SOFE 4840U Software and Computer Security

Identity and Access Management

Flutterflow UI https://github.com/tiwaojo/iam_cs
Spring Boot App: https://github.com/preetpatel87/IAM-spring-app

Final Report
Date: March 27, 2023
Project Group: Group 8

Name	Student Id
Preet Patel	100708239
Tiwaloluwa Ojo	100700622
Waleed El Alawi	100764573
Aaditya Rajput	100622434

Abstract

IAM is an acronym for Identity and Access Management (IAM) and as the name suggests, it handles access to critical information based on the Identity of the user and their permissions within the organization. IAM will help prevent mistakes where otherwise lower privileged users in the system get access to higher sensitive information. The goal of this project is to implement an IAM system that implements role-based access control in an application server to prevent users from accessing APIs that they do not have permission to access. This will help protect the application by preventing lower privileged users from accessing APIs that control the application data. This report will introduce the concept of IAM systems and will highlight the objectives of the project. Furthermore, the report also explains the methodology that has been used to implement the IAM system and provides justifications for design decisions. It provides a detailed explanation of the results of this implementation, its feasibility, and lastly, any limitations of this approach.

Table of Contents

List of Illustrations

Introduction and Background

Identity and Access Management (IAM) is a framework of business processes, policies, and technologies that facilitates the management and access to digital intellectual property based on digital identities. IAM provides additional security to any system or application on top of simple user authentication. The use of IAM provides benefits such as preventing the risk of any internal attacks where a user is attempting to gain access to resources that they are not authorized to access [2]. This security service is very beneficial to all organizations, from small to large, which require multiple layers of security in which access to corporate information is utilized. The goal of this project is to build an IAM system that applies role-based access on a forum (such as reddit) web application. The forum application will allow privileged users to create threads for discussions and let other users interact and participate in discussions by providing them the ability to post comments on the threads. The forum application is just used as an example for how the implemented IAM system would work with a web application. For this project, the team will focus on implementing authorization using IAM and therefore will limit the functionality of the forum application to a basic implementation used only for demonstration purposes. IAM is a critical component of software security. This system will demonstrate the layers of protection against unauthorized access by authenticated users to various actions and pages of the web forum. This can be achieved by managing user identities, authentication, authorization, and access control. The IAM system will shield critical functionalities and information that require higher level of security privileges from being accessed by the lower privileged users making the forum application resistant towards unauthorized access, internal attacks, data breaches, and cyber-attacks. We will be using different tools and frameworks to accomplish the objective of this project.

The IAM framework for the forum platform we would like to develop would include key components such as:

1. **User Authentication:** Users can login and register using a username and password. The scope of this project does not include implementing 2-factor authentication or any advanced form of authentication. The project will also not focus on utilizing authorization tokens such as JWT tokens. We will be focusing on implementing counterattack measures using the authorization and role-based access that is associated with IAM.
2. **Role-based Access Control:** Once a user is registered their access tier is determined. The roles and their privileges are listed below:
    - Viewer: Viewers can view threads, see comments under threads, and comment under threads themselves.
    - Creator: Creators can do everything GRs can as well as start their own thread.
    - Moderator: Moderators can delete threads and comments as well as everything Creator and Viewer can do.
    - Administrator: Admins can edit threads/comments, as well as everything else Moderators, Creators, and Viewers can do.
3. **API access authorization:** The IAM system will ensure that users cannot perform unauthorized API calls which are not part of their role’s privileges.

Project Objective

The purpose of this project is to build an IAM framework that implements role-based access control for a forum web application. The forum application web application will use APIs to function, and these will require users to have valid permissions to be able to call these APIs to perform various actions in the application. The IAM framework will be responsible for authorization of the users to ensure that users can only access APIs that they are authorized to perform. The IAM framework will deny access to lower privileged critical to perform functions and access information that require the user to have higher privileged role to access. Such a framework will protect the forum application from internal attacks and limit the potential damage if the authentication of user accounts with less privileges are compromised by protecting sensitive data and preventing users to perform powerful functions. The deliverables for this project will consist of a web application consisting of a frontend and backend, and a database. We aim to achieve these objectives using various tools and frameworks such as a relational database, framework for creating the web server, and a framework for implementing the user interface.

Project Methodology

To implement the IAM system in this project, we will utilize a relational database that will be deployed locally to create an access matrix which will contain the data such as the defined roles that users can have in the forum application and the API access permissions for each of the roles. This will create a role-based access control system that provides permissions to users based on their roles and rules stating what accesses are allowed to users in each defined role. Figure 1 below depicts a deployment diagram that describes the various components that will be required to implement the system is provided in the following image:

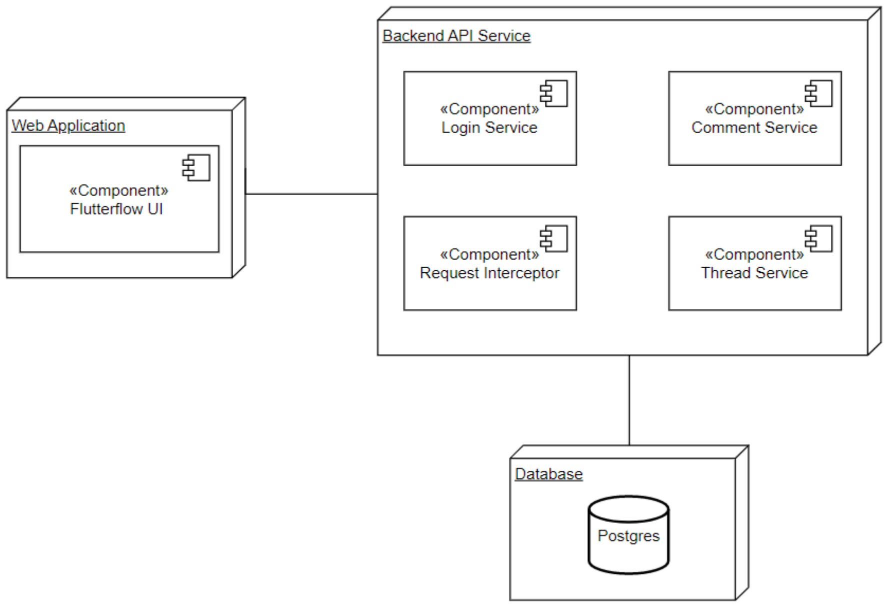

Backend API Service
«Component» Login Service
«Component» Comment Service
«Component» Request Interceptor
«Component» Thread Service
Web Application
«Component» Flutterflow UI
Database
Postgres

Figure 1 Deployment Diagram

The first and most important component of the system that we will discuss is the database. The database will be used to implement and store the access matrix required to implement the IAM system. It will also be used to hold the application data such as the threads and comments which are essential to implement the forum application. Moreover, our database of choice is PostgreSQL. This DB was chosen for its unique array data type which will aid in meeting the functionality requirements of the access matrix. MySQL and NoSQL were also considered when choosing what type of database, we wanted to implement but PostgreSQL was chosen due to its advanced features that are not available on MySQL or NoSQL such as indexing options, advanced query optimization, and support for stored procedures and triggers. Another factor we considered was the ability of PostgreSQL to enforce data integrity because it has strict data type checking. MySQL and NoSQL databases on the contrary are more permissive with data types which can lead to data corruption if not managed carefully. PostgreSQL also allows for horizontal scalability and can handle large scale datasets well. The entity relationship diagram below shows the structure of the data that will be stored in the database as well as their relationships to each other:

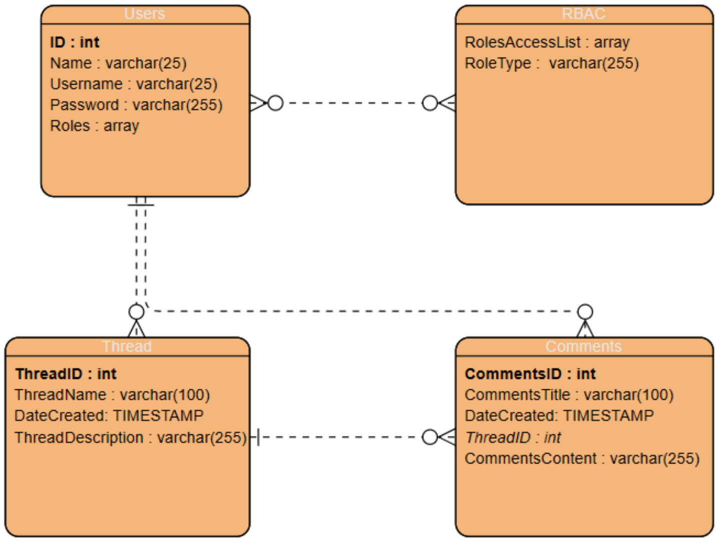


Figure 2 Entity Relationship Diagram

- **Users**: These represent the person(s) who have created an account with the application and are registered in the database. They are identified with a name, username, password, and an array of 1 or more roles. These roles include but are not limited to Admin, Moderator, Creator, and Viewer.
- **Thread**: This entity represents a conversation thread that multiple users have made comments on. Each thread is identified in the database by their thread name, the timestamp the thread was created, and a description of the threads. A single thread may have 1 to many comments.
- **Comments**: This entity represents the conversation item the user appends to the Thread. A comment can only have one thread.
- **RBAC**: Role based access control entity represents the type of roles as well as the resource they have access to.

The next component in the system that we will discuss is the web application that will be implemented using FlutterFlow which will act as the user interface for the forum application. FlutterFlow is a visual development platform that allows developers to create mobile and web applications without writing code by enabling drag and drop components. Creating UI from scratch can be time-consuming and would take time and effort away from developing the actual computer security aspects of the project so we decided to go with FlutterFlow, the Flutter based platform that leverages all Flutter functionalities and provides a code editor for customization to create the UI. We went with FlutterFlow over other options such as Adalo, Bubble, Glide, and manual UI development because of its speed of development, customization, cross-platform compatibility and because of our prior experience with Flutter. The users will be able to interact with this interface to trigger various API requests that will be made to the backend service. Users that attempt to perform actions that they cannot perform due to their limited privileges determined by their role will be shown an error message that will be returned by the backend API service. An example of a “happy” interaction with the user interface involves the user attempting to view all the threads in the forum which every role of the system should have the permission to access. An example of an “unhappy” interaction would involve the user attempting to create a new thread without having the correct permission as this action requires a role of a creator or a role with higher level of security clearance.

Lastly, the final component will be the backend service which will include all the APIs that the system will require to implement the functionalities of the forum application involving login, threads, and comments. It also includes the request interceptor which will be used to implement the authorization performed by the IAM system. This implementation will utilize Java Spring Boot Framework for developing the various endpoints in the API. Figure 2 below shows component diagram of each of services and APIs that will be implemented in the backend service:

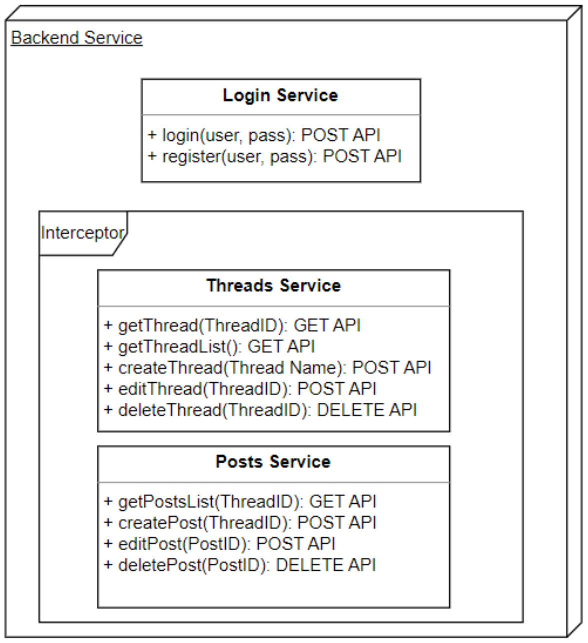

Backend Service
Login Service
+ login(user, pass): POST API
+ register(user, pass): POST API
Interceptor
Threads Service
+ getThread(ThreadID): GET API
+ getThreadList(): GET API
+ createThread(Thread Name): POST API
+ editThread(ThreadID): POST API
+ deleteThread(ThreadID): DELETE API
Posts Service
+ getPostsList(ThreadID): GET API
+ createPost(ThreadID): POST API
+ editPost(PostID): POST API
+ deletePost(PostID): DELETE API

Figure 3 Backend Service Structure and APIs

The backend service will include the following three main service components:

1. Login Service - This service will include the login and register APIs which will call to register new users and allow existing users to login to the forum application.
2. Threads Service - This service includes all the APIs that are required in the forum application to create, view, edit, and delete the discussion threads.
3. Comments Service - This service includes all the APIs that are required in the application to create, view, edit, and delete comments on discussion threads.

The user will first have to login to the system to authenticate themselves to interact with the forum application. The user can then interact with the user interface to perform actions related to threads and comments which are the basic functionalities of the forum application. In order to implement the IAM system, we will utilize a request interceptor which will intercept any requests made to the thread or comments service to check if the user that is making the request has the authorization to perform that API request. The interceptor will query the database to obtain the user's role and the corresponding permissions of the role from the access matrix to see if the user is authorized for the API call or not. If the user does not have the role with the permission to perform the API request, the interceptor will return an unauthorized error message and the request will not be performed. Whereas, if the user does have the correct permissions, then the request will be forwarded to the appropriate service to be handled correctly. The following sequence diagram illustrates a happy interaction between the user and the system when trying to obtain a list of all the threads on the forum:

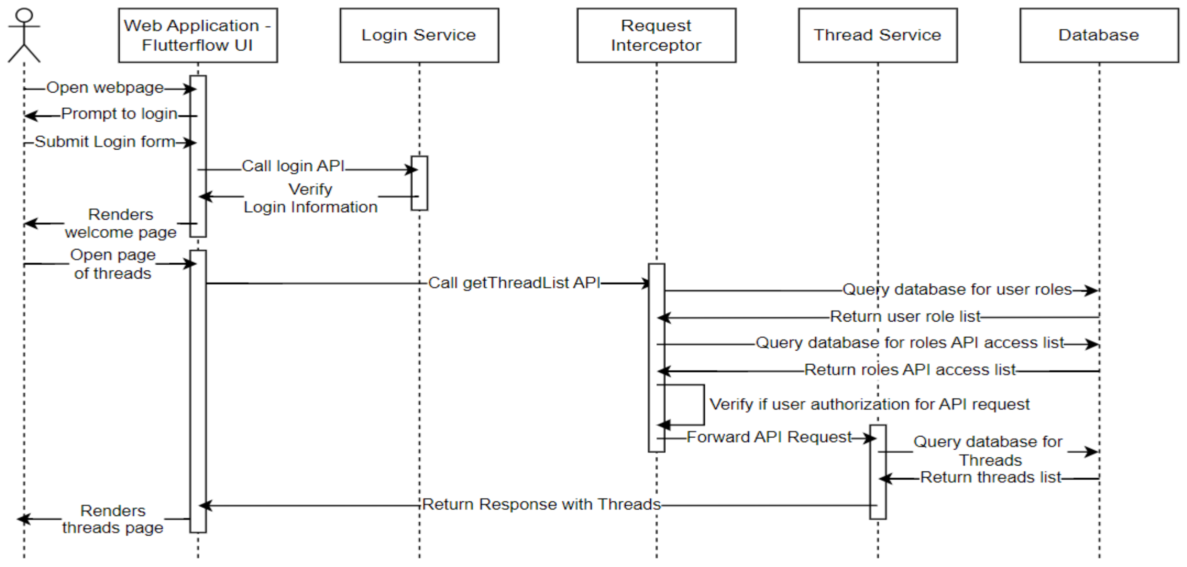
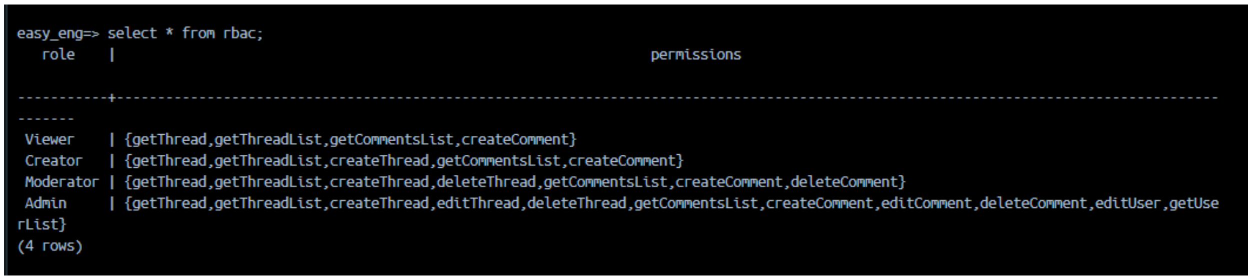


Figure 4 Sequence Diagram for getThreadList API in the Forum Application

As laid out in this literature, our system will utilize various techniques and methods to achieve a robust system in which a user may only access what they have been granted permission to access.

Results

Although we encountered a few issues during development, we were able to successfully develop an IAM system that implemented a role-based access system in a forum application. The team was able to use the Spring Boot to create a request interceptor that intercepts any requests being made to the APIs of the forum application and query the PostgreSQL database that holds the access matrix to check the user's permission. The interceptor will either deny the request or allow the request to pass depending on if the user has the permission to perform the API call. Moreover, such interactions were made available to the user through an intuitive User Interface designed on Flutter. The following image shows the access matrix in the PostgreSQL database.



```
easy_eng=> select * from rbac;
   role    |                                                    permissions

-----------+-------------------------------------------------------------------------------------------------------------------------------
-------
 Viewer    | {getThread,getThreadList,getCommentsList,createComment}
 Creator   | {getThread,getThreadList,createThread,getCommentsList,createComment}
 Moderator | {getThread,getThreadList,createThread,deleteThread,getCommentsList,createComment,deleteComment}
 Admin     | {getThread,getThreadList,createThread,editThread,deleteThread,getCommentsList,createComment,editComment,deleteComment,editUser,getUse
rList}
(4 rows)
```

Figure 5 Access Matrix Table in PostgreSQL Database

The table above is the access matrix containing different roles and the list of APIs that the user with the role is permitted to access. Please refer to the images in Appendices that demonstrates an API call being performed by different users with different roles which are either performed or denied by the backend application depending on if the users have permissions to access the API.

Our system relies on PostgreSQL and Spring Boot (Java) which operate asynchronously and secluded from the host with the help of Docker Containers, and Flutter (FlutterFlow CMS) for our frontend application. Our development and program execution encountered a few issues such as Cross Origin Resource Sharing errors and inconsistent data types. The following segments below further discuss the results of our approach to implementing IAM, issues encountered and how we solved them.

Spring Boot Application and PostgreSQL Database

Spring Boot application was the framework that the team used to develop the backend application that would contain the implementation of the API and the request interceptor that is responsible for checking if the user making the request has the permission to make the API request. The PostgreSQL database was used to hold the access matrix which is core component of implementing the IAM system. There were no issues using the PostgreSQL database over MySQL or any other types of databases. The biggest issues faced when implementing the Spring Boot application were related to the interceptor implementation. The interceptor, which is a one of the two main components of the IAM system was implemented using a preHandle method. Such a method always runs before the request reaches the API, so it can protect the system from any unauthorized access as the project required. However, by implementing the IAM system interceptor using the preHandle method, we lost the backend application's ability to throw appropriate errors that are naturally handled. Example, when making a POST request to a GET HTTP method API the application will throw an unauthorized error instead of throwing the default error that is handled by Spring Boot itself.

FlutterFlow User Interface

FlutterFlow was a means of reducing the development overhead required for this project. It allowed us to cut down on our development time. This CMS system interacts with our Spring Boot API service through the exposed endpoints from Spring Boot application to the CMS API calls tab. By carefully editing the UI, we were able to call our API endpoints as FlutterFlow actions or Backend Queries. These actions provided us the ability to relate the results and manage the request body with the inputs we required. Due to FlutterFlow existing as an online service provided at a different origin from our local computers, we were not able to test our application with the provided API test tool on FlutterFlow. We had to either publish our API to the cloud or result to publishing the program on GitHub, cloning, and running locally. Any attempts at using the test tool on FlutterFlow would result in CORS errors. By running the application locally, we were also able to debug the API and examine the database of the entries made. Although we encountered a few issues during development, we were able to successfully deploy the system.

Conclusion and Recommendations

In conclusion, our team has successfully developed an Identity and Access Management (IAM) framework for a web forum application using FlutterFlow and Spring Boot. FlutterFlow allowed us to create an intuitive and efficient way to create the frontend whereas Spring Boot and a PostgreSQL database allowed us to create the backend of the application. PostgreSQL provided a reliable and secure database for storing application data as well as the access matrix. The access matrix model enabled us to implement a granular system of permissions and privileges, ensuring that each user had access only to the resources that were necessary for their roles. Spring Boot allowed us to implement the application APIs and a request interceptor which queries the access matrix to deduce if the user making the request has the permissions to perform that API request. Our Project has achieved its objectives, which included creating a secure role-based access system and a simple forum webpage to display threads and comments. Additionally, our project has demonstrated how the use of modern development tools and frameworks can simplify the development process and increase productivity.

In order to scale this application for the future, two things should be implemented. Adding 2 factor authentication and using JWT tokens instead of usernames and passwords in the interceptor for authorization. Adding two-factor authentication on top of the IAM framework would enhance the security of our web forum by adding an additional layer of protection to the authentication process. This would prevent attackers that look to gain access to user accounts.

Using JWT tokens instead of usernames and passwords in the interceptor of the web forum application would improve its security by reducing the amount of sensitive information that is transmitted over the network. JWT tokens are generated by the server after a successful user authentication, and they contain a unique and encrypted representation of the user’s identity. This further secures the user’s login details as well as improves the scalability of authentication and authorization. These two recommendations would make the application less susceptible to security threats and allow the application to be more scalable.

Contribution Matrix

Table 1 Final Report Contribution Matrix

Task	Preet Patel	Aaditya Rajput	Tiwaloluwa Ojo	Waleed El-Alawi
Abstract	85%	5%	5%	5%
Introduction and background	5%	45%	5%	45%
Project Objective	5%	5%	5%	85%
Project Methodology	40%	10%	40%	10%
Results	25%	5%	45%	25%
Conclusion and Recommendations	5%	85%	5%	5%
Appendices	5%	5%	85%	5%
Total **Approximately equal contribution amongst team members**	25%	25%	25%	25%

Table 2 Overall Project Contribution Matrix

Task	Preet Patel	Aaditya Rajput	Tiwaloluwa Ojo	Waleed El-Alawi
Phase 1: Project Planning & Requirements Gathering	25%	25%	25%	25%
Phase 2: Development & Design	25%	25%	25%	25%
Project Development and Testing	25%	25%	25%	25%
Phase 3: Final Report	25%	25%	25%	25%
Project Presentation	25%	25%	25%	25%
Total **Approximately equal contribution amongst team members**	25%	25%	25%	25%

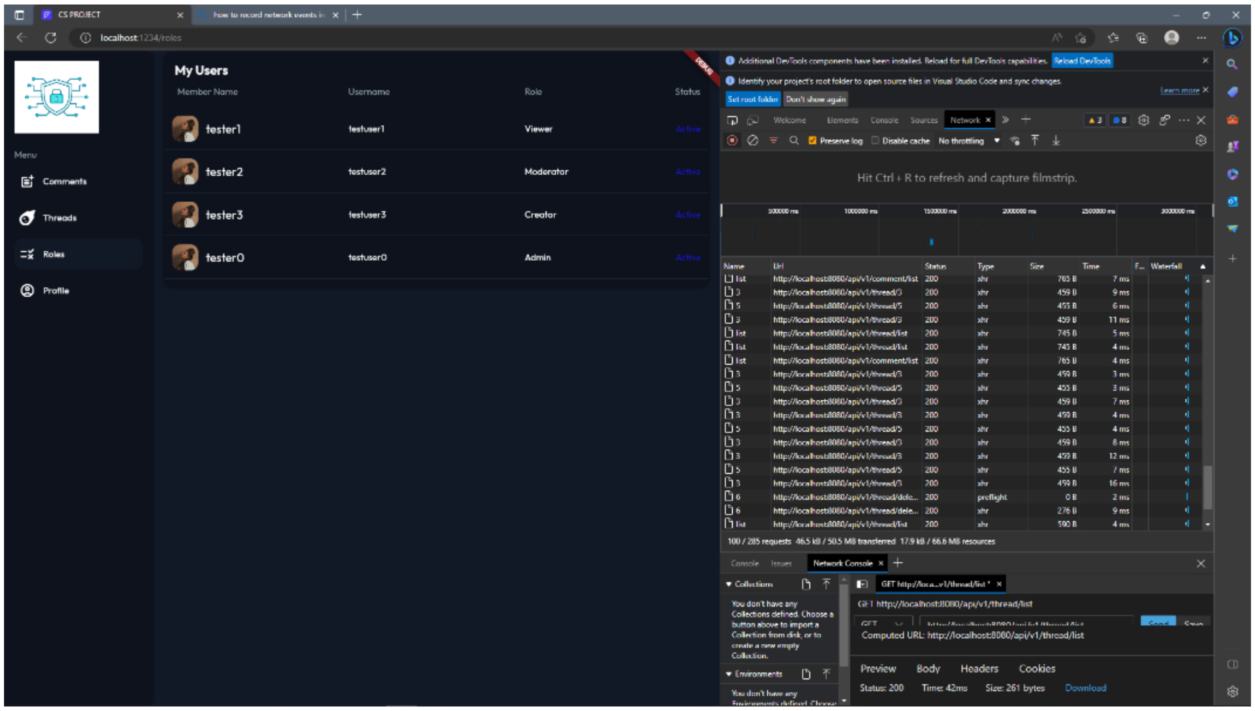
References

[1] S. Gittlen and L. Rosencrance, “What is identity and access management? guide to IAM,” *TechTarget*, 10-Aug-2021. [Online]. Available: https://www.techtarget.com/searchsecurity/definition/identity-access-management-IAM-system. [Accessed: 05-Feb-2023].

[2] I. A. Mohammed, “SYSTEMATIC REVIEW OF IDENTITY ACCESS MANAGEMENT IN INFORMATION SECURITY,” SSRN Electronic Journal, vol. 4, pp. 1–7, Jul. 2017. Available: https://www.researchgate.net/publication/353887659_SYSTEMATIC_REVIEW_OF_IDENTITY_ACCESS_MANAGEMENT_IN_INFORMATION_SECURITY/citation/download. [Accessed: 05-Feb-2023].

Appendices

Admin Role



localhost:1234/roles

My Users

Member Name	Username	Role	Status
tester1	testuser1	Viewer	Active
tester2	testuser2	Moderator	Active
tester3	testuser3	Creator	Active
tester0	testuser0	Admin	Active

Additional DevTools components have been installed. Reload for full DevTools capabilities.

Identify your project's root folder to open source files in Visual Studio Code and sync changes.

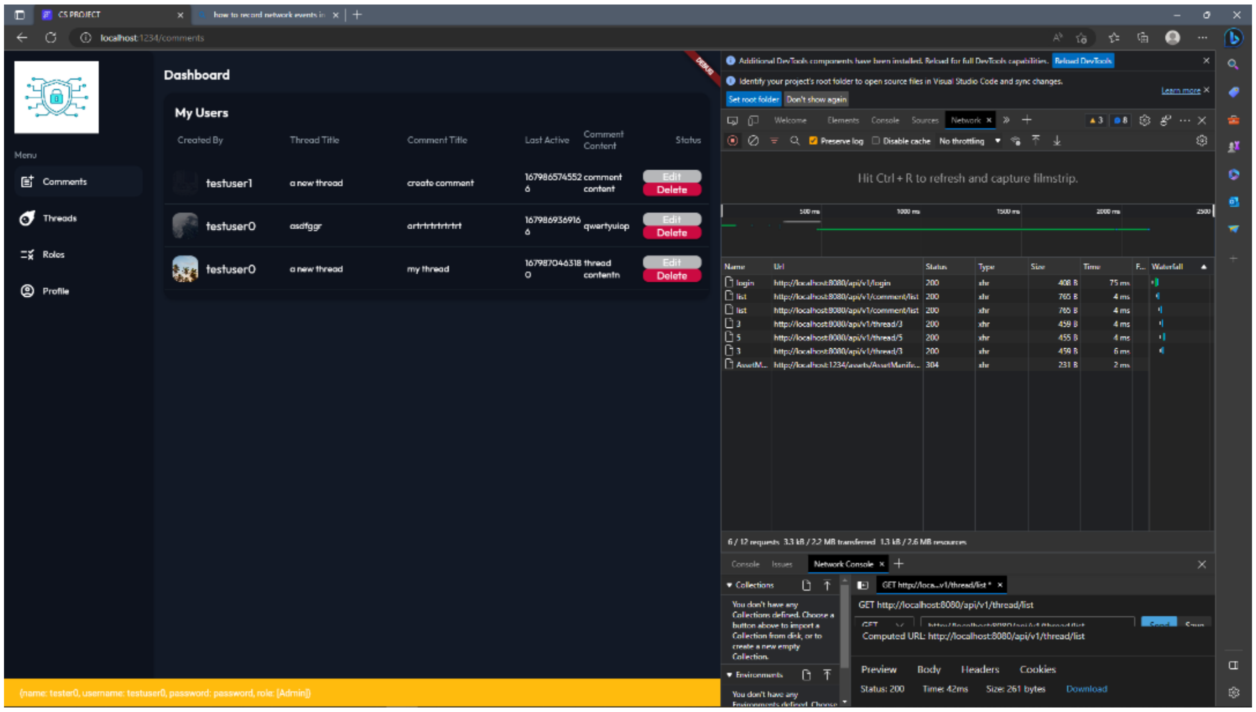
Hit Ctrl + R to refresh and capture filmstrip.

```
Name  Url                                          Status  Type       Size    Time
list  http://localhost8080/api/v1/comment/list     200     xhr        765 B   7 ms
3     http://localhost8080/api/v1/thread/3         200     xhr        459 B   9 ms
5     http://localhost8080/api/v1/thread/5         200     xhr        455 B   6 ms
3     http://localhost8080/api/v1/thread/3         200     xhr        459 B   11 ms
list  http://localhost8080/api/v1/thread/list      200     xhr        745 B   5 ms
list  http://localhost8080/api/v1/thread/list      200     xhr        745 B   4 ms
list  http://localhost8080/api/v1/comment/list     200     xhr        765 B   4 ms
3     http://localhost8080/api/v1/thread/3         200     xhr        459 B   3 ms
5     http://localhost8080/api/v1/thread/5         200     xhr        455 B   3 ms
3     http://localhost8080/api/v1/thread/3         200     xhr        459 B   7 ms
3     http://localhost8080/api/v1/thread/3         200     xhr        459 B   4 ms
5     http://localhost8080/api/v1/thread/5         200     xhr        455 B   4 ms
3     http://localhost8080/api/v1/thread/3         200     xhr        459 B   8 ms
3     http://localhost8080/api/v1/thread/3         200     xhr        459 B   12 ms
5     http://localhost8080/api/v1/thread/5         200     xhr        455 B   7 ms
3     http://localhost8080/api/v1/thread/3         200     xhr        459 B   16 ms
6     http://localhost8080/api/v1/thread/dele...   200     preflight  0 B     2 ms
6     http://localhost8080/api/v1/thread/dele...   200     xhr        276 B   9 ms
list  http://localhost8080/api/v1/thread/list      200     xhr        590 B   4 ms
100 / 285 requests  46.5 kB / 50.5 MB transferred  17.9 kB / 66.6 MB resources
```

You don't have any Collections defined. Choose a button above to import a Collection from disk, or to create a new empty Collection.

```
GET http://localhost:8080/api/v1/thread/list
Computed URL: http://localhost:8080/api/v1/thread/list
Status: 200   Time: 42ms   Size: 261 bytes
```

Figure 6 Users and their respective roles in the Forum



localhost:1234/comments

Dashboard

My Users

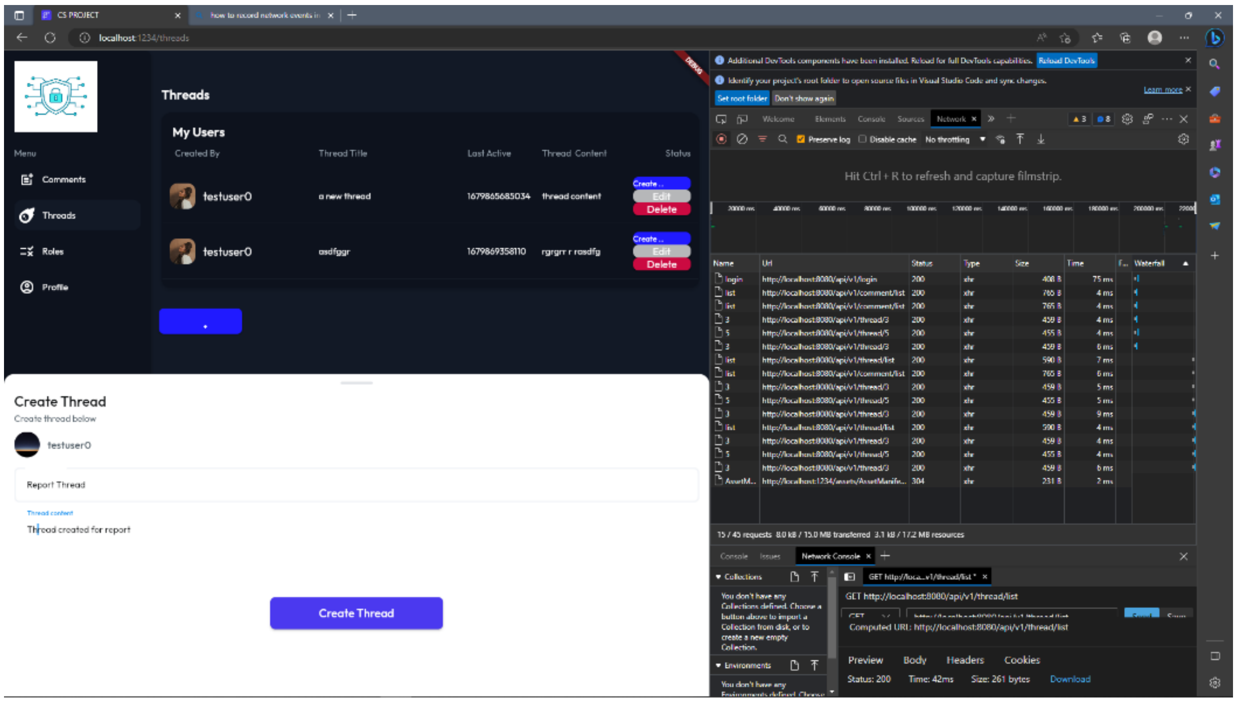
Created By	Thread Title	Comment Title	Last Active	Comment Content	Status
testuser1	a new thread	create comment	1679865745526	comment content	Edit Delete
testuser0	asdfggr	artrtrtrtrtrtrt	1679869369166	qwertyuiop	Edit Delete
testuser0	a new thread	my thread	1679870463180	thread contentn	Edit Delete

```
Name       Url                                          Status  Type  Size   Time
login      http://localhost:8080/api/v1/login           200     xhr   408 B  75 ms
list       http://localhost:8080/api/v1/comment/list    200     xhr   765 B  4 ms
list       http://localhost:8080/api/v1/comment/list    200     xhr   765 B  4 ms
3          http://localhost:8080/api/v1/thread/3        200     xhr   459 B  4 ms
5          http://localhost:8080/api/v1/thread/5        200     xhr   455 B  4 ms
3          http://localhost:8080/api/v1/thread/3        200     xhr   459 B  6 ms
AssetM...  http://localhost:1234/assets/AssetManife...  304     xhr   231 B  2 ms
6 / 12 requests  3.3 kB / 2.2 MB transferred  1.3 kB / 2.6 MB resources
```

```
GET http://localhost:8080/api/v1/thread/list
Computed URL: http://localhost:8080/api/v1/thread/list
Status: 200   Time: 42ms   Size: 261 bytes
```

```
{name: tester0, username: testuser0, password: password, role: [Admin]}
```

Figure 7 Logging into the web app with admin role



Threads

My Users

Created By	Thread Title	Last Active	Thread Content
testuser0	a new thread	1679865685034	thread content
testuser0	asdfggr	1679869358110	rgrgrr r rasdfg

Create Thread

Create thread below

testuser0

Report Thread

Thread content

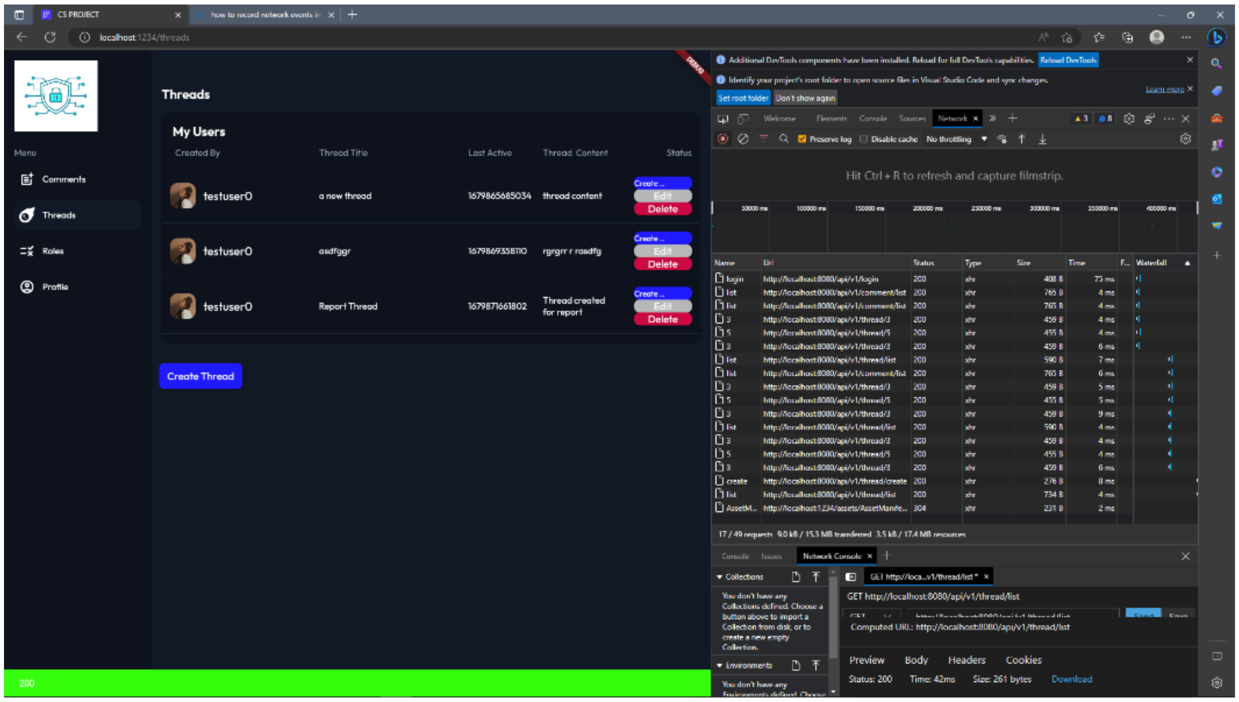
Thread created for report

Hit Ctrl + R to refresh and capture filmstrip.

Name	Url	Status	Type	Size	Time
login	http://localhost:8080/api/v1/login	200	xhr	408 B	75 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	4 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	4 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	6 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	7 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	6 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	5 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	5 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	9 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	4 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	6 ms
AssetM...	http://localhost:1234/assets/AssetManife...	304	xhr	231 B	2 ms

```
15 / 45 requests  8.0 kB / 15.0 MB transferred  3.1 kB / 17.2 MB resources
GET http://localhost:8080/api/v1/thread/list
Computed URL: http://localhost:8080/api/v1/thread/list
Status: 200   Time: 42ms   Size: 261 bytes
```

Figure 8 Creating a new Thread



Threads

My Users

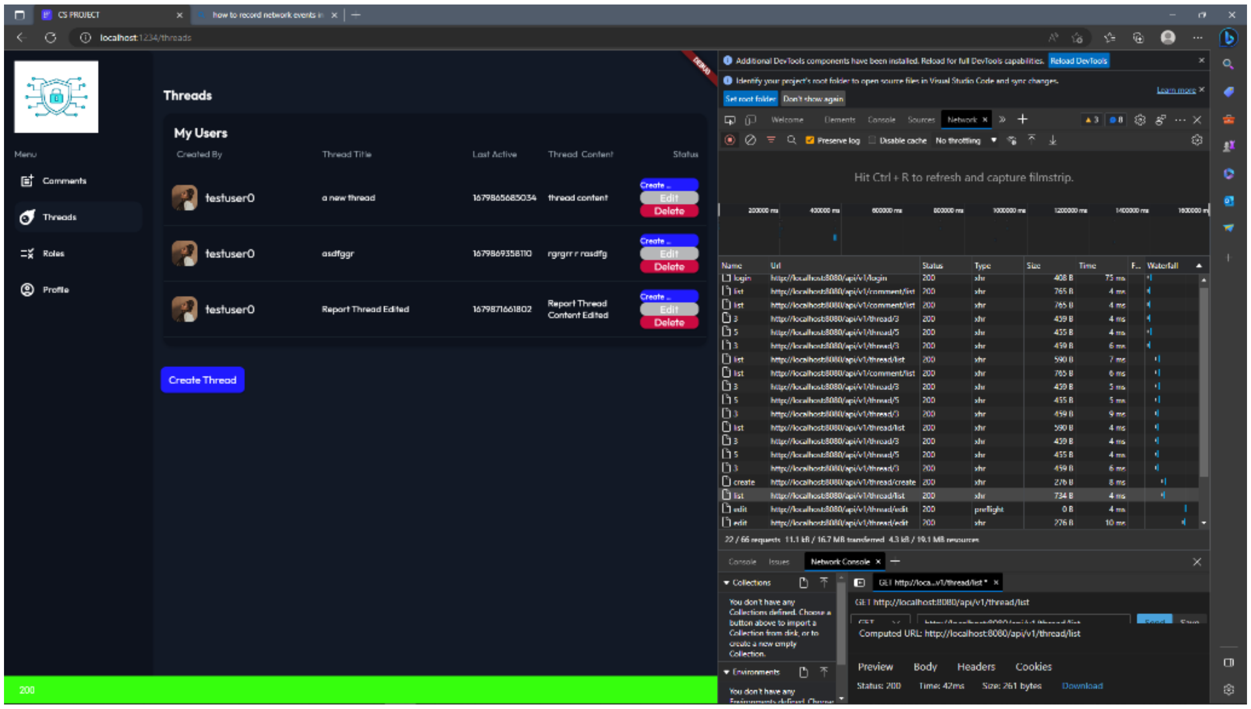
Created By	Thread Title	Last Active	Thread Content
testuser0	a new thread	1679865685034	thread content
testuser0	asdfggr	1679869358110	rgrgrr r rasdfg
testuser0	Report Thread	1679871661802	Thread created for report

Hit Ctrl + R to refresh and capture filmstrip.

Name	Url	Status	Type	Size	Time
login	http://localhost:8080/api/v1/login	200	xhr	408 B	75 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	4 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	4 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	6 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	7 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	6 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	5 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	5 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	9 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	4 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	6 ms
create	http://localhost:8080/api/v1/thread/create	200	xhr	276 B	8 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	734 B	4 ms
AssetM...	http://localhost:1234/assets/AssetManife...	304	xhr	231 B	2 ms

```
17 / 49 requests  9.0 kB / 15.3 MB transferred  3.5 kB / 17.4 MB resources
GET http://localhost:8080/api/v1/thread/list
Computed URL: http://localhost:8080/api/v1/thread/list
Status: 200   Time: 42ms   Size: 261 bytes
```

Figure 9 New thread was created and Thread list updated



Threads

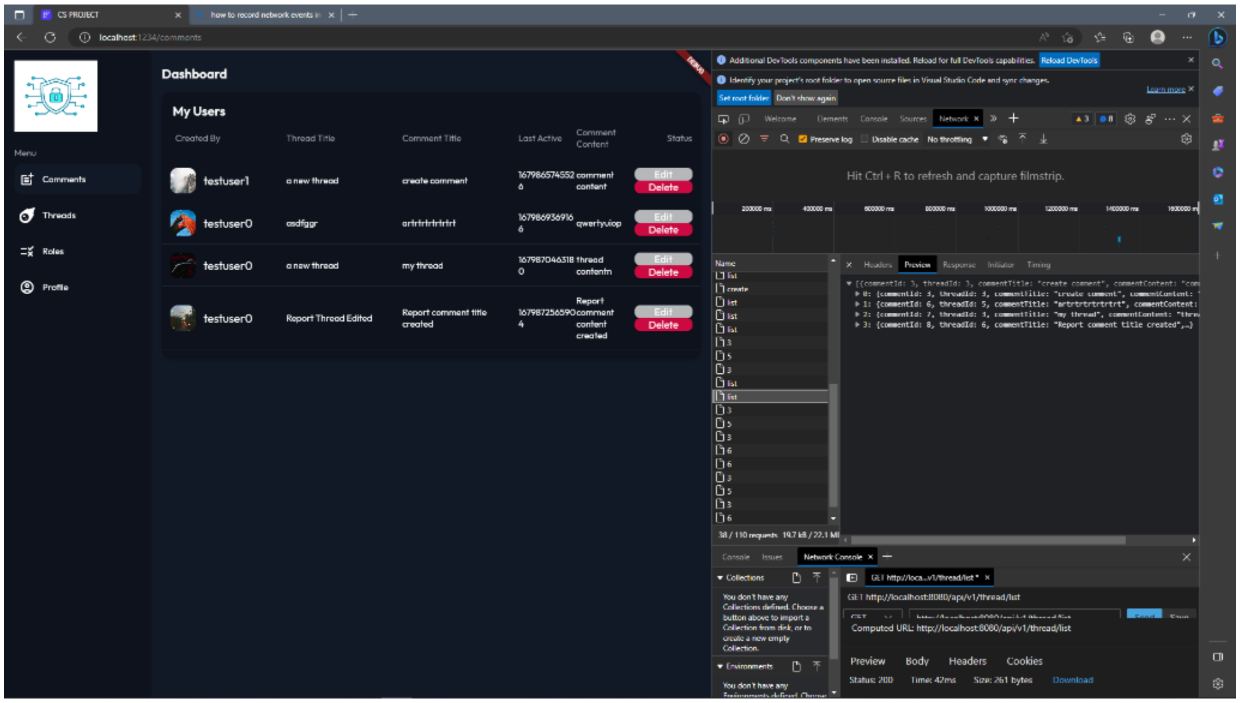
My Users

Created By	Thread Title	Last Active	Thread Content	Status
testuser0	a new thread	1679865685034	thread content	Create ... Edit Delete
testuser0	asdfggr	1679869358110	rgrgrr r rasdfg	Create ... Edit Delete
testuser0	Report Thread Edited	1679871661802	Report Thread Content Edited	Create ... Edit Delete

```
Name    Url                                          Status  Type       Size    Time
login   http://localhost:8080/api/v1/login           200     xhr        408 B   75 ms
list    http://localhost:8080/api/v1/comment/list    200     xhr        765 B   4 ms
list    http://localhost:8080/api/v1/comment/list    200     xhr        765 B   4 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        439 B   4 ms
5       http://localhost:8080/api/v1/thread/5        200     xhr        435 B   4 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B   6 ms
list    http://localhost:8080/api/v1/thread/list     200     xhr        590 B   7 ms
list    http://localhost:8080/api/v1/comment/list    200     xhr        765 B   6 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        439 B   5 ms
5       http://localhost:8080/api/v1/thread/5        200     xhr        455 B   5 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B   9 ms
list    http://localhost:8080/api/v1/thread/list     200     xhr        590 B   4 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        439 B   4 ms
5       http://localhost:8080/api/v1/thread/5        200     xhr        455 B   4 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B   6 ms
create  http://localhost:8080/api/v1/thread/create   200     xhr        276 B   8 ms
list    http://localhost:8080/api/v1/thread/list     200     xhr        734 B   4 ms
edit    http://localhost:8080/api/v1/thread/edit     200     preflight  0 B     4 ms
edit    http://localhost:8080/api/v1/thread/edit     200     xhr        276 B   10 ms
22 / 66 requests  11.1 kB / 16.7 MB transferred  4.3 kB / 19.1 MB resources

GET http://localhost:8080/api/v1/thread/list
Computed URL: http://localhost:8080/api/v1/thread/list
Status: 200   Time: 42ms   Size: 261 bytes
```

Figure 10 Successfully created a comment from an existing thread



Dashboard

My Users

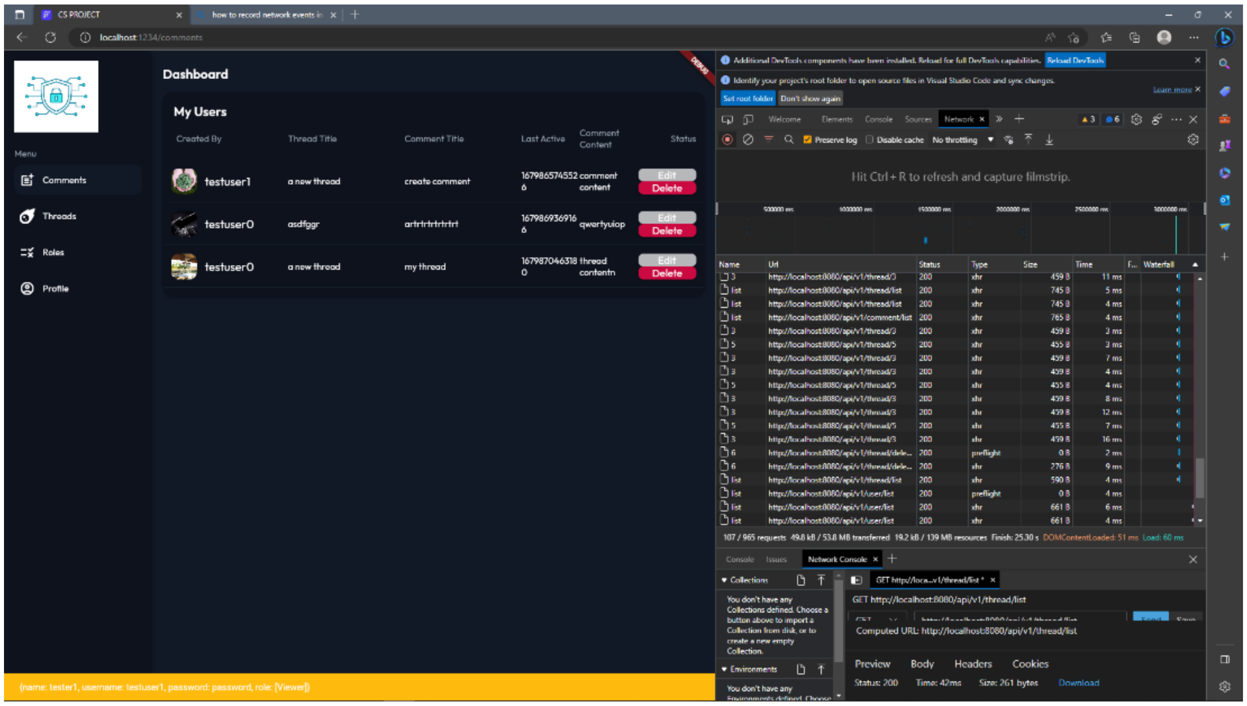
Created By	Thread Title	Comment Title	Last Active	Comment Content	Status
testuser1	a new thread	create comment	1679865745526	comment content	Edit Delete
testuser0	asdfggr	artrtrtrtrtrtrt	1679869369166	qwertyuiop	Edit Delete
testuser0	a new thread	my thread	1679870463180	thread contentn	Edit Delete
testuser0	Report Thread Edited	Report comment title created	1679872565904	Report comment content created	Edit Delete

```
▼ [{commentId: 3, threadId: 3, commentTitle: "create comment", commentContent: "com
  ▶ 0: {commentId: 3, threadId: 3, commentTitle: "create comment", commentContent: "
  ▶ 1: {commentId: 6, threadId: 5, commentTitle: "artrtrtrtrtrtrt", commentContent:
  ▶ 2: {commentId: 7, threadId: 3, commentTitle: "my thread", commentContent: "thre
  ▶ 3: {commentId: 8, threadId: 6, commentTitle: "Report comment title created",…}
38 / 130 requests  19.7 kB / 22.1 MB

GET http://localhost:8080/api/v1/thread/list
Computed URL: http://localhost:8080/api/v1/thread/list
Status: 200   Time: 42ms   Size: 261 bytes
```

Figure 11 List of comments

Viewer Role



Dashboard

My Users

Created By	Thread Title	Comment Title	Last Active	Comment Content	Status
testuser1	a new thread	create comment	1679865745526	comment content	Edit Delete
testuser0	asdfggr	artrtrtrtrtrtrt	1679869369166	qwertyuiop	Edit Delete
testuser0	a new thread	my thread	1679870463180	thread contentn	Edit Delete

{name: tester1, username: testuser1, password: password, role: [Viewer]}

Additional DevTools components have been installed. Reload for full DevTools capabilities.

Identify your project's root folder to open source files in Visual Studio Code and sync changes.

Hit Ctrl + R to refresh and capture filmstrip.

```
Name  Url                                          Status  Type       Size    Time
3     http://localhost:8080/api/v1/thread/3        200     xhr        459 B   11 ms
list  http://localhost:8080/api/v1/thread/list     200     xhr        745 B    5 ms
list  http://localhost:8080/api/v1/thread/list     200     xhr        745 B    4 ms
list  http://localhost:8080/api/v1/comment/list    200     xhr        765 B    4 ms
3     http://localhost:8080/api/v1/thread/3        200     xhr        459 B    3 ms
5     http://localhost:8080/api/v1/thread/5        200     xhr        455 B    3 ms
3     http://localhost:8080/api/v1/thread/3        200     xhr        459 B    7 ms
3     http://localhost:8080/api/v1/thread/3        200     xhr        459 B    4 ms
5     http://localhost:8080/api/v1/thread/5        200     xhr        455 B    4 ms
3     http://localhost:8080/api/v1/thread/3        200     xhr        459 B    8 ms
3     http://localhost:8080/api/v1/thread/3        200     xhr        459 B   12 ms
5     http://localhost:8080/api/v1/thread/5        200     xhr        455 B    7 ms
3     http://localhost:8080/api/v1/thread/3        200     xhr        459 B   16 ms
6     http://localhost:8080/api/v1/thread/dele...  200     preflight    0 B    2 ms
6     http://localhost:8080/api/v1/thread/dele...  200     xhr        276 B    9 ms
list  http://localhost:8080/api/v1/thread/list     200     xhr        590 B    4 ms
list  http://localhost:8080/api/v1/user/list       200     preflight    0 B    4 ms
list  http://localhost:8080/api/v1/user/list       200     xhr        661 B    6 ms
list  http://localhost:8080/api/v1/user/list       200     xhr        661 B    4 ms
107 / 965 requests  49.8 kB / 53.8 MB transferred  19.2 kB / 139 MB resources  Finish: 25.30 s  DOMContentLoaded: 51 ms  Load: 60 ms
```

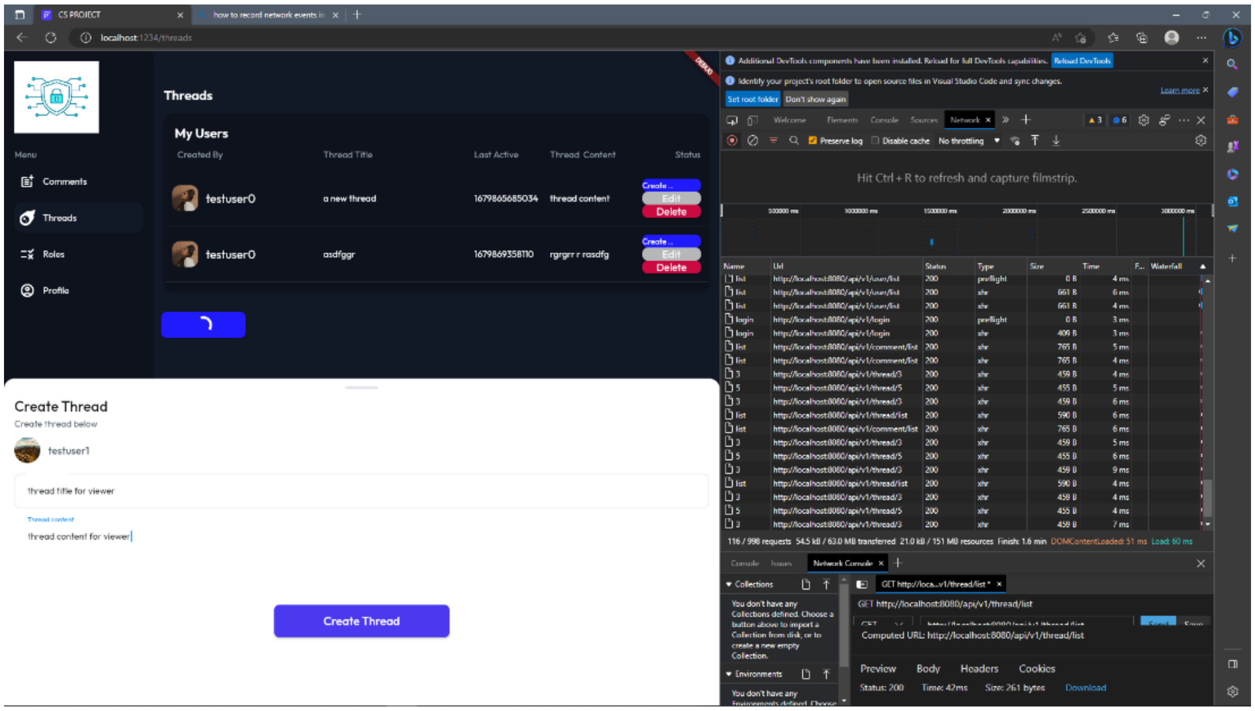
You don't have any Collections defined. Choose a button above to import a Collection from disk, or to create a new empty Collection.

GET http://localhost:8080/api/v1/thread/list

Computed URL: http://localhost:8080/api/v1/thread/list

Status: 200 Time: 42ms Size: 261 bytes

Figure 12 Signed in with viewer role.



Threads

My Users

Created By	Thread Title	Last Active	Thread Content	Status
testuser0	a new thread	1679865685034	thread content	Create ... Edit Delete
testuser0	asdfggr	1679869358110	rgrgrr r rasdfg	Create ... Edit Delete

Create Thread

Create thread below

testuser1

thread title for viewer

Thread content

thread content for viewer

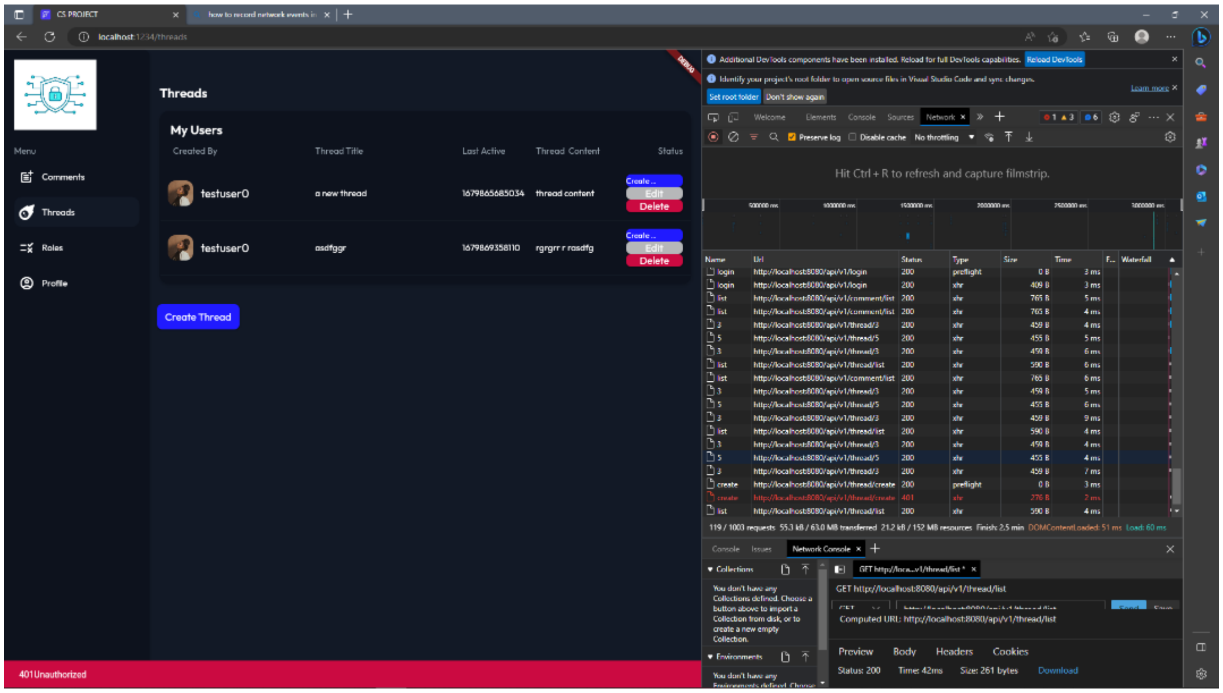
```
Name   Url                                          Status  Type       Size    Time
list   http://localhost:8080/api/v1/user/list       200     preflight    0 B    4 ms
list   http://localhost:8080/api/v1/user/list       200     xhr        661 B    6 ms
list   http://localhost:8080/api/v1/user/list       200     xhr        661 B    4 ms
login  http://localhost:8080/api/v1/login           200     preflight    0 B    3 ms
login  http://localhost:8080/api/v1/login           200     xhr        409 B    3 ms
list   http://localhost:8080/api/v1/comment/list    200     xhr        765 B    5 ms
list   http://localhost:8080/api/v1/comment/list    200     xhr        765 B    4 ms
3      http://localhost:8080/api/v1/thread/3        200     xhr        459 B    4 ms
5      http://localhost:8080/api/v1/thread/5        200     xhr        455 B    5 ms
3      http://localhost:8080/api/v1/thread/3        200     xhr        459 B    6 ms
list   http://localhost:8080/api/v1/thread/list     200     xhr        590 B    6 ms
list   http://localhost:8080/api/v1/comment/list    200     xhr        765 B    6 ms
3      http://localhost:8080/api/v1/thread/3        200     xhr        459 B    5 ms
5      http://localhost:8080/api/v1/thread/5        200     xhr        455 B    6 ms
3      http://localhost:8080/api/v1/thread/3        200     xhr        459 B    9 ms
list   http://localhost:8080/api/v1/thread/list     200     xhr        590 B    4 ms
3      http://localhost:8080/api/v1/thread/3        200     xhr        459 B    4 ms
5      http://localhost:8080/api/v1/thread/5        200     xhr        455 B    4 ms
3      http://localhost:8080/api/v1/thread/3        200     xhr        459 B    7 ms
116 / 998 requests  54.5 kB / 63.0 MB transferred  21.0 kB / 151 MB resources  Finish: 1.6 min  DOMContentLoaded: 51 ms  Load: 60 ms
```

GET http://localhost:8080/api/v1/thread/list

Computed URL: http://localhost:8080/api/v1/thread/list

Status: 200 Time: 42ms Size: 261 bytes

Figure 13 Creating a thread from a viewer role



Threads

My Users

Created By	Thread Title	Last Active	Thread Content	Status
testuser0	a new thread	1679865685034	thread content	Create ... Edit Delete
testuser0	asdfggr	1679869358110	rgrgrr r rasdfg	Create ... Edit Delete

401 Unauthorized

Name	Url	Status	Type	Size	Time
login	http://localhost:8080/api/v1/login	200	preflight	0 B	3 ms
login	http://localhost:8080/api/v1/login	200	xhr	409 B	3 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	5 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	4 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	5 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	6 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	6 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	6 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	5 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	6 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	9 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	4 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	7 ms
create	http://localhost:8080/api/v1/thread/create	200	preflight	0 B	3 ms
create	http://localhost:8080/api/v1/thread/create	401	xhr	276 B	2 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	4 ms

119 / 1003 requests 55.3 kB / 63.0 MB transferred 21.2 kB / 152 MB resources Finish: 2.5 min DOMContentLoaded: 51 ms Load: 60 ms

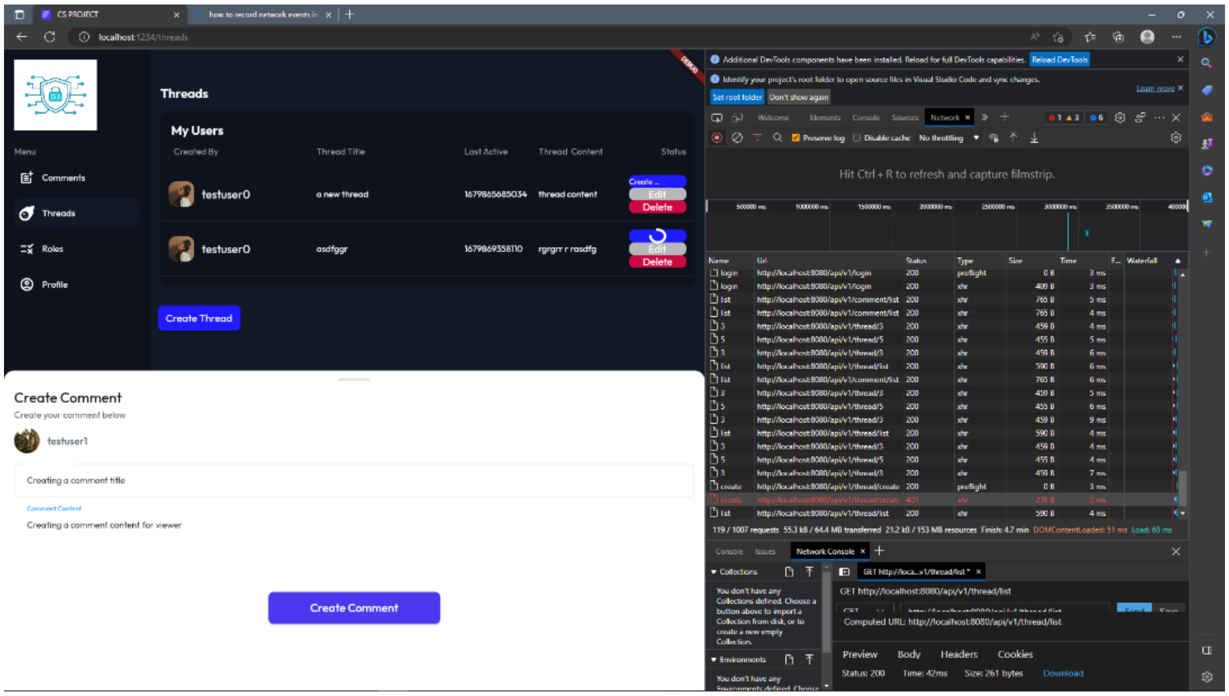
You don't have any Collections defined. Choose a button above to import a Collection from disk, or to create a new empty Collection.

GET http://localhost:8080/api/v1/thread/list

Computed URL: http://localhost:8080/api/v1/thread/list

Status: 200 Time: 42ms Size: 261 bytes

Figure 14 Unauthorized access due to insufficient authorization access



Threads

My Users

Created By	Thread Title	Last Active	Thread Content	Status
testuser0	a new thread	1679865685034	thread content	Create ... Edit Delete
testuser0	asdfggr	1679869358110	rgrgrr r rasdfg	Edit Delete

Create Comment

Create your comment below

testuser1

Creating a comment title

Comment Content

Creating a comment content for viewer

Create Comment

Name	Url	Status	Type	Size	Time
login	http://localhost:8080/api/v1/login	200	preflight	0 B	3 ms
login	http://localhost:8080/api/v1/login	200	xhr	409 B	3 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	5 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	4 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	5 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	6 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	6 ms
list	http://localhost:8080/api/v1/comment/list	200	xhr	765 B	6 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	5 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	6 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	9 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	4 ms
5	http://localhost:8080/api/v1/thread/5	200	xhr	455 B	4 ms
3	http://localhost:8080/api/v1/thread/3	200	xhr	459 B	7 ms
create	http://localhost:8080/api/v1/thread/create	200	preflight	0 B	3 ms
create	http://localhost:8080/api/v1/thread/create	401	xhr	276 B	2 ms
list	http://localhost:8080/api/v1/thread/list	200	xhr	590 B	4 ms

119 / 1007 requests 55.3 kB / 64.4 MB transferred 21.2 kB / 153 MB resources Finish: 4.7 min DOMContentLoaded: 51 ms Load: 60 ms

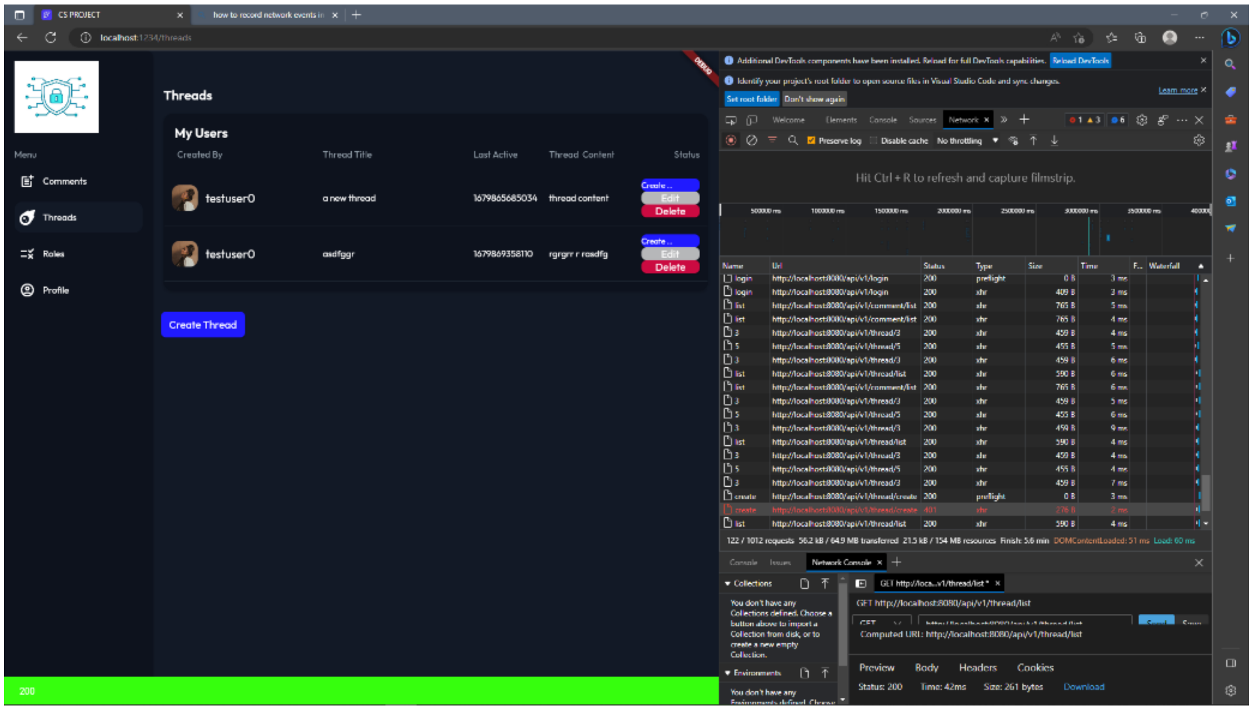
You don't have any Collections defined. Choose a button above to import a Collection from disk, or to create a new empty Collection.

GET http://localhost:8080/api/v1/thread/list

Computed URL: http://localhost:8080/api/v1/thread/list

Status: 200 Time: 42ms Size: 261 bytes

Figure 15 Creating a comment from viewer role



localhost:1234/threads

Threads

My Users

Created By	Thread Title	Last Active	Thread Content	Status
testuser0	a new thread	1679865685034	thread content	Create ... Edit Delete
testuser0	asdfggr	1679869358110	rgrgrr r rasdfg	Create ... Edit Delete

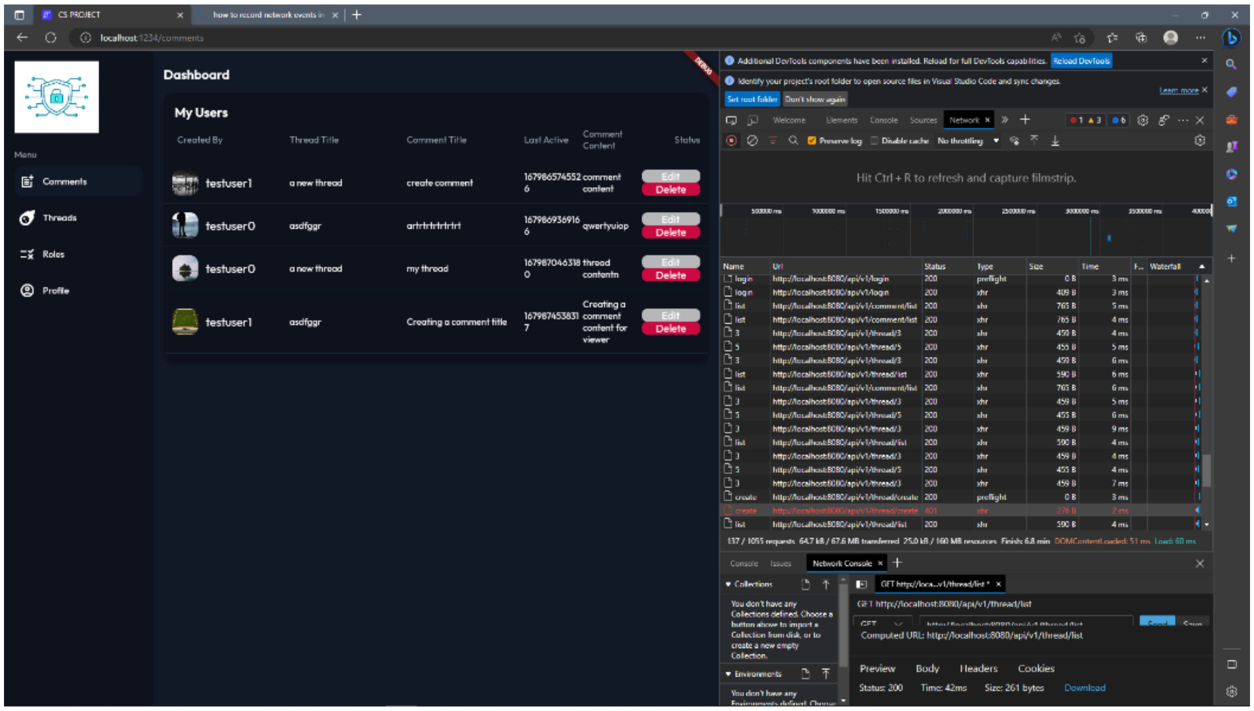
Create Thread

200

```
Name    Url                                          Status  Type       Size   Time
login   http://localhost:8080/api/v1/login           200     preflight  0 B    3 ms
login   http://localhost:8080/api/v1/login           200     xhr        409 B  3 ms
list    http://localhost:8080/api/v1/comment/list    200     xhr        765 B  5 ms
list    http://localhost:8080/api/v1/comment/list    200     xhr        765 B  4 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B  4 ms
5       http://localhost:8080/api/v1/thread/5        200     xhr        455 B  5 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B  6 ms
list    http://localhost:8080/api/v1/thread/list     200     xhr        590 B  6 ms
list    http://localhost:8080/api/v1/comment/list    200     xhr        765 B  6 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B  5 ms
5       http://localhost:8080/api/v1/thread/5        200     xhr        455 B  6 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B  9 ms
list    http://localhost:8080/api/v1/thread/list     200     xhr        590 B  4 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B  4 ms
5       http://localhost:8080/api/v1/thread/5        200     xhr        455 B  4 ms
3       http://localhost:8080/api/v1/thread/3        200     xhr        459 B  7 ms
create  http://localhost:8080/api/v1/thread/create   200     preflight  0 B    3 ms
create  http://localhost:8080/api/v1/thread/create   401     xhr        276 B  2 ms
list    http://localhost:8080/api/v1/thread/list     200     xhr        590 B  4 ms
122 / 1012 requests  56.2 kB / 64.9 MB transferred  21.5 kB / 154 MB resources  Finish: 5.6 min  DOMContentLoaded: 51 ms  Load: 60 ms

GET http://localhost:8080/api/v1/thread/list
Computed URL: http://localhost:8080/api/v1/thread/list
Status: 200   Time: 42ms   Size: 261 bytes
```

Figure 16 Successfully created a comment from viewer role



localhost:1234/comments

Dashboard

My Users

Created By	Thread Title	Comment Title	Last Active	Comment Content	Status
testuser1	a new thread	create comment	1679865745526	comment content	Edit Delete
testuser0	asdfggr	artrtrtrtrtrtrt	1679869369166	qwertyuiop	Edit Delete
testuser0	a new thread	my thread	1679870463180	thread contentn	Edit Delete
testuser1	asdfggr	Creating a comment title	1679874538317	Creating a comment content for viewer	Edit Delete

```
137 / 1055 requests  64.7 kB / 67.6 MB transferred  25.0 kB / 160 MB resources  Finish: 6.8 min  DOMContentLoaded: 51 ms  Load: 60 ms

GET http://localhost:8080/api/v1/thread/list
Computed URL: http://localhost:8080/api/v1/thread/list
Status: 200   Time: 42ms   Size: 261 bytes
```

Figure 17 New comment by viewer role