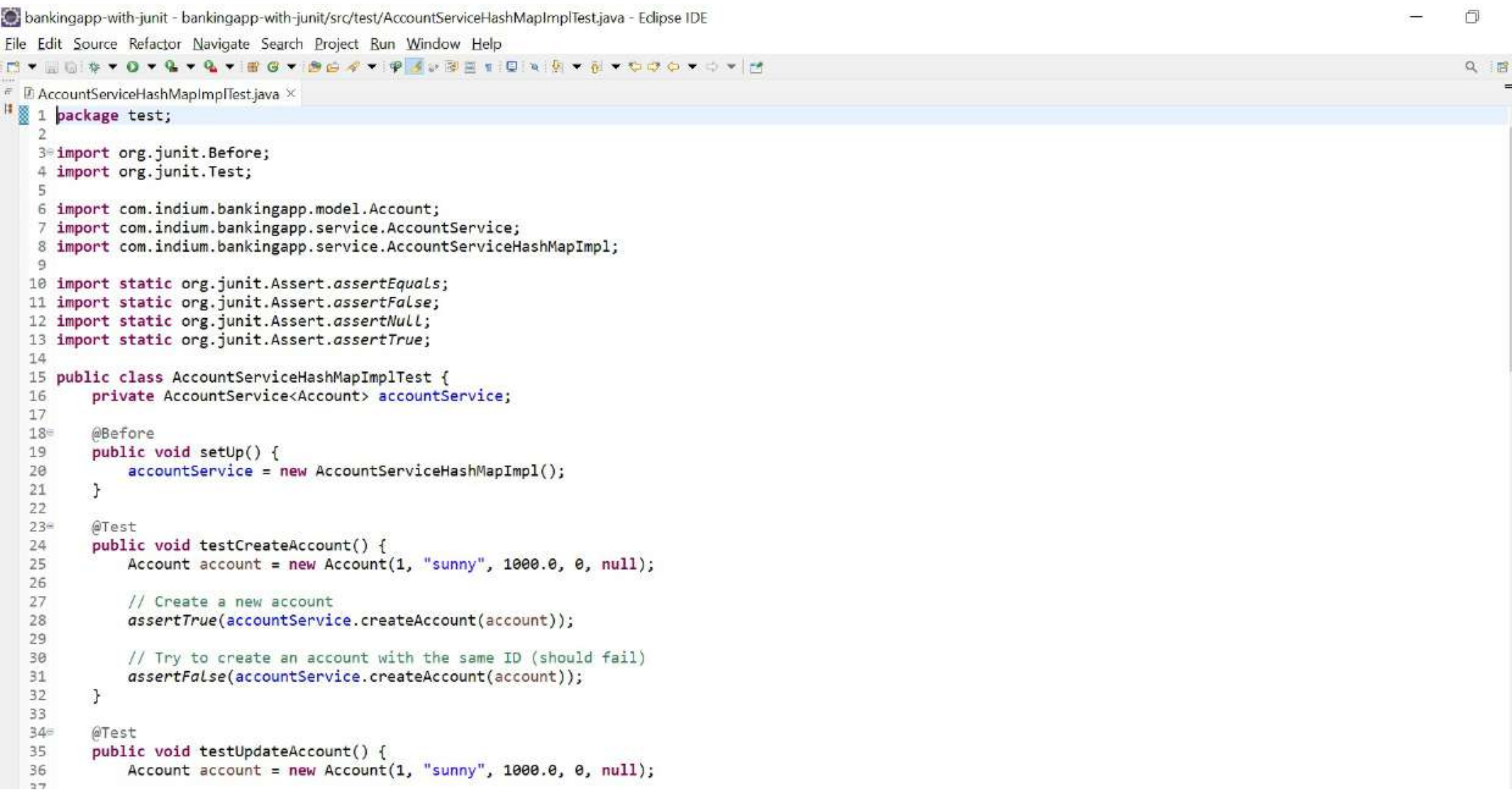
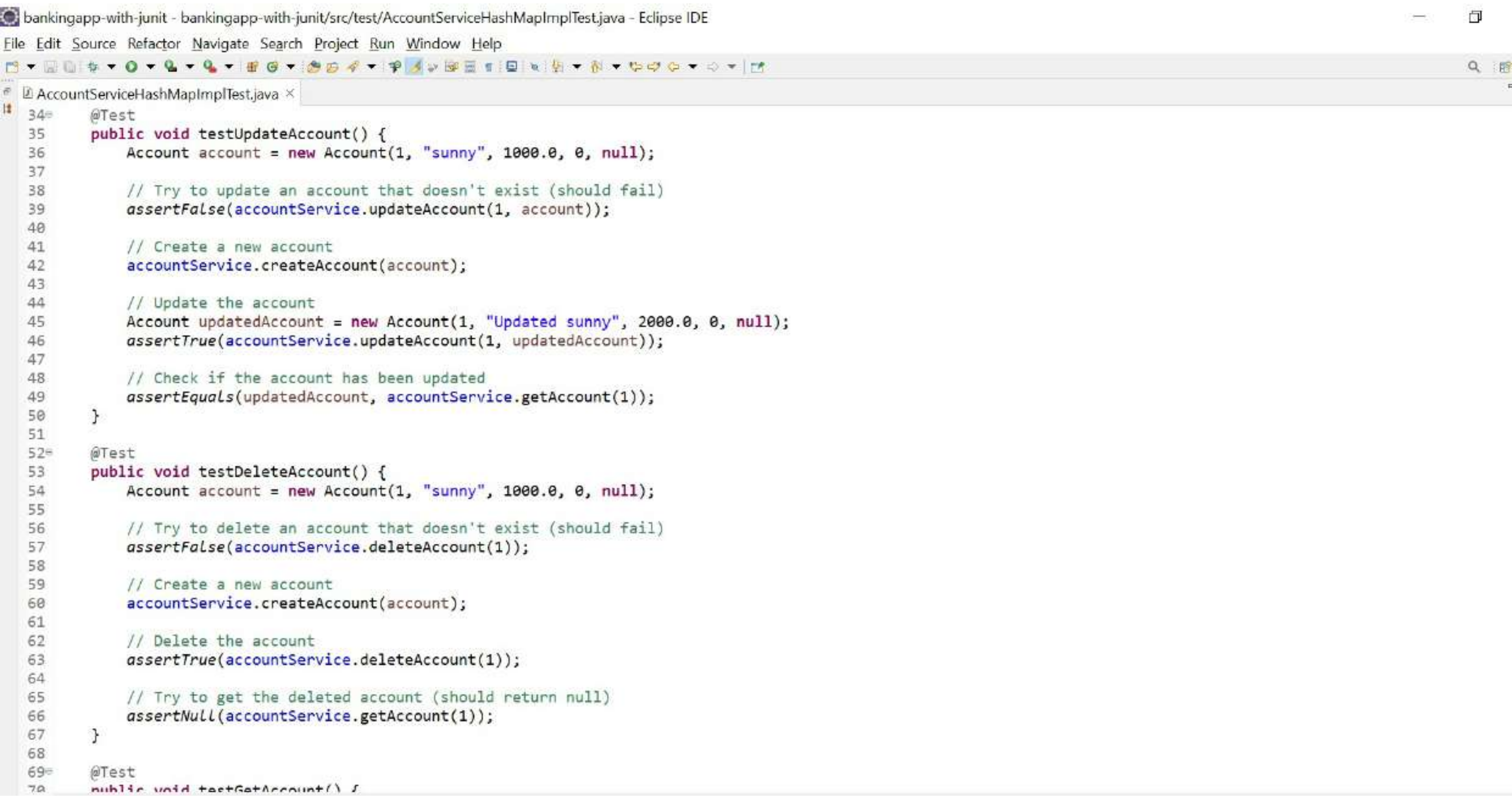
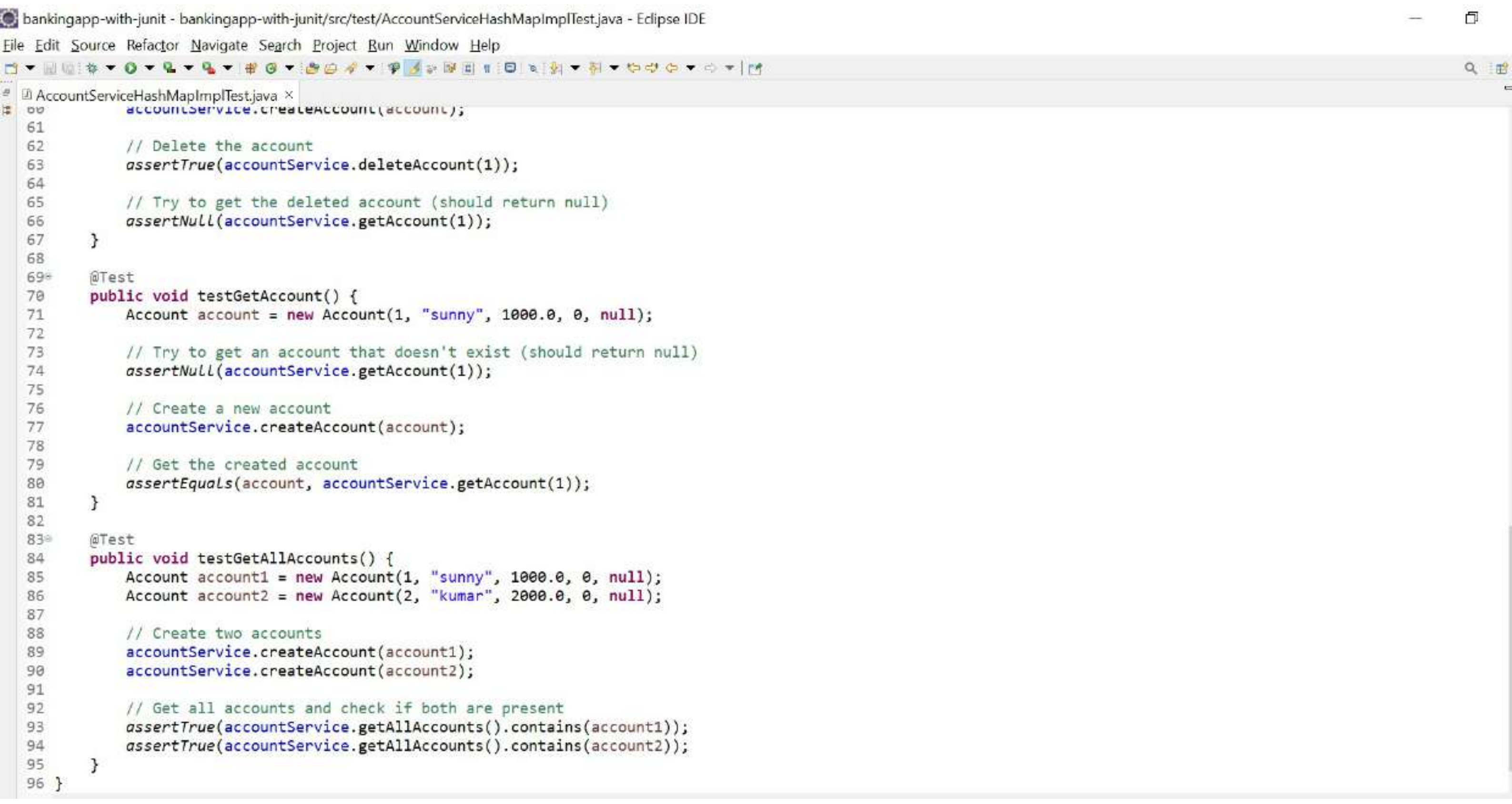

bankingapp-with-junit - bankingapp-with-junit/src/test/AccountServiceHashMapImplTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
AccountServiceHashMapImplTest.java

```java
package test;

import org.junit.Before;
import org.junit.Test;

import com.indium.bankingapp.model.Account;
import com.indium.bankingapp.service.AccountService;
import com.indium.bankingapp.service.AccountServiceHashMapImpl;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertFalse;
import static org.junit.Assert.assertNull;
import static org.junit.Assert.assertTrue;

public class AccountServiceHashMapImplTest {
    private AccountService<Account> accountService;

    @Before
    public void setUp() {
        accountService = new AccountServiceHashMapImpl();
    }

    @Test
    public void testCreateAccount() {
        Account account = new Account(1, "sunny", 1000.0, 0, null);

        // Create a new account
        assertTrue(accountService.createAccount(account));

        // Try to create an account with the same ID (should fail)
        assertFalse(accountService.createAccount(account));
    }

    @Test
    public void testUpdateAccount() {
        Account account = new Account(1, "sunny", 1000.0, 0, null);
```


bankingapp-with-junit - bankingapp-with-junit/src/test/AccountServiceHashMapImplTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
AccountServiceHashMapImplTest.java

```java
    @Test
    public void testUpdateAccount() {
        Account account = new Account(1, "sunny", 1000.0, 0, null);

        // Try to update an account that doesn't exist (should fail)
        assertFalse(accountService.updateAccount(1, account));

        // Create a new account
        accountService.createAccount(account);

        // Update the account
        Account updatedAccount = new Account(1, "Updated sunny", 2000.0, 0, null);
        assertTrue(accountService.updateAccount(1, updatedAccount));

        // Check if the account has been updated
        assertEquals(updatedAccount, accountService.getAccount(1));
    }

    @Test
    public void testDeleteAccount() {
        Account account = new Account(1, "sunny", 1000.0, 0, null);

        // Try to delete an account that doesn't exist (should fail)
        assertFalse(accountService.deleteAccount(1));

        // Create a new account
        accountService.createAccount(account);

        // Delete the account
        assertTrue(accountService.deleteAccount(1));

        // Try to get the deleted account (should return null)
        assertNull(accountService.getAccount(1));
    }

    @Test
    public void testGetAccount() {
```


bankingapp-with-junit - bankingapp-with-junit/src/test/AccountServiceHashMapImplTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
AccountServiceHashMapImplTest.java

```java
        accountService.createAccount(account);

        // Delete the account
        assertTrue(accountService.deleteAccount(1));

        // Try to get the deleted account (should return null)
        assertNull(accountService.getAccount(1));
    }

    @Test
    public void testGetAccount() {
        Account account = new Account(1, "sunny", 1000.0, 0, null);

        // Try to get an account that doesn't exist (should return null)
        assertNull(accountService.getAccount(1));

        // Create a new account
        accountService.createAccount(account);

        // Get the created account
        assertEquals(account, accountService.getAccount(1));
    }

    @Test
    public void testGetAllAccounts() {
        Account account1 = new Account(1, "sunny", 1000.0, 0, null);
        Account account2 = new Account(2, "kumar", 2000.0, 0, null);

        // Create two accounts
        accountService.createAccount(account1);
        accountService.createAccount(account2);

        // Get all accounts and check if both are present
        assertTrue(accountService.getAllAccounts().contains(account1));
        assertTrue(accountService.getAllAccounts().contains(account2));
    }
}
```