ResolveIT: IT Helpdesk

HW 6 - Team 5

Client Server Architecture

- **User:** Initiates actions or requests by interacting with the client application's graphical user interface.
- **Client Application (Java Swing):** Presents the GUI to the user and communicates user actions to the server. It sends login details or service requests and receives responses from the server.
- **Server:** Processes requests from the client application. It validates login credentials, handles service requests, and interacts with the database. Then, it sends the results back to the client application.
- **SQLite Database:** Stores user data, login credentials, and service request details. It is accessed and modified by the server based on client requests.

ServerMain

- **Initialization:** Server creates a ServerSocket on port 6868 to listen for client connections.
- **Listening for Clients:** Server enters an infinite loop, waiting for clients using the blocking accept() method of ServerSocket.
- **Handling Client Connections:** Each client connection returns a new Socket object for two-way communication.
- **Multithreading:** Server spawns a new ServerThread for each client to handle concurrent processing.
- **Client Communication:** ServerThread processes client requests (e.g., login) and sends responses back.
- **Robustness:** Error handling ensures server stability and graceful handling of unexpected situations.

```
/Users/tanishqpadwal/Library/Java/JavaVirtualMachines/openjdk-21.0.2/Contents/Home/bin
 /Users/tanishqpadwal/Desktop/Helpdesk/out/production/Helpdesk:/Users/tanishqpadwal/Do
 .jar:/Users/tanishqpadwal/Downloads/javax.mail.jar:/Users/tanishqpadwal/Downloads/mai
Server started and listening on port 6868
Message from client: login,user,user
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Message from client: login,user,user
```

ServerThread

- **SocketManagement:** Each ServerThread gets a dedicated Socket for two-way communication.
- **CommunicationChannels:** Sets up input and output streams using BufferedReader and PrintWriter.
- **RequestProcessing:** Thread continuously listens for client messages in a predefined format.
- **CommandHandling:** Parses messages, checks for login requests, and validates credentials.
- **ResponseSending:** Sends success/failure messages based on login outcome or error messages for invalid requests.
- **ConnectionClosure:** Closes socket after client disconnects or on exceptions for resource release, ending the session.

ClientMain

- **Client Application Launch:** Uses SwingUtilities.invokeLater for GUI creation on the EDT.
- **User Interaction:** LoginFrame presents login interface; user inputs credentials.
- **Server Communication:** Client initiates Socket connection to server on port 6868.
- **Request Handling:** ServerThread processes login request, validates credentials.
- **Response:** ServerThread sends success/failure response to client.
- **Client Processing:** LoginFrame processes server response, transitions accordingly.

Limitations Faced

- **Distribution Challenges:** The necessity for manual configuration of the database path makes it difficult to seamlessly distribute our application to a broader audience.
- **User Experience Impact:** This manual setup process can be a barrier for less technically inclined users, potentially limiting the application's accessibility and usability.
- **Scalability Concerns:** As our application grows in features and complexity, the dependency on local configurations such as database paths could further complicate deployment and scalability.
- Our team acknowledges this limitation and is exploring potential solutions, such as containerization with Docker and migrating to cloud-based database services, to make our application more accessible and easier to use for everyone.

```
19 usages   t4n15hq
public class DatabaseHelper {
    1 usage
    private static final String DB_URL = "jdbc:sqlite:/Users/tanishqpadwal/Desktop/Helpdesk/MyHelpdeskDB.db";
```

Accessing Data

Application and Database Interaction:

- Our application interacts with an SQLite database to store and retrieve data.
- Database API: We use JDBC (Java Database Connectivity) for database operations within our Java application.
- Operations: Include creating users, submitting and managing service requests, and authenticating user logins.

Connecting to the Database:

- Run DatabaseSetup.java, this should create a new db called MyHelpdesk.db on user system.
- Navigate to DatabaseHelper.java and update path in DB_URL variable.
- Run ServerMain.java followed by ClientMain.java that launches the GUI while ServerMain enables it to listen for incoming client connections.
- Application is ready to use.

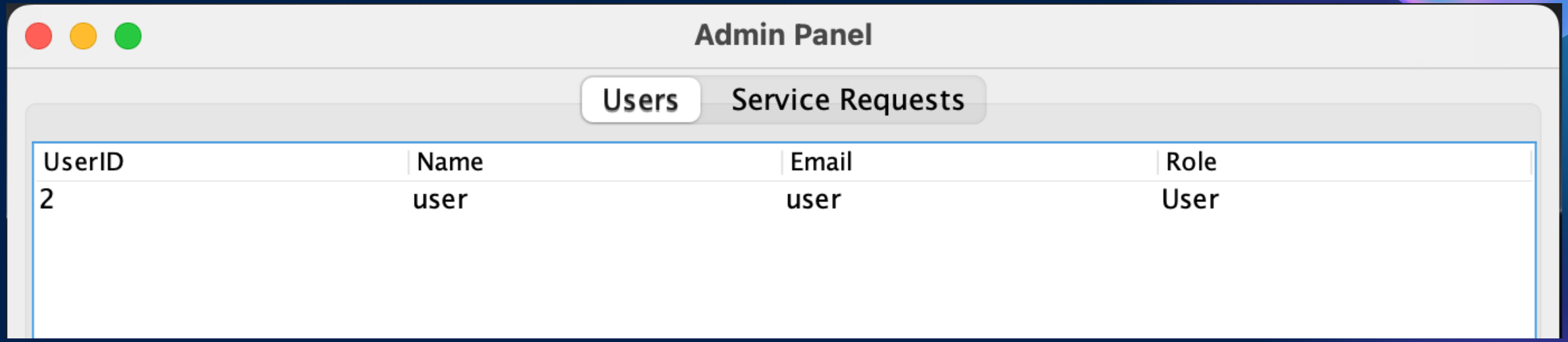
Verifying Data

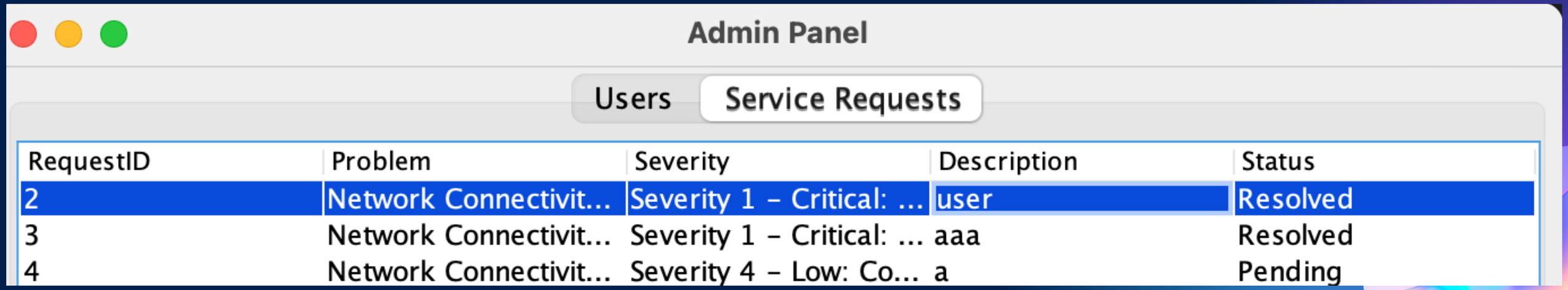
Verifying data via Admin Panel:

- Open Admin Panel on main login frame.
- The AdminPanel class uses a JTabbedPane to separate user management and service request management into two tabs, improving the organization and accessibility of these administrative functions.
- The loadUsersData() and loadServiceRequestsData() methods dynamically load data from the database and display it in their respective tables. This real-time data fetching ensures the admin panel displays the most current information.
- The data on Admin Panel is the same as the data on the database.

Verifying Data

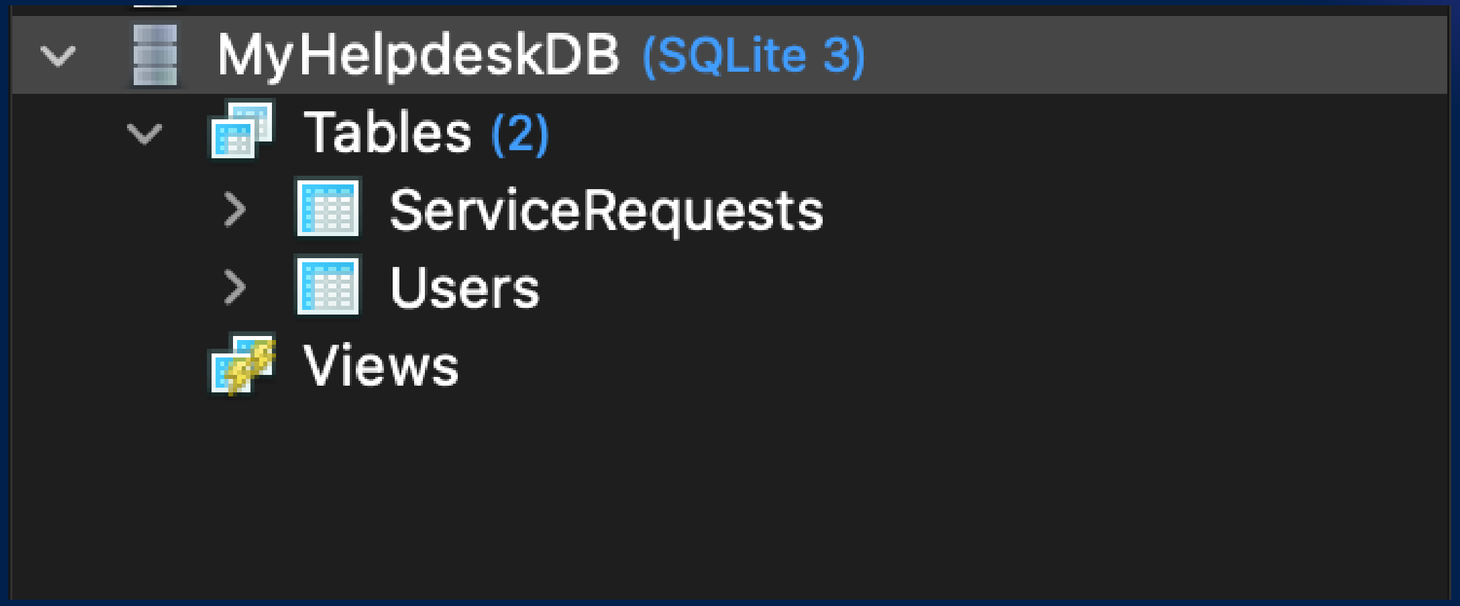
Data On Admin Panel

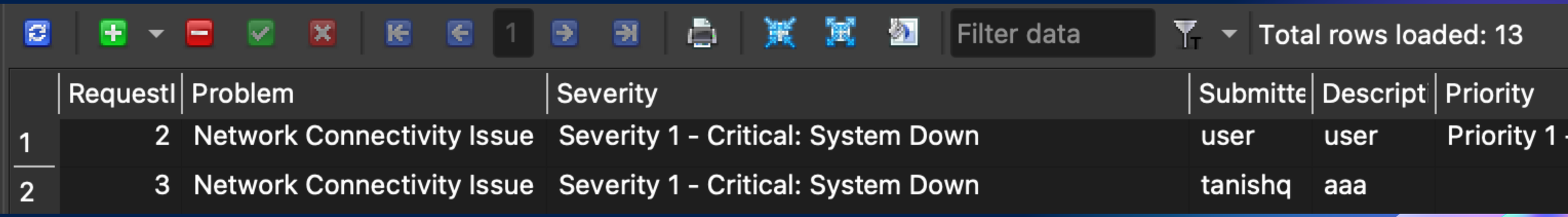




Verifying Data

Data On SQLiteStudio

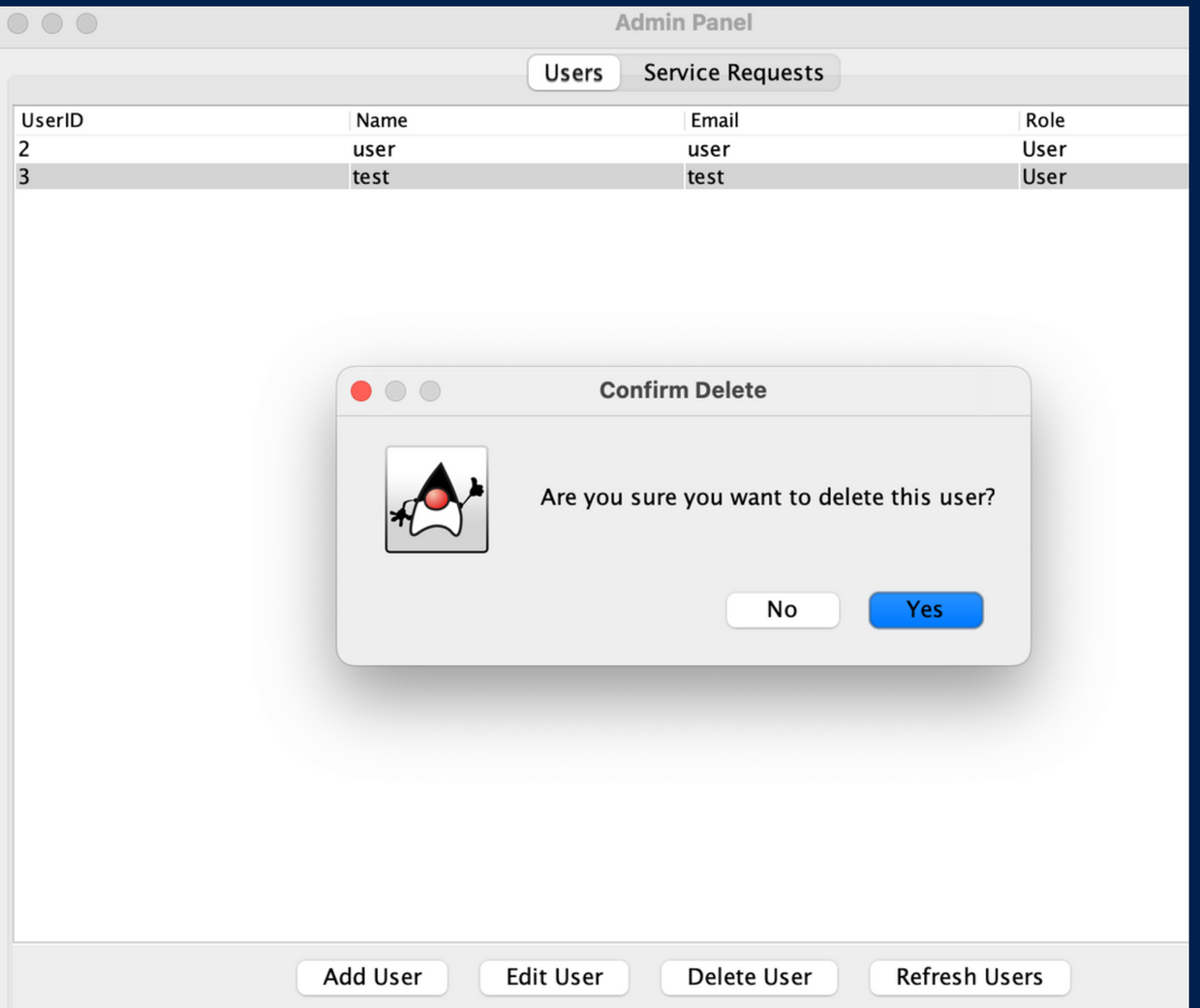


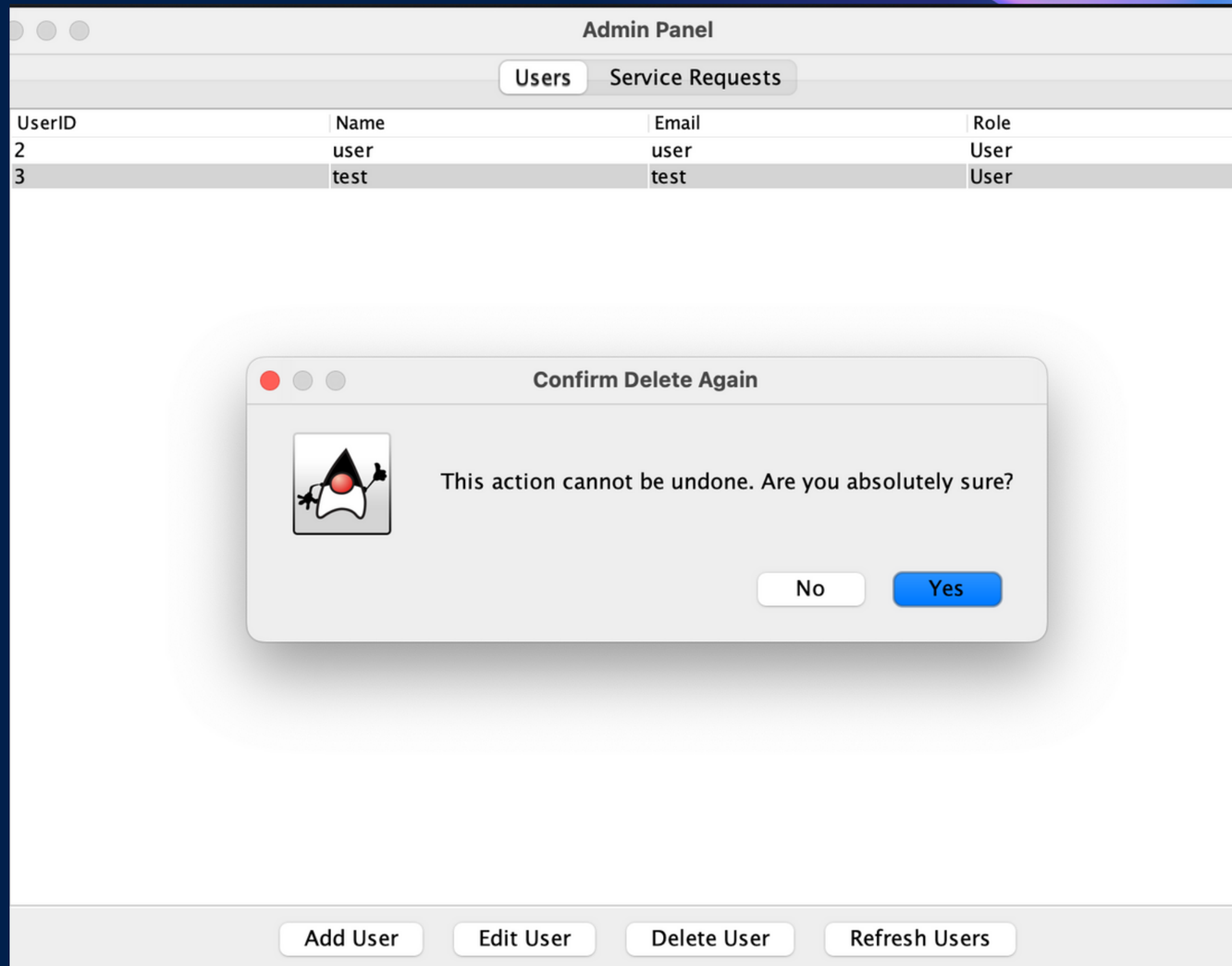


	RequestI	Problem	Severity	Submitte	Descript	Priority
1	2	Network Connectivity Issue	Severity 1 - Critical: System Down	user	user	Priority 1
2	3	Network Connectivity Issue	Severity 1 - Critical: System Down	tanishq	aaa	

Improvements from HW5 Suggestions

Extra Confirmation for deleting Users and service requests





Improvements from HW5 Suggestions

Uneditable fields are greyed out

View Requests

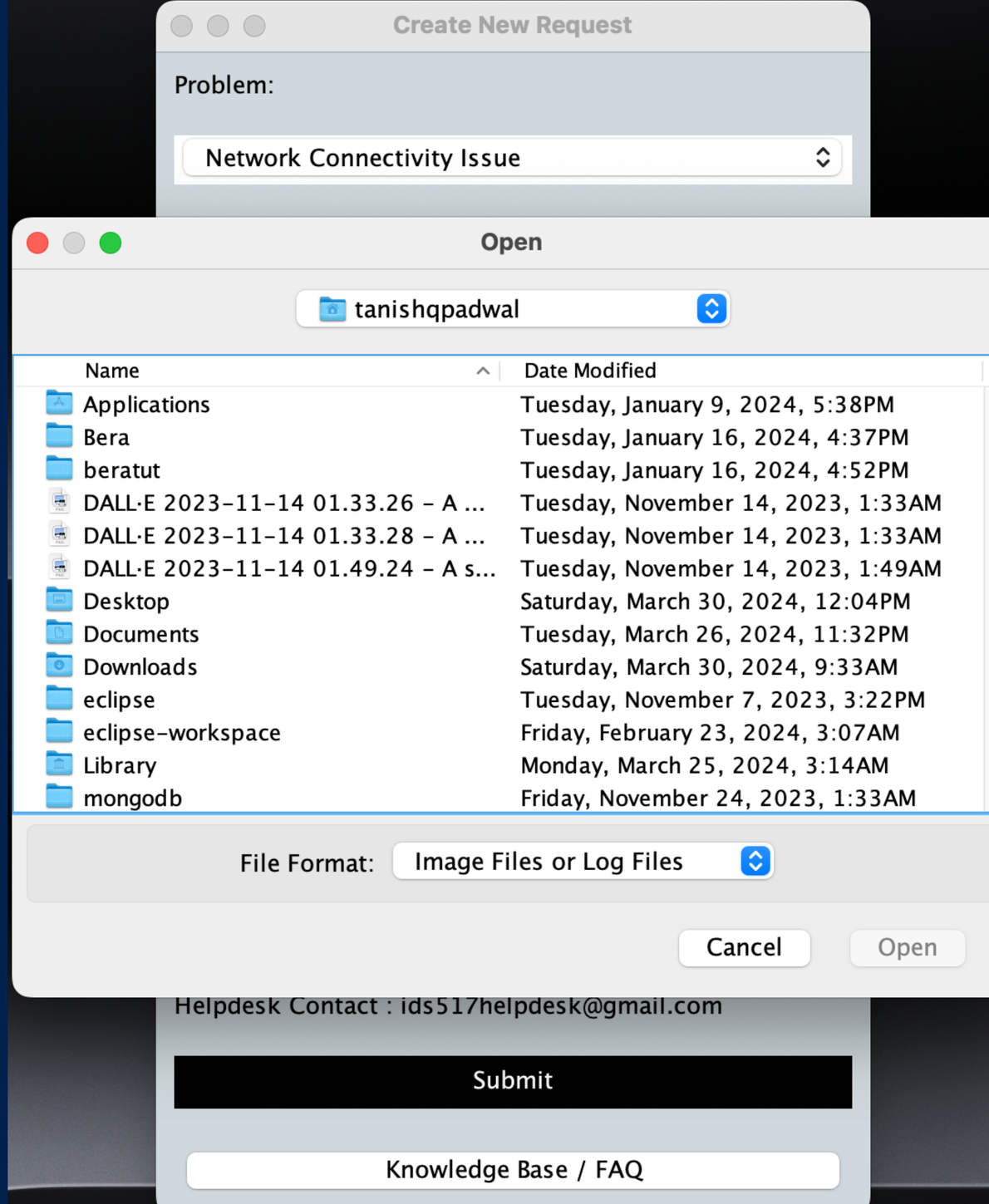
Ticket ID: Search

RequestID	Problem	Priority	Severity	Description	Status	Comment	Submitted By	Assigned To	Request Raised	Resolution Date
2	Network Connectiv...	Priority 1 - Immed...	Severity 1 - Critica...	user	Resolved	aaa	user	IT Helpdesk	2024-03-28 04:...	2024-03-28
3	Network Connectiv...		Severity 1 - Critica...	aaa	Resolved		tanishq	IT Helpdesk	2024-03-28 05:...	2024-03-28
4	Network Connectiv...		Severity 4 - Low: ...	a	Pending		a	IT Helpdesk	2024-03-28 05:...	
5	Network Connectiv...		Severity 1 - Critica...	a	Pending		a	IT Helpdesk	2024-03-28 05:...	
6	Network Connectiv...		Severity 2 - High: ...	a	Pending		a	IT Helpdesk	2024-03-28 05:...	
7	Network Connectiv...		Severity 3 - Mediu...	aa	Pending			IT Helpdesk	2024-03-28 05:...	
8	Network Connectiv...		Severity 1 - Critica...	aa	Pending			IT Helpdesk	2024-03-28 05:...	
9	Network Connectiv...	Priority 3 - Medium	Severity 4 - Low: ...	aaa	Resolved	aaaa		IT Helpdesk	2024-03-28 05:...	2024-03-28
10	Network Connectiv...		Severity 2 - High: ...	ads	Pending			IT Helpdesk	2024-03-29 20:...	
11	Network Connectiv...		Severity 1 - Critica...	aa	Pending		aa	IT Helpdesk	2024-03-29 20:...	

Supervisor Login

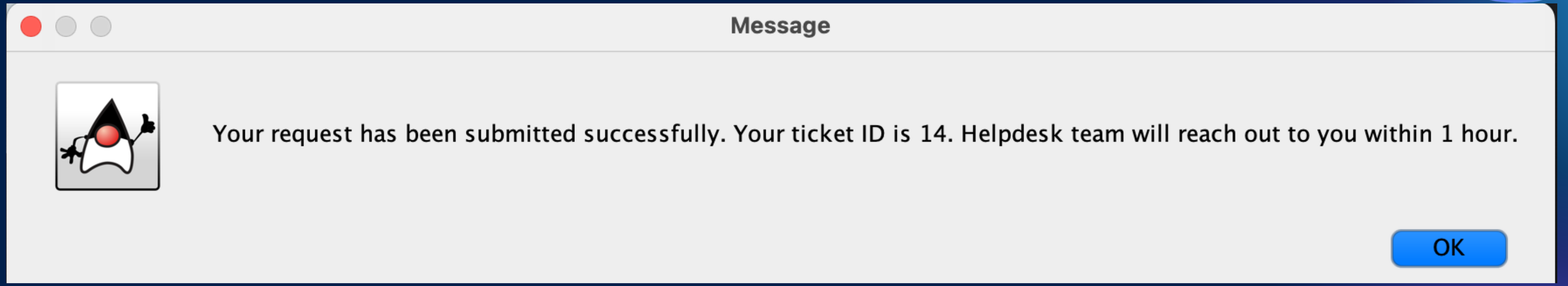
Improvements from HW5 Suggestions

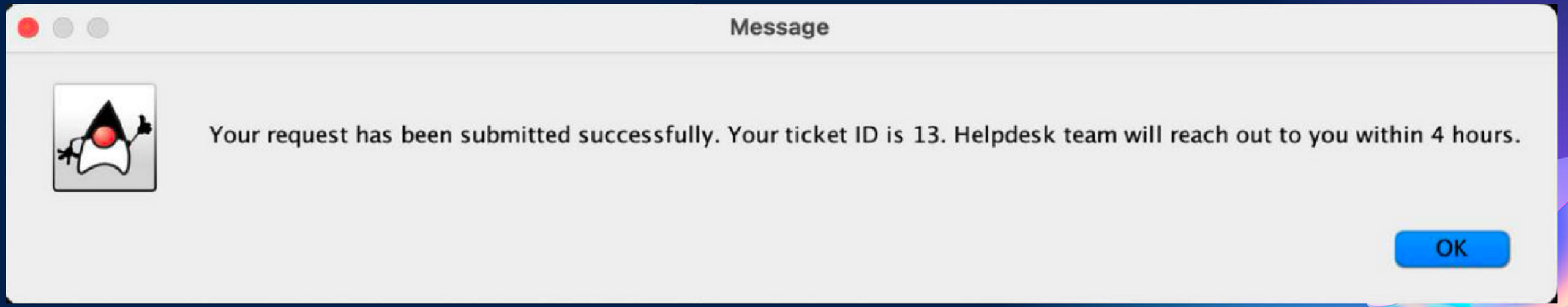
- File upload option for error logs & screenshots



Improvements from HW5 Suggestions

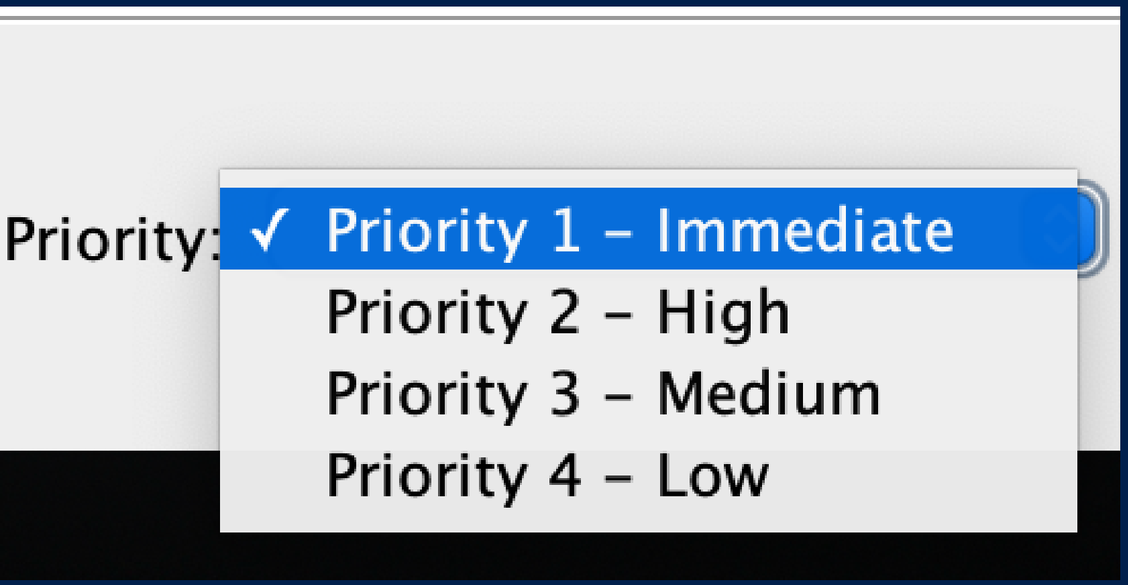
- Custom SLAs based on severity

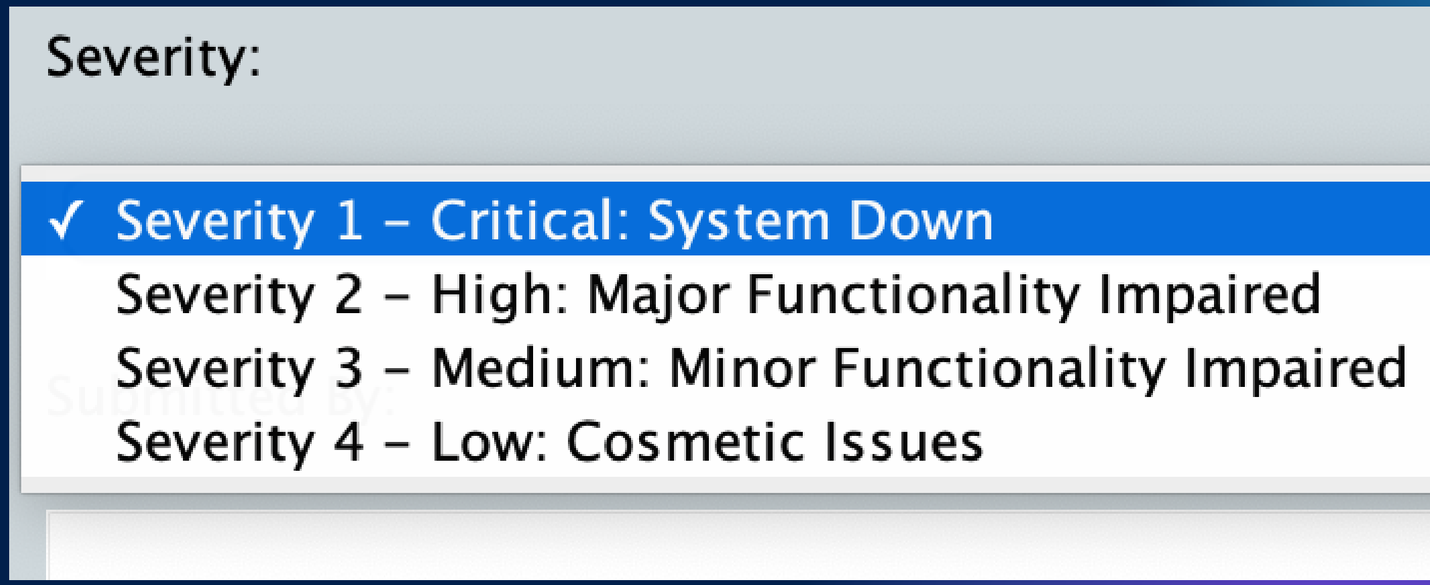




Improvements from HW5 Suggestions

- Industry Standard wording used for Severity and Priority Dropdowns





Thank You