SpringBoot Adding Spring Security Login, Adding - override loadUserByUsername

```
public class AccountService implements UserDetailsService
{
}
```

Implemented Methods - UserDetailsService

```
@Override
    public UserDetails loadUserByUsername(String username) throws
UsernameNotFoundException {
        // TODO Auto-generated method stub
        throw new UnsupportedOperationException("Unimplemented method
'loadUserByUsername'");
    }
```

AccountService.java Updated

```
package org.studyeasy.SpringStarter.services;

import java.util.ArrayList;
import java.util.Optional;

import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.User;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;
import org.studyeasy.SpringStarter.models.Account;
import org.studyeasy.SpringStarter.repositories.AccountRepository;

@Service
public class AccountService implements UserDetailsService{
    @Autowired
    private AccountRepository accountRepository;
    @Autowired
    private PasswordEncoder passwordEncoder;
```

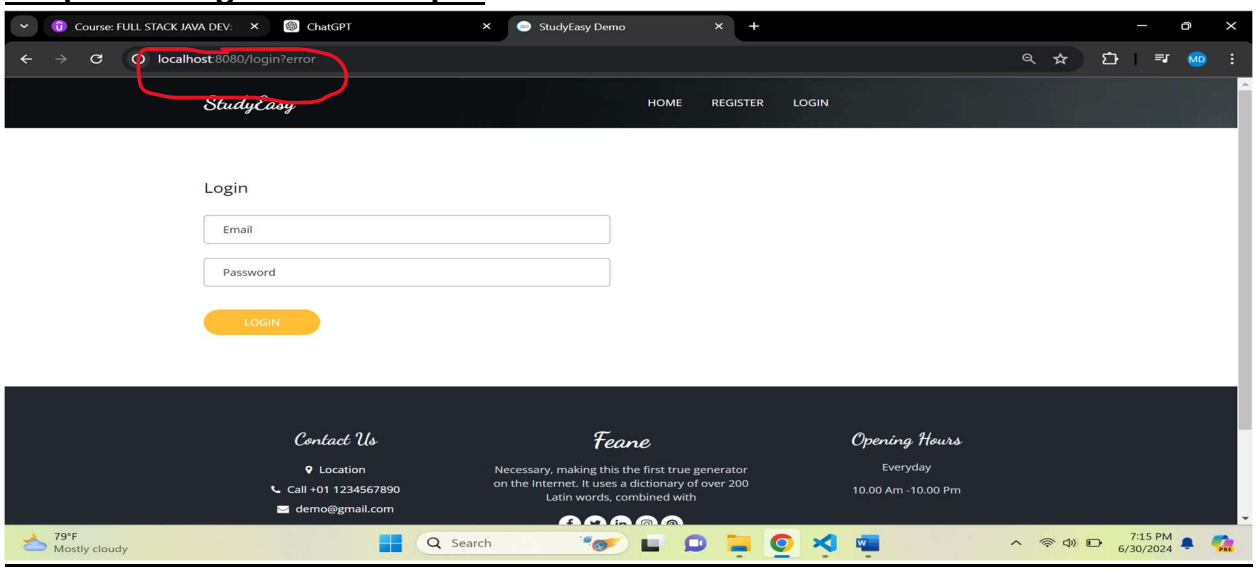
```
    public Account save(Account account){
        account.setPassword(passwordEncoder.encode(account.getPassword()));
        return accountRepository.save(account);
    }
 @Override
    public UserDetails loadUserByUsername(String email) throws UsernameNotFoundException {
        Optional<Account> optionalAccount = accountRepository.findOneByEmailIgnoreCase(email);
        if(!optionalAccount.isPresent()){
            throw new UsernameNotFoundException("Account not found!..");
        }
        Account account = optionalAccount.get();
        return new User(account.getEmail(),account.getPassword(),null);
    }
}
```

AccountRepository.java

```
@Repository
public interface AccountRepository extends JpaRepository<Account, Long>{

    @Query("SELECT a FROM Account a WHERE LOWER(a.email) = LOWER(:email)")
    Optional<Account> findOneByEmailIgnoreCase(@Param("email") String email);
}
```

Output – You get below output



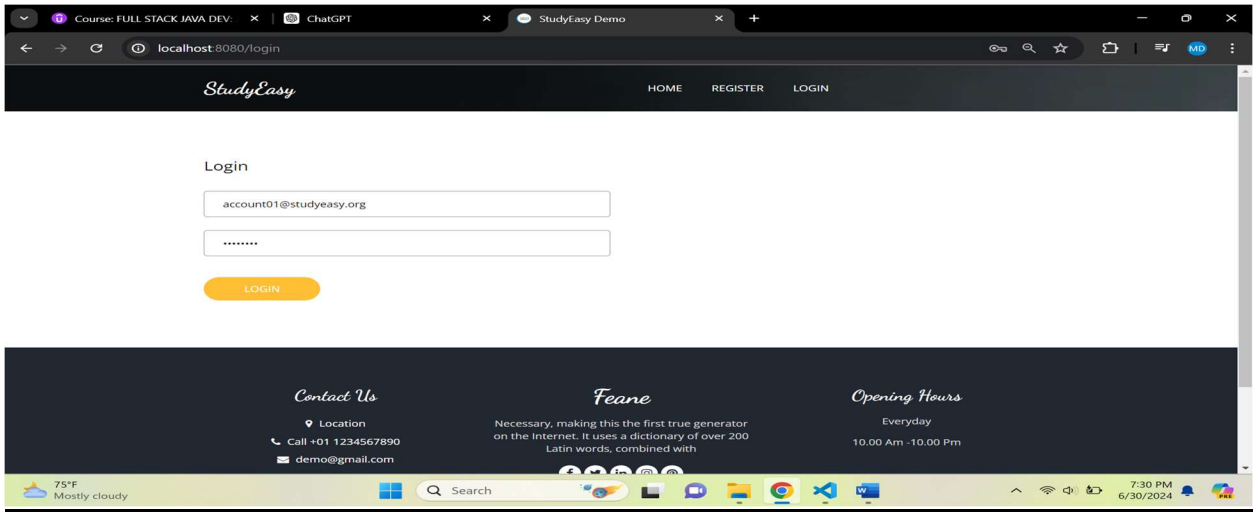
Solution:-

Updated loadUserByUsername()

```
@Override
    public UserDetails loadUserByUsername(String email) throws UsernameNotFoundException {
        Optional<Account> optionalAccount = accountRepository.findOneByEmailIgnoreCase(email);
        if(!optionalAccount.isPresent()){
            throw new UsernameNotFoundException("Account not found!..");
        }
        Account account = optionalAccount.get();

        List<GrantedAuthority> grantedAuthority = new ArrayList<>();
        grantedAuthority.add(new SimpleGrantedAuthority("Allow"));

        return new User(account.getEmail(),account.getPassword(),grantedAuthority);
    }
```



After Logging in, the default page will be opened.