الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة الشهيد حمه لخضر الوادي
كلية العلوم الدقيقة
قسم الاعلام الآلي

مذكرة نهاية تخرج ضمن متطلبات للحصول على شهادة
ليسانس أكاديمي

تصميم ونشر نظام تسجيل دخول موحّد (SSO)

من إنجاز الطلبة

- منير سراوي
- اسراء بوغزالة حمد
- آمنة كنيوة

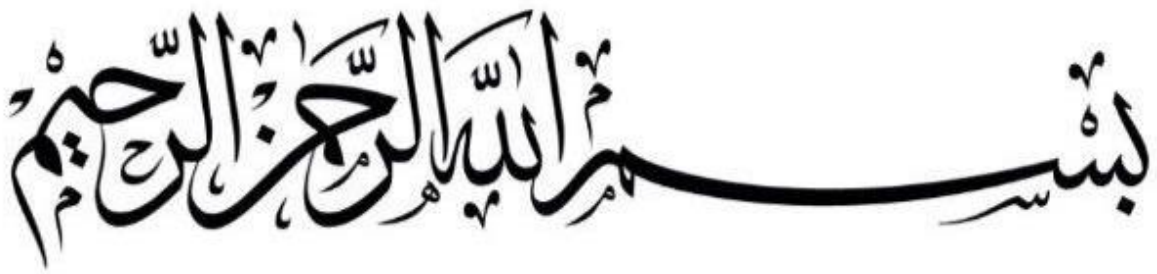
تحت اشراف الاستاذ

- معاذ بالي

لجنة المناقشة

الاسم واللقب	الرتبة	الجامعة	الصفة
			رئيسا
			ممتحنا

السنة الجامعية: 2025/2024


بسم الله الرحمن الرحيم

الملخص

تواجه المؤسسات التعليمية الحديثة تحديات متزايدة في إدارة الدخول إلى أنظمتها الرقمية المتعددة، حيث يجد المستخدم نفسه مجبرًا على التعامل مع حسابات وكلمات مرور متكررة، ما يؤدي إلى تشتّت التجربة، ضعف الأمان، وتضخم في الأعباء التقنية. في جامعة الوادي، تتجلى هذه الإشكالية بوضوح من خلال صعوبة الوصول الموحد إلى خدمات مثل البريد الجامعي، التعليم عن بعد، ونظام الموظفين.

انطلاقًا من هذه الحاجة الملحّة، تقترح هذه المذكرة رؤية متكاملة لحل فعّال وذكي، يتمثل في نظام تسجيل دخول موحد (SSO) يُمكّن المستخدم من الوصول إلى كل خدماته من خلال عملية دخول واحدة فقط. هذا الحل لا يُبسّط فقط تجربة الاستخدام، بل يرفع من الكفاءة، يُعزّز الأمان، ويمنح الإدارة قدرة أكبر على التحكم في الهويات والصلاحيات.

تجمع هذه الدراسة بين تشخيص دقيق للواقع، وتحليل منهجي للحلول الممكنة، وصولًا إلى تصميم عملي يراعي خصوصيات البيئة الجامعية. إنها دعوة لانتقال سلس نحو بيئة رقمية أكثر تماسكًا وذكاءً، يكون فيها الدخول إلى المعرفة والخدمة أبسط وأكثر أمانًا.

الكلمات المفتاحية:

سجيل الدخول الموحد (SSO)، الهوية الرقمية، المصادقة والتفويض، الخدمات الرقمية، نظام (Keycloak)، التحول الرقمي.

Abstract

Modern educational institutions face increasing challenges in managing access to their multiple digital systems, where users often find themselves compelled to handle numerous accounts and passwords. This leads to a fragmented experience, weakened security, and increased technical burdens. At the University of El Oued, this issue is clearly evident in the difficulty of unified access to services such as university email, distance learning platforms, and staff management systems.

In response to this pressing need, this thesis proposes an integrated, intelligent solution in the form of a Single Sign-On (SSO) system that enables users to access all their services through a single login process. This solution not only simplifies the user experience but also enhances efficiency, strengthens security, and provides the administration with greater control over identities and permissions.

This study combines a precise diagnosis of the current situation with a systematic analysis of potential solutions, leading to a practical design tailored to the specific needs of the university environment. It is a call for a smooth transition toward a more cohesive and intelligent digital ecosystem, where access to knowledge and services becomes simpler and more secure.

Keywords :

Single Sign-On (SSO), Digital Identity, Authentication and Authorization, Digital Services, Keycloak, Digital Transformation.

الشكر والعرفان

الحمد لله الذي بنعمته تتمّ الصالحات،
والصلاة والسلام على سيدنا محمد، وعلى آله وصحبه أجمعين.

أتقدّم بخالص الشكر وعظيم التقدير لأستاذي المشرف بالي معاذ،
الذي لم يبخل عليّ بعلمه وخبرته، ووجّهني طيلة فترة إعداد هذه المذكرة بكل
حكمة وصبر، فله مني كل الامتنان والعرفان.

كما أعبّر عن بالغ امتناني للسادة مسؤولي مركز شبكات الأنظمة، على
ما قدّموه من تعاون ودعم تقني أسهم بشكل فعّال في إنجاز هذا المشروع،
فلهم كل التقدير على جهودهم المتفانية.

ولا يفوتني أن أخصّ بالشكر والتقدير "جنديّ الخفاء"،
الذي كان له دور محوري في تسهيل العديد من التحديات خلف الكواليس،
دون أن يسعى إلى الظهور أو الإشادة، فجزاه الله عنّي كل خير.

وختامًا، أتوجّه بالشكر لكل من ساندني ووقف
إلى جانبي خلال مسيرتي الأكاديمية، وأخص بالذكر عائلتي الكريمة
التي كانت دومًا السند والداعم.

الاهداء

بسم الله الرحمن الرحيم

﴿وَقُلِ ٱعۡمَلُواْ فَسَيَرَى ٱللَّهُ عَمَلَكُمۡ وَرَسُولُهُۥ وَٱلۡمُؤۡمِنُونَ﴾

الحمد لله الذي بنعمته تتمّ الصالحات، وبتوفيقه تكتمل البدايات والنهايات، نحمده حمد الشاكرين، ونستعينه استعانة الراجين، ونسأله الإخلاص في القول والعمل، والثبات بعد كل طريقٍ قطعناه بشغفٍ وأمل.

نهدي هذا العمل المتواضع، الذي كان ثمرة جهدٍ وتعبٍ وسهرٍ طويل، ومزيجًا من الأحلام والتحديات، إلى من كان لهم الفضل – بعد الله – في أن يرى هذا المشروع النور، وإلى من يستحقون أن يُخلَّد ذكرهم في أولى صفحات نجاحنا:

إلى من إذا أراد شيئًا قال له: كن فيكون،

إلى من لولاه ما سلكت درب العلم، ولا أدركت معنى التوفيق،

إلى الله جلّ في علاه...

أهدي ثمرة هذا الجهد، سائلًا أن يجعلها خالصة لوجهه الكريم، وينفع بها.

إلى النبع الذي لا ينضب، والعطاء الذي لا ينتهي،

إلى من غرسا في قلبي الإيمان، وفي روحي الطمأنينة،

إلى والديّ العزيزين...

يا من حملتموني حبًا، وعلّقتم آمالكم بي،

لكما أهدي هذا الإنجاز، عربون وفاء وامتنان، وعطر دعاء لا ينقطع.

إلى نفسي...

يا من ذاقت مرارة التعب، وتجاوزت العثرات، وثابرت حين تساقطت الهمم،

أهديكِ هذا الجنيّ من بستان الصبر.

إلى من كان السند حين ثقلت الأيام، والبلسم حين قست اللحظات،

إلى إخوتي الأعزاء...

أهديكم هذا الإنجاز، فأنتم الدفء الذي لا يُعوّض.

وإلى أولئك الذين شاركوني الحلم، ورسموا معي طريق الأمل،

إلى أصدقائي الصادقين،

أنتم النجوم التي أنارت لي عتمة المسير،

لكم من القلب كل الحب، وهذا الإهداء.

الفهرس

فهرس الاشكال

فهرس الجداول

المقدمة العامة

تخيل لو كان بإمكانك الدخول إلى جميع أنظمتك الرقمية أو الإدارية باستخدام حساب واحد فقط، دون الحاجة إلى تذكر عشرات كلمات المرور أو المرور بعمليات تسجيل دخول متكررة ومملة! تخيل بيئة رقمية متكاملة، سلسة وآمنة، حيث يتم فتح الأبواب الرقمية أمامك بنقرة واحدة. هذا ليس خيالاً تقنياً، بل هو ما يسعى إلى تحقيقه نظام الدخول الموحد (SSO).

في عصر تزداد فيه الحاجة إلى السرعة والكفاءة، وتصبح فيه البيانات والمعلومات أكثر حساسية، يظهر (SSO) كحل عبقري يوازن بين سهولة الاستخدام وتعزيز الأمان. المؤسسات الكبرى والشركات العالمية وحتى الجامعات باتت تدرك أن تجربة المستخدم لم تعد رفاهية، بل ضرورة، وأن تأمين الموارد الرقمية لا يجب أن يكون على حساب الراحة أو الإنتاجية.

من هنا تنطلق هذه المذكرة في رحلة استكشافية وعملية داخل عالم المصادقة الرقمية، لنصل إلى حل تقني حديث وفعّال يتمثل في اقتراح أداة لا تمنح للمستخدمين تجربة دخول موحدة فقط، بل يتيح أيضًا مرونة هائلة في إدارة الهويات، التحكم في الصلاحيات، وتطبيق السياسات الأمنية.

فما هو الدخول الموحد؟ كيف يعمل؟ وماذا يُعد الخيار الامثل لتطبيقه في بيئة ذات أنظمة رقمية متعددة تطلب تسجيل دخول مثل جامعة الوادي؟ هذه الأسئلة وأكثر سنجد إجاباتها في فصول هذه المذكرة التي تمزج بين التحليل النظري والتطبيق العملي بطريقة مفصلة من التخطيط الإنجاز.

الفصل الأول: دراسة الموجود

1.I مقدمة

في إطار إعداد هذه المذكرة، ومن أجل تعزيز الجانب التطبيقي للمشروع المقترح، تم القيام بزيارة ميدانية إلى مركز شبكات الأنظمة (CSRICTED) بجامعة الوادي وذلك بتاريخ [2025/03/13]، هدفت هذه الزيارة إلى الاطلاع عن قرب على البنية التحتية المعلوماتية للجامعة، والتعرف على آليات عمل الشبكات المحلية، أنظمة الدخول، والتقنيات المعتمدة في إدارة الموارد المعلوماتية.

في هذا الفصل، سنقوم بعرض ما توصلنا إليه من خلال اتباع المراحل التالية: أولا، سنتعرف على المفاهيم الأساسية المتعلقة ببيئة العمل، متضمنين بذلك مفهوم المتصفح والخادم مع استعراض آلية عمل تسجيل الدخول وانواعها كاستخدام الجلسة (Session) والرموز (Token)، كما سنقوم بدراسة نظام تسجيل الدخول الموحد (SSO) وآلية عمله. ثانيا، من خلال الدراسة الميدانية سنسلط الضوء على التحديات الموجودة في سيرورة العمل الحالية المتعلقة بأنظمة تسجيل الدخول على مستوى (CSRICTED)، وأخيرا، سنسعى الى اقتراح أفكار لحلول تقنية لتحسين سيرورة العمل وجعل الأنظمة أكثر مرونة وفائدة للمستخدمين. سنكتفي في هذا الفصل بذكر الأفكار ليتم مناقشة تفاصيلها ونمذجتها في الفصل الثاني.

2.I مفهوم الويب

يُعرف بروتوكول (HTTP) ببروتوكول الويب، حيث يُعد جزءًا من مجموعة الخدمات التي تقدمها الشبكة أو الإنترنت. وفي مجال الويب وتكنولوجيا

المعلومات، يُعتبر كل من الخادم (Server) والمتصفح (Browser/Navigator) من العناصر الأساسية التي تتكامل لتوفير تجربة تصفح سلسة للمستخدمين. ورغم تكاملهما في أداء هذه المهمة، فإن لكل منهما دورًا ووظيفة تختلف تمامًا عن الآخر

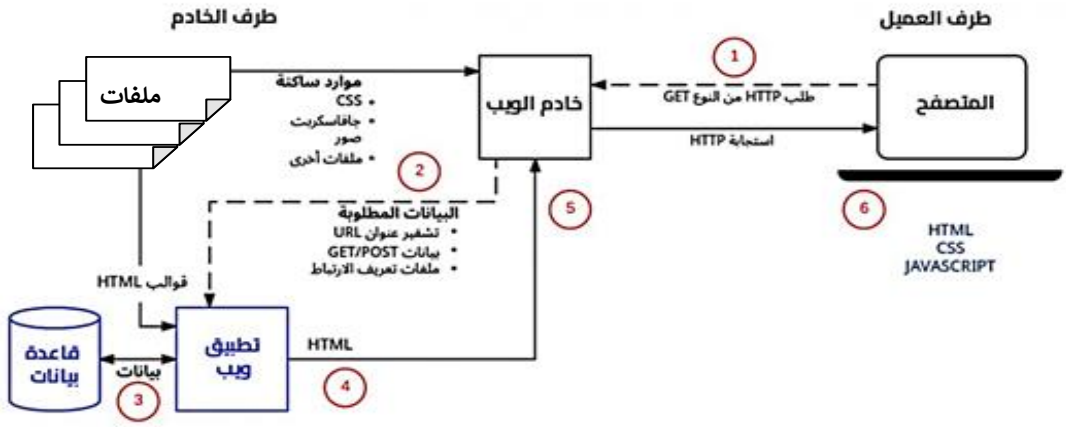
1.2.I تعريف المتصفح

المتصفح هو تطبيق يعمل على جهاز المستخدم) مثل الكمبيوتر أو الهاتف (ويسمح للمستخدمين بالوصول إلى مواقع الويب وتصفحه.[1]

2.2.I تعريف خادم الويب

الخادم هو جهاز يحوي نظاما يعمل على استضافة الموارد وتقديمها إلى المتصفح عند الطلب عبر بروتوكول (HTTP) يمكن أن يكون الخادم جهازا ماديا (Physical Server) أو إقراضيا كالخدمات السحابية (Service Cloud).[1]

3.2.I آلية تفاعل المتصفح مع الخادم



الشكل I-1: ألية تفاعل المتصفح مع الخادم، عبر أهم المراحل التالية[15]

- **زيارة الموقع**: يقوم المستخدم بفتح المتصفح وزيارة موقع الويب مثل (www.amazon.com).
- **إنشاء اتصال آمن (TCP Handshake)**

بعد الحصول على عنوان (IP)، يحاول المتصفح إنشاء اتصال آمن مع الخادم باستخدام بروتوكول (TCP). يتم إجراء "مصافحة ثلاثية" بين المتصفح والخادم لتأسيس الاتصال (Handshake Three-way).

- **ترقية الاتصال إلى (HTTPS)**

يتم ترقية الاتصال من (HTTP) إلى (HTTPS) لضمان أمان البيانات. سيتم ذكر تفاصيل أكثر حول هذا البروتوكول في الفقرات القادمة.

يقوم المتصفح والخادم بإجراء مصافحة (TLS Handshake) للاتفاق على خوارزميات التشفير وتبادل المفاتيح الامنة، لضمان اتصال امن بين المتصفح والخادم.

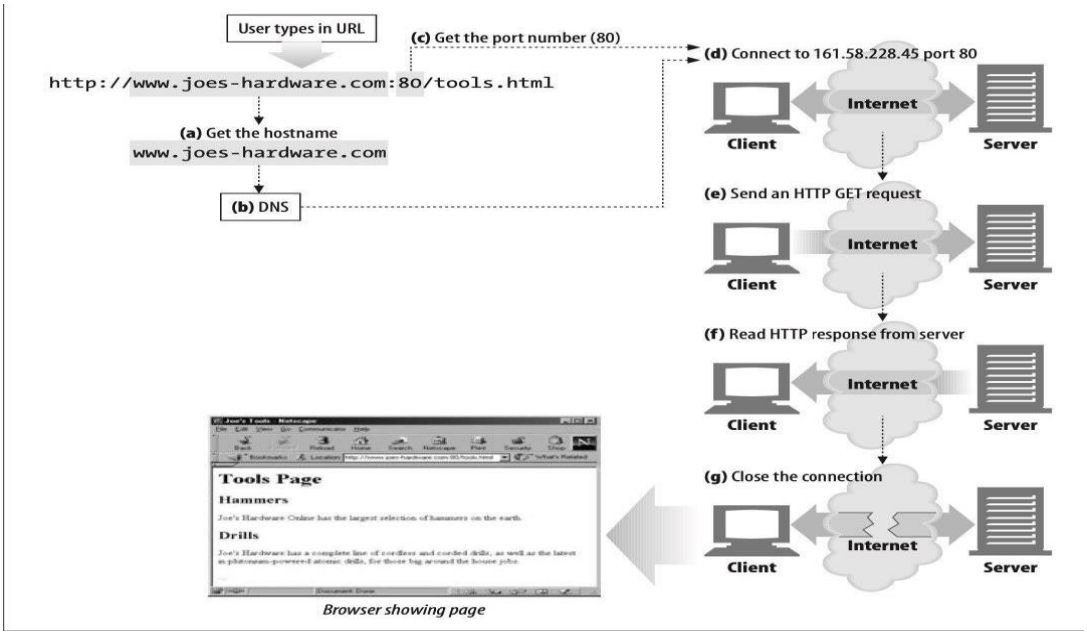
- **طلب صفحة الويب**

يرسل المتصفح طلب (HTTP Request) إلى الخادم للحصول على صفحة الويب الرئيسية.

يقوم الخادم بمعالجة الطلب والبيانات المرسلة ثم يرسل استجابة تحتوي على كود (HTML) لصفحة الويب.

- **استقبال وعرض صفحة الويب**

يستقبل المتصفح الكود ويقوم بعرض الصفحة ويحمل الموارد الإضافية مثل الصور، (JavaScript)، (CSS) من خلال طلبات (HTTP Request) إضافية.



الشكل I-2: تفاعل المتصفح مع الخادم باستخدام(HTTP) لعرض مورد(HTML)

I.4.2 أهم وظائف خادم الويب

- استقبال الطلب (HTTP Request).
- معالجة البيانات على الخادم (Processing Side-Server)
- إرسال الاستجابة (Response HTTP).
- التفاعل مع قواعد البيانات

- **إعادة توجيه الطلبيات (HTTP Request Redirect):**

فكرة إعادة توجيه الروابط (URL Redirect) تدور حول تحويل المستخدم تلقائيًا من عنوان (URL)إلى آخر، سواء بشكل مؤقت أو دائم، وذلك لأغراض مثل صيانة الموقع، أو الحفاظ على الروابط بعد تغيير هيكل الصفحات تعتمد هذه العملية على استجابة خاصة من بروتوكول (HTTP)تُعرف بـ "إعادة التوجيه".[17]

I.5.2 التفاعل مع خدمات الطرف الثالث (Third-Party Services)

1. توجيه المستخدم إلى مزود الخدمة:

عندما يختار المستخدم تسجيل الدخول عبر خدمة خارجية مثل (Google)أو (Facebook)، يتم نقله إلى صفحة تسجيل الدخول الخاصة بالمزود (على سبيل المثال، صفحة تسجيل الدخول في Google).

2. إدخال بيانات المستخدم:

على صفحة تسجيل الدخول، يقوم المستخدم بإدخال بياناته، مثل البريد الإلكتروني وكلمة المرور الخاصة بحسابه على (Google) أو (Facebook).

3. طلب التفويض:

بعد إدخال بيانات الدخول، يطلب المزود الخارجي من المستخدم السماح له بالوصول إلى بعض البيانات الشخصية، مثل البريد الإلكتروني والاسم. يمكن للمستخدم قبول أو رفض هذا الطلب.

4. إعادة التوجيه مع رمز التفويض:

إذا وافق المستخدم على التفويض، يتم إعادة توجيهه إلى الموقع الأصلي (الذي يود المستخدم تسجيل الدخول إليه) مع رمز التفويض (Authorization Token)، وهو رمز مؤقت يُستخدم للتحقق من هوية المستخدم.

5. تحويل الرمز إلى رمز الوصول:

بعد استلام رمز التفويض، يرسل الموقع الأصلي هذا الرمز إلى خدمة (OAuth) الخاصة بمزود الخدمة (مثل Google أو Facebook) عبر طلب (HTTP)، يرد المزود برمز الوصول (Access Token)، الذي يسمح للموقع الأصلي بالحصول على البيانات المصرح بها من حساب المستخدم، مثل البريد الإلكتروني والاسم.

6. التحقق من الهوية والوصول إلى البيانات:

يستخدم الموقع الأصلي رمز الوصول للتحقق من هوية المستخدم، ومن ثم يمكنه الوصول إلى البيانات التي وافق المستخدم على مشاركتها. بعد ذلك، يتم تسجيل دخول المستخدم إلى الموقع باستخدام بيانات حسابه في (Google أو Facebook).

I.3 بروتوكول (HTTP)

- إدارة الاتصالات في بروتوكول (HTTP)

يعتبر موضوع إدارة الاتصالات أمرًا حيويًا في بروتوكول (HTTP)، حيث يؤثر فتح الاتصالات والحفاظ عليها بشكل كبير على أداء مواقع وتطبيقات الويب في إصدار (HTTP/1.x)، توجد عدة نماذج لإدارة هذه الاتصالات.

I.3.1 الاعتماد على بروتوكول (TCP)

- يعتمد (HTTP)بشكل أساسي على بروتوكول (TCP)كنظام نقل، مما يوفر اتصالًا موثوقًا بين العميل والخادم.

I.3.2 الاتصالات قصيرة العمر (النموذج الأولي)

- آلية العمل: في البداية، استخدم (HTTP)نموذجًا بسيطًا حيث يتم إنشاء اتصال (TCP)جديد لكل طلب يرسله العميل، ويتم إغلاق هذا الاتصال بمجرد استلام الرد.
- القيود المتعلقة بالأداء:
- استهلاك الموارد: فتح كل اتصال (TCP) هو عملية مستهلكة للموارد.
- تبادل رسائل متعددة: يتطلب إنشاء الاتصال تبادل عدة رسائل بين العميل والخادم.
- تأثير زمن الوصول وعرض النطاق: يؤثر زمن الوصول (Latency) وعرض النطاق الترددي (Bandwidth) سلبًا على الأداء عند الحاجة لإرسال طلبات متعددة.
- عدم الفعالية للصفحات الحديثة: تتطلب الصفحات الحديثة عددًا كبيرًا من الطلبات، مما يجعل هذا النموذج غير فعال.[2]

I.4 التقنيات المستخدمة في أنظمة تسجيل الدخول

I.4.1 الجلسات (Session)

I.4.1.1 بدء الجلسة

عندما يقوم المستخدم بتسجيل الدخول أو يزور موقعًا ويب لأول مرة، ينشئ الخادم جلسة جديدة تحتوي معرف فريد للجلسة (ID Session) وهو عبارة عن سلسلة نصية عشوائية تسُتخدم لتحديد الجلسة بالإضافة إلى ذلك، يتم تخزين بيانات الجلسة مثل معلومات المستخدم، تفضيلاته، أو أي بيانات أخرى على الخادم.

I.4.1.2 استخدام (Session ID) في طلبات المتصفح :

يتم إرسال (Session ID) ليخزن في متصفح المستخدم عل شكل (Cookie) ليستخدم في كل طلب لاحق من المتصفح إلى الخادم، بحيث يتم إرسال (Session ID) تلقائيًا مع الطلب تتيح هذه العملية للخادم استعادة بيانات الجلسة المرتبطة بمتصفح المستخدم المعني.

3.1.4.I تسجيل الخروج وانهاء الجلسة (Termination Session)

تنتهي الجلسة عندما يقوم المستخدم بتسجيل الخروج (Logout)، أو تنتهي صلاحية الجلسة أليا بعد فترة زمنية محددة (Expiry.Session) بانتهاء الجلسة يتم حذف جميع معلوماتها المخزنة على الخادم [14]

2.4.I استخدام الرموز (Tokens)

1.2.4.I رموز الوصول (Access Tokens)

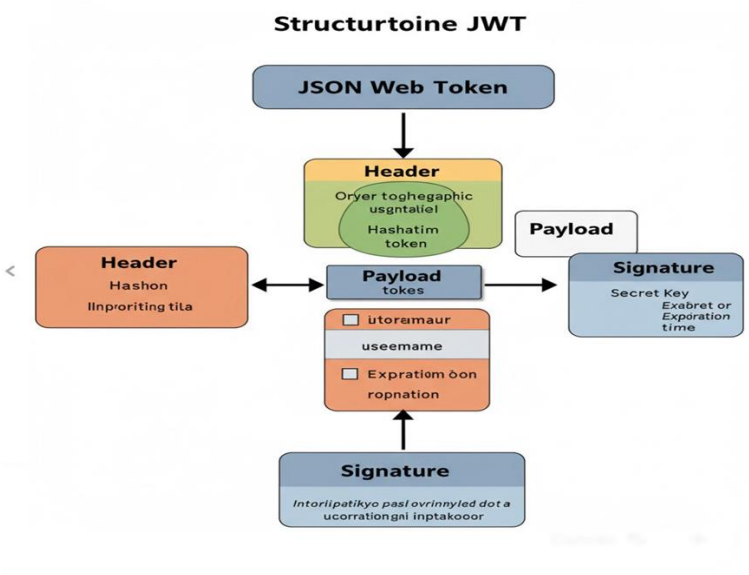
رموز الوصول (Access Tokens) تُستخدم لتفويض الوصول إلى خدمات (Google) دون أن تتضمن هوية المستخدم. عند استخدام بيانات الاعتماد الافتراضية مع مكتبات (Google)، تتم إدارة هذه الرموز تلقائيًا دون تدخل يدوي.

2.2.4.I رموز تعريفية (ID Tokens)

رموز الهوية (ID Tokens) هي رموز (JWT) تتبع معيار (OpenID Connect)، وتحتوي على بيانات يمكن للتطبيق قراءتها مثل هوية المستخدم والجهة المصدِرة. بخلاف رموز الوصول التي لا يمكن فحصها، تُستخدم رموز الهوية من قبل التطبيق للتحقق من هوية المستخدم.

3.2.4.I رموز ويب بصيغة (Json) (JWT) (Json Web Tokens)

رموز الويب (JWT) هي تقنية شائعة ومرنة لنقل المعلومات بأمان بين طرفين. تتكون من ثلاثة أجزاء (رأس، حمولة، توقيع) مرمزة بـ (Base64) تسمح بإضافة مطالبات مخصصة (بيانات إضافية). يتم توقيع حمولة (JWT)رقميًا (باستخدام JWS) لضمان صحتها تعتبر (JWT)جزءًا من (JSON Identity Suite) الأوسع، والتي تشمل أيضًا (JWT) (للتشفير)، تُستخدم هذه المجموعة بشكل كبير في بروتوكولات مثل (OpenID Connect) و(OAuth 2.0) [4]



الشكل I-3: بنية الرمز(jwt)

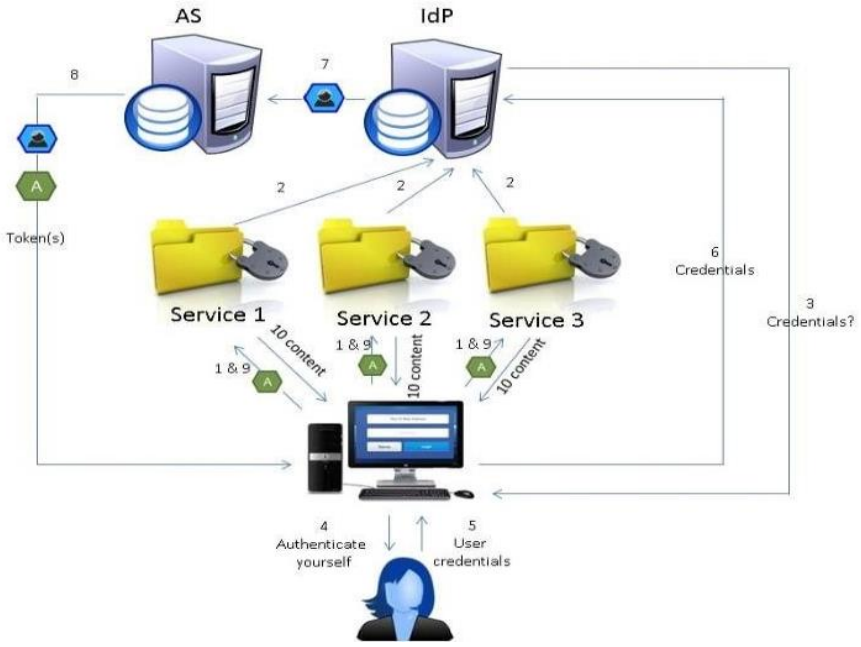
حيث نلاحظ أن رموز (JWT) هي سلسلة من الأحرف "الآمنة للعنوان" (URL) التي تقوم بترميز المعلومات تتكون الرموز من ثلاثة أجزاء مفصولة بنقاط، كما هو موضح أدناه:

- الرأس: يصف نوع الرمز والخوارزمية المستخدمة
- الحمولة: تحتوي على المعلومات أو المطالبات.
- التوقيع: يستخدم للتحقق من صحة الرمز [3]

I.5 تعريف تسجيل الدخول الموحد (SSO)

يُعدّ مصطلح "تسجيل الدخول الأحادي (SSO) "مصطلحًا راسخًا استُخدم منذ التسعينيات وحتى يومنا هذا. بدأ كحل مؤسسي يهدف إلى تسجيل دخول المستخدمين تلقائيًا إلى التطبيقات، لكن مع مرور الوقت، اتسع نطاقه ليشمل تطبيقات الويب.

تتعدد وجهات النظر حول تعريف تسجيل الدخول الأحادي. لكن يمكننا تعريفه بأنه حل يمكّن المستخدمين من تسجيل الدخول عبر صفحة واحدة، ومن ثم الوصول إلى خدمات متعددة دون الحاجة لتسجيل الدخول مرة أخرى لكل خدمة[9] كما هو موضح في الشكل (**4-1**).



الشكل I-4: المصادقة باستخدام الدخول الموحد(SSO)

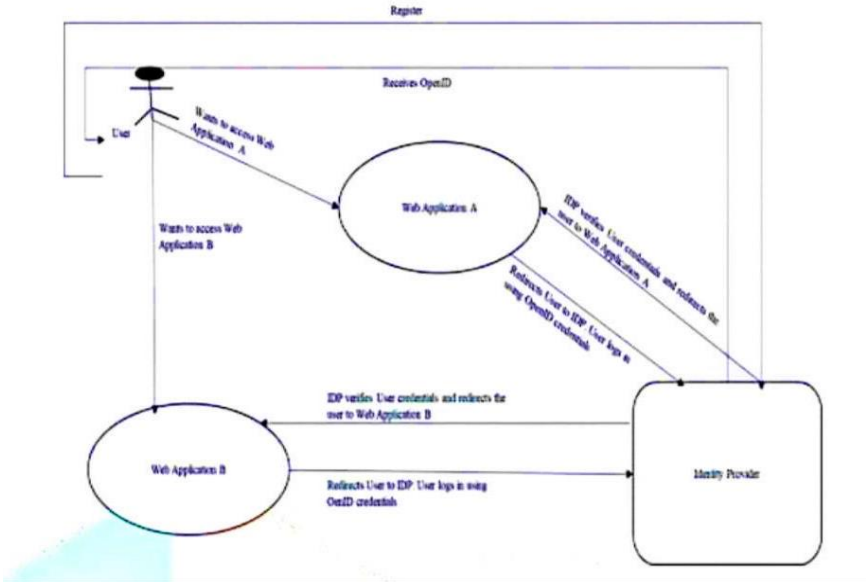
باستخدام هذا التفسير، يحتوي تسجيل الدخول الأحادي على الخطوات التالية:

1. يحاول العميل الوصول إلى خدمة. إذا كان لدى العميل بالفعل رمز وصول لهذه الخدمة، إضافة الرمز المميز إلى الطلب. بعد ذلك، انتقل إلى الخطوة 10.
2. تستدعي الخدمة موفر الهوية للتعامل مع المصادقة.

3. يطلب موفر الهوية من العميل بيانات اعتماد تسجيل الدخول.
4. يطلب العميل من المستخدم إعطاء بيانات اعتماد تسجيل الدخول.
5. يسلم المستخدم بيانات اعتماد تسجيل الدخول.
6. يرسل العميل بيانات الاعتماد هذه إلى موفر الهوية الذي يتحقق من صحة بيانات الاعتماد.
7. إذا كانت بيانات الاعتماد صحيحة، يتم إرسال رمز مميز للمعرف إلى (AS)، وإلا فإنه يعود إلى الخطوة 3.
8. يجمع (AS)الحقوق التي تم تعيينها للمستخدم وينشئ رمزا مميزا للوصول. يتم إرسال الرمز المميز للوصول والرمز المميز للمعرف إلى العميل.
9. يحاول العميل الوصول إلى خدمة باستخدام رمز الوصول.
10. تمنح الخدمة الوصول إلى الخدمة.

I.1.5 آلية عمل تسجيل الدخول الموحد

يقوم المستخدم بتسجيل نفسه مع (IDP) لتلقي بيانات اعتماد (OpenID) الآن، يريد المستخدم الوصول إلى التطبيق أ. يعيد التطبيق "أ" توجيه المستخدم إلى (IDP) يقوم المستخدم بتسجيل الدخول إلى (IDP) باستخدام بيانات اعتماد (OpenID) الخاصة به عند المصادقة الناجحة، يقوم (IDP) بإعادة توجيه المستخدم مرة أخرى إلى التطبيق A. إذا أراد المستخدم الوصول إلى تطبيق الويب "ب "، فسيرسل طلبا إلى تطبيق الويب "ب "، وعند استلام الطلب، ينتقل إلى موفر الهوية ويتحقق مما إذا كان المستخدم الذي حاول الوصول إليه نشطا أم لا. إذا تم العثور على تطبيق ويب نش ط، فسيسمح تطبيق الويب B للمستخدم بالوصول إليه تلقائيا. وبالمثل، ستتبع تطبيقات الويب الأخرى نفس العملية. لا يعرف تطبيق الويب "أ" ما يحدث في تطبيق الويب "ب "، ولا يعرف تطبيق الويب "ب " ما يحدث في تطبيق الويب "أ" وما إلى ذلك.



الشكل I-5: كيفية عمل نظام تسجيل الدخول الموحد(SSO)

I.2.5 البروتوكولات المستخدمة في (SSO)

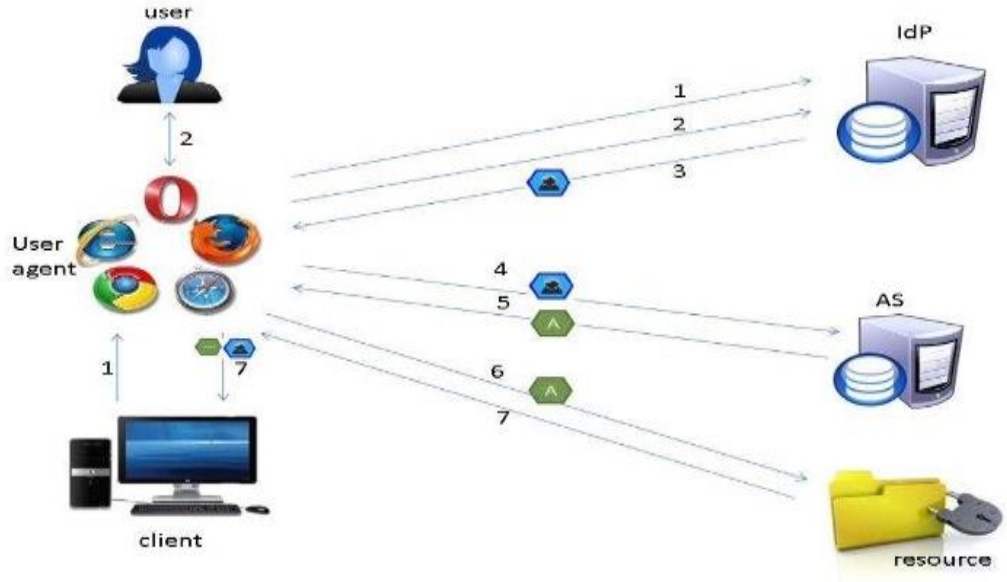
I.2.5.1 الاتصال بـ (OpenID Connect)

(OpenID Connect) (OIDC) يمكن وصفه على أنه طبقة هوية فوق (OAuth2) توفر امتدادات مختلفة، أهمها المصادقة لذا، يقدم هذا البروتوكول كلًا من التفويض (Authorization) والمصادقة (Authentication) تدفقات (OpenID Connect) (Flows) تشبه إلى حد كبير تلك الموجودة في (OAuth2)، الفرق الرئيسي هو إضافة موفر الهوية (IdP) ورمز الهوية (Token ID)

مبدأ عمله:

في OpenID Connect ، تتوفر ثلاث تدفقات رئيسية: التدفق الضمني Flow Implicit، تدفق تفويض الرمز Flow Authorization Code، والتدفق الهجين Flow Hybrid. التدفق الهجين يشبه إلى حد كبير تدفق تفويض الرمز، ولكنه يسمح بإرسال الرموز مباشرة إلى العميل دون الحاجة إلى استخدام رمز تفويض Code Authorization.[10]

بالإضافة إلى ذلك، يتيح التدفق الهجين إمكانية إرسال معلومة



الشكل I-6: التدفق المختلط OpenID Connect

1. يقوم العميل بإعداد طلب مصادقة يحتوي على معلومات الطلب المطلوبة وإرساله إلى مزود الهوية (IdP).
2. ثم يطلب مزود الهوية (IdP) بيانات اعتماد المستخدم، والتي يقوم المستخدم بتقديمها.
4. إذا كانت بيانات اعتماد المستخدم صحيحة، يعيد مزود الهوية (IdP) رمز الهوية (ID TOKEN) للحصول على حقوق الوصول، يتصل وكيل المستخدم (User Agent) بخادم التفويض (AS) مُرسلًا رمز الهوية (ID) كمرجع (TOKEN)

5. يقوم خادم التفويض (AS) بإرجاع رمز وصول (Access Token) بناءً على الهوية المقدمة من خلال رمز الهوية (ID Token)
6. يقوم وكيل المستخدم (Agent User) بإرسال طلب يحتوي على رمز الوصول (Access Token) إلى الخدمة للحصول على الوصول.
7. بناءً على رمز الوصول (Access Token)، تقوم الخدمة بإرجاع مجموعة من البيانات التي يتم عرضها على وكيل المستخدم (Agent User)

I.2.5.2 Open Authorization 2.0 (OAuth2)

رغم أن (OAuth2) لا يُعد بروتوكول مصادقة بالمعنى الدقيق، بل يركّز على التفويض، إلا أن المصادقة تعتبر ضمنيًا جزءًا من عملية إصدار رمز الوصول(Access Token)، مما يتيح استخدامه في خدمات (SSO) الفرق بين رمز الوصول ورمز الهوية (ID Token) هو أن الأول يقتصر على صلاحيات الوصول، بينما الثاني يحتوي على معلومات تعريفية عن المستخدم يعمل(OAuth2) عبر عدة كيانات، حيث يمنح خادم التفويض (AS) رموزًا مؤقتة للعميل، الذي يستخدمها للوصول إلى الموارد من خوادم تثق بـ.(AS) تُشبه آلية العمل توزيع مفاتيح موقّتة لعمال فندق. كما أن البروتوكول يتضمن عدة تدفقات تفويض، كل منها مناسب لسيناريو مختلف، وجميعها موصوفة في المواصفات الرسمية.[11]

I.2.5.3 Security Assertion Markup Language (SAML)

هو بروتوكول يُستخدم للمصادقة والتفويض بين النطاقات، ويعتمد عليه العديد من مزودي الخدمات مثل Googleو Amazonيُستخدم بشكل واسع في بيئات الأعمال والحكومات والتعليم. يوفر نوعين من وسائل نقل الرسائل HTTP Redirect :(Bindings) للرسائل القصيرة، و HTTP POSTللرسائل الأطول والموقعة. رغم فعاليته، يُعد SAML أقل ملاءمة للتطبيقات الحديثة، خاصة التطبيقات الأصلية، [12] بسبب اعتماده على POST ونموذج XMLكما أنه يفتقر إلى مرونة التدفقات المتعددة الموجودة في بروتوكولات مثل open id conectو aouth2

I.2.5.4 Lightweight Directory Access Protocol (LDAP)

يُعد LDAP (Lightweight Directory Access Protocol)بروتوكولًا يُستخدم للوصول إلى خدمات الدلائل الموزعة، وغالبًا ما يُستخدم مع Active Directoryلتوفير المصادقة (Authentication) والتفويض .(Authorization)يُشبه الدليل قاعدة بيانات مخصصة، مثل دفتر الهاتف، ويُستخدم أساسًا في البيئات التي تتطلب عمليات استعلام بسيطة وسريعة عبر الشبكة.

يتكون LDAP من عميل وخادم، حيث يُرسل العميل طلبًا (مثل البحث عن إدخال معين) في شكل رسالة LDAPتحتوي على معرّف فريد(ID-Message) ، ويقوم الخادم بمعالجة الطلب والرد برسالة تحتوي على النتيجة ورمز استجابة. تعتمد جميع التفاعلات على هذا المعرّف لتتبع الطلبات والردود[13].

I.6 بروتوكولات المصادقة والتفويض في بيئة الدخول الموحد

I.6.1 تسجيل الدخول الواحد (SSO-On-Sign Single)

- الوصف: يسمح للمستخدم بالدخول مرة واحدة باستخدام بيانات اعتماد واحدة للوصول إلى عدة أنظمة أو تطبيقات دون الحاجة لإعادة تسجيل الدخول
- المزايا: يقلل من الحاجة إلى تذكر عدة كلمات مرور، ويحسن تجربة المستخدم.
- بروتوكولات مثل SAML OpenID Connect

I.6.2 التفويض باستخدام (OAuth2)

- الوصف: بروتوكول يسمح للتطبيقات بالوصول إلى موارد المستخدم دون الحاجة إلى مشاركة بيانات الاعتماد الخاصة به.
- المزايا: يحسن الأمان ويوفر تحكمًا دقيقًا في الوصول.

I.6.3 المصادقة باستخدام OpenID Connect (OIDC)

- الوصف: طبقة هوية تعمل فوق (OAuth2) لتوفير المصادقة بالإضافة إلى التفويض.
- المزايا: يوفر مصادقة موحدة وآمنة عبر عدة أنظمة. – –

I.7 نماذج أنظمة SSO

I.7.1 تعريف (CAS)

يُعد خدمة المصادقة المركزية (CAS) بروتوكولًا لتسجيل الدخول الموحد (Single Sign-On) وتسجيل الخروج الموحد (Single Sign-Off) على الويب.

يتيح هذا البروتوكول للمستخدم الوصول إلى عدة تطبيقات مختلفة من خلال تقديم بيانات الاعتماد الخاصة به (مثل اسم المستخدم وكلمة المرور) مرة واحدة فقط إلى تطبيق خادم CAS المركزي.[16]

I.7.2 تعريف (Keycloak)

هو نظام مفتوح المصدر لإدارة الهوية والوصول، صُمم لتأمين التطبيقات الحديثة مثل تطبيقات الويب الأحادية وتطبيقات الهاتف المحمول وواجهات برمجة التطبيقاتAPIs .

ويوفر ميزات جاهزة مثل صفحات تسجيل الدخول القابلة للتخصيص، دعم المصادقة القوية ومتعددة العوامل، استعادة كلمات المرور، قبول الشروط والأحكام، وتحديث كلمة المرور بشكل دوري بدون الحاجة إلى كتابة كود. كما يتيح تسجيل الدخول الموحد (SSO) وإدارة الجلسات، ما يُمكّن المستخدم من الوصول إلى عدة تطبيقات بجلسة واحدة.

ويعتمد على بروتوكولات أمان قياسية مثل OAuth 2.0 و OpenID Connectو2.0 SAML، ويحتوي على قاعدة بيانات مستخدمين مدمجة، بالإضافة إلى إمكانية التكامل مع أنظمة هوية خارجية مثل Active DirectoryوLDAP، أو منصات الهوية الاجتماعية [18]

I.3.7 مقارنة بين (CAS) و(Keycloak)

من أجل تحديد الحل الأمثل لمشروع الدخول الموحد، تم إجراء مقارنة تقنية بين نظامي CAS وKeycloak، اعتماداً على عدة معايير

المعيار	CAS	KEYCLOAK
واجهة الإدارة	محدودة	رسومية وسهلة الاستخدام
البروتوكلات المدعومة	SAML, CAS	SAML, OpenID Connect, OAuth2
سهولة التكامل	يتطلب إعدادات يدوية كثيرة	تكامل بسيط نسبيًا مع تطبيقات الويب
دعم التوثيق المتعدد	محدود	متقدم (يدعم MFA بسهولة)
التوثيق الموحد (SSO)	مدعوم	مدعوم
الدعم المجتمعي	متوسط	نشيط وواسع الانتشار

جدول I-1: مقارنة بين (CAS) و(Keycloak)

I.8 الفرق بين تسجيل الدخول الكلاسيكي والموحد

الخاصية	تسجيل دخول العادي	تسجيل دخول الموحد(**sso**)
• عدد مرات تسجيل الدخول	• يحتاج المستخدم لتسجيل الدخول في كل تطبيق بشكل منفصل	• تسجيل دخول مرة واحدة فقط لجميع التطبيقات المتصلة
• الامان	• يعتمد على قوة التطبيق الفردي في تأمين تسجيل الدخول	• يوفر مركزية في إدارة الهوية ويمكن ضبط سياسات أمان موحدة
• إدارة الجلسات	• الجلسات تدار على مستوى فقط	• الجلسة تدار مركزيا، ويمكن مشاركتها بين التطبيقات

جدول I-2:مقارنة بين تسجيل دخول عادي والموحد

I.9 الدراسة الميدانية

I.9.1 تقديم مركز شبكات الأنظمة لجامعة الوادي (CSRCTED)

يُعد مركز تقنيات الأنظمة بجامعة الوادي وحدة تقنية متخصصة، تهدف إلى إدارة وتشغيل شبكات المعلومات، وتطوير الأنظمة والتطبيقات الرقمية، إضافة إلى دعم التعليم عن بُعد، وذلك في إطار تعزيز التحول الرقمي وتحسين جودة الخدمات الأكاديمية والبحثية داخل الجامعة

I.9.2 المواقع المستخدمة

تسمية الموقع	لغة البرمجة	متاح خدمة تسجيل الدخول
my.univ-eloued.dz	PHP	الاساتذة اداريين وعمال
social.univ-eloued.dz		الاساتذة اداريين وعمال
e-learning.univ-eloued.dz		الطلبة الاساتذة ومسؤولي المنصة
plagiarism.univ-eloued.dz	PHP او Python	الاساتذة الباحثين ومسؤولي المنصة
dspace.univ-eloued.dz	Java	مسؤولي النظام وموظفي المكتبة المركزية

جدول I-3: المواقع المستخدمة

I.9.3 المشاكل

في جامعة الوادي، يعاني الطلاب والأساتذة والموظفون من الحاجة لتسجيل الدخول إلى عدة منصات مختلفة باستخدام بيانات اعتماد منفصلة لكل منها، مثل موقع الموظف، ودروس على الخط، والخدمات الاجتماعية، و(WEB TV)، و(EDEX) هذا الوضع يؤدي إلى مشاكل متعددة مثل نسيان كلمات المرور لكثرتها، وعدم وجود طريقة موحدة لاسترجاعها. كما يؤدي إلى ضعف الأمان بسبب استخدام كلمات مرور بسيطة أو مكررة. إضافة إلى ذلك، لا يوجد نظام مركزي يمكن من خلاله إدارة الحسابات أو تعطيلها عند الاشتباه في تعرضها للاختراق، مما يشكل تهديداً لأمن وسلامة الأنظمة الجامعية ويزيد من العبء على فرق الدعم التقني.

I.9.4 الحلول

لتجاوز التحديات المذكورة سابقًا، نقترح اعتماد نظام تسجيل الدخول الموحد(SSO) ، الذي يتيح للمستخدمين تسجيل الدخول مرة واحدة فقط للوصول إلى جميع الخدمات الجامعية. هذا النظام يُقلل من الحاجة إلى تذكّر كلمات مرور متعددة، ويُقلل من احتمالية نسيان كلمات المرور. كما يُسهّل على مسؤولي الشبكة إدارة صلاحيات الوصول وتعليق الحسابات بسرعة في حال وجود أي تهديد أمني، مما يُعزز مستوى الحماية. بالإضافة إلى ذلك، يُخفف (SSO) من الضغط على فرق الدعم التقني من خلال تقليل عدد التدخلات الفنية المرتبطة بمشاكل تسجيل الدخول.

I.10 الخاتمة

من خلال هذا الفصل، تمكّنا من تسليط الضوء على الوضع الحالي المعتمد في جامعة الوادي فيما يخص نظام تسجيل الدخول والخدمات الرقمية المرتبطة به. أظهرت الدراسة الميدانية أن تعدد أنظمة المصادقة وتكرار عمليات تسجيل الدخول يشكّل تحدياً حقيقياً للمستخدمين، سواء من حيث تجربة الاستخدام أو من ناحية الأمان. كما تم التطرق إلى البنية الأساسية لنظام المصادقة التقليدي وآلية التفاعل بين المتصفح والخادم، ما سمح بفهم أعمق لكيفية معالجة الطلبات وتأكيد هوية المستخدم.

هذا التحليل يبرز بوضوح الحاجة إلى اعتماد حل أكثر فعالية وتكاملاً، مثل نظام الدخول الموحد (SSO)، الذي من شأنه تبسيط تجربة المستخدم، وتعزيز أمان الأنظمة، وتحقيق كفاءة في إدارة الهويات الرقمية وعليه، يشكّل هذا الفصل الأساس الذي ستُبنى عليه المراحل القادمة من المشروع، والتي ستركز على تصميم وتطبيق حل عملي باستخدام تقنيات حديثة مثل (Keycloak)، بما يتماشى مع احتياجات المؤسسة.

الفصل الثاني: تصميم النظام

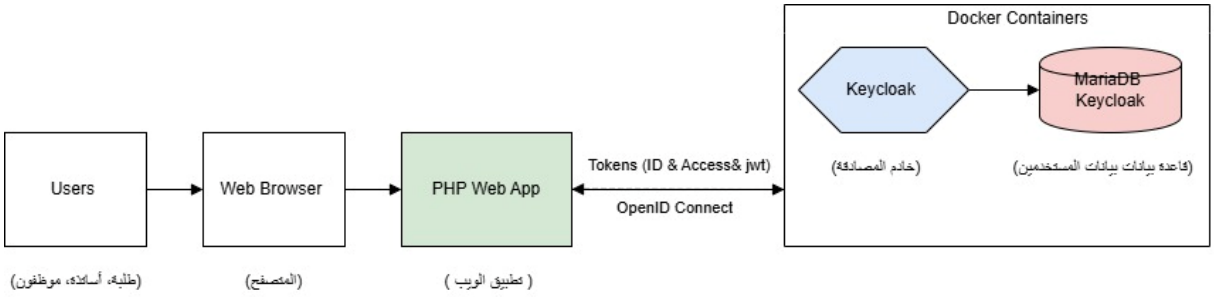
II.1 مقدمة

بعد الانتهاء من دراسة الموجود ومعرفة أكبر حجم من المعلومات حول النظام، يمكننا الان الانتقال الى مرحلة النمذجة والتصور التي تعد حجر الأساس في رحلة تطوير المشروع .وفي هذا الفصل، سنغوص في عالم تحليل متطلبات المشروع سنستعين بلغة (UML) في هذا الفصل الى مخططين رئيسيين:

(Use Case Diagram): يتم في هذه المخطط وصف الأنشطة والمهام التي يقوم بها المستخدمون أو الأنظمة الخارجية والتفاعلات المختلفة مع النظام المراد تطويره.

(**Sequence Diagram**): يستخدم لعرض تسلسل التفاعلات بين الكائنات (Objects أو Actors) في نظام ما مع مرور الزمن، ويُظهر كيف تتبادل الكائنات الرسائل (Messages) لتنفيذ وظيفة معينة

3.II مكونات بيئة العمل



الشكل II-1: مكونات بيئة العمل

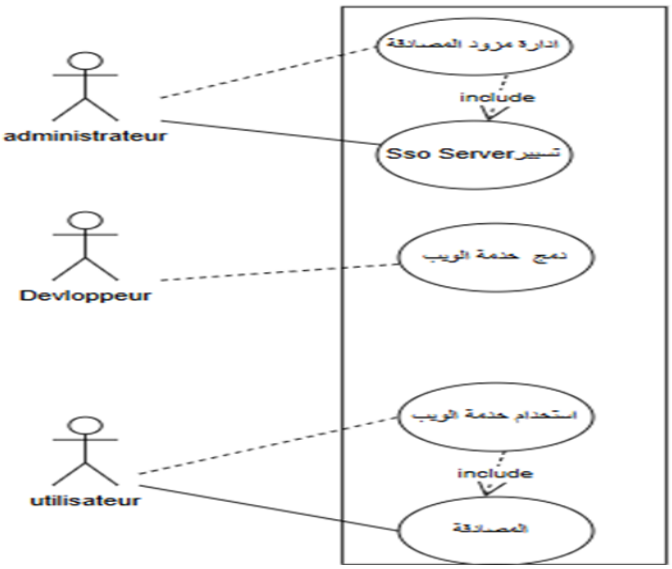
3.II تعريف لغة (UML)

هي لغة نمذجة معيارية تُستخدم لتصور وتصميم وتوثيق نظم البرمجيات توفر (UML) مجموعة من الرسومات البيانية التي تساعد في توضيح هيكل النظام، سلوك النظام، والتفاعلات بين مكونات النظام. تُستخدم (UML) بشكل واسع في هندسة البرمجيات لتسهيل فهم وتطوير نظم البرمجيات المعقدة[5]

4.II مخططات حالة الاستخدام

مخطط الحالة في (UML) هو أحد أنواع مخططات (UML) الخمسة الأساسية وهو يركز على تصور حالات الاستخدام وسلوك النظام يُستخدم مخطط الحالة لوصف كيفية تفاعل النظام مع المستخدمين أو مع العناصر الخارجية الأخرى من خلال تعريف حالات مختلفة تقع فيها النظام. تعتمد مقدمة مخطط الحالة عادة على تحديد الكيفية التي يتغير فيها حال الكائنات أو الكيانات في النظام بناء على المحفزات الخارجية أو الأحداث الداخلية، مما يوفر فهً ما أعمق لسلوك النظام واحتمالات التفاعل معه.

1.4.II مخطط حالة الاستخدام العام



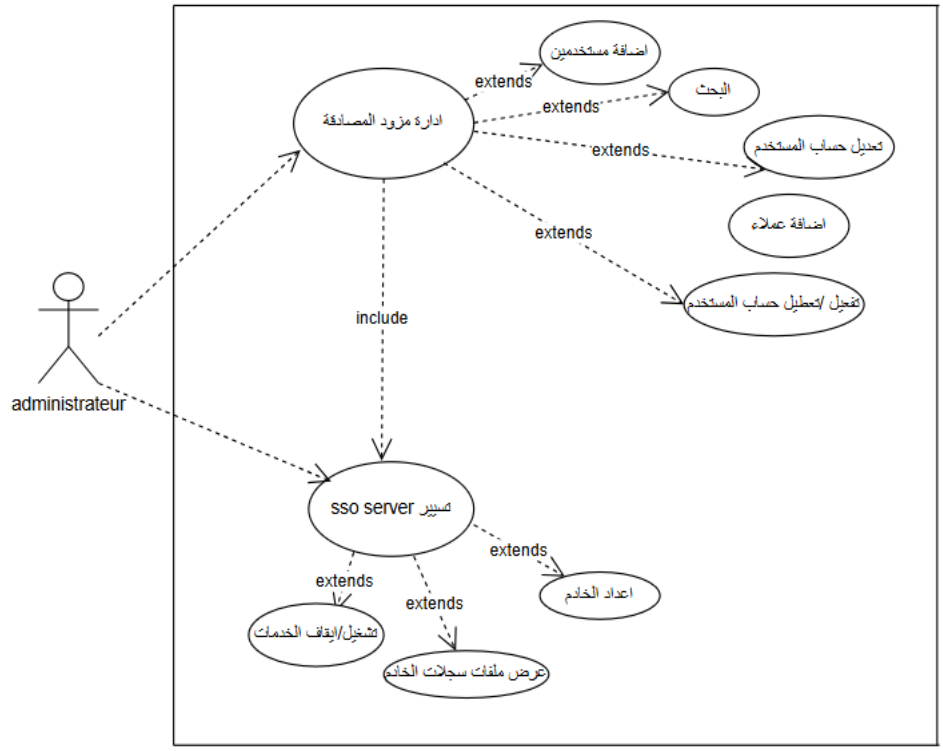
الشكل II-2: مخطط حالة الإستخدام العام

5.II تصميم المشروع

Actor	Cas d'utilisation	Description
Administrateur	إدارة مزود المصادقة	يقوم المسؤول (L'administrateur) بإدارة حسابات المستخدمين، وإجراء العمليات التالية: إضافة، تعديل، بحث، تفعيل، تعطيل، عرض، وتحديد صلاحيات حساب المستخدم
	إدارة خادم SSO	"يجب على مسؤول خادم SSO أن يقوم بالمصادقة مرتين: الأولى عبر مزود المصادقة، والثانية عبر قاعدة بيانات الخادم SSO الذي يديره.
Développeur	دمج خدمة الويب	تحديد إعدادات الخدمة الخاصة به (عنوان IP، اسم النطاق)
Utilisateur	المصادقة	يجب على جميع الجهات المعنية إجراء عملية المصادقة (التحقق من الهوية) قبل القيام بأي عملية
	استخدام الخدمة الويب	تختلف من مستخدم لآخر، حسب الوظيفة التي يشغلها (الأستاذ يقوم برفع نقاط الوحدة التي يُدرسها، والطالب يطّلع على نقاطه خلال السنة الجامعية الجارية)."

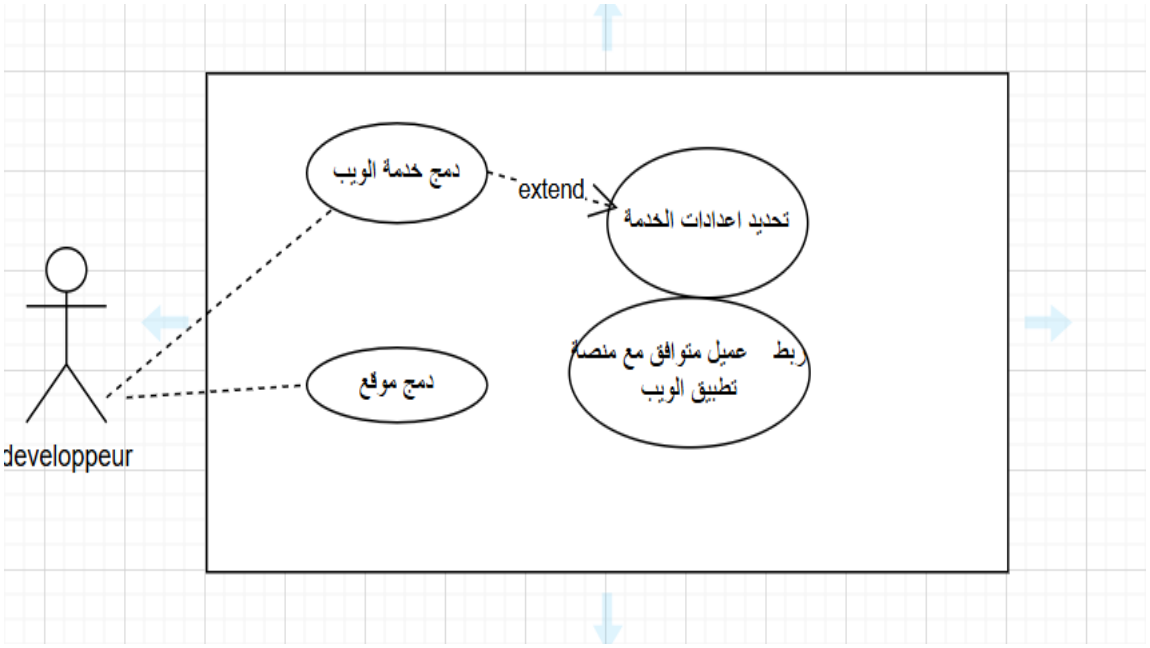
جدول II-1: تصميم المشروع

1.5.II مخطط حالة الاستخدام الخاص بالمسؤول



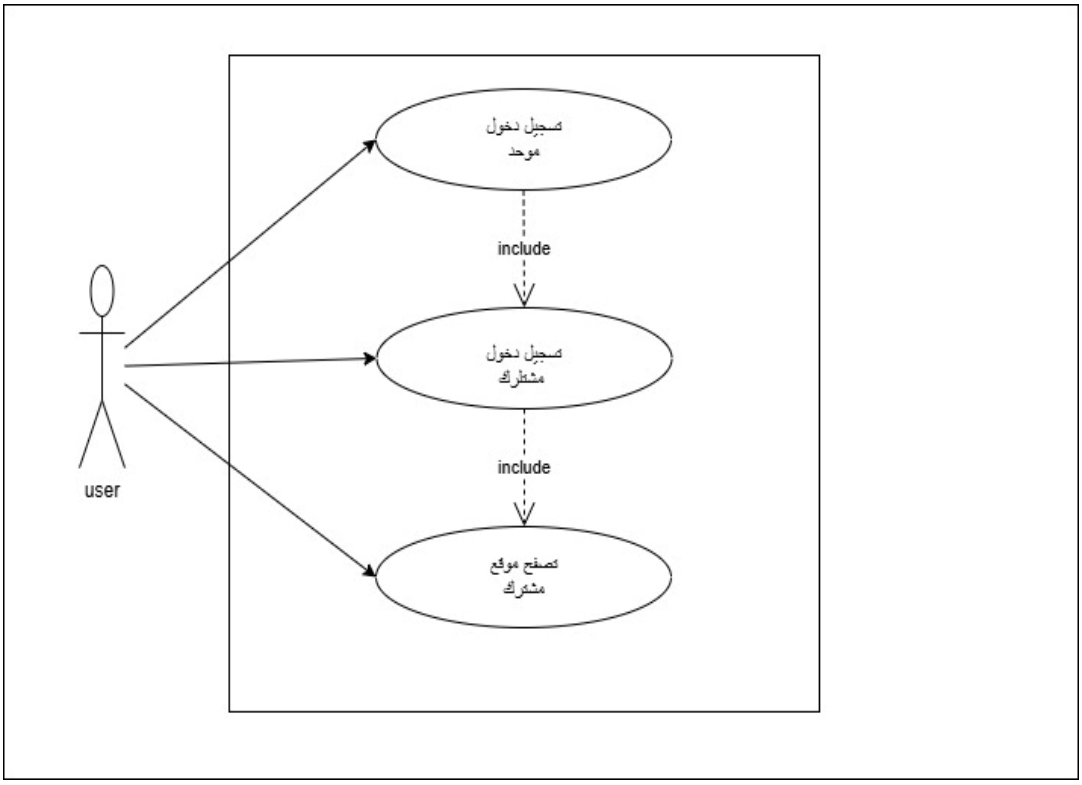
الشكل II-3: مخطط حالة الاستخدام الخاص بالمسؤول

II.2.5 مخطط حالة الاستخدام الخاص بالمطور



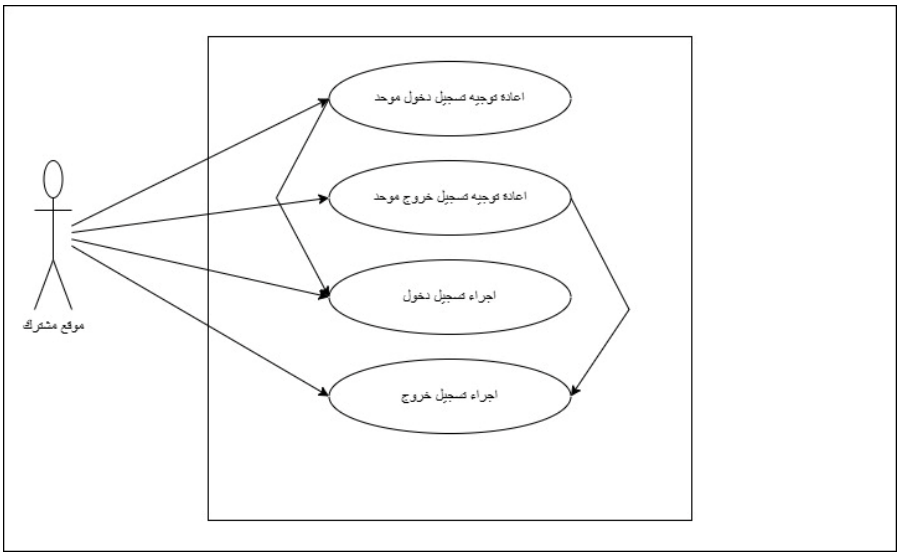
الشكل II-4: مخطط حالة الاستخدام الخاص بالمطور

II.3.5 مخطط حالة الاستخدام الخاص بالمستخدم



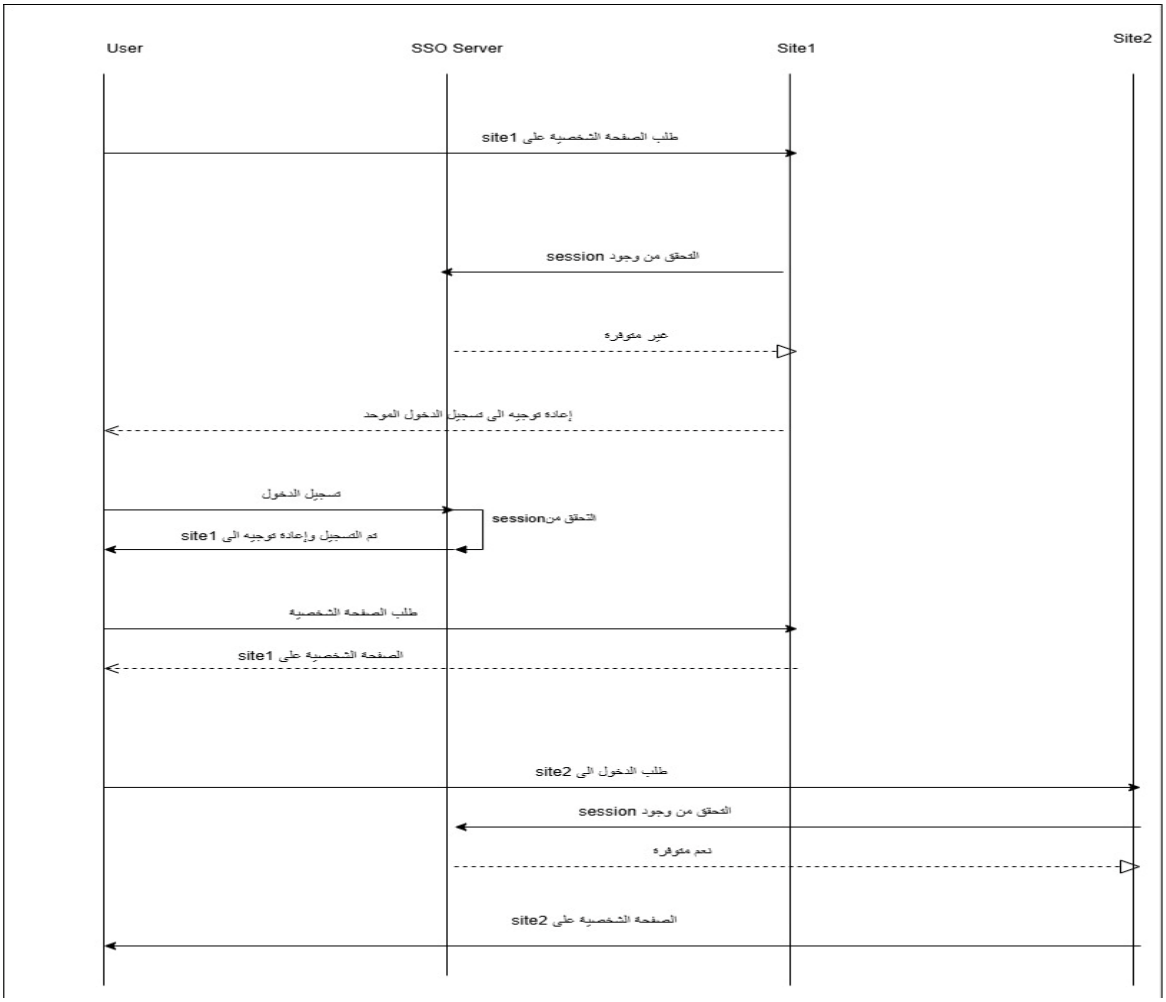
الشكل II-5: مخطط حالة الإستخدام الخاص بالمستخدم

II.4.5 مخطط حالة الاستخدام الخاص بالموقع المشترك



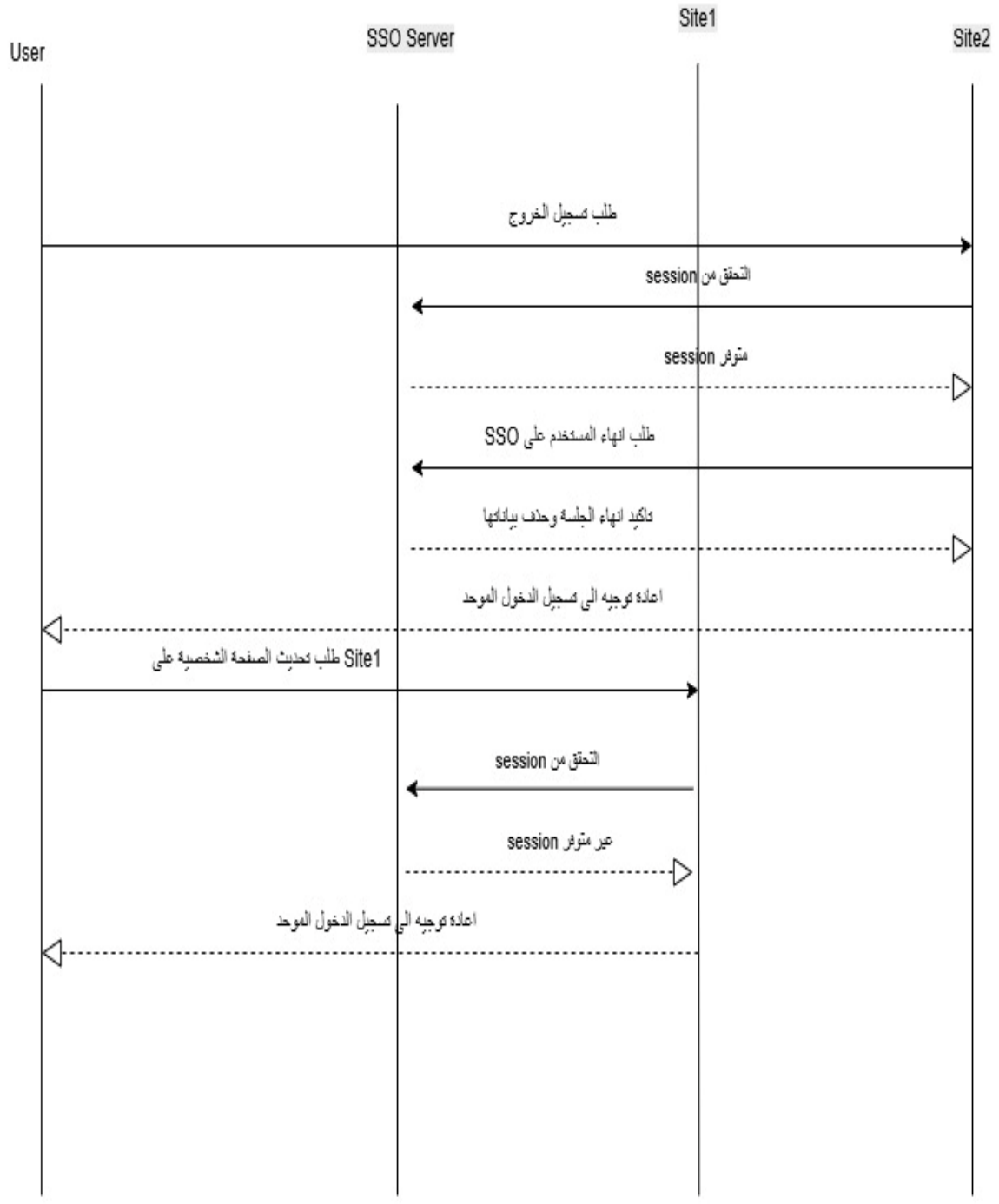
الشكل II-6: مخطط حالة الإستخدام الخاص بالموقع المشترك

II.5.5 مخطط التسلسل لعملية تسجيل الدخول



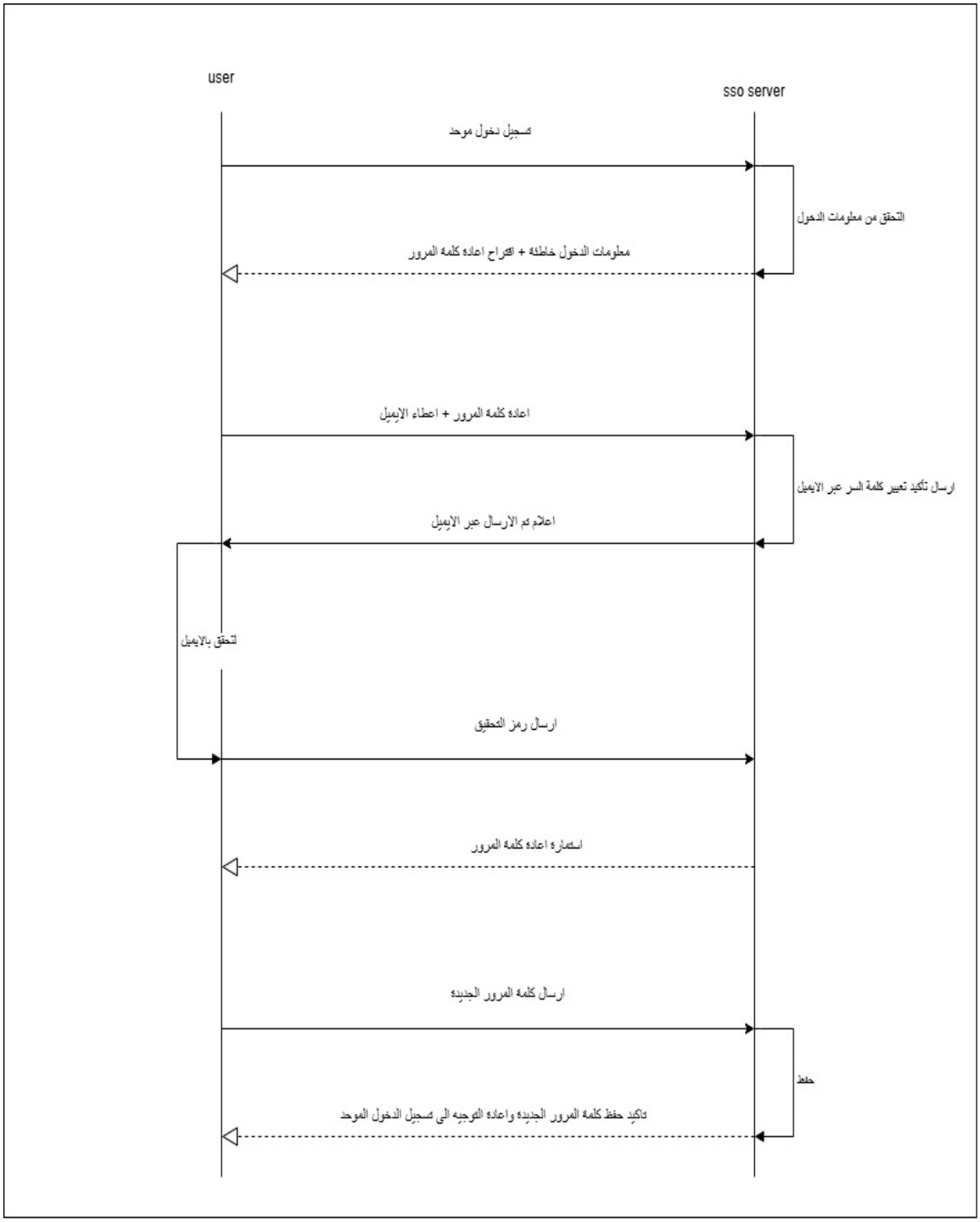
الشكل II-7: مخطط التسلسل لعملية تسجيل الدخول

II.6.5 مخطط التسلسل لعملية تسجيل الخروج



الشكل II-8: مخطط التسلسل لعملية تسجيل الخروج

II.7.5 مخطط التسلسل لعملية استرجاع كلمة المرور



الشكل II-9: مخطط التسلسل لعملية استرجاع كلمة المرور

II.5 الخاتمة

ختامًا، في هذا الفصل، قمنا بإجراء تحليل شامل ونمذجة دقيقة لمشروعنا بهدف تطوير التطبيق قمنا بتقديم إجابات شاملة عن التساؤلات المتعلقة بالتصور والنمذجة، حيث اعتمدنا على استخدام لغة (UML) كأداة رئيسية في تصميم مجموعة مخططات متنوعة تتضمن مخطط حالة الاستخدام، ومخططات النشاط، ومخطط الفئات تُعَد هذه النماذج خطوة مهمة نحو تحقيق أهداف التطوير بكفاءة ودقة.

الفصل الثالث: الإنجاز

III.1 مقدمة

بعد الانتهاء من تصميم ونمذجة النظام باستخدام لغة النمذجة الموحدة (UML)، يركّز هذا الفصل على الجوانب التقنية لتطوير التطبيق، مع استعراض الأدوات والتقنيات المعتمدة. تم اعتماد (Keycloak)كنظام لإدارة الهوية وتوفير آلية تسجيل الدخول الموحد (SSO) باستخدام بروتوكول (OpenID Connect)، مما أتاح تأمين المصادقة وتفويض الوصول بشكل مركزي. تم بناء الجانب الخلفي من التطبيق باستخدام لغة البرمجة (PHP)، مع توظيف قاعدة البيانات (MYSQL) لتخزين معلومات المستخدمين والتهيئة الخاصة بـ (Keycloak). تم استخدام Docker لتشغيل خدمات (Keycloak) و(MariaDB)ضمن حاويات معزولة، مما وفّر بيئة تطوير مرنة وسهلة النشر. كما تم تطوير التطبيق وتحرير الشيفرات البرمجية باستخدام بيئة التطوير (Visual Studio Code)، التي سهلت عمليات التكامل مع (Docker) وأدوات التحكم في الإصدارات. هذا التكوين التقني ساهم في تسريع عمليات التطوير، وتوفير بيئة موحّدة تضمن سهولة التشغيل والاختبار عبر مختلف المنصات.

2.III مكونات بيئة العمل

3.III اختيار (Keycloak)

بناءً على نتائج المقارنة، تم اختيار نظام Keycloak كحل معتمد لتنفيذ المشروع المقترح، نظراً لما يوفره من مرونة، دعم بروتوكولات حديثة، وتكامل سهل مع الأنظمة المختلفة، إضافة إلى كونه مدعوماً من مجتمع قوي ويملك وثائق تقنية شاملة

4.III الأدوات ولغات البرمجة المستخدمة

1.4.III (PHP)

هي لغة برمجة سكريبت تستخدم أساسا لتطوير تطبيقات الويب وتفاعلها مع قواعد البيانات[6]

2.4.III جهاز خادم

في إطار تنفيذ وتقييم النظام المقترح، تم استخدام خادم (Server) يتمتع بالمواصفات التقنية التالية:

- **اسم المضيف**: (Hostname) pais-server
- **نظام التشغيل**: Ubuntu 22.04.5 LTS
- **إصدار النواة**: (Kernel) 6.8.0-57-generic
- **المعمارية**: x86_64
- **نوع المعالج**: Intel(R) Core (TM) i5-8400 CPU بسرعة GHz2.80
- **عدد الأنوية**: 6 أنوية
- **الذاكرة العشوائية**: 7.6 (RAM) جيجابايت
- **سعة القرص (الجذر)**: 916 جيجابايت

III.4.3 حاسوب شخصي

تم تنفيذ الاختبارات والتطوير البرمجي للمشروع على جهاز حاسوب محمول من نوعMacBook Pro 13 ، والذي يتميز بالمواصفات التقنية التالية:

- المعالج: (Intel® Core i7-1068NG7) الجيل العاشر، بسرعة أساسية(GHz2.30) وتصل إلى (GHz4.10) باستخدام تقنية (Turbo Boost)، مزود بذاكرة ذكية (Smart Cache) بحجم 8 ميجابايت.
- الذاكرة العشوائية :32 (RAM) جيجابايت.
- وحدة التخزين: قرص صلب بسعة تخزين 1000 جيجابايت نوع.(SSD)
- بطاقة الرسومات:(Intel Iris Plus Graphics)
- الشاشة: (Retina)بقياس 13.3 بوصة، وبدقة عالية تبلغ 2880 × 1800 بكسل.
- نظام التشغيل (macOS Big Sur :أحدث إصدار وقت تنفيذ المشروع.)

تساهم هذه المواصفات في توفير أداء مستقر وسلس أثناء العمل على المشروع، خاصة في المهام المتعلقة ببيئة التطوير، اختبار الأنظمة، وتكامل الخدمات

III.5 التكنولوجيا المستخدمة

III.5.1 (Docker)

هو منصة تكنولوجيا الحاويات التي تستخدم لتطوير ونشر وتشغيل التطبيقات بسرعة[7]

III.5.2 (GitHub)

هو منصة قائمة على السحابة حيث يمكنك تخزين الأكواد البرمجية ومشاركتها والعمل مع الآخرين لكتابتها بشكل تعاوني[8]

III.5.3 (Keycloak)

هو نظام مفتوح المصدر لإدارة الهوية والوصول (Identity and Access Management - IAM) يسمح لك بإضافة ميزة تسجيل الدخول والتفويض لتطبيقاتك بسهولة، دون الحاجة إلى بناء نظام مصادقة من الصفر.

III.5.4 (Visual Studio Code)

هو محرر شيفرة مصدرية (Code editor) مجاني ومفتوح المصدر، تم تطويره من قبل شركة (Microsoft) يستخدم لكتابة، تعديل، وتصحيح الاكواد البرمجية بلغات متعددة مثل(php)، (java) وغيرها

III.5.5 (MySql)

هو نظام إدارة قواعد البيانات الشهير ومفتوح المصدر. يتميز بسهولة الاستخدام والأداء العالي. يستخدم على نطاق واسع في تطبيقات الويب وغيرها [19]

III.6 شرح النظام

في إطار تطوير نظام تسجيل دخول موحد (Single Sign-On) يعتمد على بروتوكول (OpenID Connect)، تم اعتماد منصة (Keycloak) كنظام لإدارة الهوية والوصول. ولضمان بيئة تطوير وتشغيل مرنة، موحدة، وقابلة للنقل، تم توظيف تقنية (Docker) لتغليف وتشغيل مكونات النظام داخل حاويات (Containers) معزولة.

أُنشئت حاوية خاصة بخادم (MariaDB) لتعمل كقاعدة بيانات تخزّن معلومات التهيئة والمستخدمين الخاصة بـ (Keycloak) كما تم إعداد حاوية أخرى مخصصة لـ (Keycloak) نفسه، حيث تم ضبطها للاتصال بقاعدة البيانات عبر شبكة داخلية باستخدام (Docker Compose)، مما سهّل عملية التهيئة والتشغيل.

تم استعمال أداة Docker Compose لتجميع جميع الخدمات الضرورية وتشغيلها بشكل متزامن، من خلال ملف تكوين موحّد، مما مكّن الفريق من تشغيل كامل النظام عبر أمر واحد فقط، دون الحاجة إلى تثبيت كل خدمة على حدة.

بعد تشغيل النظام، تم الولوج إلى لوحة تحكم (Keycloak) عبر منفذ(https://41.111.217.17:8080/)، وإنشاء (Realm) خاص بالمشروع، بالإضافة إلى تعريف (Client) يمثل التطبيق الخارجي. في هذا السياق، تم تطوير تطبيق ويب بلغة (PHP) واستعمال مكتبة (jumbojett/openid-connect-php) للتكامل مع(Keycloak) عبر بروتوكول (OpenID Connect) يُوجَّه المستخدم إلى خادم (Keycloak) لتسجيل الدخول، وبعد المصادقة الناجحة يتم إعادة توجيهه إلى التطبيق مرفقًا برموز المصادقة (Tokens).

بفضل استخدام (Docker)، تم ضمان قابلية التشغيل الموحد عبر مختلف البيئات (تطوير، اختبار، إنتاج)، وتقليص الوقت والجهد المرتبطين بإعداد البنية التحتية، بالإضافة إلى تسهيل مشاركة النظام بين أعضاء الفريق.

III.7 طريقة عملية تسجيل الدخول الموحد

- المستخدم يطلب صفحة الويب من (Site1):

أولاً، يقوم المستخدم بفتح المتصفح ويطلب صفحة من (Site1).

- خادم المصادقة (SSO Server) يتعامل مع الطلب:

عند محاولة الوصول إلى (site1) لأول مرة، إذا لم يكن المستخدم قد سجل الدخول مسبقًا، سيُوجهه خادم المصادقة (Keycloak) لصفحة تسجيل الدخول.

- إعادة توجيه إلى صفحة تسجيل الدخول:

يُرسل المستخدم إلى خادم المصادقة (Keycloak) حيث يُطلب منه إدخال بياناته (اسم المستخدم وكلمة المرور) بعد التحقق الناجح، يتم إنشاء جلسة (Session) جديدة للمستخدم.

- عودة المستخدم إلى (site1) بعد تسجيل الدخول:

بمجرد التحقق بنجاح من (Keycloak)، يتم إعادة توجيه المستخدم إلى(site1) مع جلسة (Session) نشطة، وبالتالي يُعتبر المستخدم مسجل دخول إلى الموقع.

- -طلب صفحة الويب من site2 :

بعد ذلك، إذا حاول المستخدم الوصول إلى site2، يتم التحقق من وجود جلسة نشطة في خادم المصادقة.

- -إعادة توجيه المستخدم إلى site2 :

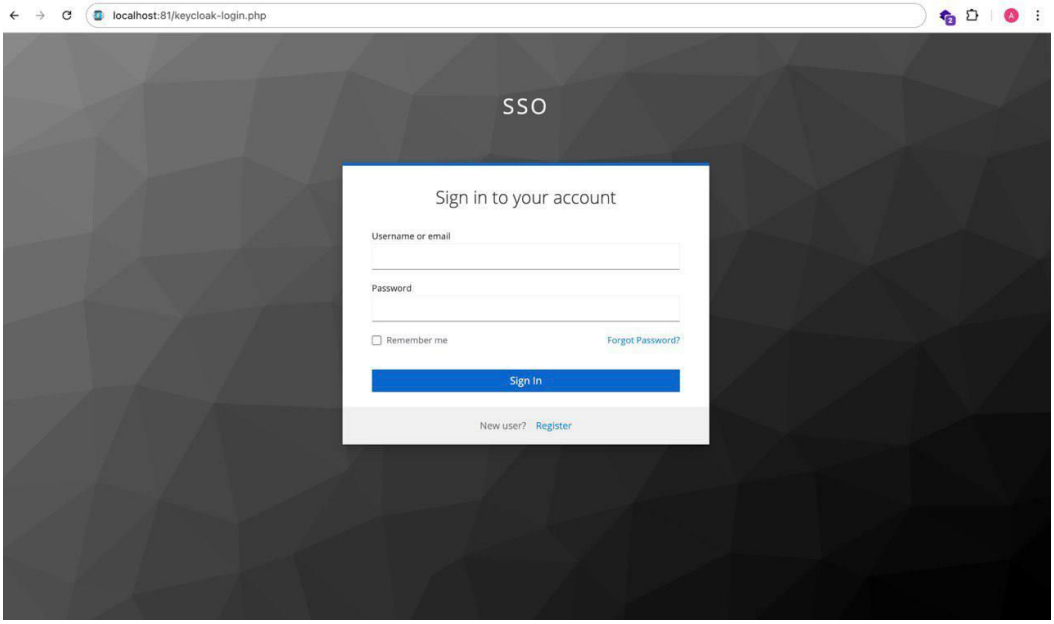
بما أن المستخدم قد سجل دخوله مسبقًا في (Keycloak) (خدمة المصادقة المركزية)، سيقوم الخادم Keycloak بتمرير جلسة المصادقة نفسها إلى site2 دون الحاجة لإدخال بيانات تسجيل الدخول مجددًا.

- -إنشاء جلسة على(site2) :

يتم إنشاء جلسة جديدة على (site2) بناءً على الجلسة السابقة من (Keycloak)، مما يتيح للمستخدم الوصول بسهولة إلى site2 بدون الحاجة لتسجيل الدخول مرة أخرى.

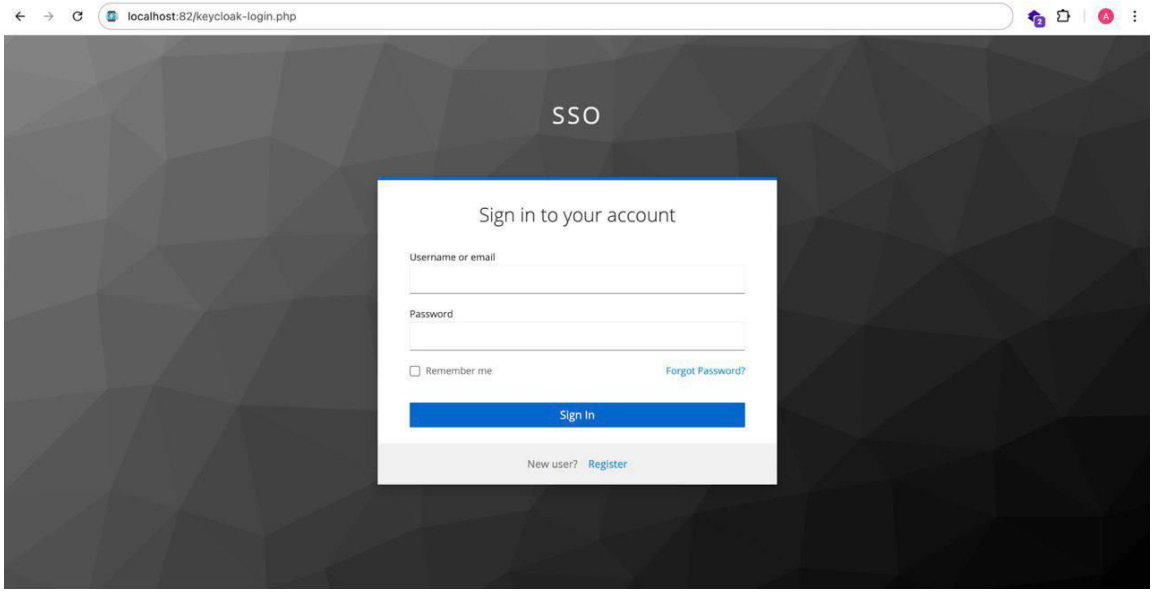
III.8 الواجهات الرئيسية للتسجيل دخول موحد

III.8.1 واجهة تسجيل دخول موقع



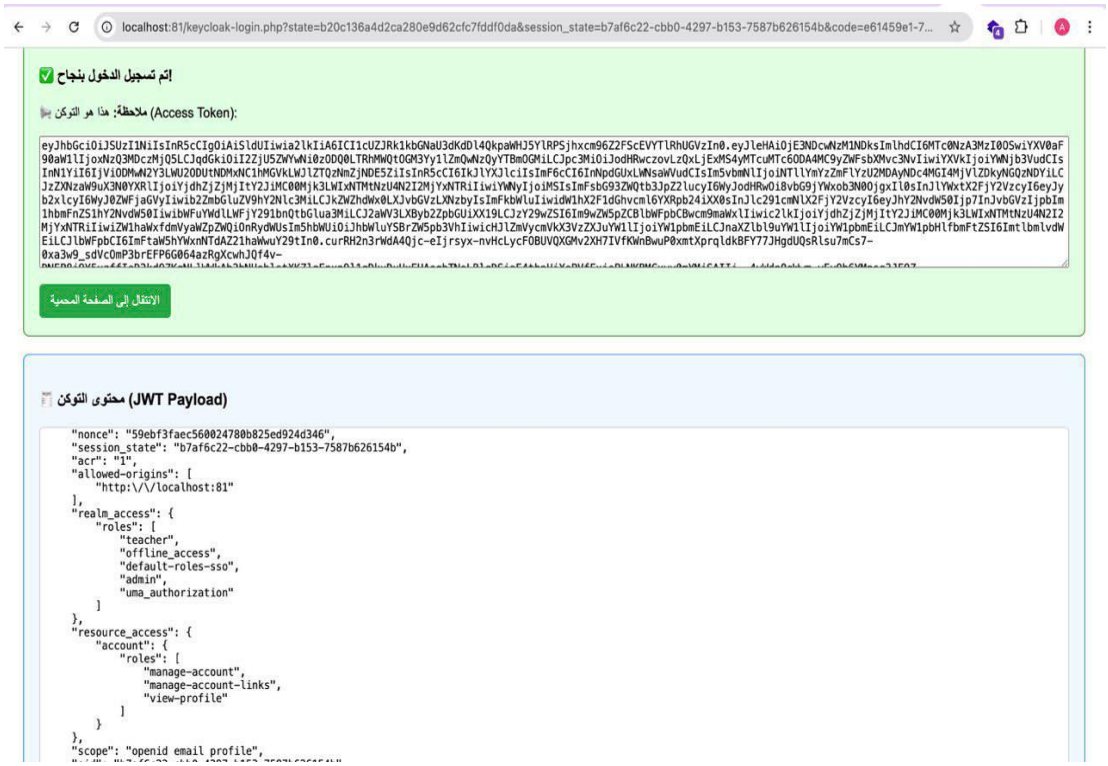
الشكل III-1: واجهة تسجيل دخول موقع 1

III.2.8 واجهة تسجيل دخول موقع 2



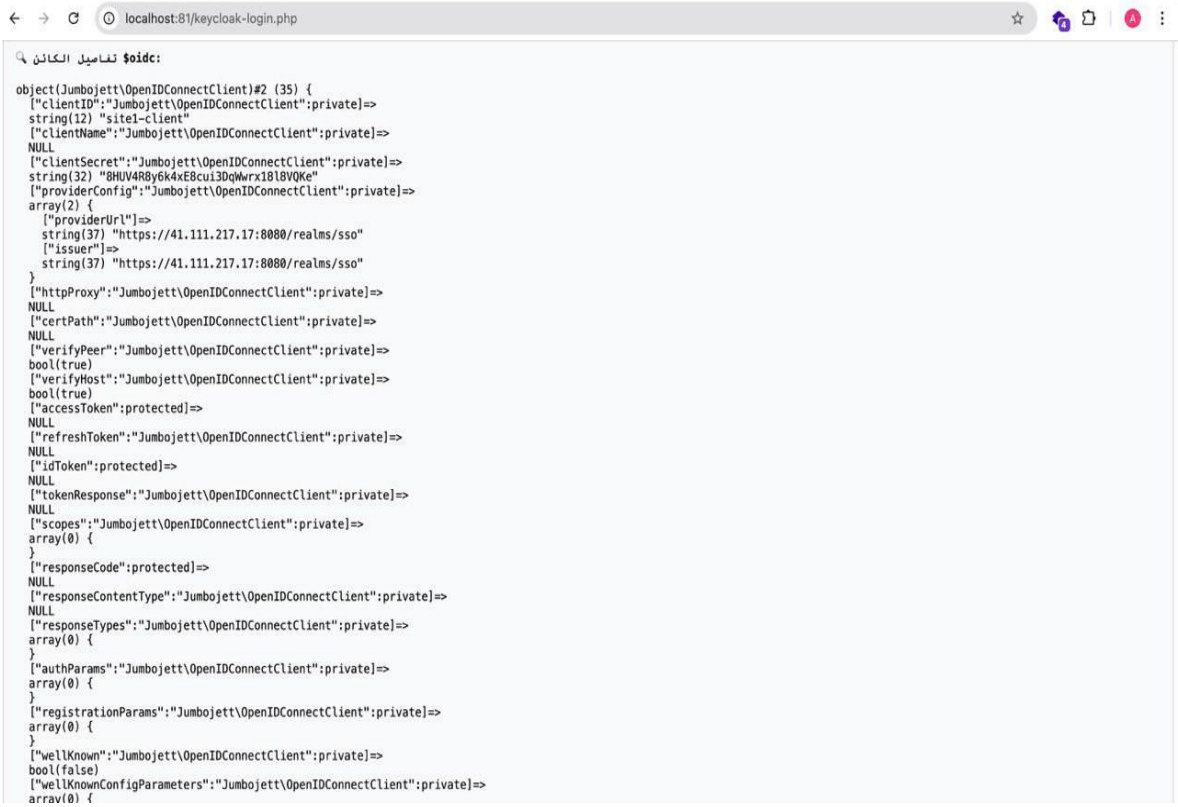
الشكل III-2: واجهة تسجيل دخول موقع 2

III.3.8 واجهة تسجيل الدخول الناجح باستخدام (Keycloak)وعرض رمز التوكن ومحتواه (JWT Payload)



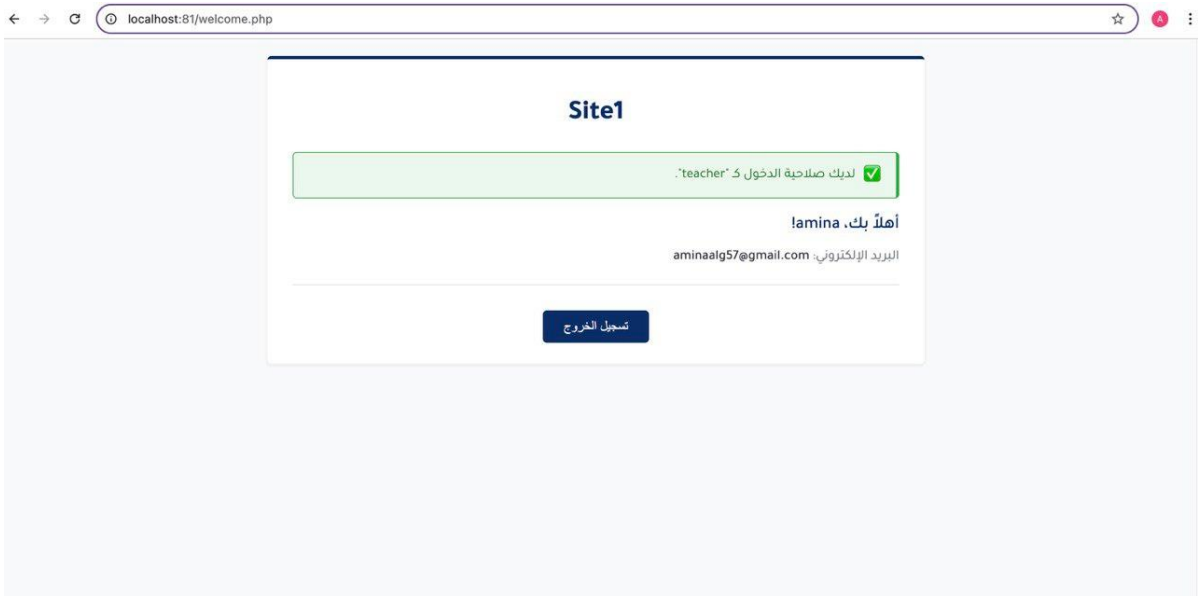
الشكل III-3: واجهة تسجيل الدخول الناجح باستخدام (Keycloak)وعرض رمز التوكن ومحتواه (JWT Payload)

4.8.III jwt session token ,(user logged in)



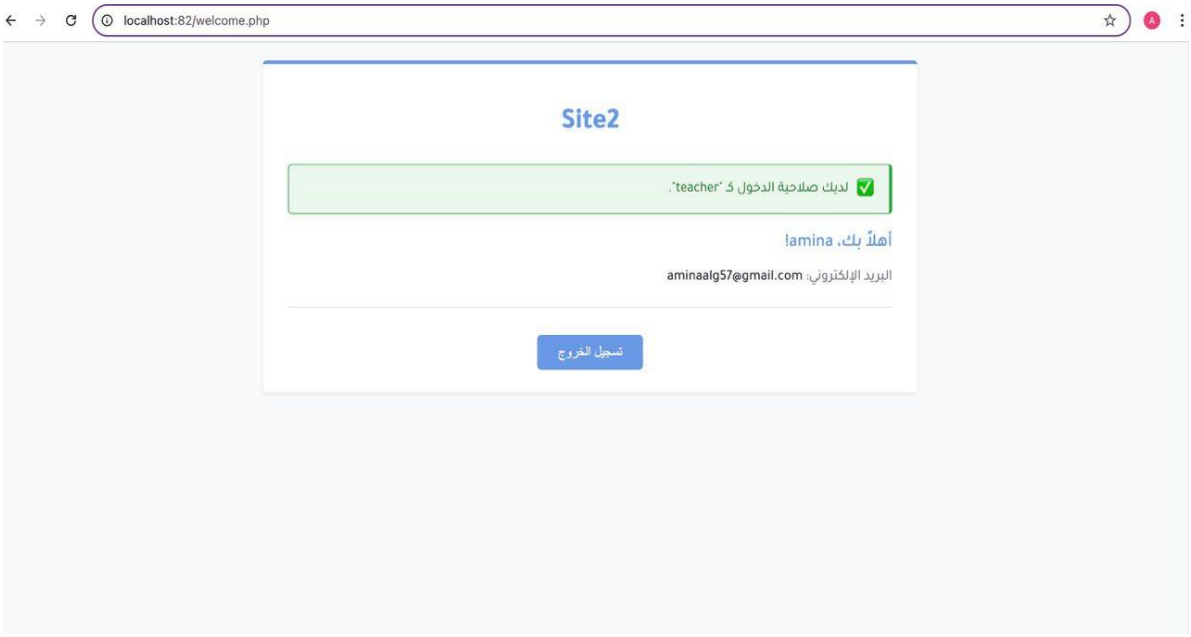
الشكل III-4: jwt session token ,(user logged in)

5.8.III واجهة welcome موقع1 لديه صلاحية



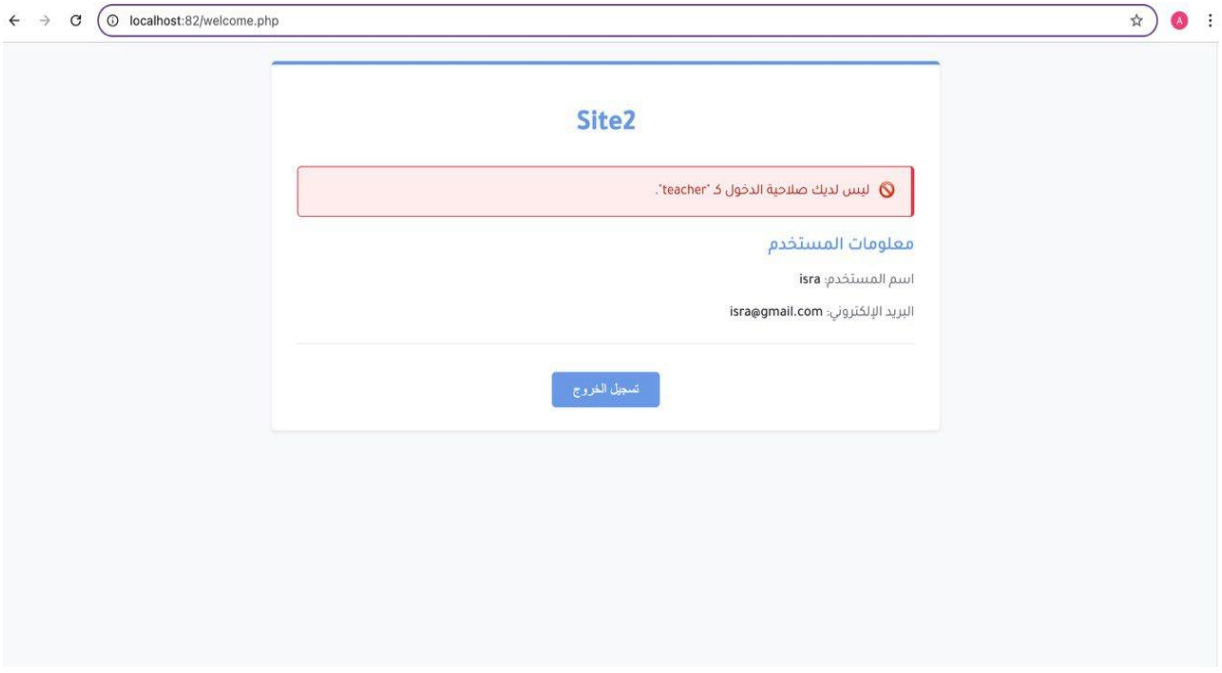
الشكل III-5:واجهة welcome موقع1 لديه صلاحية

III.8.6 واجهة welcome موقع 2 لديه صلاحية



الشكل III-6: واجهة welcome موقع 2 لديه صلاحية

III.8.7 واجهة welcome موقع 2ليست لديه صلاحية



الشكل III-7: واجهة welcome موقع 2ليست لديه صلاحية

III.8.8 واجهة welcome موقع 1ليست لديه صلاحية

الشكل III-8: واجهة welcome موقع 1ليست لديه صلاحية

III.9 الخاتمة

في ختام هذا الفصل، تم التطرق بشكل مفصل إلى مراحل تطوير النظام وتطبيق مفهوم تسجيل الدخول الموحد (SSO) باستخدام (Keycloak)، مع استعراض الأدوات والتقنيات المعتمدة مثل (PHP)، (Docker)، و(Spring Boot)، بالإضافة إلى بيئة التطوير (Visual Studio Code) ومنصة (GitHub) وقد تم التركيز على البنية التقنية التي تم من خلالها إنجاز التطبيق، وعلى كيفية دمج (Keycloak) كنظام إدارة الهوية والوصول ضمن البنية الحالية.

كما تم تقديم شرح للواجهات الرئيسية للتطبيق من خلال أمثلة مرئية توضّح عملية تسجيل الدخول، إنشاء الحساب، والتحقق من الصلاحيات ضمن بيئات متعددة. هذه الواجهات تعكس التطبيق العملي لنموذج المصادقة الموحدة، ومدى فاعليته في تحسين تجربة المستخدم وضمان التحكم الآمن في الوصول إلى الموارد.

يمكن القول إن هذا الفصل يجسّد الجانب التطبيقي من الدراسة، حيث تُرجم التصميم المفاهيمي إلى نظام فعلي قابل للاستخدام، ويُعتبر خطوة عملية مهمة نحو اعتماد حلول المصادقة المركزية في بيئات المؤسسات الحديثة.

الخاتمة العامة

في عصر يتسارع فيه التحول الرقمي، لم يعد الحديث عن الأمان وسهولة الاستخدام في الأنظمة المعلوماتية ترفًا تقنيًا، بل ضرورة حتمية لنجاح المؤسسات واستدامة خدماتها. ومن هذا المنطلق، جاءت هذه المذكرة لتتناول واحدة من أبرز الإشكاليات التي تواجه المؤسسات التعليمية والإدارية على حد سواء، والمتمثلة في تعدد أنظمة المصادقة وصعوبة إدارتها، لتُقترح في المقابل حلاً ذكيًا وفعالًا يتمثل في نظام الدخول الموحد (Single Sign-On - SSO).

انطلقت الدراسة من تشخيص واقعي للوضع الحالي في جامعة الوادي، من خلال تحليل البنية التحتية الرقمية المعتمدة، وتحديد مكامن الضعف في آليات المصادقة التقليدية. ثم تم التعمق في النمذجة المنهجية للنظام باستخدام أدوات (UML)، مما أتاح رسم تصور شامل لبنية النظام من وجهة نظر تحليلية وهندسية دقيقة. وبعدها، تُوجت الدراسة بمرحلة الإنجاز العملي، حيث تم تطوير وتنفيذ نظام (SSO)اعتمادًا على منصة (Keycloak)المفتوحة المصدر، مستغلين ما توفره من مرونة، أمان، ودعم للبروتوكولات العالمية مثل (OpenID Connect) و(OAuth 2.0) و(SAML 2.0).

ولم تكن هذه الرحلة الأكاديمية خالية من التعقيدات، بل واجهنا خلالها جملة من التحديات التقنية والعملية، بدءًا من استيعاب المفاهيم المعقدة في مجال إدارة الهوية والوصول، مرورًا بتركيب البيئة البرمجية وتكامل الأدوات، وانتهاءً بمعالجة المشاكل غير المتوقعة أثناء التجارب والاختبارات. لكن بالرغم من هذه العراقيل، تمكنا بعون الله، ثم بالإصرار والمثابرة، من الوصول إلى نموذج أولي فعّال يحقق الأهداف المرجوة ويوفر تجربة استخدام سلسة وآمنة.

هذا المشروع لا يقتصر على كونه تمرينًا أكاديميًا، بل يُعد نواة حقيقية قابلة للتوسيع والتبني على نطاق واسع داخل الجامعة أو حتى في مؤسسات أخرى تسعى لتوحيد إجراءات الدخول لمستخدميها. ومن خلال هذا الإنجاز، أُثبت أن دمج التكنولوجيا الحديثة، مثل (Keycloak)، مع فهم دقيق لاحتياجات المستخدم والهيكل التنظيمي، يمكن أن ينتج عنه نظام آمن، مرن، وسهل الاستخدام في آن واحد.

بناءً على ما تم إنجازه في هذا المشروع، نوصي بالتوسع في اعتماد نظام الدخول الموحد (SSO) ليشمل مختلف الخدمات الرقمية بالجامعة، مع تعزيز الأمان عبر دمج المصادقة متعددة العوامل (MFA). كما يُنصح بتحسين تجربة المستخدم من خلال واجهات مبسطة وتوثيق مرئي، واعتماد منصة موحدة للدعم التقني. ومن المهم متابعة تحديثات (Keycloak) واستغلال ميزاته الأمنية، إضافة إلى إنشاء وحدة تحليل للولوجات المشبوهة. مستقبلاً، يمكن تعميم هذا النموذج على مؤسسات تعليمية أو إدارية أخرى ذات بيئة رقمية مشابهة لتحقيق إدارة مركزية وآمنة للهويات الرقمية

ختامًا، تمثل هذه المذكرة تجسيدًا لفلسفة "الهندسة النافعة"، حيث لا ينحصر الهدف في حل مشكلة تقنية فحسب، بل في تقديم قيمة حقيقية تُمكّن المستخدم وتُبسط الإدارة وتُحصّن المنظومة الأمنية للمؤسسة. ويبقى الأمل أن تكون هذه الخطوة بداية لمزيد من مشاريع التحول الرقمي الفعّالة في جامعتنا ومؤسساتنا الوطنية.

المراجع

[1] Mozilla Contributors. (n.d.). MDN Web Docs – HTTP Guides. Mozilla. https://developer.mozilla.org/en-US/docs/Web/HTTP

[2] Gourley, D., & Totty, B. (2002). HTTP: The Definitive Guide. O'Reilly Media. https://dl.ebooksworld.ir/books/HTTP.The.Definitive.Guide.Brian.Totty.David.Gourley.OReilly.9781565925090.EBooksWorld.ir.pdf

[3] DocDis. (n.d.). Learn JSON Web Tokens. GitHub. https://github.com/docdis/learn-json-web-tokens

[4] Bilbie, A. (n.d.). A Guide To OAuth 2.0 Grants. alexbilbie.com. Retrieved ~2014. https://alexbilbie.com/guide-to-oauth-2-grants

[5] Object Management Group (OMG). (2017). UML Specification Version 2.5.1. https://www.omg.org/spec/UML/2.5.1

[6] The PHP Group. (n.d.). PHP: Hypertext Preprocessor. php.net. /https://www.php.net

[7] Docker Inc. (n.d.). Docker: Accelerated Container Application /Development. https://www.docker.com

[8] GitHub. (n.d.). About GitHub and Git – GitHub Docs. GitHub Docs. https://docs.github.com/en/get-started/using-git/about-git

[9] Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., & Jay, E. (2013). OpenID Connect. Mutually Human Blog. /http://www.mutuallyhuman.com/blog/2013

[10] Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., & Jay, E. (2011). OpenID Connect Standard 1.0 – Draft 20. OpenID Foundation. https://openid.net/specs/openid-connect-core-1_0.html

[11] Reede, J. (2007). On A-Select and Federated Identity Management /Systems [Unpublished paper]. http://www.mutuallyhuman.com/blog/2013

[12] Howes, T. A., Smith, M. C., & Good, G. S. (2003). Understanding and .Deploying LDAP Directory Services. Addison-Wesley

[13] Vanmeegern. (n.d.). Web Development with Node and Express [PDF]. https://www.vanmeegern.de/fileadmin/user_upload/PDF/Web_Developme nt_with_Node_Express.pdf

[14] الزبيدي، أ., وآخرون. (2020). نظرة على تفاعلات الخادم مع العميل في موقع ويب ديناميكي. أكاديمية حسوب. https://academy.hsoub.com/devops/servers/نظرة-على-تفاعلات-الخادم-مع-العميل-في-موقع-ويب-ديناميكي-r782

[15] Jasig Community. (n.d.). CAS Protocol Specification – CAS Server. Jasig. https://jasigcas.readthedocs.io/en/latest/cas-server-documentation/protocol/CAS-Protocol-Specification.html

[16] Mozilla Contributors. (n.d.). Redirections – HTTP Guides. Mozilla. https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Redirections

[17] Packt Publishing. (2023). Web Development [Free eBook]. https://packt.link/free-ebook/9781804616444

[18] Oracle Corporation. (n.d.). MySQL Documentation. mysql.com. https://dev.mysql.com/doc/

[19] Oracle Corporation (MySQL Team). (n.d.). MySQL Documentation. /Oracle / mysql.com. https://dev.mysql.com/doc

الملحقة

حمل Docker Desktop:

من الموقع الرسمي: https://www.docker.com/products/docker-desktop/2.

ثبت Docker Desktop:

افتح ملف التثبيت (Installer) واتبع التعليمات.

بعد التثبيت، شغل Docker وانتظر حتى تظهر أيقونة الحوت في شريط المهام.

تحقق من التثبيت:

افتح Terminal أو CMD ونفّذ:

```
docker --version
docker run hello-world
```

تثبيت وتشغيل Keycloak

في هذا القسم، ستتعلّم بسرعة كيفية تثبيت وتشغيل Keycloak. وبمجرد تشغيل Keycloak، سنلقي نظرة على وحدة تحكم المسؤول (Keycloak Admin Console) ووحدة تحكم الحساب (Keycloak Account Console).

يوفر Keycloak عدة خيارات للتثبيت، ومنها ما يلي:

التشغيل كحاوية باستخدام Docker

التثبيت والتشغيل محليًا (ويتطلب وجود آلة Java الافتراضية مثل OpenJDK)

التشغيل على Kubernetes

استخدام مشغّل Keycloak الخاص بـ Kubernetes (Keycloak Kubernetes Operator)

إذا كان Docker مثبتًا على جهازك، فإن هذا الخيار يُعد الأفضل لأنه الأسهل والأسرع لبدء الاستخدام.

أما إذا لم يكن Docker مثبتًا، فمن الأسهل البدء بتثبيت Keycloak وتشغيله محليًا، حيث إن الاعتماد الوحيد المطلوب هو وجود آلة Java الافتراضية.

كما يمكن نشر Keycloak بسهولة على Kubernetes، مع إمكانية استخدام مشغّل Keycloak الخاص بـ Kubernetes، والذي يجعل من عملية التثبيت والتكوين والإدارة أمرًا أكثر بساطة.

ومع ذلك، لن نقدم تعليمات حول Kubernetes في هذا الكتاب، لأن تركيزنا سيكون على Keycloak وميزاته إذا كنت مهتمًا بمعرفة كيفية تشغيل Keycloak على Kubernetes، فإن موقع Keycloak يوفر أدلة ممتازة للبدء من خلال الرابط التالي:

https://www.keycloak.org/guides#getting-started

في القسم التالي، سنستعرض كيفية تشغيل Keycloak كحاوية باستخدام Docker.

وإذا كنت تفضل تشغيله محليًا، يمكنك الانتقال مباشرة إلى القسم المعنون تثبيت وتشغيل Keycloak باستخدام OpenJDK.

تشغيل Keycloak باستخدام Docker

باستخدام Docker، يصبح من السهل جدًا تشغيل Keycloak، حيث أنك لست بحاجة إلى تثبيت آلة جافا الافتراضية (JVM) بنفسك، ولا إلى تحميل وفك ضغط توزيعة Keycloak يدويًا.

لتشغيل Keycloak على Docker، ببساطة نفذ الأمر التالي:

```
$ docker run -e KEYCLOAK_ADMIN=admin -e KEYCLOAK_ADMIN_PASSWORD=admin -p
8080:8080 quay.io/keycloak/keycloak:22.0.0 start-dev
```

ملاحظة:

بما أن Keycloak لا يأتي مع حساب مسؤول (Admin) افتراضي، فإن تمرير متغيرات البيئة KEYCLOAK_ADMIN وKEYCLOAK_ADMIN_PASSWORD يسمح بإنشاء حساب مسؤول ابتدائي بسهولة.

كما أننا نستخدم الخيار p 8080:8080- لنشر المنفذ الذي يستخدمه Keycloak على الجهاز المضيف، مما يسهل الوصول إلى Keycloak.

بعد بضع ثوانٍ من تشغيل الأمر، يمكنك التحقق من أن Keycloak يعمل عبر فتح الرابط التالي في المتصفح:

http://localhost:8080/admin

ثم تسجيل الدخول باستخدام:

اسم المستخدم: admin

كلمة المرور: admin

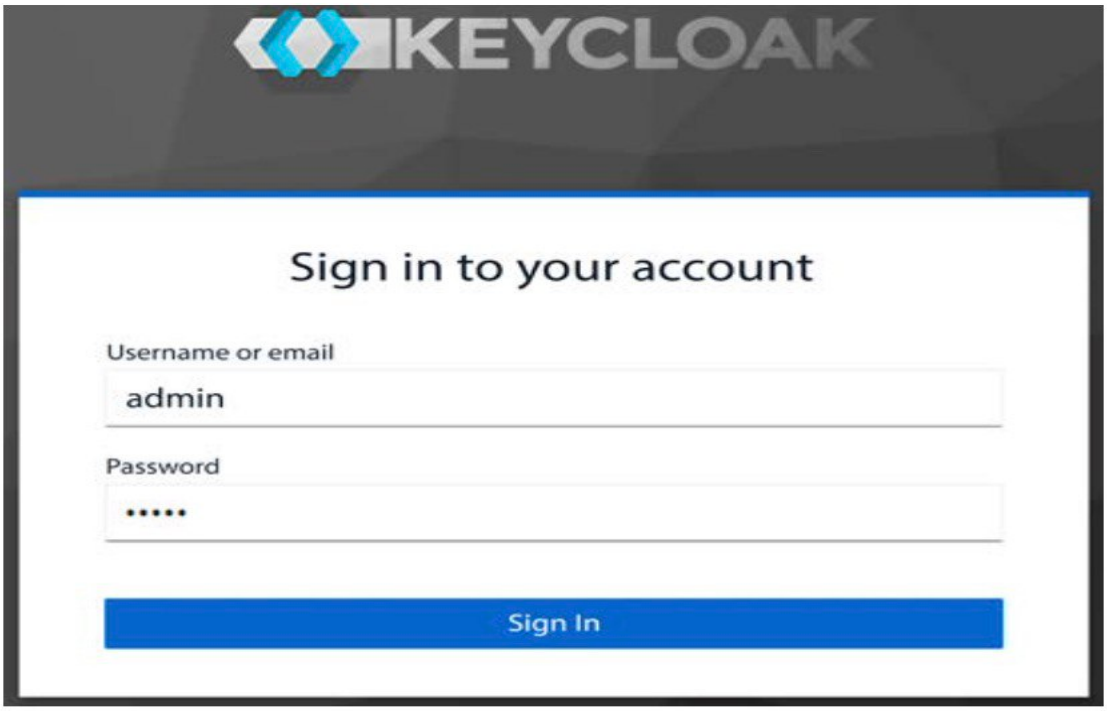


Figure 1.1: Log in to the administration console as the admin user

البدء باستخدام وحدة تحكم المسؤول في Keycloak

توفر وحدة تحكم المسؤول في Keycloak واجهة شاملة وسهلة الاستخدام للمسؤولين والمطورين لتكوين وإدارة Keycloak.

للوصول إلى وحدة تحكم المسؤول، افتح الرابط التالي في المتصفح:

http://localhost:8080/admin

ستتم إعادة توجيهك إلى صفحة تسجيل الدخول الخاصة بـ Keycloak، حيث يمكنك تسجيل الدخول باستخدام اسم المستخدم وكلمة المرور (admin/admin) اللذين قمت بإنشائهما أثناء تثبيت Keycloak في القسم السابق.

بمجرد تسجيل الدخول، ستتمكن من رؤية إعدادات التكوين الخاصة بـ النطاق الرئيسي (Master Realm) في Keycloak، كما هو موضح في لقطة الشاشة التالية:

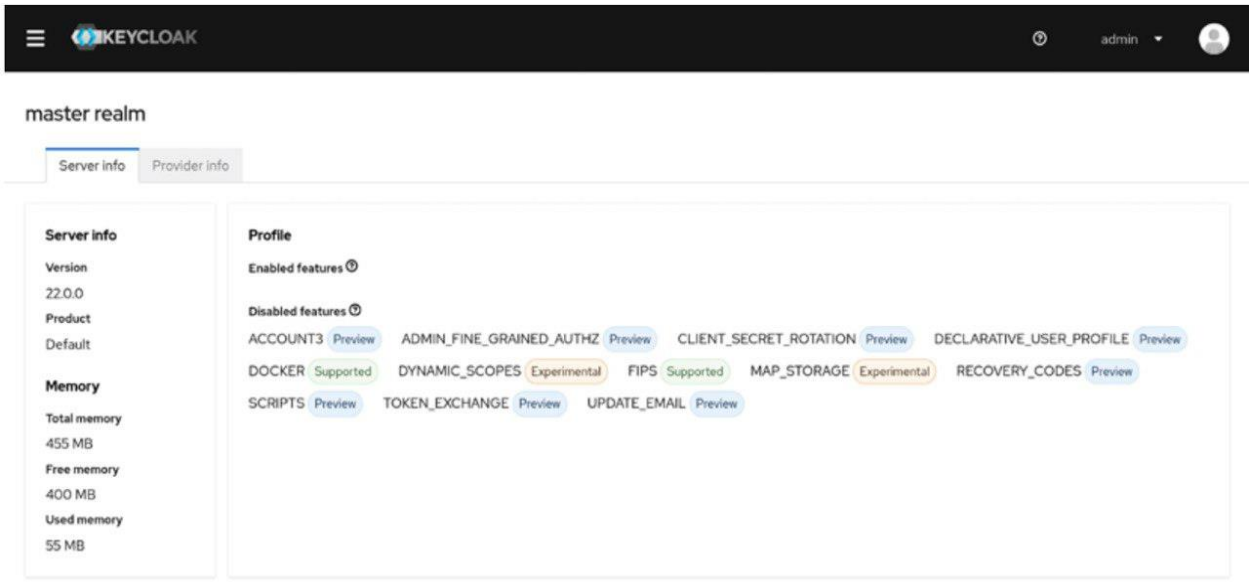


Figure 1.2: The Keycloak admin console

تثبيت وتشغيل Keycloak باستخدام OpenJDK

نظرًا لأن Keycloak مبرمج بلغة Java، فإنه من السهل تشغيله على أي نظام تشغيل دون الحاجة إلى تثبيت تبعيات إضافية. كل ما تحتاج إليه هو وجود آلة Java الافتراضية (Java Virtual Machine) مثبتة، مثل OpenJDK.

في القسم التالي، سنقوم بتثبيت OpenJDK، والذي يُعد ضروريًا قبل تشغيل Keycloak. إذا كان لديك بالفعل آلة Java الافتراضية مثبتة، يمكنك تخطي القسم التالي والانتقال مباشرة إلى القسم المعنون تثبيت Keycloak.

تثبيت OpenJDK

لتثبيت Keycloak، تحتاج إلى تثبيت الإصدار 17 من بيئة تشغيل Java (JRE).

لتثبيت OpenJDK، قم بتنزيل أحد الإصدارات الجاهزة من الموقع التالي:

adoptium.net/temurin/archive

افتح هذه الصفحة في متصفحك، ثم قم بتنزيل الإصدار 17 من OpenJDK.

قم بتنزيل الإصدار المناسب لنظام التشغيل الخاص بك، ثم فك ضغطه في مكان مناسب على جهازك.

بعد فك الضغط، قم بتعيين متغير البيئة JAVA_HOME ليشير إلى المجلد الذي يحتوي على الملفات التي تم استخراجها.

توضح الأوامر التالية مثالاً على تثبيت OpenJDK على نظام Linux باستخدام إصدار جاهز.

```
$ mkdir ~/kc-book
$ cd ~/kc-book
$ tar xfvz ~/Downloads/OpenJDK17U-jdk_x64_linux_hotspot_17.0.6_10.tar.gz
$ export JAVA_HOME=~/kc-book/jdk-17.0.6+10
$ $JAVA_HOME/bin/java-version
```

الأمر الأخير (java –version) يقوم بالتحقق من أن Java تعمل بشكل صحيح.

الآن بعد أن قمت بتثبيت OpenJDK، سننتقل إلى تثبيت Keycloak

تثبيت Keycloak

بمجرد أن تقوم بتثبيت آلة Java الافتراضية على جهازك، تكون الخطوة التالية هي تنزيل توزيعة Keycloak من موقع Keycloak الرسمي.

افتح الرابط التالي في متصفحك: https://www.keycloak.org/downloads، ثم قم بتنزيل إما ملف ZIP أو TAR.GZ الخاص بالخادم (توزيعة الخادم المستقل).

بعد تنزيل الملف، ما عليك سوى فك ضغطه في موقع مناسب على جهازك.

```
$ unzip ~/Downloads/keycloak-22.0.0.zip
$ export KC_HOME=~/kc-book/keycloak-22.0.0
```

أنت الآن جاهز لبدء تشغيل Keycloak، وهو ما سنتناوله في القسم التالي.

بدء تشغيل Keycloak

بعد أن تكون قد قمت بتثبيت Keycloak وإنشاء حساب المسؤول الأولي، يصبح من السهل بدء تشغيل Keycloak.

على نظام Linux أو macOS، يمكنك بدء تشغيل Keycloak باستخدام الأمر التالي:

```
$ export KEYCLOAK_ADMIN=admin
$ export KEYCLOAK_ADMIN_PASSWORD=admin
$ cd $KC_HOME
$ bin/kc.sh start-dev
```

أو، على نظام Windows، نفذ الأمر التالي:

```
> set KEYCLOAK_ADMIN=admin
> set KEYCLOAK_ADMIN_PASSWORD=admin
> cd %KC_HOME%
> bin\kc.bat start-dev
```

نظرًا لأن Keycloak لا يأتي مع حساب مسؤول افتراضي، فإن تمرير المتغيرات البيئية KEYCLOAK_ADMIN وKEYCLOAK_ADMIN_PASSWORD يسهل إنشاء حساب المسؤول الأولي.

بعد بضع ثوانٍ، يمكنك التحقق من أن Keycloak يعمل بفتح الرابط التالي: http://localhost:8080/admin، ثم قم بتسجيل الدخول باستخدام اسم المستخدم admin وكلمة المرور admin.

تهانينا! أصبح لديك الآن Keycloak يعمل على جهازك ويمكنك البدء في تجربة Keycloak من خلال استكشاف وحدة تحكم المسؤول (Keycloak Admin Console) ووحدة تحكم الحساب (Keycloak Account Console).

إعداد Keycloak باستخدام Docker

قم بإنشاء ملف docker-compose.yml في الدليل الجذري:

```
▷Run Service
keycloak:
  image: quay.io/keycloak/keycloak:21.1.1
  ports:
    - 8080:8080
  command: -v start-dev --import-realm
  environment:
    KEYCLOAK_ADMIN: admin
    KEYCLOAK_ADMIN_PASSWORD: admin
    KC_DB: mariadb
    KC_HOSTNAME_URL: http://localhost:8080
    KC_DB_URL_HOST: mariadb-server
    KC_DB_URL_DATABASE: keycloak
    KC_DB_URL_PORT: 3306
    KC_DB_USERNAME: keycloak_user
    KC_DB_PASSWORD: xyz123ABC$
  volumes:
    - ./keycloak/init_config:/opt/keycloak/data/import
  networks:
    - global-network
```

قم بتشغيل خادم Keycloak باستخدام:

docker-compose up -d

Configure Keycloak

افتح المتصفح واذهب إلى:

http://localhost:8088

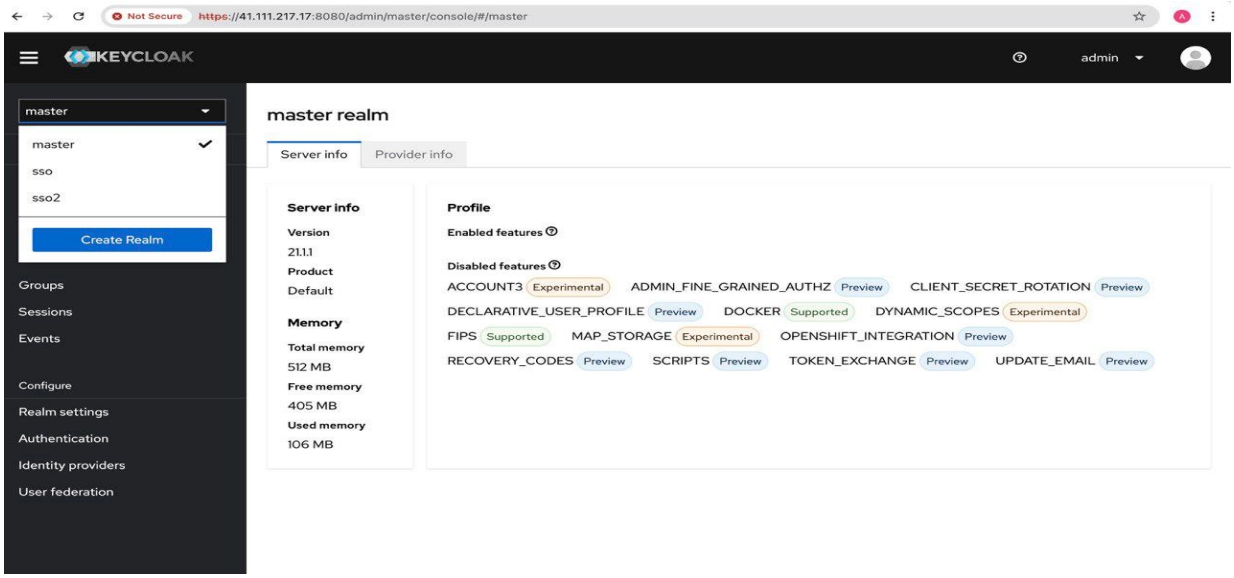
سجّل الدخول باستخدام:

اسم المستخدم: admin

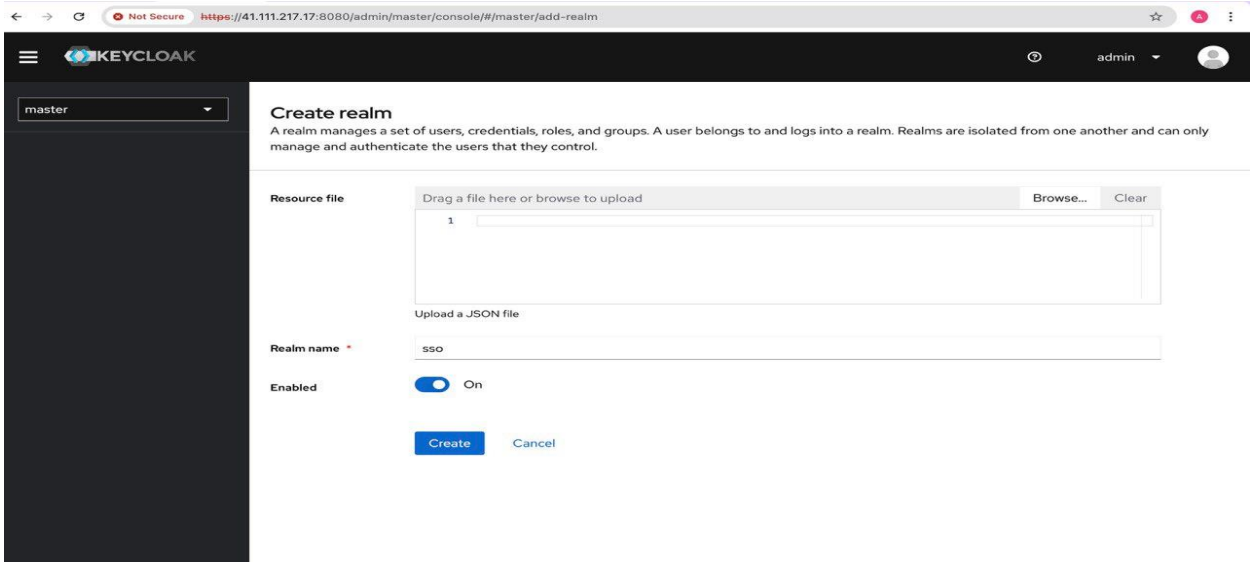
كلمة المرور: admin

إنشاء (**Realm**)

انقر على القائمة المنسدلة على اليسار، ثم اختر إنشاء نطاق (Create Realm) لبدء إعداد نطاق جديد في Keycloak.

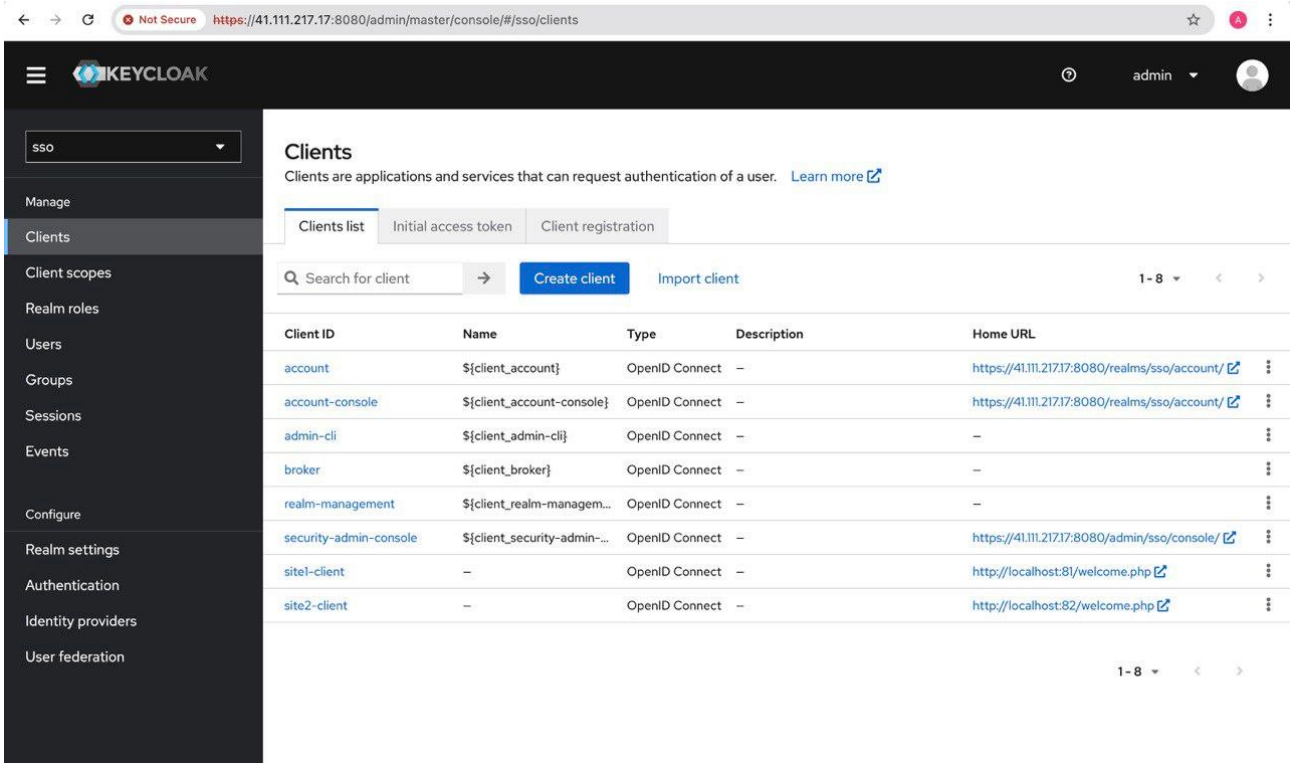


قم بتسمية النطاق ثم انقر على إنشاء (Create).

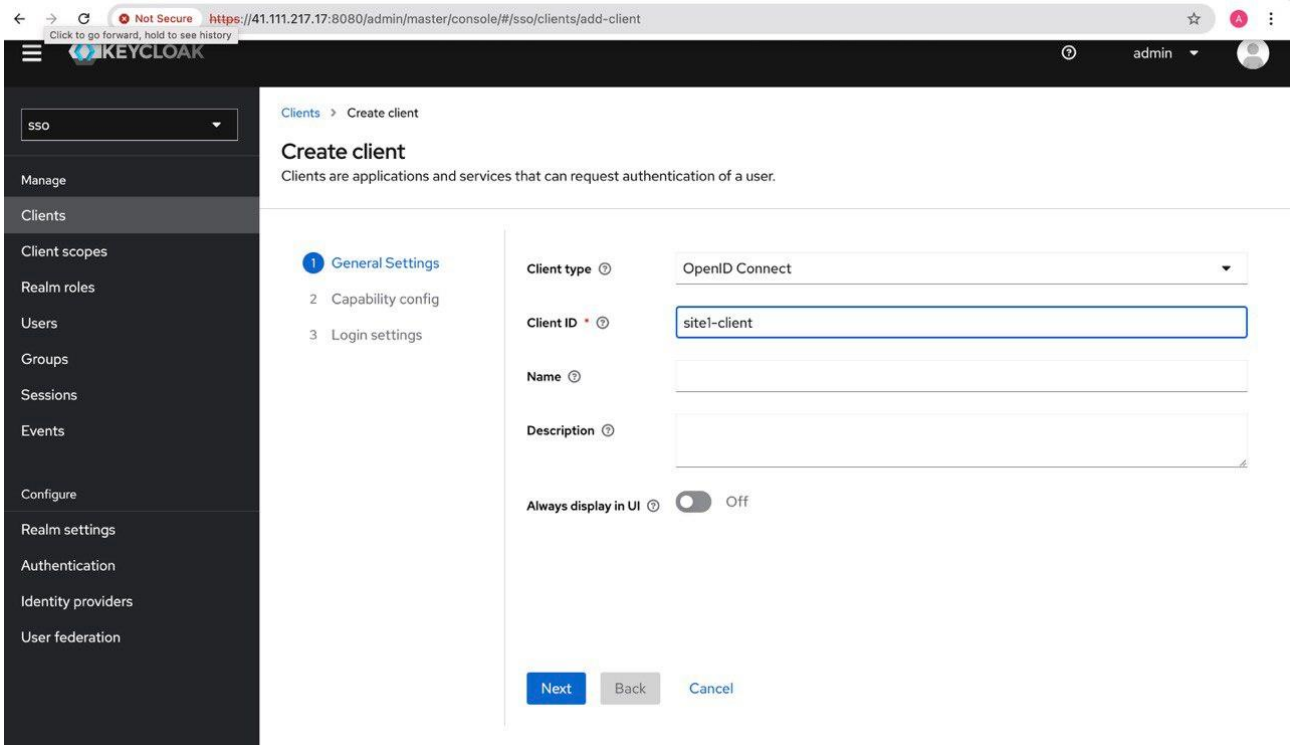


إنشاء عميل (**Client**)

داخل Realm الذي تم إنشاؤه، انقر على العملاء (Clients)، ثم اختر إنشاء عميل (Create Client) لإضافة تكوين عميل جديد في Keycloak.



قم بتسمية العميل ثم انقر على التالي (Next)، ثم انقر على التالي (Next) مرة أخرى.

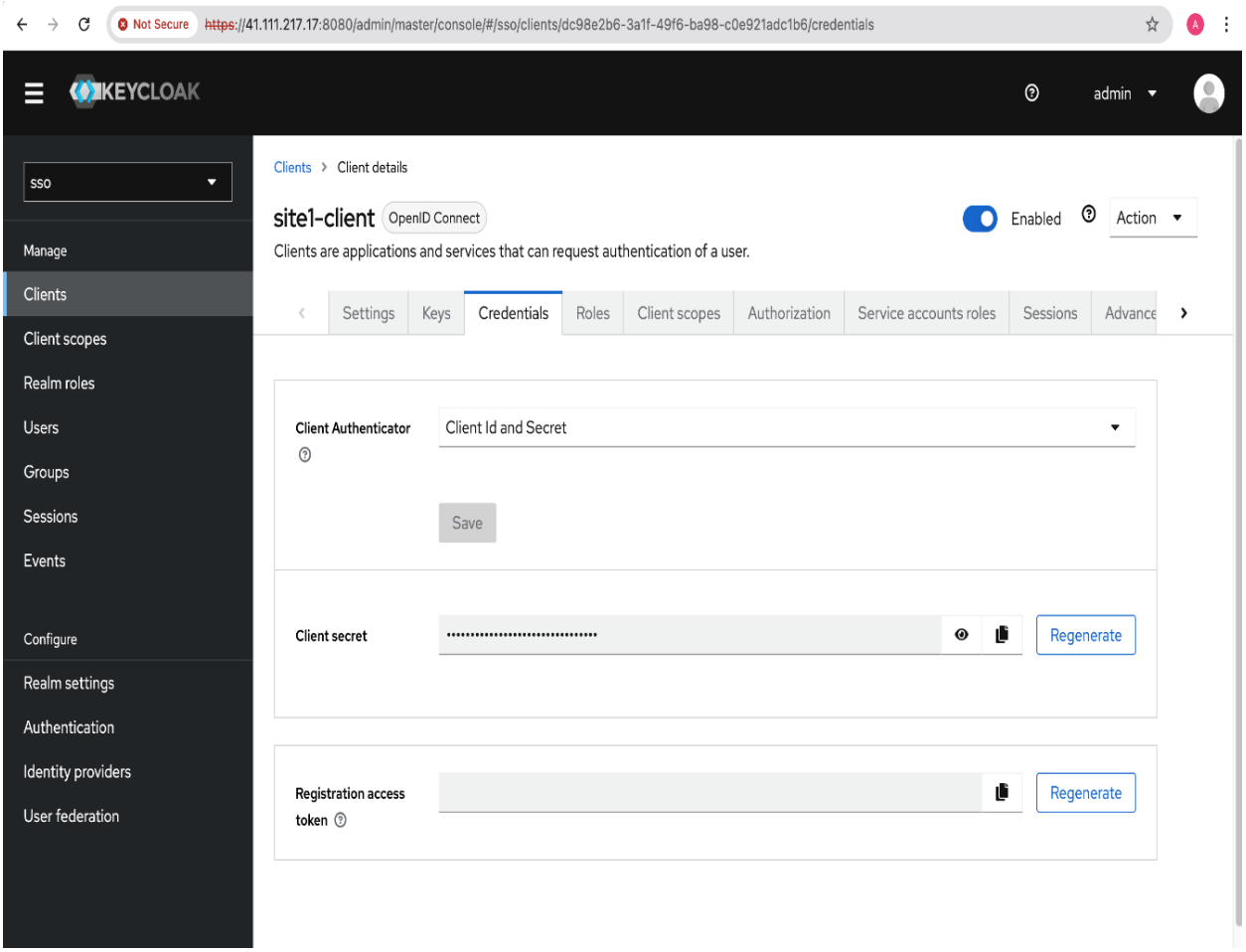


في حقل Valid Redirect URIs، اكتب http://localhost:81/keycloak-login.php هذا هو المنفذ الذي سيعمل عليه تطبيق. ثم، انقر على حفظ (Save) لإكمال تكوين العميل.

تأكد من تفعيل مصادقة العميل في صفحة إعدادات العميل

احفظ التغييرات، ثم قم بتحديث الصفحة.

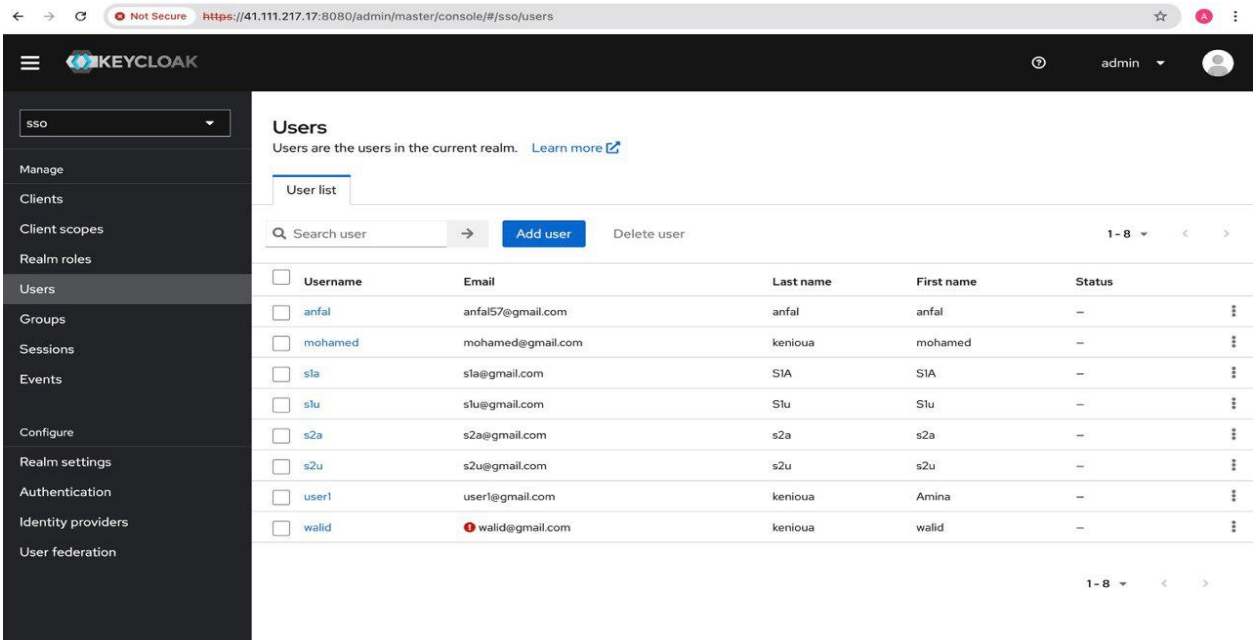
حدد قسم بيانات الاعتماد (Credentials) وانقر عليه لعرض بيانات اعتماد العميل."



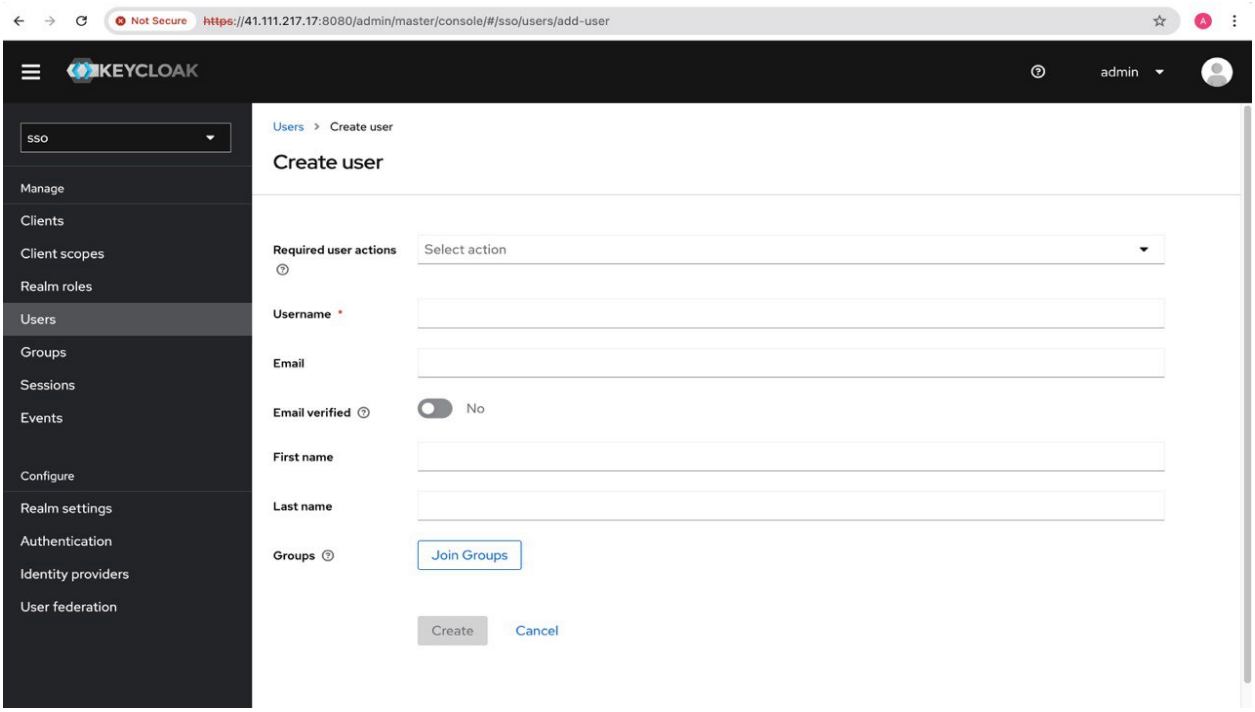
تأكد من عرض السر الخاص بالعميل واحتفظ به في مكان آمن، لأننا سنحتاجه في الخطوات التالية!

إنشاءUser

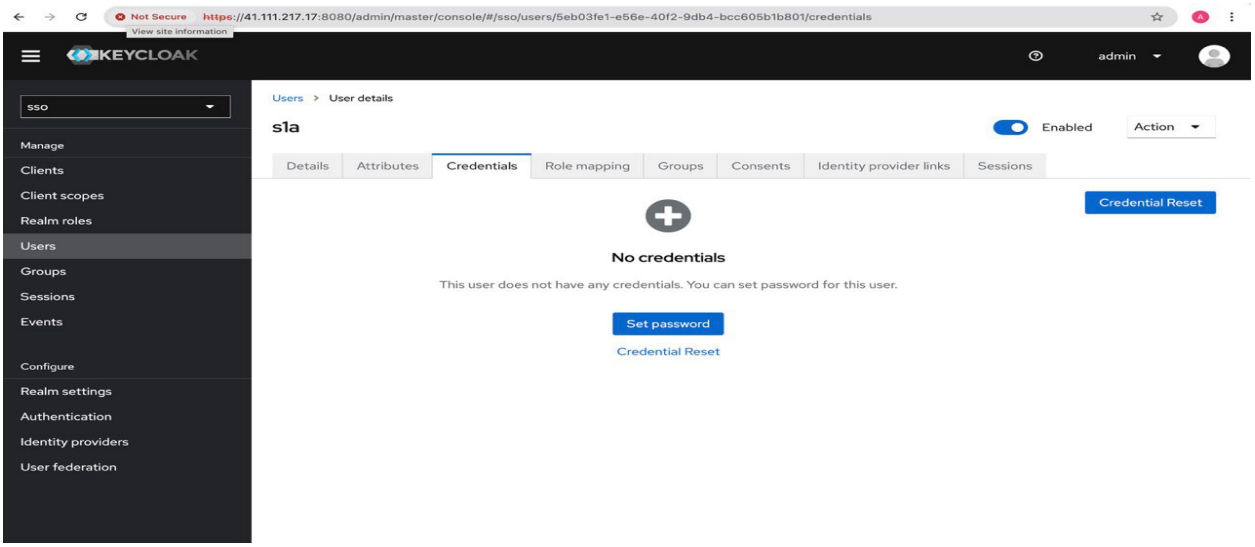
انتقل إلى قسم المستخدمين (Users)، ثم انقر على إنشاء مستخدم (Add User) لإضافة مستخدم جديد في Keycloak.



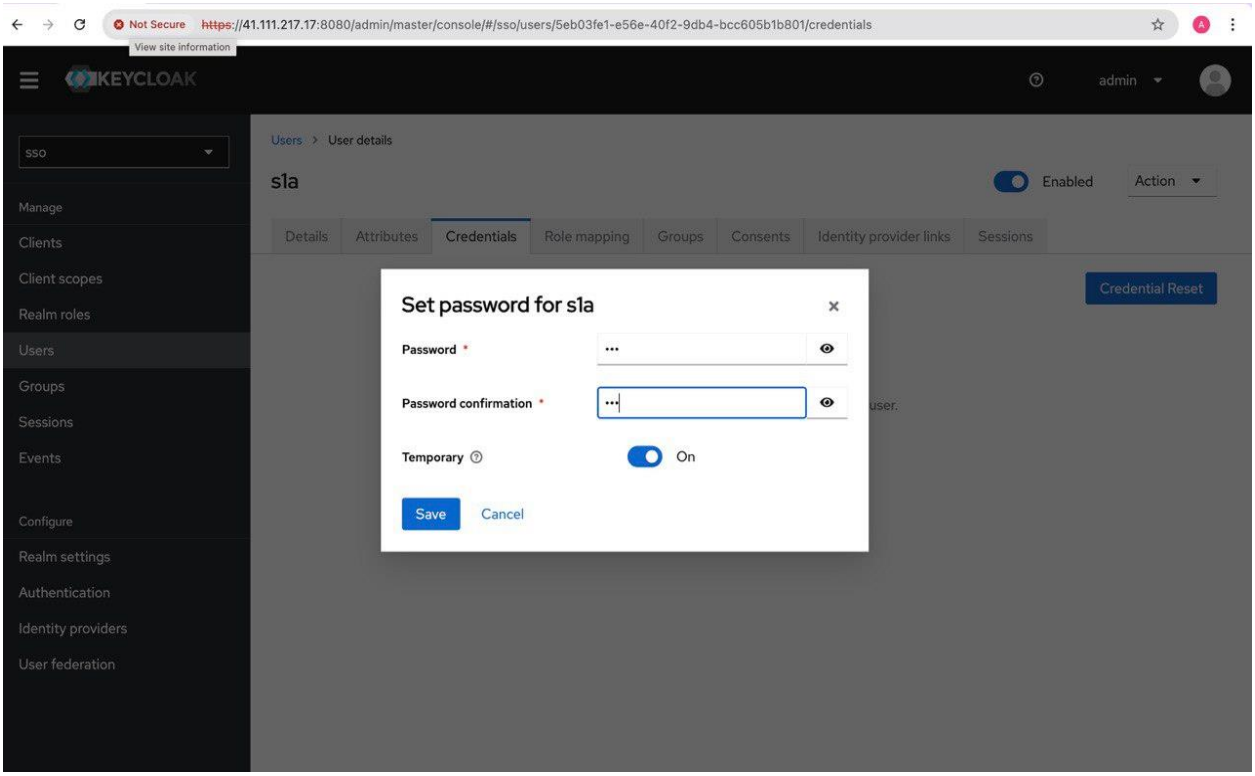
املأ المعلومات المطلوبة للمستخدم الجديد، ثم انقر على إنشاء (Create).



اذهب إلى قسم بيانات الاعتماد (Credentials)، ثم قم بتعيين كلمة مرور

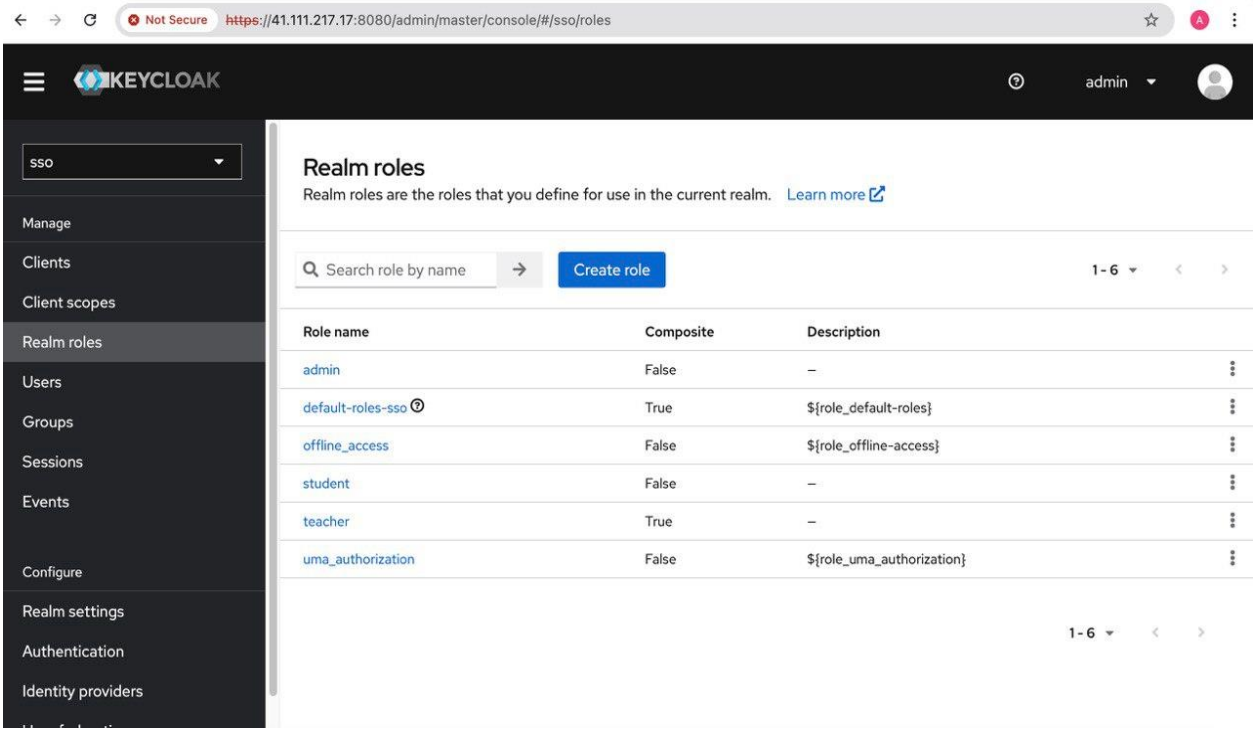


قم بإدخال كلمة مرور ثم انقر على حفظ (Save).



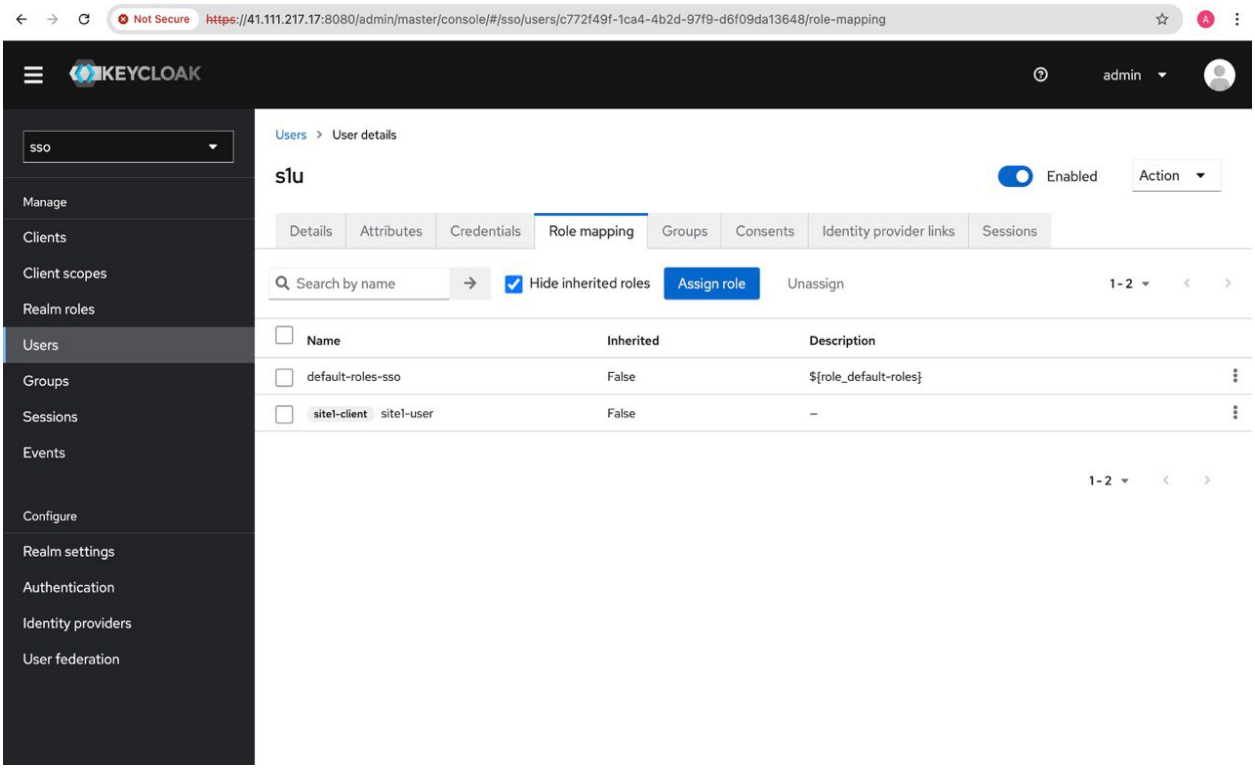
. **إنشاء الأدوار (Realm Roles)**

في الشريط الجانبي، اضغط على create Role لإضافة دور جديد.



تخصيص الأدوار للمستخدمين: قم بتخصيص الأدوار المناسبة للمستخدمين المسجلين، مما يضمن الوصول الصحيح إلى الوظائف والأقسام المختلفة في البوابة.

بعد تسجيل مستخدم جديد، يمكنك تخصيص أي دور متاح لذلك المستخدم.



الخلاصة: يوفر Keycloak قدرات مرنة وقوية لإدارة الهوية والوصول. يعد فهم الـ Realms و الـ Clients و الـ Roles أمرًا أساسيًا لإعداد واستخدام Keycloak بشكل فعال. تسمح هذه المفاهيم بإنشاء بنية تحتية آمنة وقابلة للتوسع للمصادقة والتفويض لأنواع مختلفة من التطبيقات والخدمات.

للمزيد من الدراسة حول Keycloak، يُنصح بمراجعة الوثائق الرسمية ومحاولة إعداد Keycloak لتطبيقاتك، مع اتباع الأمثلة وأفضل الممارسات.