CODE:

```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>
using namespace std;

class User {
public:
    string username;
    string password;
    string role;
};

void createUser(vector<User>& users) {
    User newUser;
    cout << "Enter username: ";
    cin >> newUser.username;
    cout << "Enter password: ";
    cin >> newUser.password;
    cout << "Enter role (administrator, regular user, guest): ";
    cin >> newUser.role;
    users.push_back(newUser);
    cout << "User created successfully.\n";
}

void deleteUser(vector<User>& users) {
    string username;
    cout << "Enter username to delete: ";
```

```
    cin >> username;
    auto it = find_if(users.begin(), users.end(), [username](const User& user) {
        return user.username == username;
    });
    if (it != users.end()) {
        users.erase(it);
        cout << "User deleted successfully.\n";
    } else {
        cout << "User not found.\n";
    }
}

void changePassword(vector<User>& users) {
    string username, newPassword;
    cout << "Enter username: ";
    cin >> username;
    auto it = find_if(users.begin(), users.end(), [username](const User& user) {
        return user.username == username;
    });
    if (it != users.end()) {
        cout << "Enter new password: ";
        cin >> newPassword;
        it->password = newPassword;
        cout << "Password changed successfully.\n";
    } else {
        cout << "User not found.\n";
    }
}
```

```
void createPasswordPolicy() {
    cout << "Password policy created.\n";
}
void assignRole(vector<User>& users) {
    string username, newRole;
    cout << "Enter username: ";
    cin >> username;
    auto it = find_if(users.begin(), users.end(), [username](const User& user) {
        return user.username == username;
    });
    if (it != users.end()) {
        cout << "Enter new role: ";
        cin >> newRole;
        it->role = newRole;
        cout << "Role assigned successfully.\n";
    } else {
        cout << "User not found.\n";
    }
}

void assignPrivileges() {
    cout << "Privileges assigned.\n";
}

void checkPasswordPolicy() {
    cout << "Checking password policy.\n";
}

int main() {
```

```
vector<User> users;
char choice;
do {
    cout << "\nMenu:\n"
         << "1. Create User\n"
         << "2. Delete User\n"
         << "3. Change Password\n"
         << "4. Create Password Policy\n"
         << "5. Assign Role to User\n"
         << "6. Assign Privileges to Role\n"
         << "7. Check Password Policy\n"
         << "8. Exit\n"
         << "Enter your choice: ";
    cin >> choice;

    switch (choice) {
    case '1':
        createUser(users);
        break;
    case '2':
        deleteUser(users);
        break;
    case '3':
        changePassword(users);
        break;
    case '4':
        createPasswordPolicy();
        break;
    case '5':
```

```
            assignRole(users);
            break;
        case '6':
            assignPrivileges();
            break;
        case '7':
            checkPasswordPolicy();
            break;
        case '8':
            cout << "Exiting...\n";
            break;
        default:
            cout << "Invalid choice. Please try again.\n";
        }
    } while (choice != '8');
    return 0;
}
```

OUTPUT:

```
Menu:
1. Create User
2. Delete User
3. Change Password
4. Create Password Policy
5. Assign Role to User
6. Assign Privileges to Role
7. Check Password Policy
8. Exit
Enter your choice: 1
```

Enter username: Jay

Enter password: jay1234

Enter role (administrator, regular user, guest): administrator

User created successfully.

Menu:

1. Create User
2. Delete User
3. Change Password
4. Create Password Policy
5. Assign Role to User
6. Assign Privileges to Role
7. Check Password Policy
8. Exit

Enter your choice: 1

Enter username: Madhavi

Enter password: pass1234

Enter role (administrator, regular user, guest): guest

User created successfully.

Menu:

1. Create User
2. Delete User
3. Change Password
4. Create Password Policy
5. Assign Role to User
6. Assign Privileges to Role
7. Check Password Policy
8. Exit

Enter your choice: 2

Enter username to delete: Madhavi

User deleted successfully.

Menu:

1. Create User

2. Delete User

3. Change Password

4. Create Password Policy

5. Assign Role to User

6. Assign Privileges to Role

7. Check Password Policy

8. Exit

Enter your choice: 7

Checking password policy.

Menu:

1. Create User

2. Delete User

3. Change Password

4. Create Password Policy

5. Assign Role to User

6. Assign Privileges to Role

7. Check Password Policy

8. Exit

Enter your choice: 8

Exiting...