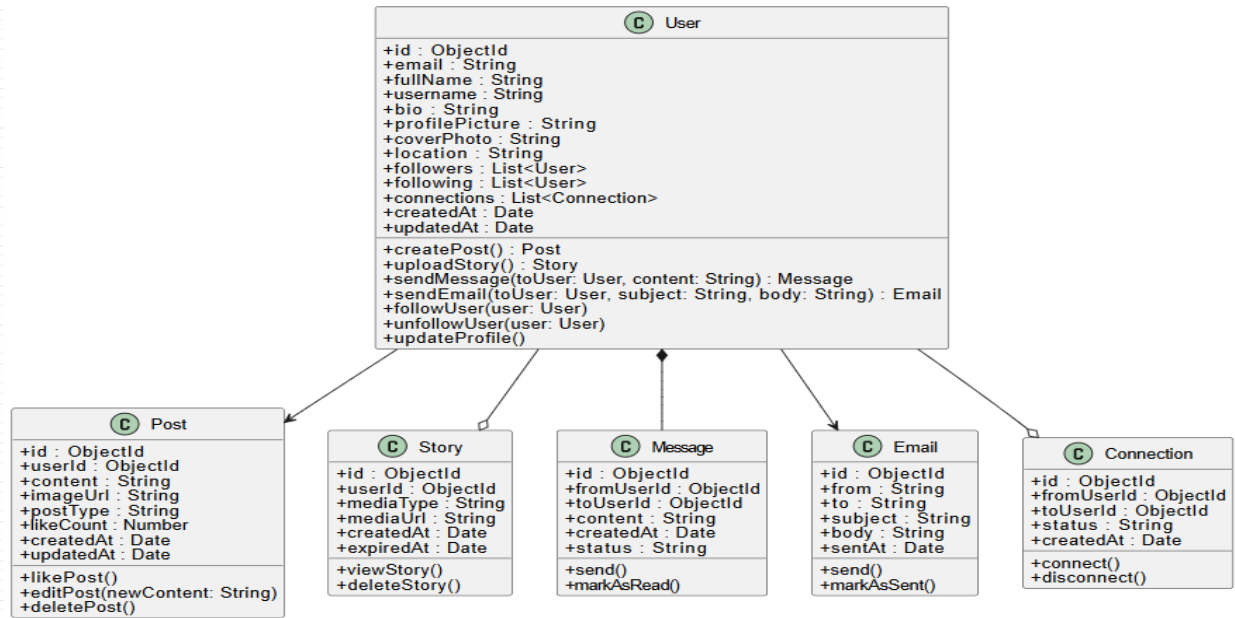
٣. الفصل الثالث : الدراسة التصميمية

١.٣ مخطط قاعدة البيانات ERD



٣٨ مخطط ال ERD

٢.٣ مخطط الصفوف Class Diagram



٣٩ مخطط ال Class Diagram

٣.٣ جدول Test Cases

Test Case ID	Related Req ID	Test Case Description	Preconditions	Test Steps	Expected Result
TC-01	ID_01	User registration using email or phone	User not registered	Open app → Register → Enter email/phone → Submit	Account created successfully
TC-02	ID_02	Login using Google/GitHub	User has valid OAuth account	Open app → Login → Select Google/GitHub	User logged in successfully

TC-03	ID_03	Reset forgotten password	User registered	Click Forgot Password → Enter email	Password reset link sent
TC-04	ID_04	Logout from current session	User logged in	Click Logout	User session terminated
TC-05	ID_05	View user profile information	User logged in	Open Profile Page	Profile details displayed
TC-06	ID_06	Edit and update profile	User logged in	Edit profile → Save changes	Profile updated successfully
TC-07	ID_07	View other users' profiles	User logged in	Search user → Open profile	Other user profile displayed
TC-08	ID_08	Display online/offline status	Multiple users exist	View users list	Correct status shown
TC-09	ID_09	Create new post	User logged in	Create post → Publish	Post published
TC-10	ID_10	Edit or delete post	Post exists	Edit/Delete selected post	Post updated or deleted
TC-11	ID_11	View feed sorted chronologically	Multiple posts exist	Open feed	Posts ordered by time
TC-12	ID_12	Like a post	Post exists	Click Like	Like registered
TC-13	ID_13	Save or share post	Post exists	Click Save/Share	Post saved/shared
TC-14	ID_14	Report inappropriate post	Post violates policy	Click Report	Report submitted
TC-15	ID_15	Upload story (image/video)	User logged in	Upload story	Story uploaded
TC-16	ID_16	View stories from followed users	Stories available	Open stories section	Stories displayed

TC-17	ID_17	Auto delete story after 24 hours	Story older than 24h	System auto process	Story removed
TC-18	ID_18	View story viewers list	Story has viewers	Open viewers list	Viewers displayed

جدول ٢٠ ال Test Cases

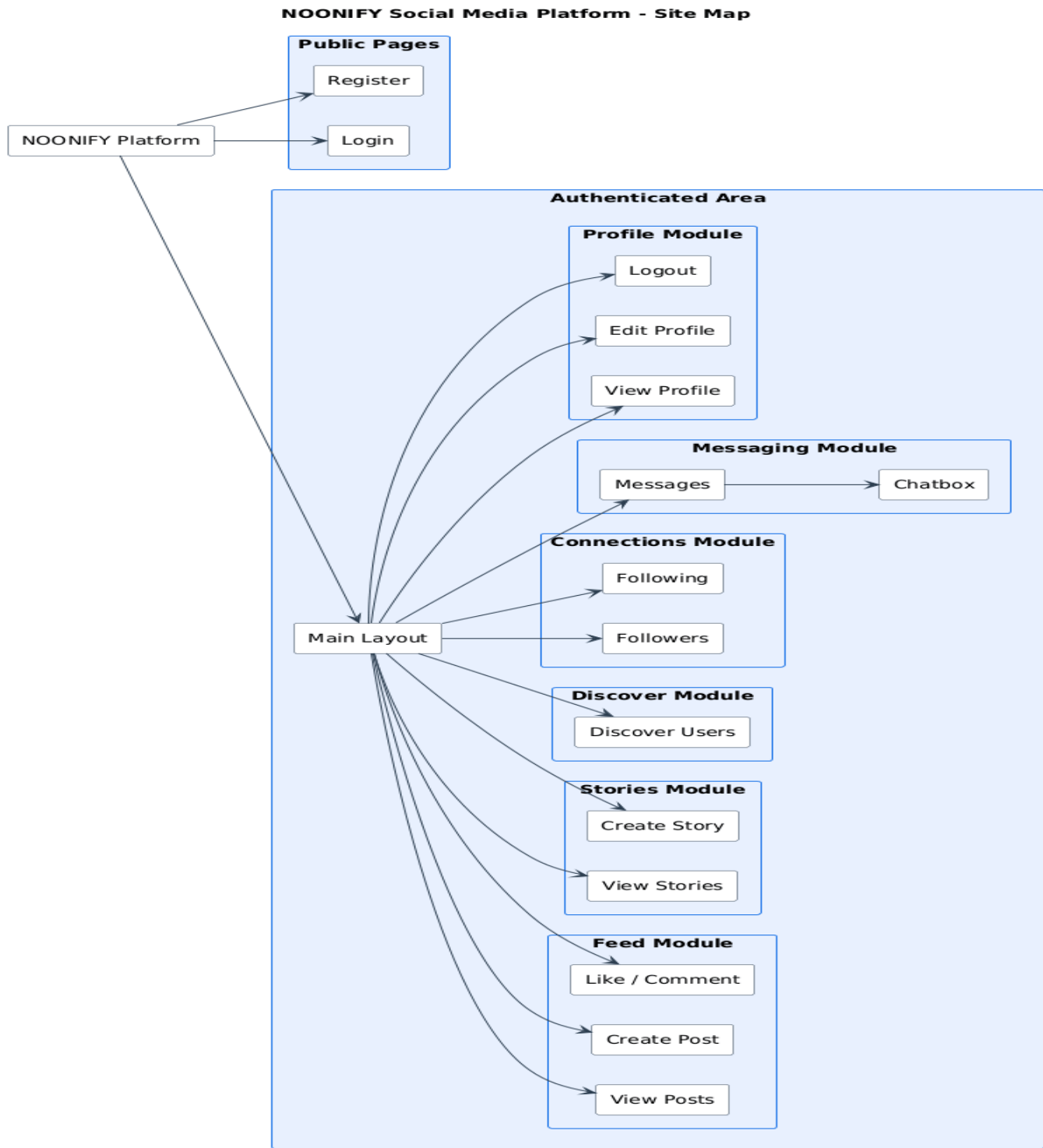
٤.٣ جدول RTM (Requirements Traceability Matrix)

Req ID	Title and Dependency	SRS Section	Coding Module	System Test Case
FR-01	Register New Account – depends on Auth module	3.1	auth_service, user_model	TC-01
FR-02	Login using Email/Google/GitHub – depends on Auth module	3.2	auth_service, oauth_service	TC-02
FR-03	Reset Password – depends on Auth module	3.3	password_service	TC-03
FR-04	Logout – depends on Session manager	3.4	session_manager	TC-04
FR-05	View User Profile – depends on Profile module	3.5	profile_service	TC-05
FR-06	Edit User Profile – depends on Profile module	3.6	profile_service	TC-06
FR-07	View Other Users Profiles – depends on Profile module	3.7	profile_service	TC-07
FR-08	Show Online Status – depends on Presence module	3.8	presence_service	TC-08

FR-09	Create Post – depends on Post module	3.9	post_service	TC-09
FR-10	Edit/Delete Post – depends on Post module	3.10	post_service	TC-10
FR-11	View Feed Chronologically – depends on Feed module	3.11	feed_service	TC-11
FR-12	Like Post – depends on Interaction module	3.12	interaction_service	TC-12
FR-13	Save/Share Post – depends on Interaction module	3.13	interaction_service	TC-13
FR-14	Report Post – depends on Moderation module	3.14	moderation_service	TC-14
FR-15	Upload Story – depends on Story module	3.15	story_service	TC-15
FR-16	View Stories – depends on Story module	3.16	story_service	TC-16
FR-17	Delete Story after 24h – depends on Story module	3.17	story_service, scheduler	TC-17
FR-18	View Story Viewers – depends on Story module	3.18	story_service	TC-18

RTM ال 21 جدول

٥.٣ مخطط ال Site Map



٤٠ مخطط ال Site Map

٤. الفصل الرابع : تنفيذ المشروع

١.٤ الأدوات المستخدمة

يعرض هذا الفصل الأدوات والتقنيات التي تم الاعتماد عليها في تطوير منصة التواصل الاجتماعي **Noonify**، حيث تم تقسيم الأدوات إلى ثلاث فئات رئيسية: أدوات الواجهة الأمامية (Front-End) ، وأدوات الواجهة الخلفية (Back-End) ، وأدوات قاعدة البيانات (Database) ، وذلك بما يتوافق مع معمارية **MERN Stack**.

١.١.٤ أدوات الواجهة الأمامية (Front-End Tools)

تم تطوير الواجهة الأمامية للتطبيق باستخدام مجموعة من الأدوات الحديثة التي تهدف إلى بناء واجهة مستخدم تفاعلية وسريعة الاستجابة، ومن أبرزها:

• **React.js**

مكتبة JavaScript تُستخدم لبناء واجهات المستخدم المعتمدة على المكونات (Component-Based Architecture)، مما يسهل إعادة استخدام الكود وتحسين قابلية الصيانة.

• **Vite**

أداة بناء (Build Tool) حديثة تُستخدم لتسريع عملية التطوير وتشغيل المشروع بكفاءة أعلى مقارنة بالأدوات التقليدية.

• **JavaScript (ES6+)**

لغة البرمجة الأساسية المستخدمة في تطوير منطق الواجهة الأمامية والتعامل مع الأحداث والتفاعل مع المستخدم.

• **HTML5**

تُستخدم لبناء الهيكل العام لصفحات الويب.

• CSS3

تُستخدم لتنسيق واجهات المستخدم وتحسين التجربة البصرية.

• Axios

مكتبة تُستخدم لإرسال واستقبال طلبات HTTP بين الواجهة الأمامية والواجهة الخلفية.

• ESLint

أداة لتحليل الكود واكتشاف الأخطاء البرمجية وتحسين جودة الكود.

• Vercel

منصة تُستخدم لنشر تطبيق الواجهة الأمامية وتشغيله على بيئة إنتاجية.

٢.١.٤ أدوات الواجهة الخلفية (Back-End Tools)

تم بناء الواجهة الخلفية للتطبيق باستخدام بيئة تشغيل وتقنيات تضمن الأداء العالي وتنظيم منطق النظام، ومن أهم هذه الأدوات:

• Node.js

بيئة تشغيل تعتمد على JavaScript تُستخدم لتنفيذ كود الخادم (Server-Side) بكفاءة عالية.

• Express.js

إطار عمل (Framework) مبني على Node.js يُستخدم لإنشاء واجهات برمجية (RESTful APIs) وتنظيم المسارات (Routes).

• JavaScript

تُستخدم كلغة أساسية لتطوير منطق الخادم.

• JWT (JSON Web Token)

تُستخدم لتنفيذ نظام المصادقة (Authentication) والتحقق من هوية المستخدم.

- **Multer**

مكتبة تُستخدم لمعالجة ورفع الملفات (مثل الصور) من المستخدم إلى الخادم.

- **NodeMailer**

مكتبة تُستخدم لإرسال رسائل البريد الإلكتروني، مثل رسائل التحقق أو الإشعارات.

- **ImageKit**

خدمة تُستخدم لإدارة الصور ورفعها وتخزينها بشكل سحابي.

- **Middleware**

تُستخدم لمعالجة الطلبات الوسيطة مثل التحقق من الصلاحيات وتأمين المسارات.

٣.١.٤ أدوات قاعدة البيانات (Database Tools)

تم استخدام قاعدة بيانات مرنة وقابلة للتوسع لتخزين بيانات المستخدمين والمحتوى، وتشمل الأدوات التالية:

- **MongoDB**

قاعدة بيانات NoSQL تعتمد على المستندات (Documents) وتتميز بالمرونة وسهولة التعامل مع البيانات غير المهيكلة.

- **Mongoose**

مكتبة تُستخدم كنظام نمذجة (ODM) للتعامل مع MongoDB ، حيث تُسهل إنشاء النماذج (Schemas) وتنفيذ العمليات على البيانات.

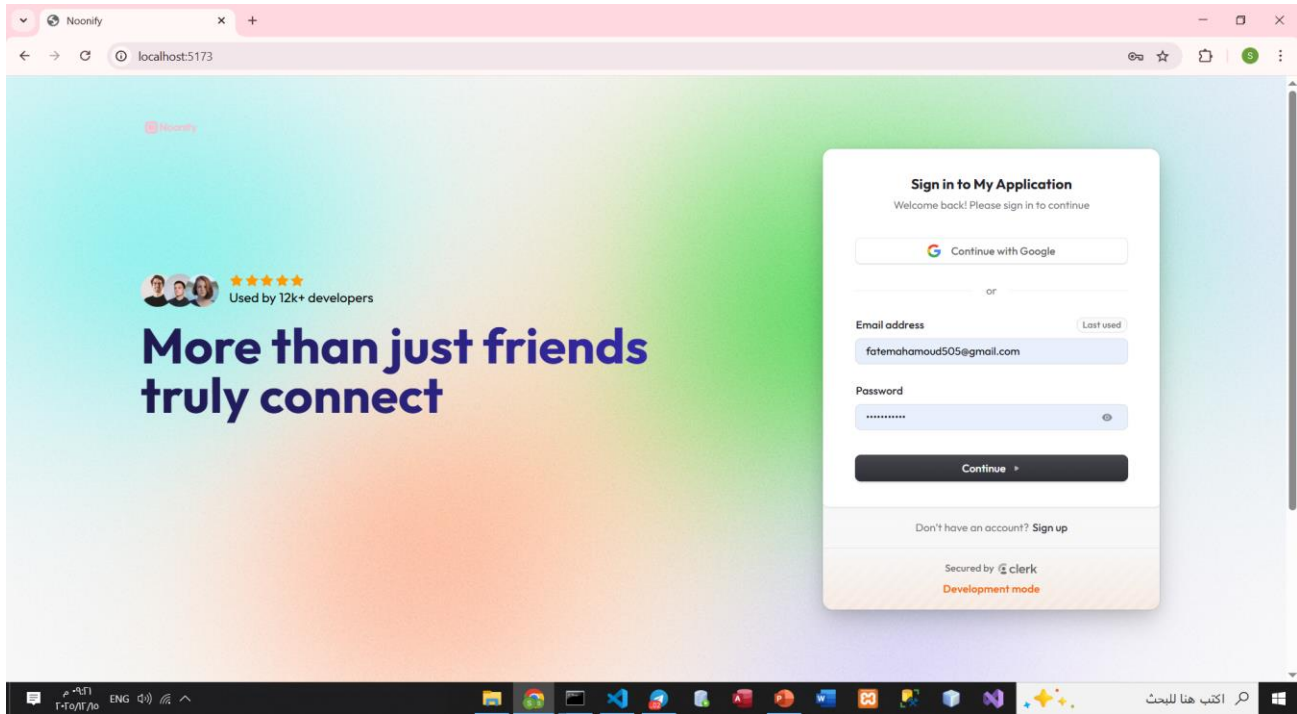
٤.١.٤ ملخص الأدوات المستخدمة

اعتمد المشروع على **MERN Stack** كمعمارية رئيسية، حيث تم تحقيق التكامل بين الواجهة الأمامية، الواجهة الخلفية، وقاعدة البيانات، مما ساهم في بناء منصة تواصل اجتماعي حديثة، منظمة، وقابلة للتطوير، وتلبي المتطلبات الأكاديمية والعملية للمشروع.

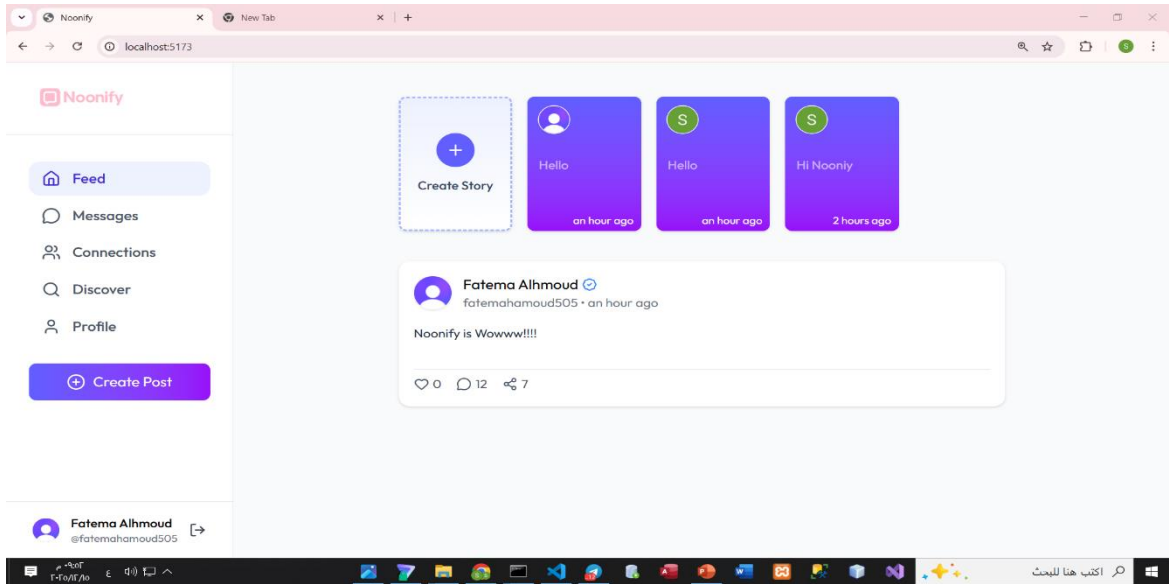
٢.٤ الواجهات

The screenshot displays a web browser window with a single tab titled 'My account | My Application'. The address bar shows the URL 'light-walrus-75.accounts.dev/sign-up'. Below the browser window, a 'Create your account' form is visible. The form has a heading 'Create your account' and a sub-heading 'Welcome! Please fill in the details to get started.' It features a 'Continue with Google' button, followed by a separator line and the word 'or'. Below this, there are input fields for 'First name' (containing 'Sara') and 'Last name' (containing 'Eyad'), both marked as 'Optional'. There is also an 'Email address' field containing 'saraob771@gmail.com' and a 'Password' field with a masked password '*****' and a green checkmark indicating it meets requirements. A 'Continue' button is at the bottom of the form. At the very bottom of the browser window, there is a link 'Already have an account? Sign in'.

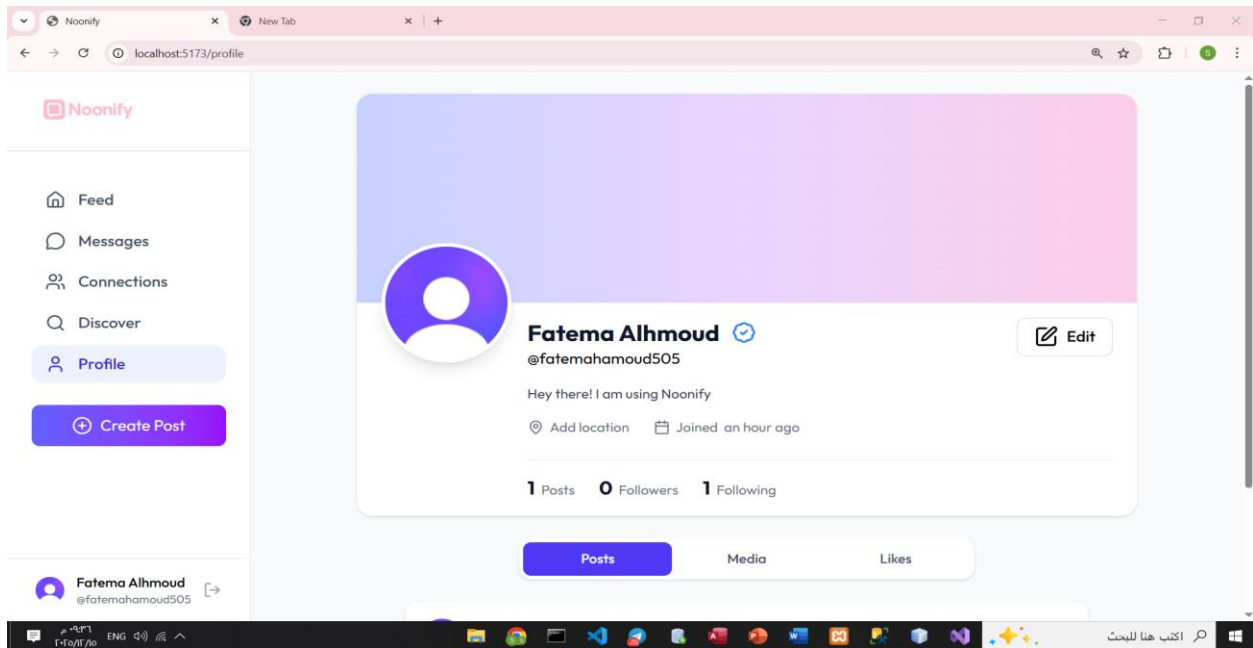
١٤ الشكل (١, ١) واجهة تسجيل حساب جديد



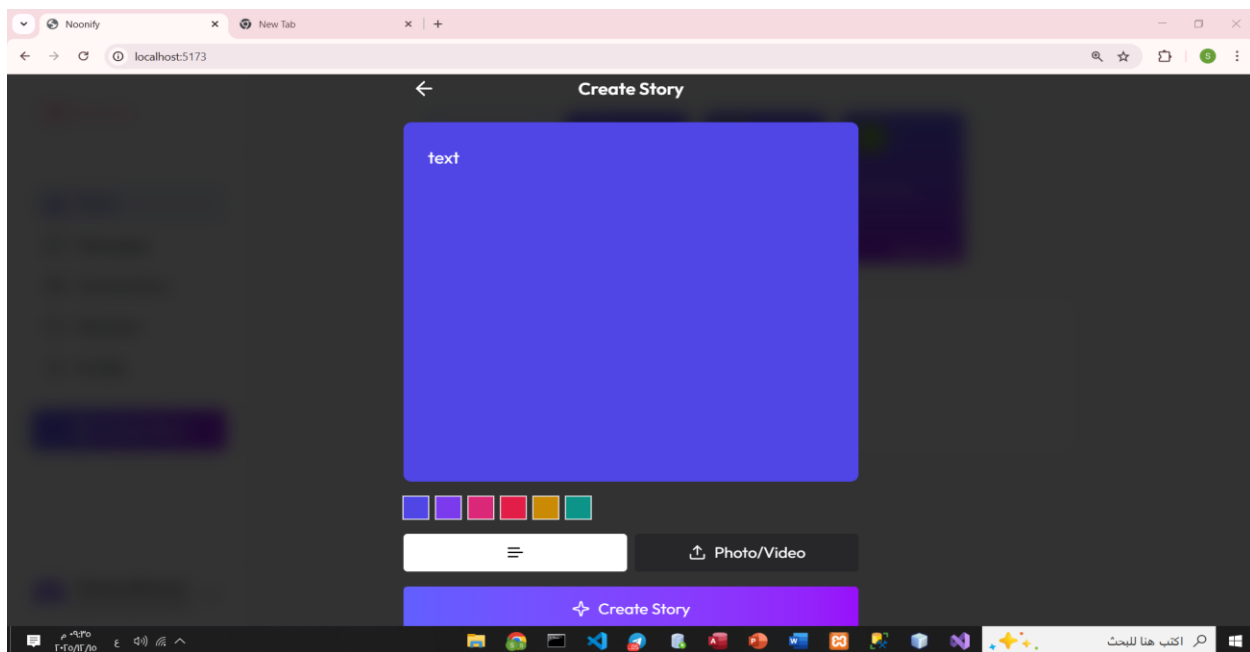
٢٤ الشكل (١، ٢) واجهة تسجيل الدخول



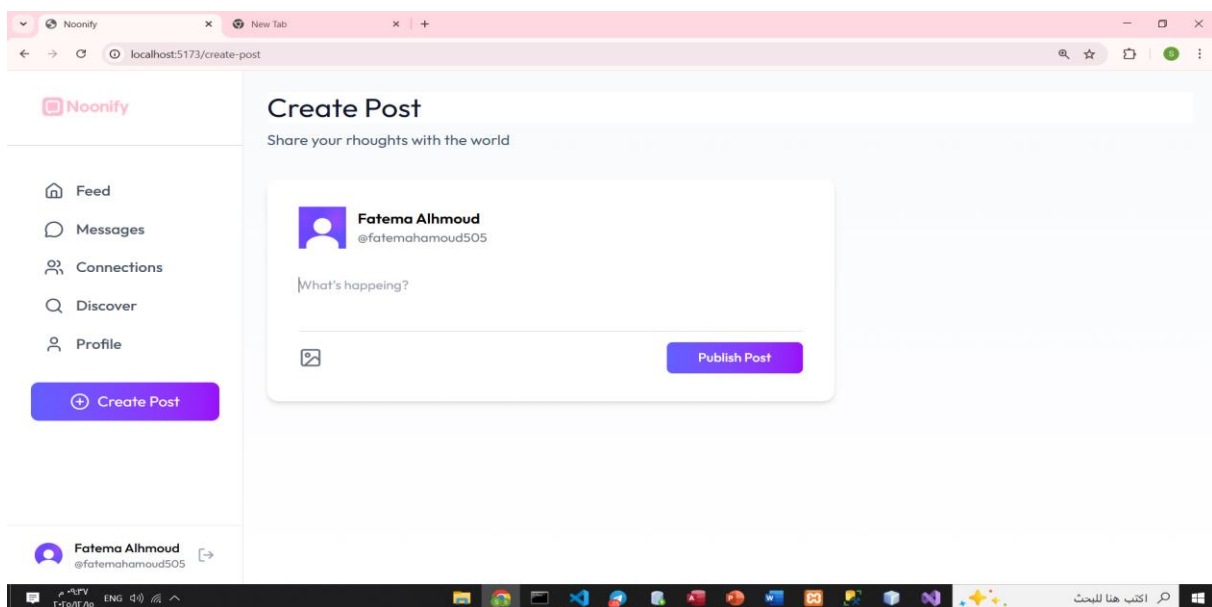
43 الشكل (١,٣) زر تسجيل الخروج



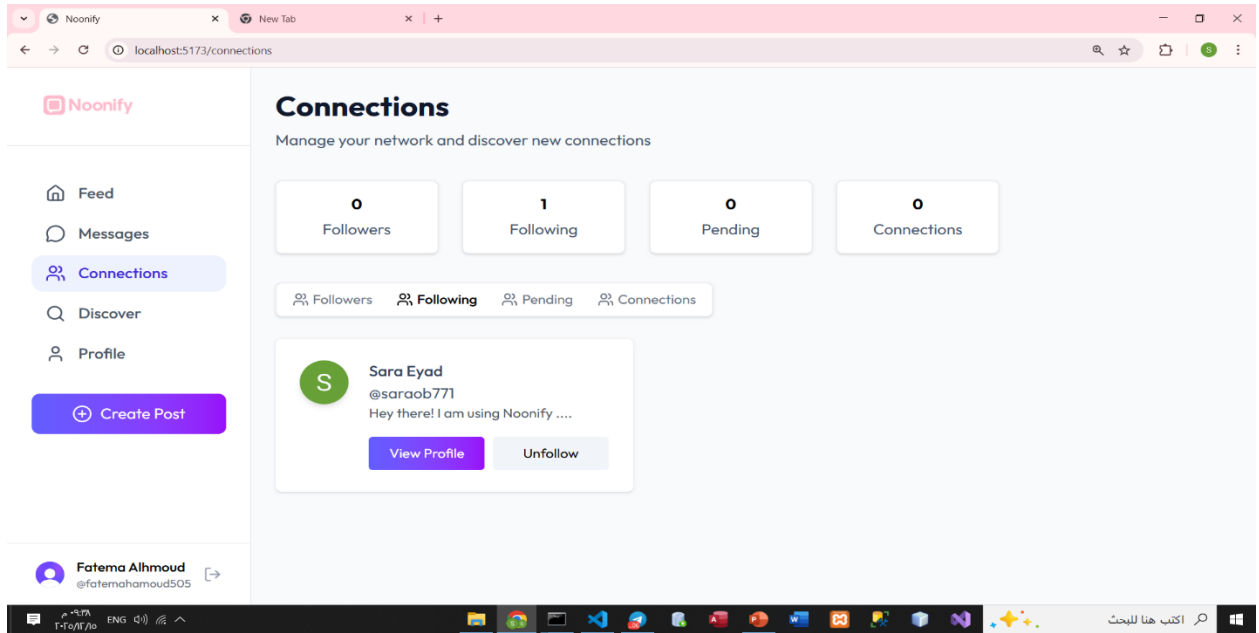
44 الشكل (١,٤) عرض المعلومات الأساسية للمستخدم



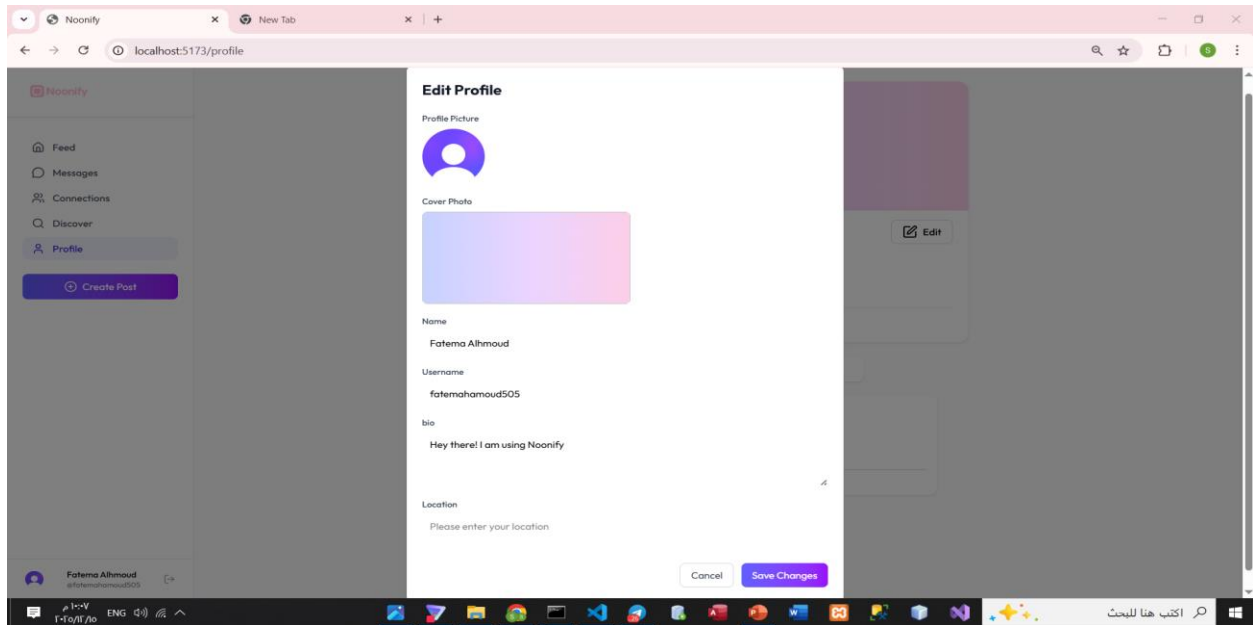
٥٤ الشكل (١,٥) رفع قصة جديدة



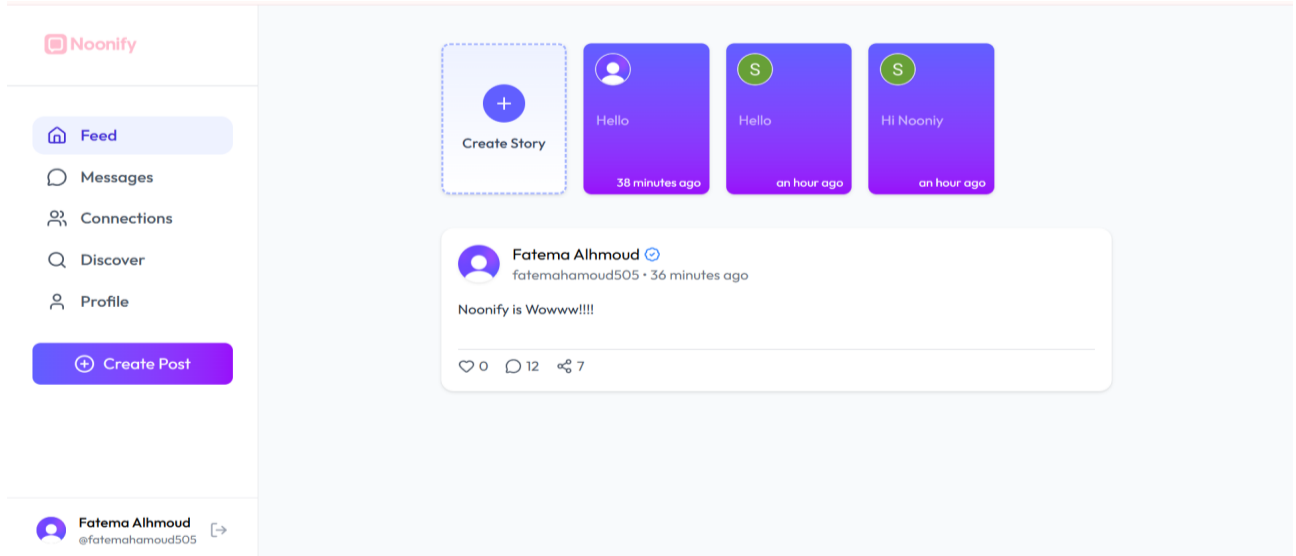
٥٦ الشكل (١,٦) إنشاء منشورات جديد



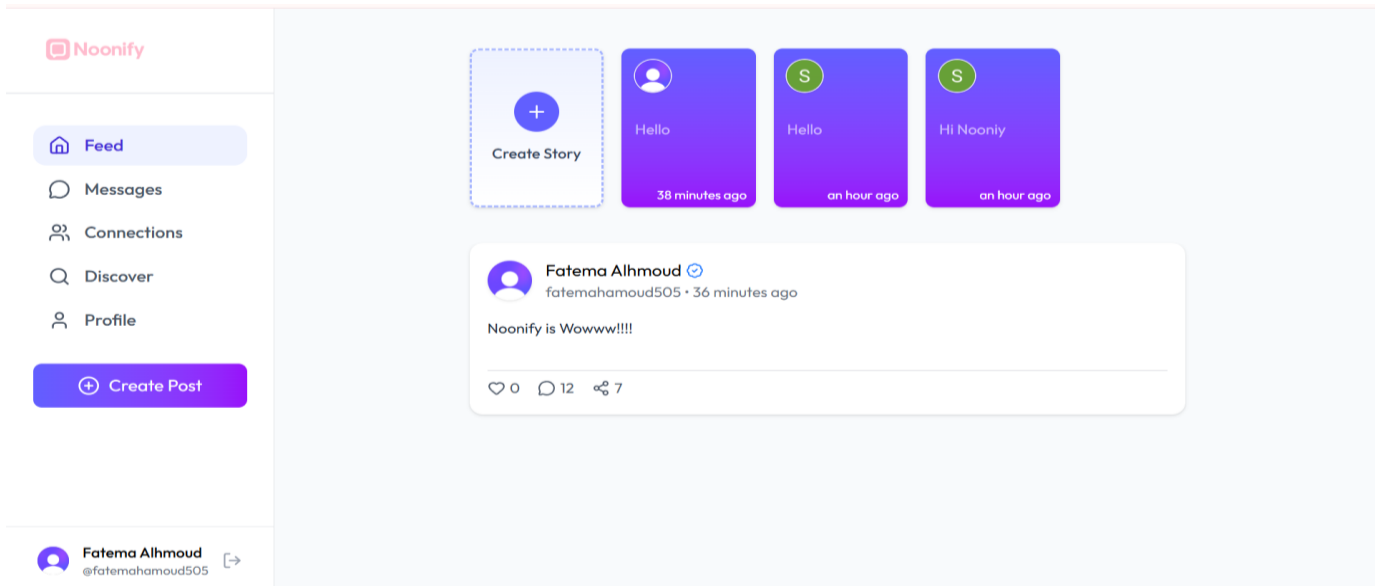
47 الشكل (١,٧) عرض ملفات المستخدمين الآخرين



48 الشكل (١,٨) تعديل الملف الشخصي



٩ ء الشكل (٩, ١) عرض القصص



٥٠ الشكل (١٠, ١) عرض منشورات الآخرين

٣.٤ الخاتمة

في ختام هذا المشروع، تم تصميم وتطوير منصة تواصل اجتماعي متكاملة تحت اسم **NOONIFY**، تهدف إلى محاكاة الخصائص الأساسية لشبكات التواصل الاجتماعي الحديثة مع مراعاة الجوانب الأكاديمية والعملية في آن واحد. اعتمد المشروع على معمارية **MERN Stack** التي تجمع بين الواجهة الأمامية التفاعلية باستخدام **React.js**، والواجهة الخلفية المبنية باستخدام **Node.js** و **Express.js**، إضافة إلى قاعدة البيانات **MongoDB**، مما ساهم في بناء نظام مرن، منظم، وقابل للتوسع.

نجح المشروع في تحقيق الأهداف المحددة له، حيث أتاح للمستخدمين إنشاء حساباتهم الشخصية، إدارة ملفاتهم الشخصية، مشاركة المحتوى النصي والمرئي، التفاعل مع المنشورات، استخدام نظام القصص، إضافة إلى التواصل عبر الرسائل الخاصة. كما تم تطبيق مفاهيم معمارية مهمة مثل **Client-Server Architecture** و **MVC Architecture**، الأمر الذي ساعد على تنظيم الكود، تحسين قابلية الصيانة، وتسهيل تطوير الميزات المستقبلية.

علاوةً على ذلك، ساهم هذا المشروع في تعزيز المهارات العملية في مجالات تطوير تطبيقات الويب الحديثة، العمل ضمن منهجية **Scrum**، وتحليل المتطلبات، مما جعله نموذجًا تطبيقيًا يجمع بين الجانبين النظري والعملية في هندسة البرمجيات ونظم المعلومات.

٤.٤ الآفاق المستقبلية (Future Work)

على الرغم من تحقيق المشروع لأهدافه الأساسية، إلا أن هناك العديد من التحسينات والتوسعات التي يمكن تنفيذها مستقبلاً لرفع كفاءة المنصة وتوسيع نطاق استخدامها، ومن أبرز هذه الآفاق:

١. تعزيز الأمان
إضافة تقنيات أمان متقدمة مثل المصادقة الثنائية (Two-Factor Authentication) وتحسين سياسات التشفير لحماية بيانات المستخدمين.
٢. تحسين الأداء وقابلية التوسع
دعم تقنيات التخزين المؤقت (Caching) واستخدام خدمات سحابية متقدمة لتمكين المنصة من التعامل مع عدد أكبر من المستخدمين.
٣. إضافة الذكاء الاصطناعي
دمج خوارزميات التوصية لاقتراح الأصدقاء والمحتوى المناسب بناءً على اهتمامات المستخدم.
٤. تطوير تطبيقات الهواتف الذكية
إنشاء تطبيقات مخصصة لأنظمة **Android** و **iOS** لتوفير تجربة استخدام أكثر سلاسة.
٥. توسيع خصائص التفاعل الاجتماعي
دعم البث المباشر (Live Streaming)، الإشعارات الفورية (Push Notifications)، وتحسين نظام التعليقات والتفاعلات.
٦. تحسين تجربة المستخدم (UX/UI)
إجراء دراسات استخدام حقيقية لتحسين تصميم الواجهات وجعلها أكثر سهولة وجاذبية.

References

1.Books and Articles

1. Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner's Approach* (9th ed.). McGraw–Hill Education.
2. Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson Education.

2.Online Resources

4. MongoDB Inc. (n.d.). *MongoDB Documentation*. Retrieved from <https://www.mongodb.com/docs/>
5. React. (n.d.). *React Official Documentation*. Retrieved from <https://react.dev/>
6. Node.js Foundation. (n.d.). *Node.js Documentation*. Retrieved from <https://nodejs.org/en/docs>
7. Express.js. (n.d.). *Express Framework Documentation*. Retrieved from <https://expressjs.com/>

3.Tools and Software Documentation

8. Vite. (n.d.). *Vite – Next Generation Frontend Tooling*. Retrieved from <https://vitejs.dev/>
9. Mongoose. (n.d.). *Mongoose ODM Documentation*. Retrieved from <https://mongoosejs.com/docs/>
10. NodeMailer. (n.d.). *NodeMailer Documentation*. Retrieved from <https://nodemailer.com/about/>

4.Additional References

- 11.Scrum.org. (n.d.). *Scrum Guide*. Retrieved from <https://www.scrumguides.org/>
- 12.Mozilla Developer Network (MDN). (n.d.). *Web Technologies Documentation*. Retrieved from <https://developer.mozilla.org/>
- 13.ImageKit. (n.d.). *ImageKit Media Optimization Documentation*. Retrieved from <https://imagekit.io/docs>