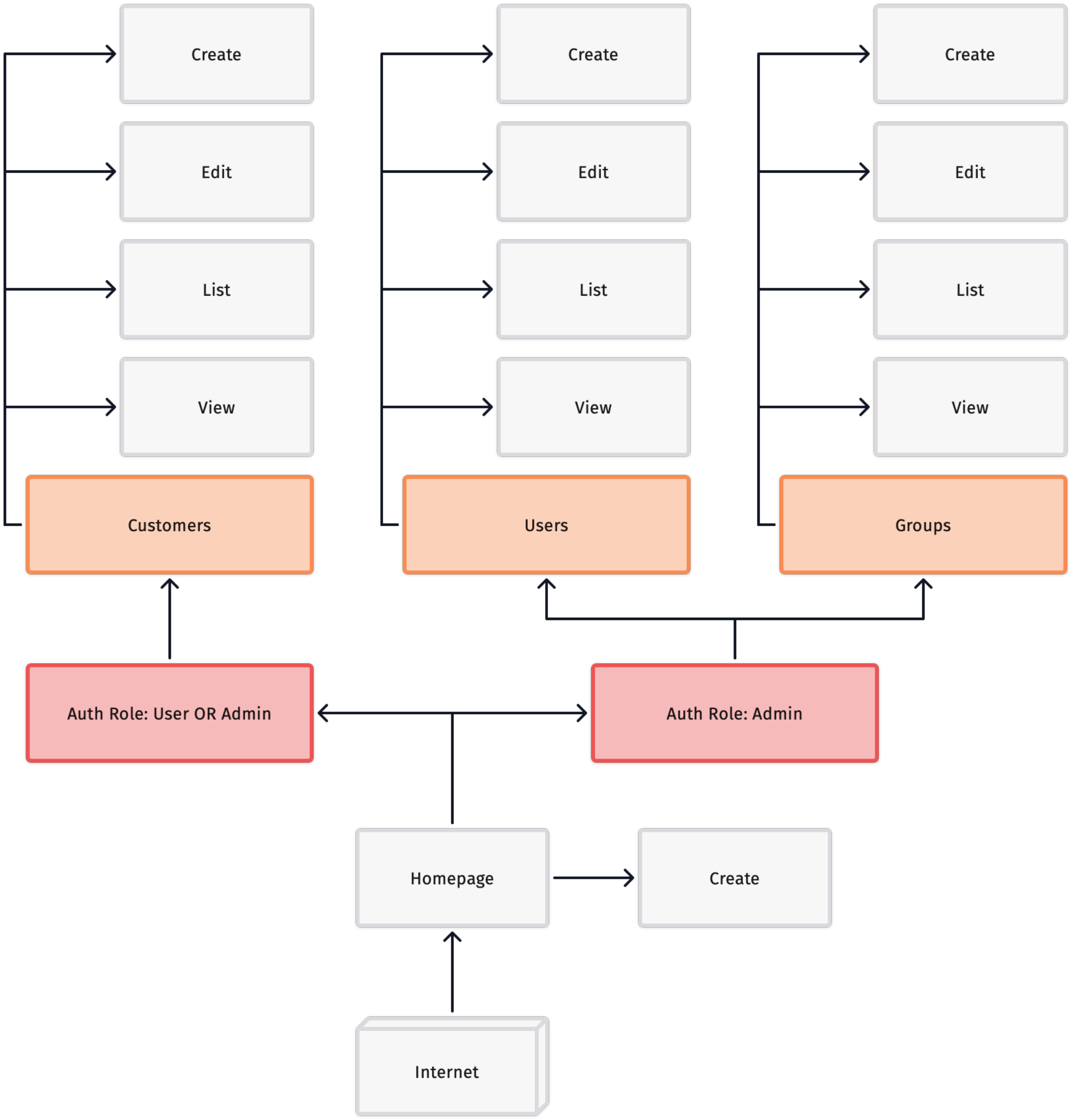
Description

The project contained in this repository functions as a CRM that allows for authenticated users to manage a set of customers and balances in a centralized place.

Features

- Authentication
    - New user creation
    - Registered user login
    - Cascading permission levels (Admin>User>Visitor)
    - Only Admins can change passwords
    - User/session logout
    - `whoami` status on the main page
- Customer management
    - Customer details table
    - Users can add/edit/remove customers and balances

Architecture



Setup

Requirements

To run this project, the server must be running **Glassfish 7** with an accessible **MySQL 9** server. The following instructions will connect Glassfish to a JDBCRealm database inside the MySQL server.

Database

The MySQL database is assumed to be running on port 3306.

Three SQL scripts are located under `configuration/database/*`. Run these scripts against a fresh MySQL database to prepare the data.

Users

```
create table if not exists USERS
(
    USERID   varchar(255) not null
```

```
        primary key,
    PASSWORD varchar(255) not null
);


INSERT INTO jdbcrealm.USERS (USERID, PASSWORD) VALUES ('admin', 'admin1234');
INSERT INTO jdbcrealm.USERS (USERID, PASSWORD) VALUES ('callum', 'ward');
INSERT INTO jdbcrealm.USERS (USERID, PASSWORD) VALUES ('carter', 'password');
INSERT INTO jdbcrealm.USERS (USERID, PASSWORD) VALUES ('jerry', 'jerry1234');
INSERT INTO jdbcrealm.USERS (USERID, PASSWORD) VALUES ('sanjay', 'gupta');
```

Groups

```
create table if not exists USERS_GROUPS
(
    GROUPID varchar(20)  not null,
    USERID  varchar(255) not null
        primary key
);
```

```
INSERT INTO jdbcrealm.USERS_GROUPS (GROUPID, USERID) VALUES
('admin', 'admin');

INSERT INTO jdbcrealm.USERS_GROUPS (GROUPID, USERID) VALUES
('user', 'callum');

INSERT INTO jdbcrealm.USERS_GROUPS (GROUPID, USERID) VALUES
('user', 'carter');

INSERT INTO jdbcrealm.USERS_GROUPS (GROUPID, USERID) VALUES
('user', 'jerry');

INSERT INTO jdbcrealm.USERS_GROUPS (GROUPID, USERID) VALUES
('user', 'sanjay');
```

Customers

```
create table if not exists CUSTOMERS
(
    ID      int          not null
        primary key,
    Name    varchar(255) null,
    Balance float        null
);
```

```
INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(1, 'Kratos Concrete', 2500);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(2, 'BW\'s Pub', 700);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(3, 'The Curiosity Shop', 57650);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(4, 'Aperture Science', 13000);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(5, 'Umbrella Corporation', 98750);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(6, 'Chocobo Farm', 3200);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(7, 'Rapture Fisheries', 6700);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(8, 'Black Mesa Research', 42000);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(10, 'Shinra Electric Power Company', 150000);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(11, 'Vault-Tec Corporation', 76500);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(12, 'Nook\'s Cranny', 25600);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(13, 'Tricell Pharmaceuticals', 87300);

INSERT INTO jdbcrealm.CUSTOMERS (ID, Name, Balance) VALUES
(14, 'Freddy Fazbear\'s Pizza', 4200);
```

Glassfish

The following instructions assume the admin console is available on port 4848. We will need to login to the console to setup the database and authentication.

Driver

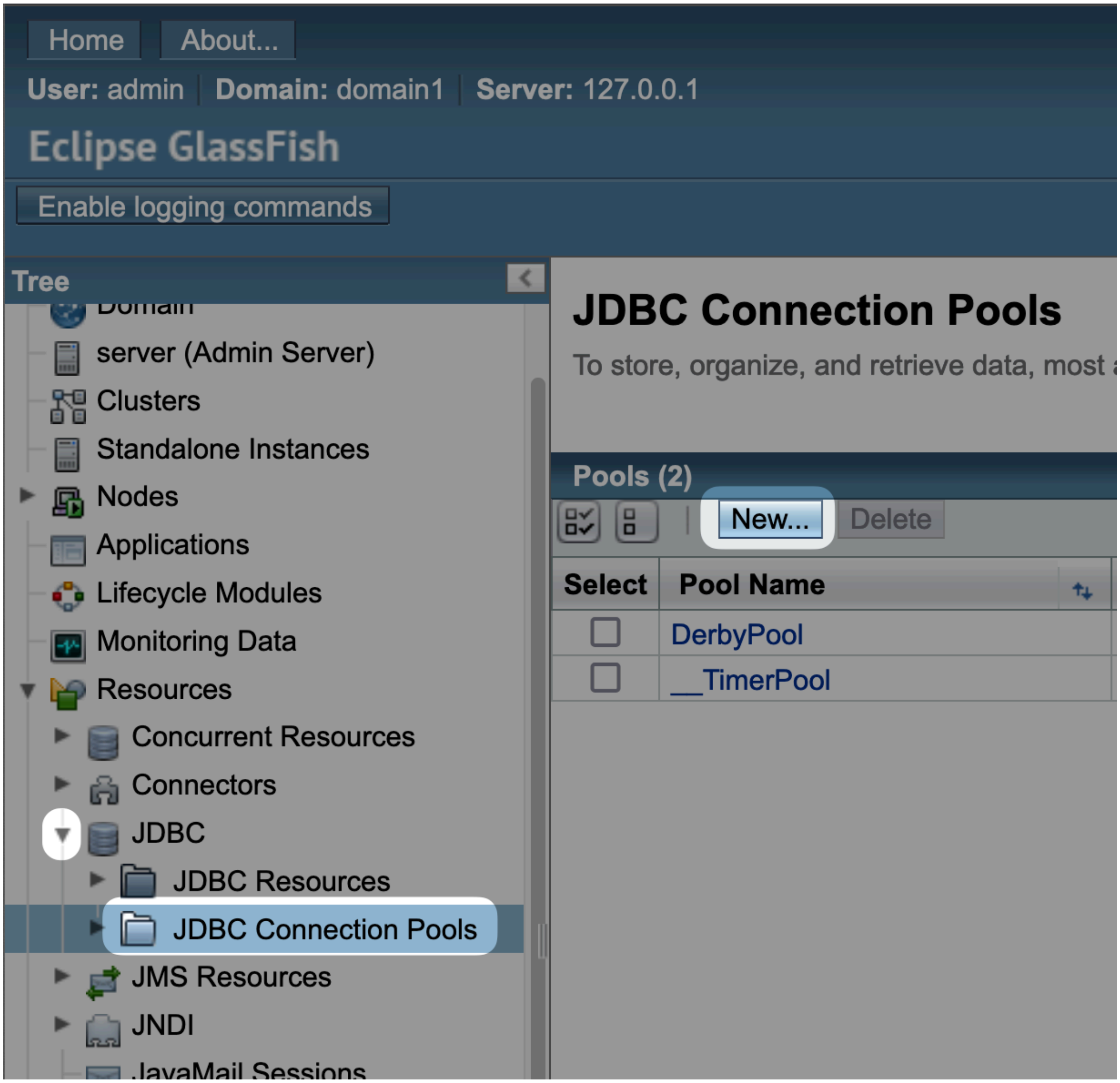
Before connecting to the MySQL server, the Glassfish server must be provided with the required driver class. The connector can be downloaded from the MySQL Website as a `Platform Independent (Architecture Independent), ZIP Archive`.

Inside the root level of the zip file, take the file named `mysql-connector-j-9.1.0.jar` and move it to `/$GLASSFISH_HOME/glassfish/lib/`. Reboot Glassfish to properly load the driver.

Connection Pool

First, a connection pool will need to be configured to connect to a MySQL database in which to store the user and customer accounts.

Navigate to the JDBC Connection Pools folder, then click to add a new Connection Pool.

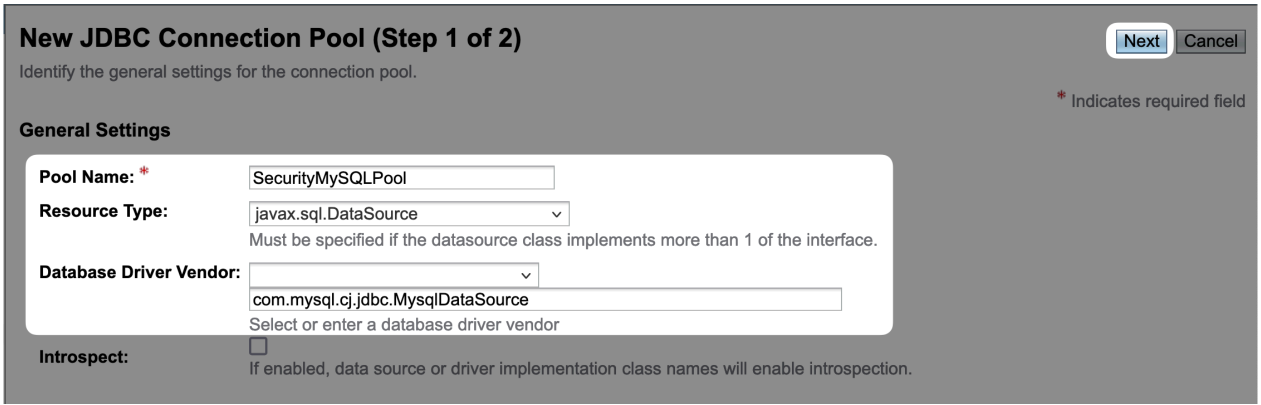


JDBC Connection Pools

To store, organize, and retrieve data, most a

Select	Pool Name
☐	DerbyPool
☐	__TimerPool

Fill in the following details for the Connection Pool, then click Next.



New JDBC Connection Pool (Step 1 of 2)

Identify the general settings for the connection pool.

* Indicates required field

General Settings

Pool Name: * SecurityMySQLPool

Resource Type: javax.sql.DataSource

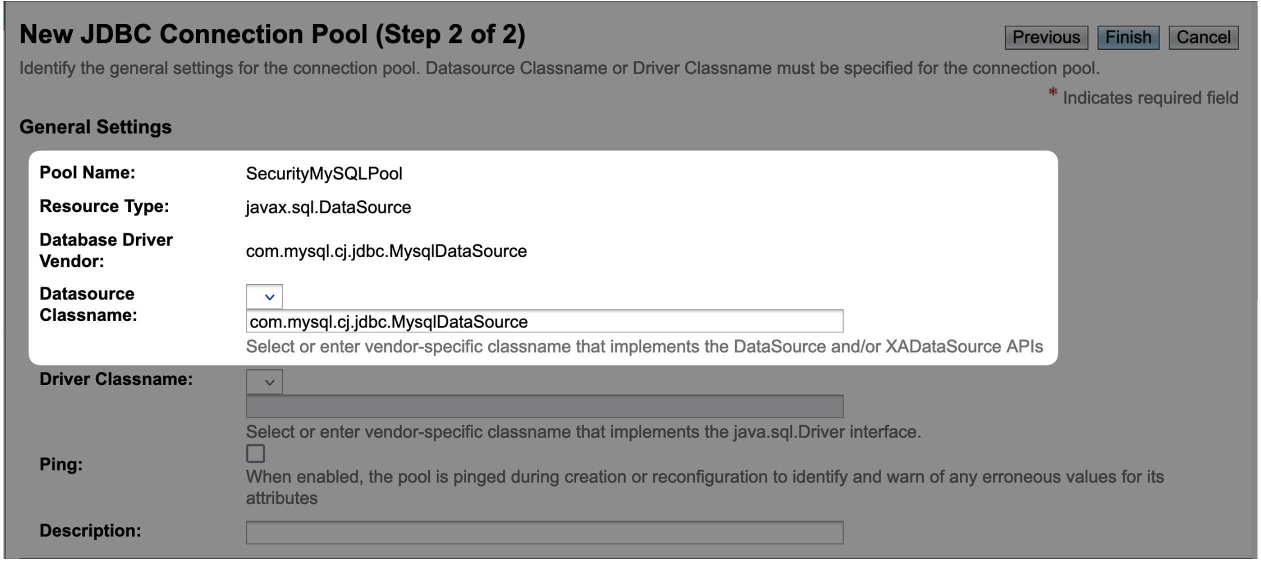
Must be specified if the datasource class implements more than 1 of the interface.

Database Driver Vendor: com.mysql.cj.jdbc.MysqlDataSource

Select or enter a database driver vendor

Introspect: □

If enabled, data source or driver implementation class names will enable introspection.



New JDBC Connection Pool (Step 2 of 2)

Identify the general settings for the connection pool. Datasource Classname or Driver Classname must be specified for the connection pool.

* Indicates required field

General Settings

Pool Name: SecurityMySQLPool

Resource Type: javax.sql.DataSource

Database Driver Vendor: com.mysql.cj.jdbc.MysqlDataSource

Datasource Classname: com.mysql.cj.jdbc.MysqlDataSource

Select or enter vendor-specific classname that implements the DataSource and/or XADataSource APIs

Driver Classname:

Select or enter vendor-specific classname that implements the java.sql.Driver interface.

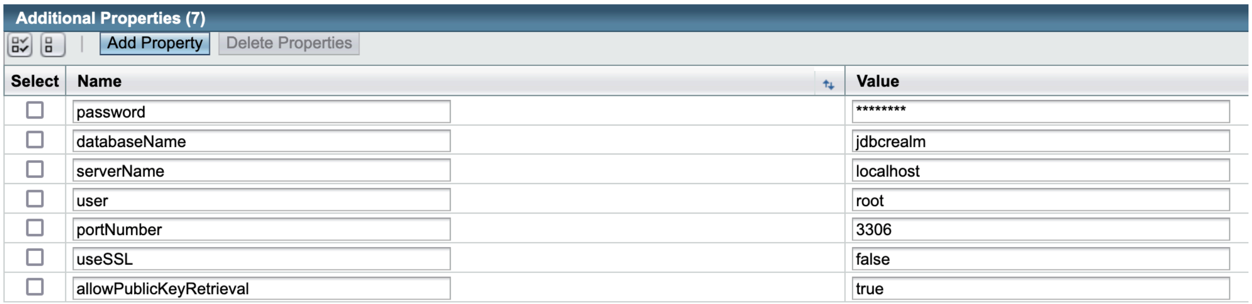
Ping: □

When enabled, the pool is pinged during creation or reconfiguration to identify and warn of any erroneous values for its attributes

Description:

Datasource Classname: `com.mysql.cj.jdbc.MysqlDataSource`

At the bottom, fill in the Additional Properties, then click Finish.



Additional Properties (7)

Select	Name	Value
□	password	********
□	databaseName	jdbcrealm
□	serverName	localhost
□	user	root
□	portNumber	3306
□	useSSL	false
□	allowPublicKeyRetrieval	true

Password: <your database password>

databaseName: `jdbcRealm`

serverName: `localhost`

user: `root`

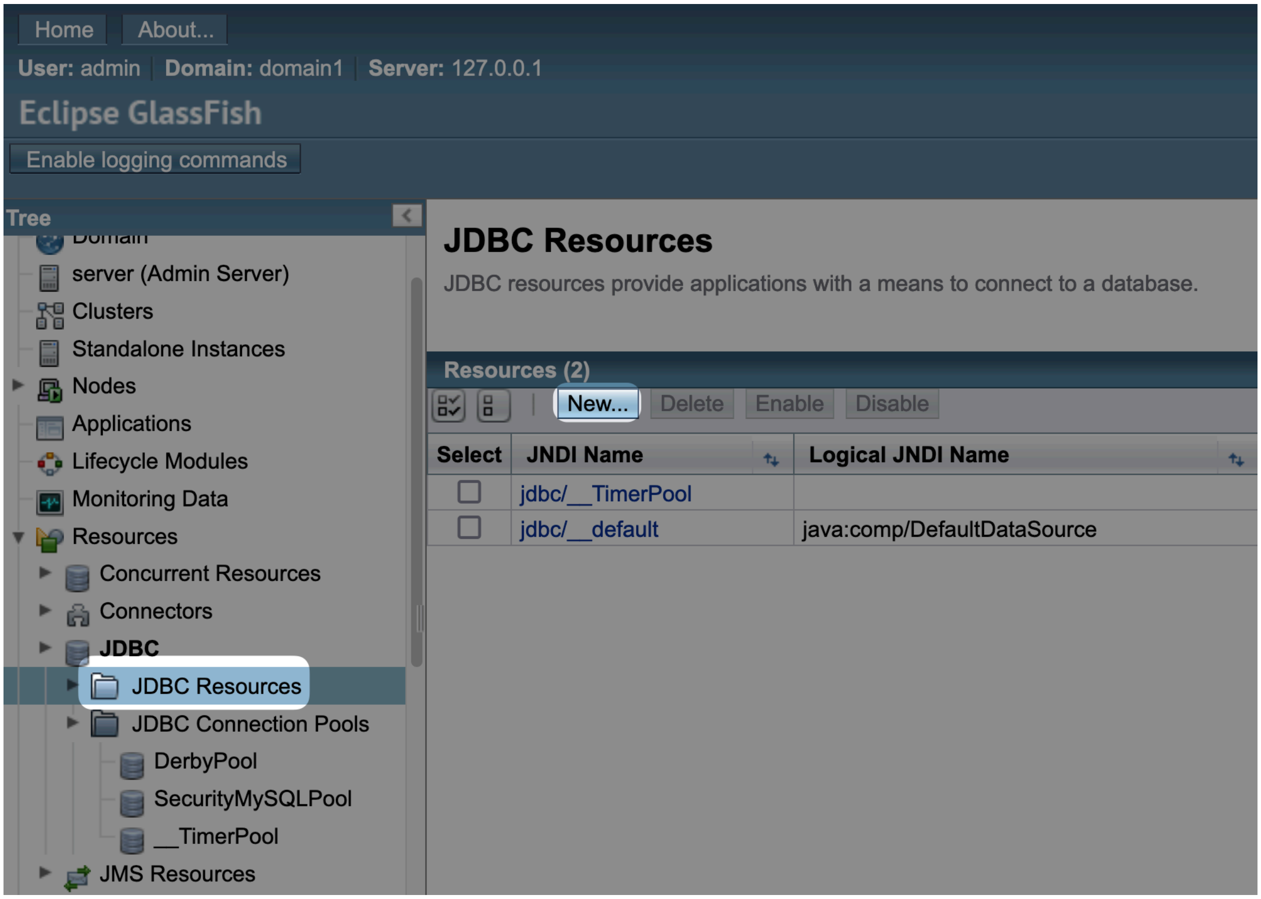
portNumber: `3306`

useSSL: `false`

allowPublicKeyRetrieval: `true`

Resource

Now that the Connection Pool has been created to the database, we must add it as a Data Source before we can use it in the security Realm. Navigate to JDBC Resources, then click New.



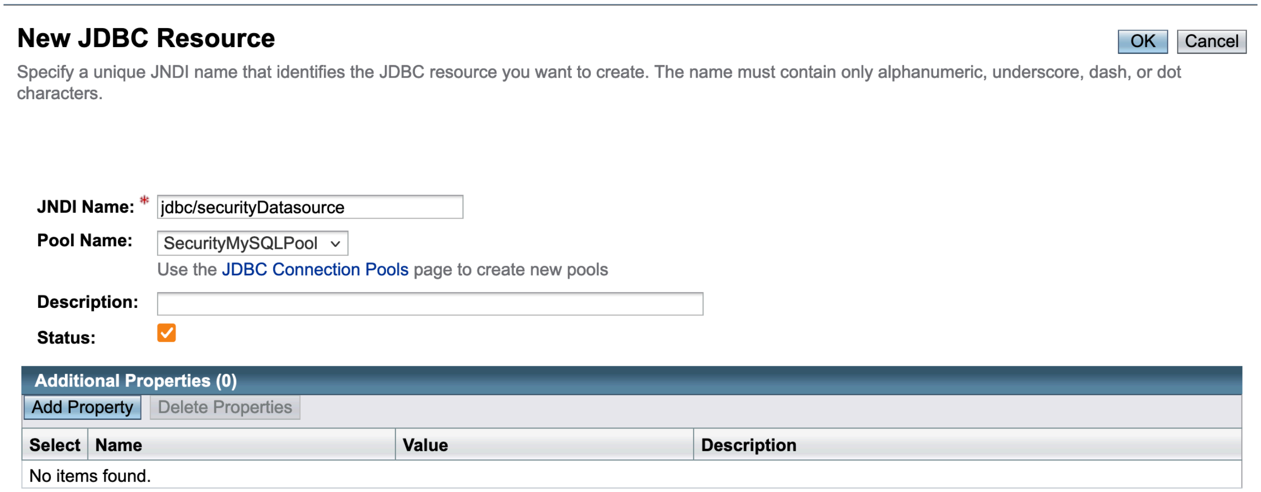
JDBC Resources

JDBC resources provide applications with a means to connect to a database.

Resources (2)

Select	JNDI Name	Logical JNDI Name
□	jdbc/__TimerPool	
□	jdbc/__default	java:comp/DefaultDataSource

Fill in the JNDI Name and set the pool to the one previously created.



New JDBC Resource

Specify a unique JNDI name that identifies the JDBC resource you want to create. The name must contain only alphanumeric, underscore, dash, or dot characters.

JNDI Name: *	jdbc/securityDatasource
Pool Name:	SecurityMySQLPool Use the JDBC Connection Pools page to create new pools
Description:	
Status:	☑

Additional Properties (0)

Select	Name	Value	Description
No items found.			

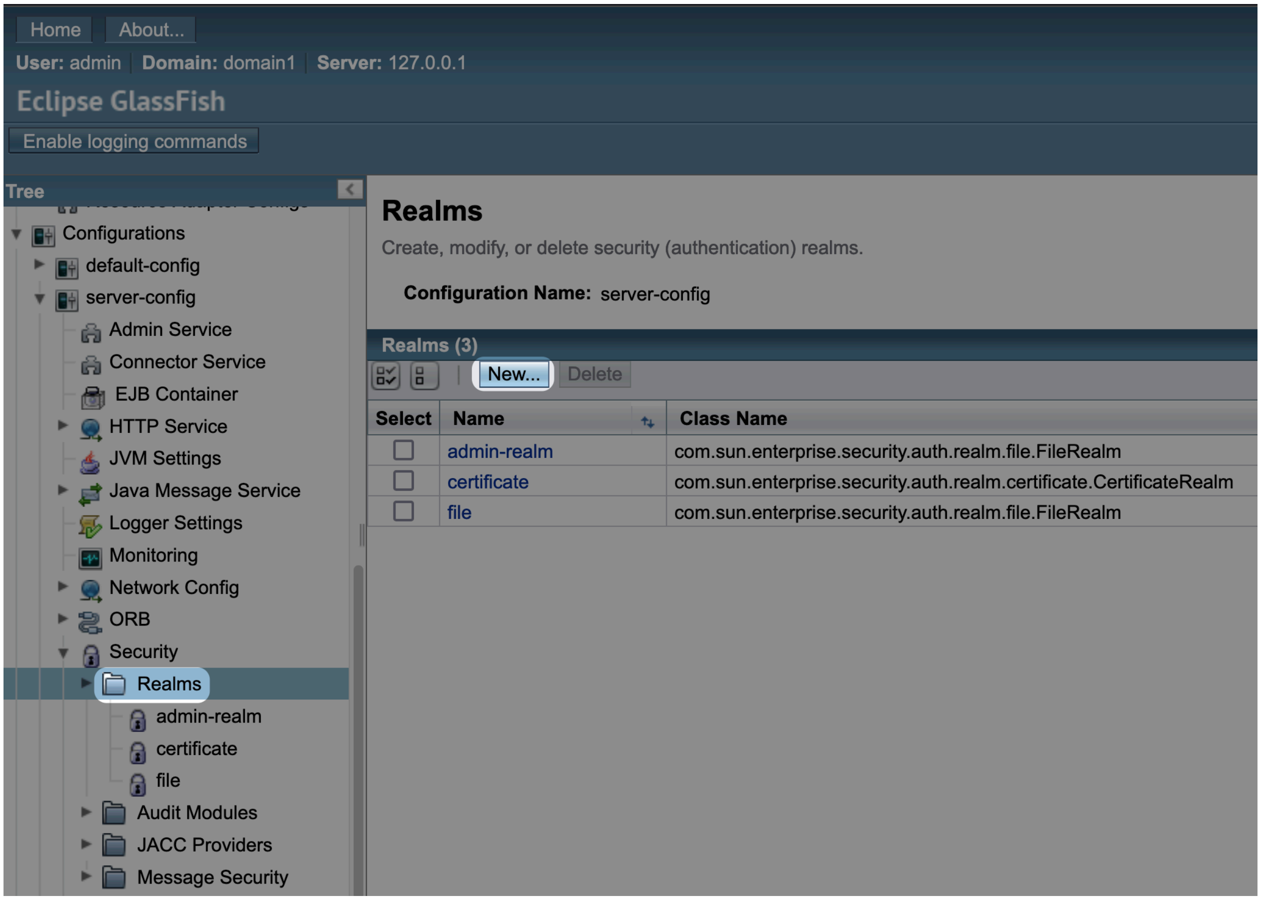
JNDI Name: `jdbc/securityDatasource`

Pool Name: `SecurityMySQLPool`

Realm

Now that the connection pool has been established and the datasource provided to Glassfish, we can create a new Realm to use for authentication.

Navigate to the Realms folder in the server-config.



Realms

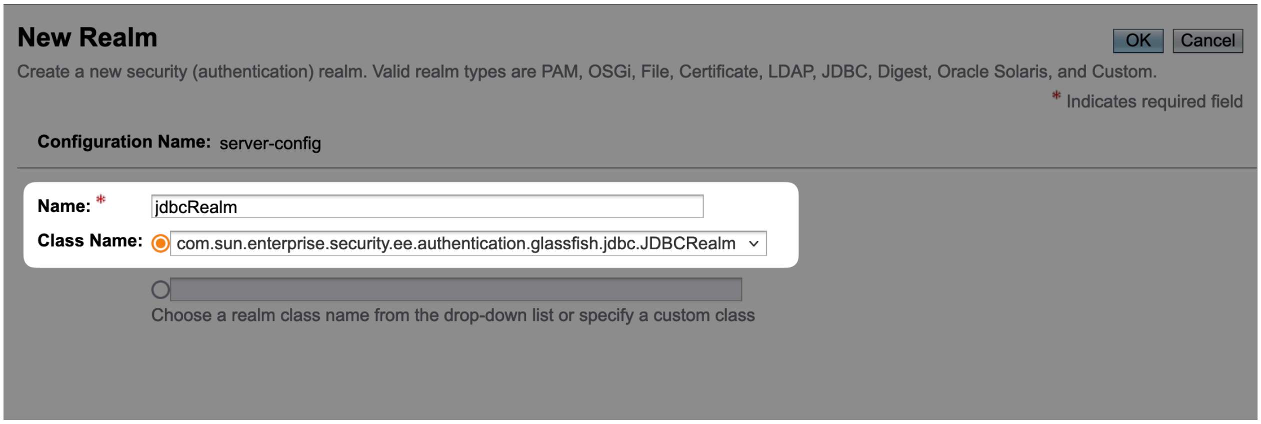
Create, modify, or delete security (authentication) realms.

Configuration Name: server-config

Realms (3)

Select	Name	Class Name
□	admin-realm	com.sun.enterprise.security.auth.realm.file.FileRealm
□	certificate	com.sun.enterprise.security.auth.realm.certificate.CertificateRealm
□	file	com.sun.enterprise.security.auth.realm.file.FileRealm

At the top, fill in the Name and Class Name.



New Realm

Create a new security (authentication) realm. Valid realm types are PAM, OSGi, File, Certificate, LDAP, JDBC, Digest, Oracle Solaris, and Custom.

* Indicates required field

Configuration Name: server-config

Name: * jdbcRealm

Class Name: ● com.sun.enterprise.security.ee.authentication.glassfish.jdbc.JDBCRealm

○

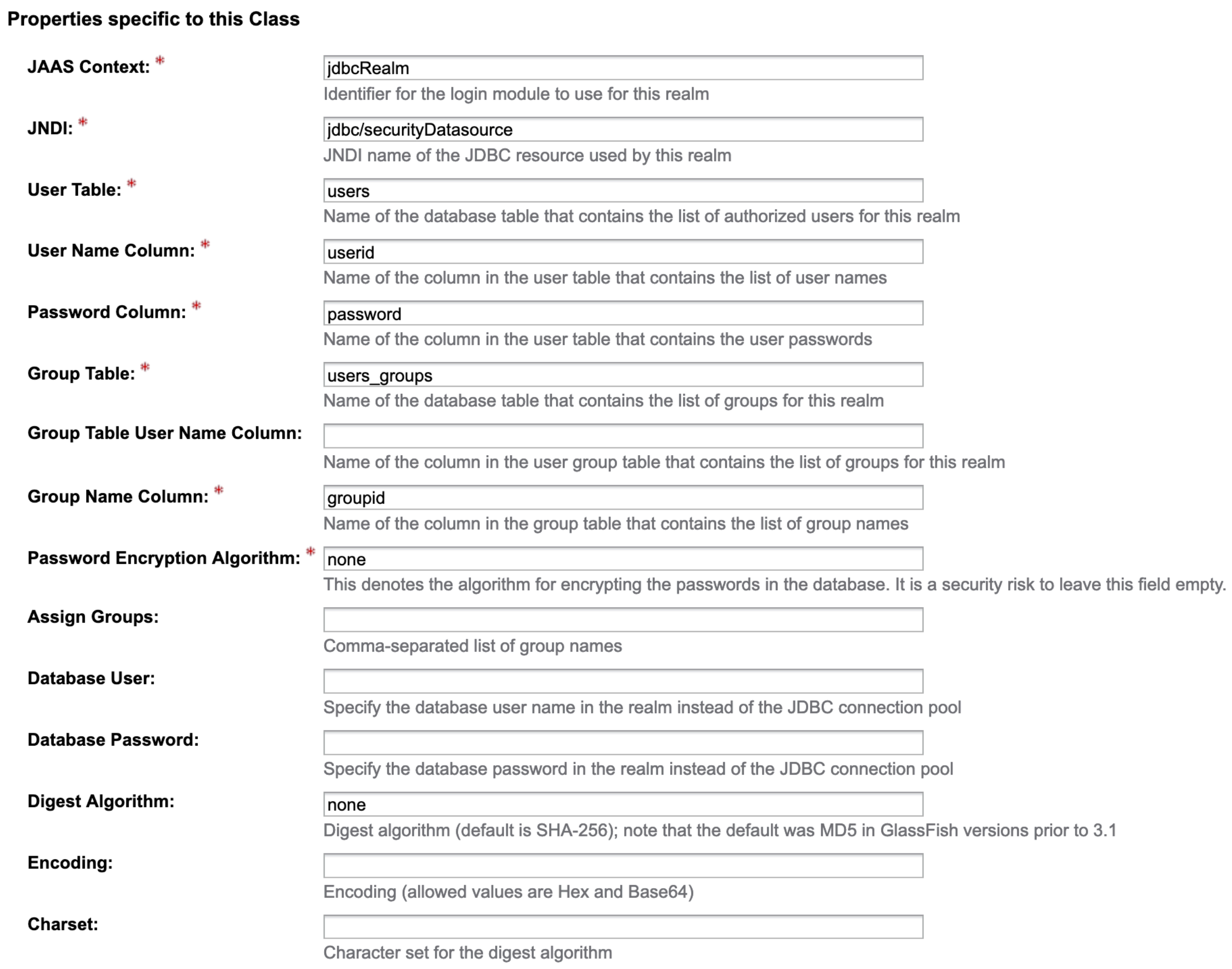
Choose a realm class name from the drop-down list or specify a custom class

Name: `jdbcRealm`

Class Name:

`com.sun.enterprise.security.ee.authentication.glassfish.jdbc.JDBCRealm`

Fill in the rest of the Realm properties.



Properties specific to this Class

JAAS Context: *	jdbcRealm	Identifier for the login module to use for this realm
JNDI: *	jdbc/securityDatasource	JNDI name of the JDBC resource used by this realm
User Table: *	users	Name of the database table that contains the list of authorized users for this realm
User Name Column: *	userid	Name of the column in the user table that contains the list of user names
Password Column: *	password	Name of the column in the user table that contains the user passwords
Group Table: *	users_groups	Name of the database table that contains the list of groups for this realm
Group Table User Name Column:		Name of the column in the user group table that contains the list of groups for this realm
Group Name Column: *	groupid	Name of the column in the group table that contains the list of group names
Password Encryption Algorithm: *	none	This denotes the algorithm for encrypting the passwords in the database. It is a security risk to leave this field empty.
Assign Groups:		Comma-separated list of group names
Database User:		Specify the database user name in the realm instead of the JDBC connection pool
Database Password:		Specify the database password in the realm instead of the JDBC connection pool
Digest Algorithm:	none	Digest algorithm (default is SHA-256); note that the default was MD5 in GlassFish versions prior to 3.1
Encoding:		Encoding (allowed values are Hex and Base64)
Charset:		Character set for the digest algorithm

JAAS Context: `jdbcRealm`

JNDI: `jdbc/securityDatasource`

User Table: `users`

User Name Column: `userid`

Password Column: `password`

Group Table: `users_groups`

Group Name Column: `groupid`

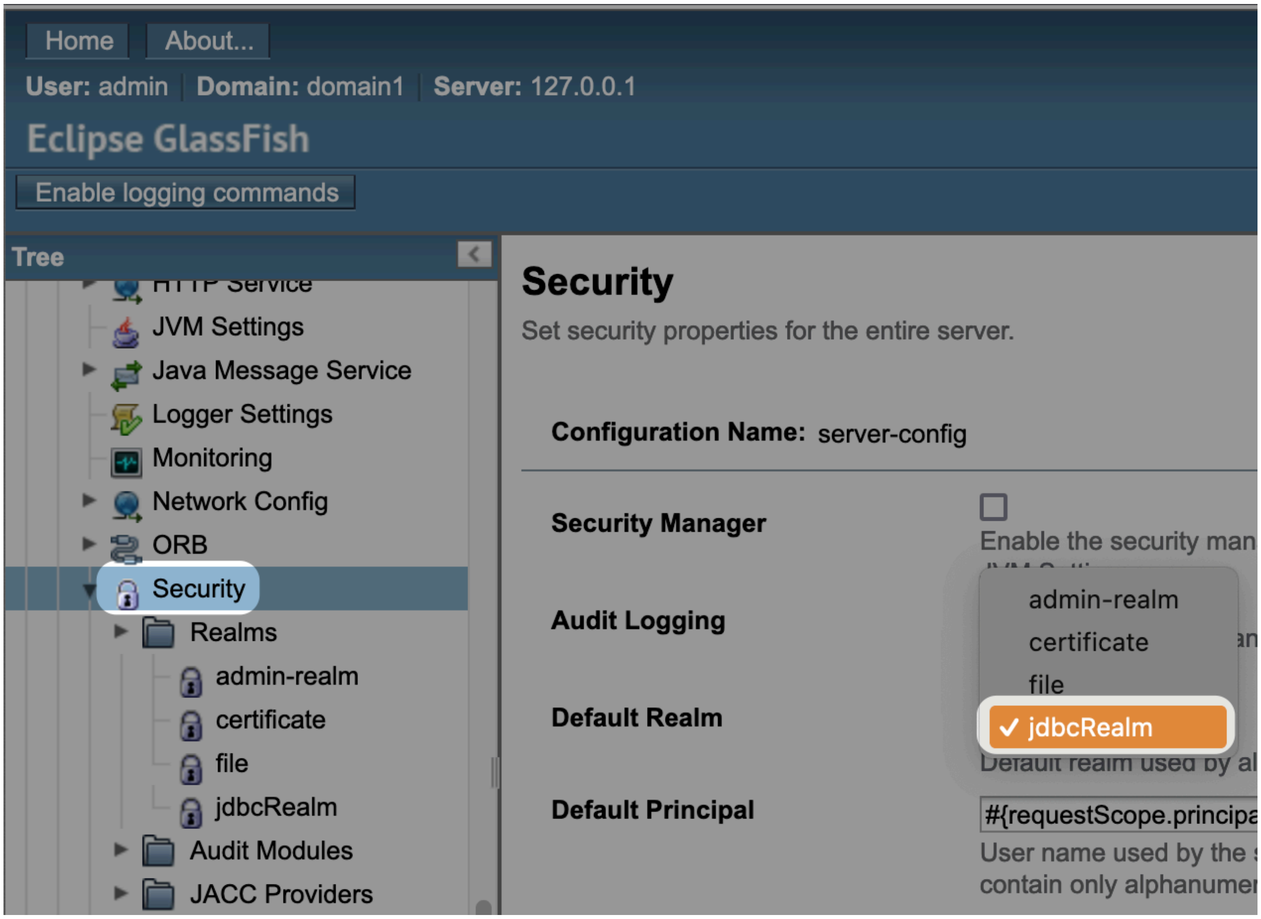
Password Encryption Algorithm: `none`

Digest Algorithm: `none`

Optional: Set to Default Realm

Normally, applications will define a level of authentication that is appropriate for them. For our use case, we can default all applications to use our new Realm.

Navigate to the Security section of the server-config and set the Default Realm to jdbcRealm, then click Save.



Security

Set security properties for the entire server.

Configuration Name: server-config

Security Manager □ Enable the security man

Audit Logging

Default Realm admin-realm / certificate / file / ✓ jdbcRealm

Default realm used by al

Default Principal #{requestScope.principa

User name used by the
contain only alphanumer

Building & Deployment

Building

This project uses Maven for build pipelines and dependency management. First, ensure Maven is installed in your environment or IDE, then run the following commands to validate and package the `.war` file for deployment.

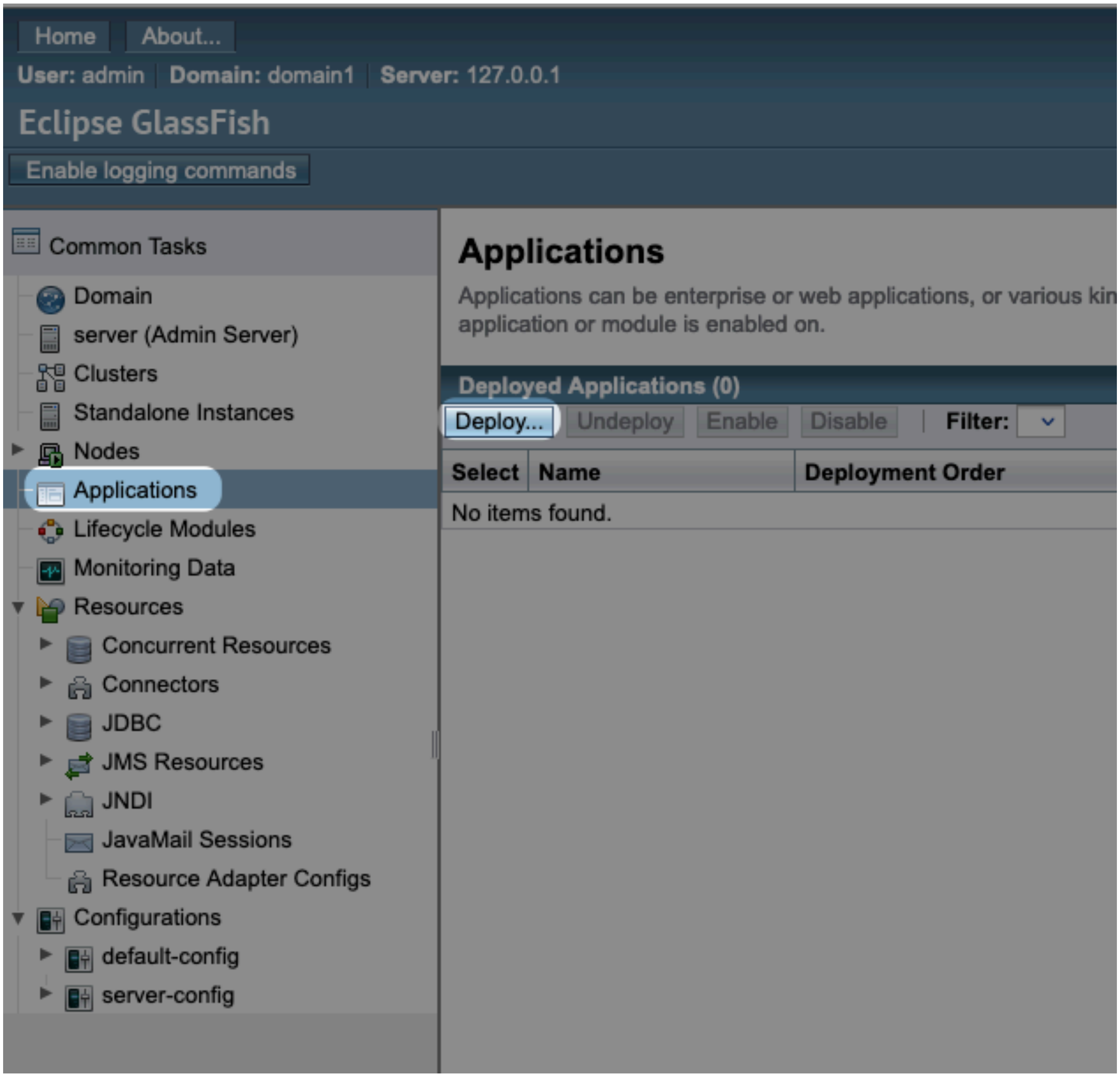
```
mvn clean

mvn validate

mvn compile

mvn package
```

When these steps are complete, the output will be located in the target folder.

```
target/csci3830-finalproject-1.0.war
```

Deployment

Deployment to Glassfish server can be done through the admin console. Navigate to Applications and Deploy a new application.



User: admin | Domain: domain1 | Server: 127.0.0.1

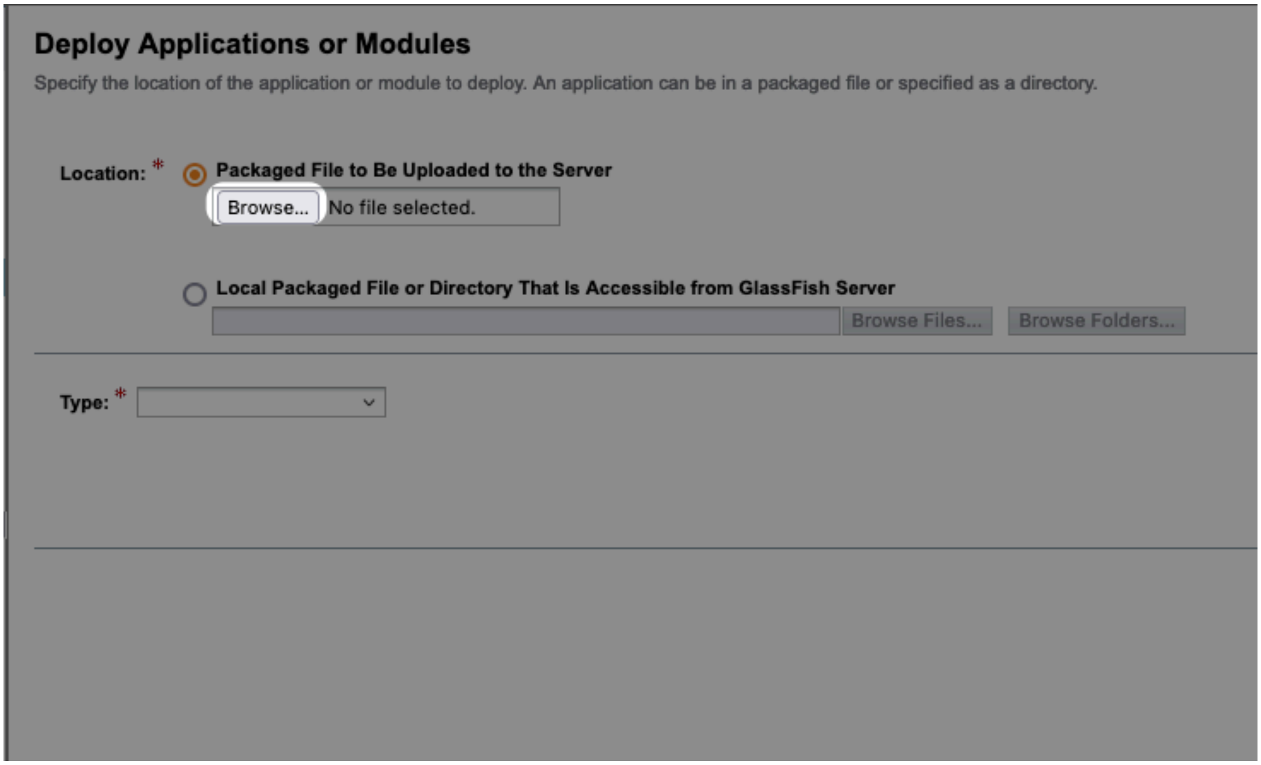
Applications

Applications can be enterprise or web applications, or various kin
application or module is enabled on.

Deployed Applications (0)

Select	Name	Deployment Order
No items found.		

Use the system file picker to choose the generated `.war`.



Deploy Applications or Modules

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

Location: * ● Packaged File to Be Uploaded to the Server

No file selected.

○ Local Packaged File or Directory That Is Accessible from GlassFish Server

Type: *

The expanded options can be safely ignored, click OK at the top right to complete the deployment.

Once the deployment is complete, the application is now available relative to the Glassfish server URL at `/csci3830-finalproject-1.0/`.