Report on Testing

# The following databases were created for testing:

**Bank Details**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Bank Name** | **State** | **Branch** | **IFSC Code** | **Note:** There are many branches of a bank in same state, they have different IFSC code and address. |
| SBI | Punjab | Ropar | 101 |
| SBI | Punjab | Ropar | 102 |
| SBI | Punjab | Ropar | 201 |
| SBI | Punjab | Ropar | 202 |
| SBI | Punjab | Ropar | 801 |
| SBI | Punjab | Ropar | 802 |
| SBI | Punjab | Ropar | 901 |

**User Identities:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **User Name** | **Password** | **Mobile no** | **User ID** | **Note:** These are the two test users created along with their identities mentioned. |
| test\_user1 | test\_passwd | 7814681937 | 1001 |
| test\_user2 | test\_passwd | 9646318665 | 2001 |

**User Accounts:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User ID** | **Account No** | **Bank IFSC code** | **A/C Type** | **Balance** | **Transaction Limit** | **No of Transaction** |
| 1001 | 10001 | 101 | 1 | 30000 | 10000 | 3 |
| 1001 | 10002 | 102 | 2 | 20000 | 10000 | 3 |
| 2001 | 20001 | 201 | 1 | 40000 | 10000 | 3 |
| 2001 | 20002 | 202 | 2 | 60000 | 10000 | 3 |

**Connected Accounts Table for third party transfer**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **C\_Host A/c ID** | **C\_Account No** | **Date of Addition** | **C\_user name** | **C\_Transaction Limit** | **C\_IFSC Code** |
| 1001 | 20001 | Current time | Connected\_1 | 10000 | 201 |
| 2001 | 10001 | Current time | Connected\_2 | 10000 | 101 |

**Connected Accounts Table for Inter-Bank Transfer**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **I\_Host A/c** | **I\_Account\_no** | **Date of Addition** | **I\_username** | **I\_Transaction**  **Limit** | **I\_IFSC code** | **Address** |
| 1001 | 90001 | CurrentTime | Connected\_1 | 10000 | 801 | a-b-c |
| 2001 | 90002 | CurrentTime | Connected\_2 | 10000 | 802 | a-b-c- |

**The following Test were stimuted on the model and view:**

**1.**

def test\_NavigationHome(self):

c=Client()

response = c.post('/Online\_transactions/')

self.assertEqual(response.status\_code, 200)

**This checks weather all pages are navigable or not.**

**Result: Passed**

**2.** def test\_simple\_funds\_transfer(self):

**It checks the fund transfer functionality by calling**

c=Client()

response = c.post('/funds\_transfer',{'account1':'10001','account2':'10002','amount\_to\_transfer':5000})

**and asserting the response code.**

self.assertEqual(response.status\_code,301)

**Result: Passed**

3. def test\_Thirdparty\_transfer(self):

**It checks the third party transfer functionality by calling**

response = c.post('/home\_page',{'user\_id':'test\_user1', 'passwd':'test\_passwd'})

**and asserting the response code.**

self.assertEqual(response.status\_code, 301)

response=c.post('/third\_party\_transfer/')

response = c.post('/third\_party\_transfer/',{'user\_id':'1001','account1':'10001', 'account2':'20001','amount\_to\_transfer':5000})

account = Bank\_Account.objects.get(ba\_acc\_no="10001")

self.assertEqual(account.ba\_acc\_bal,Decimal(25000))

**Result: Passed**

4. def test\_Thirdparty\_transfer\_transaction\_limit\_exceed(self):

**It checks whether the third party transaction limit has been exceeded on not by calling**

response = c.post('/home\_page',{'user\_id':'test\_user1', 'passwd':'test\_passwd'})

**and asserting the response code** .

self.assertEqual(response.status\_code, 301)

response=c.post('/third\_party\_transfer/')

response = c.post('/third\_party\_transfer/',{'user\_id':'1001','account1':'10001', 'account2':'20001','amount\_to\_transfer':15000})

self.assertEqual(response.status\_code,200)

**Result: Passed**

5. def test\_interbank\_transfer(self):

**Check the functionality of interbank transfer by calling post and asserting response code**

c = Client()

response = c.post('/home\_page',{'user\_id':'test\_user1', 'passwd':'test\_passwd'})

self.assertEqual(response.status\_code, 301)

response=c.post('/interbank\_transfer2/')

response = c.post('/interbank\_transfer/',{'user\_id':'1001','account1':'10002', 'account2':'9001','amount\_to\_transfer':5000})

account = Bank\_Account.objects.get(ba\_acc\_no="10002")

self.assertEqual(account.ba\_acc\_bal,Decimal(15000))

**Result: Passed**

**6.** def test\_interbank\_transfer\_limit\_exceed(self):

**Check the interbank transafer party when the transaction limit is breached**

c = Client()

response = c.post('/home\_page',{'user\_id':'test\_user1', 'passwd':'test\_passwd'})

self.assertEqual(response.status\_code, 301)

response=c.post('/interbank\_transfer2/')

response = c.post('/interbank\_transfer/',{'user\_id':'1001','account1':'10002', 'account2':'9001','amount\_to\_transfer':15000})

self.assertEqual(response.context['error'],"You entered amount more than connected accounts transaction limit")

**Result: Passed**

**7.**  **def test\_interbank\_transfer\_amount\_numeric(self):**

**Check whether the amount to tranferred is numeric or not**

c = Client()

response = c.post('/home\_page',{'user\_id':'test\_user1', 'passwd':'test\_passwd'})

self.assertEqual(response.status\_code, 301)

response=c.post('/interbank\_transfer2/')

response = c.post('/interbank\_transfer/',{'user\_id':'1001','account1':'10002', 'account2':'9001','amount\_to\_transfer':5000.3453})

self.assertEqual(response.context['error'],"Please enter amount in numeric only")

**Result: Passed**

**8.**  def test\_interbank\_transfer\_amount\_validation(self):

**Check whether the amount is valid or not**

c = Client()

response = c.post('/home\_page',{'user\_id':'test\_user1', 'passwd':'test\_passwd'})

self.assertEqual(response.status\_code, 301)

response=c.post('/interbank\_transfer2/')

response = c.post('/interbank\_transfer/',{'user\_id':'1001','account1':'10002', 'account2':'9001','amount\_to\_transfer':-5000.78})

self.assertEqual(response.context['error'],"Please enter valid amount")

**Result: Passed**

**9.** def test\_add\_third\_party(self):

**Test case for checking Addtion of a third party**

c = Client()

response = c.post('/home\_page',{'user\_id':'test\_user1', 'passwd':'test\_passwd'})

self.assertEqual(response.status\_code, 301)

response=c.post('/third\_party\_transfer/')

response = c.post('/add\_third\_party/',{'name':'test\_user2','account\_no':'20002', 'account\_no\_2':'20002','limit':15000,})

account = Connected\_Account.objects.filter(ca\_connected\_acc\_no="20002")

self.assertEqual(account.name,'test\_user2')

**Result: Passed**

**10.**  def test\_add\_interbank\_party(self):

**Test case for adding a interbank party**

c = Client()

response = c.post('/home\_page',{'user\_id':'test\_user1', 'passwd':'test\_passwd'})

self.assertEqual(response.status\_code, 301)

response=c.post('/interbank\_transfer2/')

response = c.post('/add\_other\_bank\_account/',{'name':'interbank\_beneficiery\_1','account\_no':'50001','line1':'flat no 7','line2':'J 8/1 Paras','line3':'delhi', 'account\_no\_2':'50001','limit':15000,'IFSC':'901'})

account = Connected\_Account.objects.filter(ca\_connected\_acc\_no="50001")

self.assertEqual(account.name,'interbank\_beneficiery\_1')

**Result: Passed**