Wir haben beschlossen, ein Projekt zum Thema Bankensystem zu machen. Unsere Benutzer können sich registrieren, sich dann in ihr Konto einloggen und einen bestimmten Betrag an andere Benutzer überweisen oder umgekehrt Geld erhalten.

Unser Projekt wurde mit:

Spring Boot Starter Parent; PostgreSQL JDBC Driver; Spring Boot Starter Data JPA; Spring Boot Starter Thymeleaf; Spring Boot Starter Web; Thymeleaf Extras Springsecurity6; Spring Boot Starter Test

wir haben HTML erstellt, damit Sie unser Projekt bequem im Browser anzeigen können.

Es gibt mehrere Grundlagen auf unserer Website.

/home – start seite

/hello

/register - verantwortlich für die Registrierung des Benutzers

/login – verantwortlich für die Benutzeranmeldung

/transfer –Geld Überweisung

**Authority java**

1. if (this == o) return true;: Auf Objektidentität prüfen. Wenn die Objektreferenzen identisch sind (dh sich auf dasselbe Objekt beziehen), gibt die Methode sofort true zurück.

2. wenn (o == null |/ getClass() != o.getClass ()) return false;: Auf null prüfen und die Klassen von Objekten vergleichen. Wenn das übergebene Objekt o gleich null ist oder zu einer anderen Klasse als das aktuelle Objekt gehört, gibt die Methode false zurück.

Objekte zurückgeben.ist gleich(id, authority.id ) && Objekte.entspricht (Benutzername, Autorität.benutzername) && Objekte.entspricht (Benutzername, Autorität.autorität);: Vergleich von Objektfeldern. Die Methode gibt das Ergebnis des logischen UND (&&) zum Vergleichen jedes Felds des aktuellen Objekts (this) mit dem entsprechenden Feld des Berechtigungsobjekts zurück. Gegenst.equals wird für einen sicheren Vergleich unter Berücksichtigung von Nullwerten verwendet.

**Login controller**

Zusammenfassend verarbeitet diese Methode eine Anmeldeanforderung, versucht, den angegebenen Benutzernamen und das Kennwort mithilfe eines Dienstes (CustomerService) zu authentifizieren, und leitet den Benutzer bei Erfolg auf die Startseite um. Wenn die Authentifizierung fehlschlägt, fügt sie dem Modell ein Fehlermeldungsattribut hinzu und kehrt für einen weiteren Versuch zur Anmeldeseite zurück.

**Registration controller**

Zusammenfassend behandelt diese Methode die Registrierung eines neuen Kunden. Es empfängt ein Kundenobjekt als Parameter, registriert es über einen Dienst (Kundendienst) und leitet den Benutzer dann auf die Startseite weiter. Die Besonderheiten des Registrierungsprozesses und der Servicelogik würden in der registerNewCustomer Methode des CustomerService implementiert.

**Customer service**

1. Zusammenfassend zieht diese Methode einen bestimmten Betrag vom Kontostand eines Kunden ab und führt die erforderlichen Überprüfungen auf die Gültigkeit der Konto-ID und die ausreichende Deckung durch. Bei Erfolg aktualisiert es den Kontostand des Kunden im Repository.

2. Zusammenfassend fügt diese Methode dem Kontostand eines Kunden einen bestimmten Betrag hinzu und führt die erforderlichen Überprüfungen auf die Gültigkeit der Konto-ID durch. Bei Erfolg aktualisiert es den Kontostand des Kunden im Repository.

Übersicht der Endpunkte

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| URL | Methode | Status | Params | Description |
| /login | GET | 200 | - | so erhalten Sie eine HTML-Seite für die Anmeldung in Ihrem Konto |
| /login | POST | 200 | Username: String  Password: String | Sendet Daten und leitet Sie an /home weiter |
| /login | POST | 401 | Username: String  Password: String | wird zurückgegeben, wenn der Benutzer falsche Daten eingegeben hat |
| /register | GET | 200 | - | so erhalten Sie eine HTML-Seite für die Registrierung in Ihrem Konto |
| /register | POST | 200 | Username: String  Password: String  Name: String  Age: Integer  Email: String | Sendet Daten und leitet Sie an /home weiter |
| /register | POST | 401 | Username: String  Password: String  Name: String  Age: Integer  Email: String | wird zurückgegeben, wenn der Benutzer falsche Daten eingegeben hat |
| /home | GET | 200 | - | Rückkehr zur HTML-Seite, auf der Sie sich registrieren oder sich anmelden können |
| /transfer | POST | 200 | Fromaccountid: integer  Toaccountid: integer  Amount: integer | Überweisung von Geld von einem Konto auf ein anderes |
| /transfer | POST | 400 | Fromaccountid: integer  Toaccountid: integer  Amount: integer | Unzureichende Mittel |
| /customer | GET | 200 | - | Gibt eine Liste von Benutzern zurück |