Bank System

Validation Document

Author	**Linshu Yang, Yiming Chen, and Ziqi Wang**
Group	**Team 5**
Date	**2022/6/5**

Catalog

1. Unit Test

This section provides information of unit tests we made for every function with statement coverage, branch coverage and condition coverage criteria. Testing cases with runnable test functions are provided in every test, you can find in corresponding files. To test those private methods in the classes, we changed the original code of the private methods and properties into comments and leave public copies un-commented in the source code.

1.1 Backend Access Test

1.1.1 Verify User

```
function [res, err] = verifyUser(obj, database)
    % check if user exists.

    % Parameters
    % ----------
    % database: instance of Database.

    % Test function:
    % tests/unitTestBackend/testAccess.m/testVerifyUser

    [user, err] = database.get("user", obj.userId);
    res = "nil";
    if err == "nil" % Branch Tcover1.1.1.1
        if isempty(user) % Branch Tcover1.1.1.2
            res = "用户不存在或银行卡无效.";
        elseif ~user{4} % Branch Tcover1.1.1.3
            res = "用户已被冻结.";
        else % Branch Tcover1.1.1.4
            res = "nil";
        end
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testAccess.m/testVerifyUser
- Test Case

	Test Case T1.1.1.1	Test Case T1.1.1.2	Test Case T1.1.1.3
Input	-----	-----	-----
Coverage Item	Tcover1.1.1.1	Tcover1.1.1.2	Tcover1.1.1.3

State	obj=GetData("123456");	obj=GetData("4173516266740291");	obj=GetData("2124099443931732");
Expected Output	res == “用户不存在或银行卡无效“	res == “用户已被冻结“	res == “nil”
Test Result	Passed	Passed	Passed

- Test Coverage: 3/3 = 100%

1.1.2 Get Data

```
function [res, err] = getData(obj, database)
    % Get data from the database.

    % Parameters
    % ----------
    % database: instance of Database.

    % Test function:
    % tests/unitTestBackend/testAccess.m/testGetData

     % Statement Tcover1.1.2.1
    [res, err] = database.get("bill", obj.userId, "time");
end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestBackend/testAccess.m/testGetData
- Test Case

	Test Case T1.1.2.1
Input	-----
Coverage Item	Tcover1.1.2.1
State	handler = GetData("2124099443931732");
Expected Output	res == “nil”; err == “nil”;
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.1.3 Update Data

```
function err = updateData(obj, database, action, content)
    % Updata data in the database.

    % Parameters
    % ----------
```

```
% database: instance of Database.
% action: string, action type of this update.
% content: instance of containers.Map, data used in this
%   update.

% Test function:
% tests/unitTestBackend/testAccess.m/testUpdateData

err = "nil";
if action == "withdraw" % Branch Tcover1.1.3.1
    value = content("value");
    bill = database.get("bill", obj.userId, "time");
    savings = bill{end, 4};
    if value > savings % Branch Tcover1.1.3.2
        err = "存款余额不足.";
    else % Branch Tcover1.1.3.3
        query = obj.userId + ...
            ", 'withdraw:" + obj.userId + "', " + value + ...
            ", " + (savings - value) + ...
            ", '" + datestr(now, 31) + "'";
        err = database.insert("bill", query);
    end
elseif action == "deposit" % Branch Tcover1.1.3.4
    value = content("value");
    bill = database.get("bill", obj.userId, "time");
    savings = bill{end, 4};
    query = obj.userId + ...
        ", 'deposit:" + obj.userId + "', " + value + ...
        ", " + (savings + value) + ...
        ", '" + datestr(now, 31) + "'";
    err = database.insert("bill", query);
elseif action == "transfer" % Branch Tcover1.1.3.5
    value = content("value");
    target = content("target");
    dist = database.get("user", target);
    if (~isempty(dist)) && (dist{4}) % Branch Tcover1.1.3.6
        bill = database.get("bill", obj.userId, "time");
        savings = bill{end, 4};
        if value > savings % Branch Tcover1.1.3.7
            err = "存款余额不足.";
        else % Branch Tcover1.1.3.8
            query = obj.userId + ...
                ", 'transfer:" + target + "', " + value + ...
                ", " + (savings - value) + ...
```

```
                    ", '" + datestr(now, 31) + "'";
                err = database.insert("bill", query);
                if err ~= "nil" % Branch Tcover1.1.3.9
                    return
                end
                tbill = database.get("bill", target, "time");
                tsavings = tbill{end, 4};
                query = target + ...
                    ", 'receive:" + obj.userId + "', " + value + ...
                    ", " + (tsavings + value) + ...
                    ", '" + datestr(now, 31) + "'";
                err = database.insert("bill", query);
                if err ~= "nil" % Branch Tcover1.1.3.10
                    query = obj.userId + ...
                        ", 'return:" + target + "', " + value + ...
                        ", " + savings + ...
                        ", '" + datestr(now, 31) + "'";
                    err = database.insert("bill", query);
                    if err ~= "nil" % Branch Tcover1.1.3.11
                        err = "退还转账金额失败.";
                    end
                end
            end
        else % Branch Tcover1.1.3.12
            err = "账户不存在或已被冻结.";
        end
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testAccess.m/testUpdateData
- Test Case

	Test Case T1.1.3.1	Test Case T1.1.3.2	Test Case T1.1.3.3
Input	(database,"withdraw", content)	(database,"withdraw", content)	(database,"deposit", content)
Coverage Item	Tcover1.1.3.1	Tcover1.1.3.2	Tcover1.1.3.3
State	handler = UpdateData("1236171647798361"); content = containers.Map(["value"], [12350]);	handler = UpdateData("1236171647798361"); content = containers.Map(["value"], [1235000000000]);	handler = UpdateData("1236171647798361"); content = containers.Map(["value"], [12350]);
Expected Output	err == “nil”	err =="存款余额不足."	err == “nil”
Test Result	Passed	Passed	Passed

	Test Case T1.1.3.4	Test Case T1.1.3.5	Test Case T1.1.3.6
Input	(database,"deposit", content)	(database,"transfer", content)	(database,"transfer", content)
Coverage Item	Tcover1.1.3.4	Tcover1.1.3.5	Tcover1.1.3.6
State	handler = UpdateData("1236171647798361"); content = containers.Map(["value"], [24900]);	handler = UpdateData("1236171647798361"); content = containers.Map(["value", "target"], [200, 2124099443931732]);	handler = UpdateData("1236171647798361"); content = containers.Map(["value", "target"], [200, 2124099443931732]);
Expected Output	err = "nil"	err = "nil"	err = "nil"
Test Result	Passed	Passed	Passed
	Test Case T1.1.3.7	Test Case T1.1.3.8	Test Case T1.1.3.12
Input	(database,"transfer", content)	(database,"transfer", content)	(database,"transfer", content)
Coverage Item	Tcover1.1.3.7	Tcover1.1.3.8	Tcover1.1.3.12
State	handler = UpdateData("1236171647798361"); content = containers.Map(["value", "target"], [2000000000, 2124099443931732]);	handler = UpdateData("1236171647798361"); content = containers.Map(["value", "target"], [200, 2124099443931732]);	handler = UpdateData("1236171647798361"); content = containers.Map(["value", "target"], [200, 1145141919810]);
Expected Output	err == "存款余额不足."	err == "nil"	err == "账户不存在或已被冻结."
Test Result	Passed	Passed	Passed

- Test Coverage: 9/12 = 75%

1.2 Backend Database Test

1.2.1 Check Mutex

```
function err = checkMutex(obj)
    % Check if the database is locked, if so then users will
    % wait to its unlock. Return timeout err when the waiting
    % process goes more than 5 minutes.

    % Test function:
    % tests/unitTestBackend/testDatabase.m/testCheckMutex

    err = "nil";
```

```
        deadline = datetime('now') + minutes(5);
        while obj.mutex
            if datetime('now') > deadline % Statement Tcover1.2.1.1
                err = "数据库响应超时.";
                break
            end
        end
    end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testCheckMutex
- Test Case

	Test Case T1.2.1.1
Input	-----
Coverage Item	Tcover1.2.1.1
State	database.mutex = true,
Expected Output	err == “数据库相应超时“
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.2.2 Get

```
    function [res, err] = get(obj, table, uid, order)
        % Get the data of user from certain table in certain
        % order using the give data.

        % Parameters
        % ----------
        % table: string, name of a table.
        % uid: string, user’s id;
        % order: SQL, order of the return data.

        % Test function:
        % tests/unitTestBackend/testDatabase.m/testGet

        err = obj.checkMutex();
        res = NaN;
        if err ~= "nil" % Branch Tcover1.2.2.1
            return
        end
        obj.mutex = true;
        query = "SELECT * FROM " + table ...
            + " WHERE id == " + uid;
```

```
        if exist('order','var') % Branch Tcover1.2.2.2
            query = query + " ORDER BY " + order + ";";
        else % Branch Tcover1.2.2.3
            query = query + ";";
        end
        try
            res = fetch(obj.database, query);
            err = "nil";
        catch exception
            notice = Notice(uid, ...
                Error(400, 1, exception) ...
            );
            notice.logError()
            err = "数据库发生错误.";
        end
        obj.mutex = false;
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testGet
- Test Case

	Test Case T1.2.2.1	Test Cast T1.2.2.2	Test Cast T1.2.2.3
Input	("user", "1236171647798361")	("bill", "1236171647798361", "time")	("user", "1236171647798361")
Coverage Item	Tcover1.2.2.1	Tcover1.2.2.2	Tcover1.2.2.3
State	database.mutex = true,	database.mutex = false	database.mutex = false
Expected Output	err == “数据库相应超时“	err == “nil”, res{1,2} == 'init:1236171647798361'	err = =“nil”, res{1} = ='1236171647798361'
Test Result	Passed	Passed	Passed

- Test Coverage: 3/3 = 100%

1.2.3 Has

```
    function [res, err] = has(obj, table, uid)
        % Check if the database has certain user in certain table.

        % Parameters
        % ----------
        % table: string, name of a table.
        % uid: string, user's id;

        % Test function:
```

```
        % tests/unitTestBackend/testDatabase.m/testHas

        err = obj.checkMutex();
        res = NaN;
        if err ~= "nil" % Branch Tcover1.2.3.1
            return
        end
        [res, err] = obj.get(table, uid);
        if err == "nil" % Branch Tcover1.2.3.2
            if isempty(res) % Branch Tcover1.2.3.3
                res = "用户不存在或银行卡无效.";
            else % Branch Tcover1.2.3.4
                res = "nil";
            end
        end
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testHas
- Test Case

	Test Case T1.2.3.1	Test Cast T1.2.3.2	Test Cast T1.2.3.3
Input	("user", "1236171647798361")	("user", "1236171647798361")	("user", "1145141919810")
Coverage Item	Tcover1.2.2.1	Tcover1.2.2.2	Tcover1.2.2.3
State	database.mutex = true,	database.mutex = false	database.mutex = false
Expected Output	err == “数据库相应超时“	err == “nil”, res = =“nil”	err == “nil”, res == "用户不存在或银行卡无效.”
Test Result	Passed	Passed	Passed
	Test Case T1.2.3.4		
Input	("user", "1236171647798361")		
Coverage Item	Tcover1.2.2.4		
State	database.mutex = false		
Expected Output	err == “nil”, res == “nil”“		
Test Result	Passed		

- Test Coverage: 4/4 = 100%

1.2.4 Update

```
    function err = update(obj, table, uid, content)
        % update data of certain user in certain table, using
        % the given data.
```

```
        % Parameters
        % ----------
        % table: string, name of a table.
        % uid: string, user's id.
        % content: instance of containers.Map, used to update.

        % Test function:
        % tests/unitTestBackend/testDatabase.m/testUpdate

        err = obj.checkMutex();
        if err ~= "nil" % Branch Tcover1.2.4.1
            return
        end
        obj.mutex = true;
        if table == "bill"
            % This is an optional function that is not used at present
            condition = content("condition");
            result = content("result");
            query = "UPDATE " + table + " SET " + ...
                result + " WHERE id == " + uid + ...
                " AND (" + condition +");";
        else % Branch Tcover1.2.4.2
            query = "UPDATE " + table + " SET " + ...
                content + " WHERE id == " + uid + ";";
        end
        try
            exec(obj.database, query);
            err = "nil";
        catch exception
            notice = Notice(uid, ...
                Error(400, 1, exception) ...
            );
            notice.logError()
            err = "数据库发生错误.";
        end
        obj.mutex = false;
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testUpdate
- Test Case

	Test Case T1.2.4.1	Test Cast T1.2.4.2

Input	("user", "1236171647798361", "isActive=1")	("user", "1236171647798361", "isActive=1")
Coverage Item	Tcover1.2.2.1	Tcover1.2.2.2
State	database.mutex = true,	database.mutex = false
Expected Output	err == “数据库相应超时.“	err == “nil”,
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.2.5 Create User

```
function err = createUser(obj, user, bill)
    % Create a user using given data.

    % Parameters
    % ----------
    % user: user's id.
    % bill: user's initial bill record.

    % Test function:
    % tests/unitTestBackend/testDatabase.m/testCreateUser

    err = obj.checkMutex();
    if err ~= "nil" % Branch Tcover1.2.5.1
        return
    end
    obj.mutex = true;
    try % Branch Tcover1.2.5.2
        exec(obj.database, ...
            "INSERT INTO " + "user" + ...
            " (" + obj.tables("user") + ...
            ") VALUES (" + user + ");" ...
        );
        exec(obj.database, ...
            "INSERT INTO " + "bill" + ...
            " (" + obj.tables("bill") + ...
            ") VALUES (" + bill + ");" ...
        );
        err = "nil";
    catch exception
        notice = Notice(0, ...
            Error(400, 1, exception) ...
        );
        notice.logError()
```

```
            err = "数据库发生错误.";
        end
        obj.mutex = false;
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testCreateUser
- Test Case

	Test Case T1.2.4.1	Test Cast T1.2.4.2
Input	(random id, bill)	(random id, bill)
Coverage Item	Tcover1.2.2.1	Tcover1.2.2.2
State	database.mutex = true,	database.mutex = false
Expected Output	err == “数据库相应超时.“	err == “nil”,
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.2.6 Insert

```
function err = insert(obj, table, content)
    % Insert a piece of data into certain table using given
    % data.

    % Parameters
    % ----------
    % table: string, name of a table.
    % content: string, content of the data used to insert.

    % Test function:
    % tests/unitTestBackend/testDatabase.m/testInsert

    err = obj.checkMutex();
    if err ~= "nil" % Branch Tcover1.2.6.1
        return
    end
    obj.mutex = true;
    try % Branch Tcover1.2.6.2
        exec(obj.database, ...
            "INSERT INTO " + table + ...
            " (" + obj.tables(table) + ...
            ") VALUES (" + content + ");" ...
        );
        err = "nil";
    catch exception
        notice = Notice(0, ...
```

```
            Error(400, 1, exception) ...
        );
        notice.logError()
        err = "数据库发生错误.";
    end
    obj.mutex = false;
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testInsert
- Test Case

	Test Case T1.2.4.1	Test Cast T1.2.4.2
Input	query = 1145141919810 + 'withdraw:'' + 1145141919810 + ''', '' + 114514 + '', '' + 1919810 + '', ''' + datestr(now, 31) + ''''; ("bill", query);	query = 1145141919810 + 'withdraw:'' + 1145141919810 + ''', '' + 114514 + '', '' + 1919810 + '', ''' + datestr(now, 31) + ''''; ("bill", query);
Coverage Item	Tcover1.2.2.1	Tcover1.2.2.2
State	database.mutex = true,	database.mutex = false
Expected Output	err == “数据库相应超时“	err == “nil”,
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.2.7 Deactive User

```
function err = deactiveUser(obj, uid)
    % Deactivate a user.

    % Parameters
    % ----------
    % uid: string, user’s id.

    % Test function:
    % tests/unitTestBackend/testDatabase.m/testDeactiveUser

    err = obj.checkMutex();
    if err ~= "nil" % Branch Tcover1.2.7.1
        return
    end
    err = obj.update("user", uid, "isActive=0");
     % Branch Tcover1.2.7.2
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testDeactiveUser
- Test Case

	Test Case T1.2.4.1	Test Cast T1.2.4.2
Input	(1145141919810)	(1145141919810)
Coverage Item	Tcover1.2.2.1	Tcover1.2.2.2
State	database.mutex = true,	database.mutex = false
Expected Output	err == “数据库相应超时“	err == “nil”,
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.2.8 Active User

```
function err = activeUser(obj, uid)
    % Activate a user.

    % Parameters
    % ----------
    % uid: string, user’s id.

    err = obj.checkMutex();
    if err ~= "nil" % Branch Tcover1.2.8.1
        return
    end
        err = obj.update("user", uid, "isActive=1");
     % Branch Tcover1.2.8.2
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testDeactiveUser
- Test Case

	Test Case T1.2.4.1	Test Cast T1.2.4.2
Input	(1145141919810)	(1145141919810)
Coverage Item	Tcover1.2.2.1	Tcover1.2.2.2
State	database.mutex = true,	database.mutex = false
Expected Output	err == “数据库相应超时“	err == “nil”,
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.3 Backend Manage Test

1.3.1 Health Check

```
function [res, err] = healthCheck(obj, database)
        % check the health status of the database.

        % Parameters
        % ----------
        % database: instance of Database.

        % Test function:
        % tests/unitTestBackend/testManage.m/testHealthCheck

        err = "nil";
        res = "nil";
        [bill, err] = database.fetch("SELECT * FROM bill");
        if err ~= "nil" % Branch Tcover1.3.1.1
            return
        end
        s = size(bill);
        for i = 1:s(1)
            if bill{i, 4} < 0 % Branch Tcover1.3.1.2
                obj.userId = bill{i, 1};
                obj.error = Error(500, 2, "savings error");
                obj.logError();
                res = "Savings Error Detected.";
            end
        end
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testHealthCheck
- Test Case

	Test Case T1.3.1.1	Test Case T1.3.1.2
Input	Database	Database
Coverage Item	Tcover1.3.1.1	Tcover1.3.1.2
State	database.mutex = true, exception = MException('MyComponent:testing', 'Error no error'); error = Error(400, 1, exception);	database.mutex = false, exception = MException('MyComponent:testing', 'Error no error'); error = Error(400, 1, exception);

	handler = Notice("1145141919810",error);	handler = Notice("1145141919810",error);
Expected Output	err == “数据库相应超时“, res == “nil”	err == “nil”, res == “Savings Error Detected.”
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.3.2 Log Error

```
function logError(obj)
        % Log errors on console.

        % Test function:
        % tests/unitTestBackend/testManage.m/testLogError

        fprintf("An Error has occurred at user " + obj.userId + "\n");
        fprintf("Code: " + obj.error.code + "\n");
        fprintf("Level: " + obj.error.level + "\n");
        fprintf("Content: ");
        if isa(obj.error.content, "string") % Branch Tcover1.3.2.1
            fprintf("%s", obj.error.content);
        else % Branch Tcover1.3.2.2
            fprintf("%s", char(obj.error.content.message));
        end
        fprintf("\nTime: " + datestr(datetime('now')));
        fprintf("\n");
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testDatabase.m/testLogError
- Test Case

	Test Case T1.3.2.1	Test Case T1.3.2.2
Input	-----	------
Coverage Item	Tcover1.3.2.1	Tcover1.3.2.2

State	exception = "error";
	error = Error(400, 1, exception);
	handler = Notice("1145141919810",error);
		exception = MException('MyComponent:testing','Testing, no error.');error = Error(400, 1, exception);
		handler = Notice("1145141919810",error);

Expected Output	An Error has occurred at user 1145141919810 Code: 400	An Error has occurred at user 1145141919810 Code: 400

	Level: 1 Content: error Time: 09-Jun-2022 21:45:17	Level: 1 Content: Testing, no error. Time: 09-Jun-2022 21:45:18
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.4 Backend Server Test

1.4.1 Router

```
function res = router(obj, req)
        % Router to distribute requests to the corresponding
        % handler.

        % Parameters
        % ----------
        % request: instance of Request.

        % Test function:
        % tests/unitTestBackend/TestServer.m/testRouter

        if req.head == "/login" % Branch Tcover1.4.1.1

            % {
            %   userId: string(16) (^[0-9]$)
            %   password: string(6-30)
            %   clientKey: string(6-30) (optional)
            % }

            res = obj.doLogin(req);
        elseif req.head == "/slogin" % Branch Tcover1.4.1.2
```

```
    % {
    %   adminId: string(16) (^[0-9]$)
    %   password: string(6-30)
    % }

    res = obj.doSLogin(req);
elseif req.head == "/logout" % Branch Tcover1.4.1.3
    res = obj.doLogout(req);
elseif req.head == "/slogout" % Branch Tcover1.4.1.4
    res = obj.doSLogout(req);
elseif req.head == "/checkData"
    % This is an optional function that is not used at present

    % {
    %   userId: string(16) (^[0-9]$)
    % }

    res = obj.doCheckData(req);
elseif req.head == "/editData"
    % This is an optional function that is not used at present

    % {
    %   userId: string(16) (^[0-9]$)
    %   content: {
    %       condition: string(STL),
    %       result: string(STL)
    %   }
    % }

    res = obj.doEditData(req);
elseif req.head == "/healthCheck" % Branch Tcover1.4.1.5
    res = obj.doHealthCheck(req);
elseif req.head == "/getData" % Branch Tcover1.4.1.6
    res = obj.doGetData(req);
elseif req.head == "/updateData" % Branch Tcover1.4.1.7

    % {
    %   action: string
    %   content: {
    %       value: float(.3)
    %       target: string(16) (^[0-9]$)
    %   }
    % }
```

```
            res = obj.doUpdateData(req);
        elseif req.head == "/createAccount" % Branch Tcover1.4.1.8

            % {
            %   password: string(6-30)
            % }

            res = obj.doCreateAccount(req);
        elseif req.head == "/deactiveAccount" % Branch Tcover1.4.1.9

            % {
            %   userId: string(16) (^[0-9]$)
            % }

            res = obj.doDeactiveAccount(req);
        elseif req.head == "/activeAccount" % Branch Tcover1.4.1.10

            % {
            %   userId: string(16) (^[0-9]$)
            % }

            res = obj.doActiveAccount(req);
        else % Branch Tcover1.4.1.11
            res = obj.makeErrorMsg("Invalid Request");
        end
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testRouter
- Test Case

	Test Case T1.4.1.1	Test Case T1.4.1.2	Test Case T1.4.1.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.1.1	Tcover1.4.1.2	Tcover1.1.3.3
State	request = Request("/login", ... containers.Map(["userId", "password"], ... ["1236171647798361", "654321"]), ... "atm", 0);	request = Request("/slogin", ... containers.Map(["adminId", "password"], ... ["1926081719260817", "123456"]), ... "admin", 0);	request = Request("/login", ... containers.Map(["userId", "password"], ... ["1236171647798361", "654321"]), ... "atm", 0);

Expected Output	res.body("msg") == "Success"	res.body("msg") == "Success"	res.body("msg") == "Success"
Test Result	Passed	Passed	Passed
	Test Case T1.4.1.4	Test Case T1.4.1.5	Test Case T1.4.1.6
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.1.4	Tcover1.4.1.5	Tcover1.4.1.6
State	Admin login. request = Request("/slogout", ... containers.Map(), "admin", sid);	Admin login. request = Request("/healthCheck", ... containers.Map(), "admin", sid);	Admin login request = Request("/getData", ... containers.Map(), "atm", sid);
Expected Output	res.body("msg") == "Success"	res.body("msg") == "Success"	res.body("msg") == "Success"
Test Result	Passed	Passed	Passed
	Test Case T1.4.1.7	Test Case T1.4.1.8	Test Case T1.4.1.9
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.1.7	Tcover1.4.1.8	Tcover1.4.1.9
State	Admin login. request = Request("/updateData", ... containers.Map(["action", "content"], ... {"withdraw", containers.Map("value", 12350)}), ..."atm", sid);	Admin login. request = Request("/createAccount", containers.Map ... ("password", "114514"), "admin", sid);	Admin login request = Request("/deactiveAccount", ... containers.Map("userId", "1236171647798361"), "admin", sid);
Expected Output	res.body("msg") == "Success"	res.body("msg") == "Success"	res.body("msg") == "Success"
Test Result	Passed	Passed	Passed
	Test Case T1.4.1.10	Test Case T1.4.1.11	
Input	(request)	(request)	
Coverage Item	Tcover1.4.1.10	Tcover1.4.1.11	
State	Admin login. request = Request("/activeAccount", containers.Map("userId", "1236171647798361"), "admin", sid);	request = Request("/invalid Request", containers.Map(), "atm", 0);	
Expected Output	res.body("msg") == "Success"	res.body("msg") == "Invalid Request"	
Test Result	Passed	Passed	

- Test Coverage: 11/11 = 100%

1.4.2 Do Login

```
function res = doLogin(obj, req)
        % Check the request is valid and do the user login
        % sequences.

        % Parameters
        % ----------
        % request: instance of Request.

        % Test function:
        % tests/unitTestBackend/TestServer.m/testDoLogin

        if (~isKey(req.body, "userId")) || (~isKey(req.body,"password"))
            % Branch Tcover1.4.2.1
            res = obj.makeErrorMsg("请求参数不足.");
            return
        end
        uid = req.body("userId"); % Branch Tcover1.4.2.2
        password = req.body("password");
        [res, err] = obj.userManager.checkLoged(uid, req.device);
        if err ~= "nil"  % Branch Tcover1.4.2.3
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.2.4
            res = obj.makeErrorMsg(res);
            return
        end
        [res, err] = obj.database.has("user", uid);
        if err ~= "nil" % Branch Tcover1.4.2.5
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.2.6
            res = obj.makeErrorMsg(res);
            return
            % User Not Exist.
```

```
end
[answer, err] = obj.database.get("user", uid);
if err ~= "nil" % Branch Tcover1.4.2.7
    res = obj.makeErrorMsg(err);
    return
end
isActive = answer{4};
if ~isActive % Branch Tcover1.4.2.8
    res = obj.makeErrorMsg("用户已被冻结.");
    return;
end
allowApp = answer{3};
answer = string(answer{2});
if req.device == "app" && ~allowApp % Branch Tcover1.4.2.9
    if ~isKey(req.body, "clientKey") % Branch Tcover1.4.2.10
        res = obj.makeErrorMsg("APP 登录未被激活.");
        return;
    end
    if obj.database.clientKey ~= req.body("clientKey")
     % Branch Tcover1.4.2.11
        res = obj.makeErrorMsg("激活码错误.");
        return;
        % App Not Allowed.
    end
    err = obj.database.update("user", uid, "allowApp=1");
    % Branch Tcover1.4.2.12
    if err ~= "nil" % Branch Tcover1.4.2.13
        res = obj.makeErrorMsg(err);
        return
    end
end
if answer ~= password % Branch Tcover1.4.2.14
    res = obj.makeErrorMsg("密码错误.");
    return
    % Wrong Password.
end
[sid, err] = obj.userManager.createSession(uid, req.device);
if err ~= "nil" % Branch Tcover1.4.2.15
    res = obj.makeErrorMsg(err);
    return
end
res = Response("/", ... % Branch Tcover1.4.2.16
    containers.Map("msg", "Success"), ...
    "server", sid, 200 ...
```

```
        );
        % 200 For Success.
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoLogin
- Test Case

	Test Case T1.4.2.1	Test Case T1.4.2.2	Test Case T1.4.2.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.2.1	Tcover1.4.2.2	Tcover1.4.2.3
State	request = Request("/login", containers.Map(), ... "atm", 0);	request = Request("/login", ... containers.Map(["userId", "password"], ... ["1236171647798361", "654321"]), ... "atm", 0);	request = Request("/login", ... containers.Map(["userId", "password"], ... ["1236171647798361", "654321"]), ... "atm", 0); userManager.mutex = true;
Expected Output	res.body("msg") == "请求参数不足."	res.body("msg") == "Success"	res.body("msg") == "服务器响应超时."
Test Result	Passed	Passed	Passed
	Test Case T1.4.2.4	Test Case T1.4.2.5	Test Case T1.4.2.6
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.2.4	Tcover1.4.2.5	Tcover1.4.2.6
State	User login. request = Request("/login", ... containers.Map(["userId", "password"], ... ["1236171647798361", "654321"]), ... "atm", 0);	request = Request("/login", ... containers.Map(["userId", "password"], ... ["1236171647798361", "654321"]), ... "atm", 0); database.mutex = true;	request = Request("/login", ... containers.Map(["userId", "password"], ... ["114514", "654321"]), ... "atm", 0);
Expected Output	res.body("msg") == "用户已登录."	res.body("msg") == "数据库相应超时."	res.body("msg") == "用户不存在或银行卡无效"
Test Result	Passed	Passed	Passed
	Test Case T1.4.2.7	Test Case T1.4.2.8	Test Case T1.4.2.9
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.2.8	Tcover1.4.2.9 Tcover1.4.2.10	Tcover1.4.2.11
State	request = Request("/login", ... containers.Map(["userId",	request = Request("/login", ...	request = Request("/login", ... containers.Map(["userId",

	"password"], ... ["4173516266740291", "654321"]), ... "atm", 0);	containers.Map(["userId", "password"], ... ["5927735005791540", "654321"]), ... "app", 0);	"password","clientKey"], ... ["5927735005791540", "654321","114514"]), ... "app", 0);
Expected Output	res.body("msg") == "用户已被冻结."	res.body("msg") == "APP登录未被激活."	res.body("msg") == "激活码错误"
Test Result	Passed	Passed	Passed
	Test Case T1.4.2.10	Test Case T1.4.2.11	
Input	(request)	(request)	
Coverage Item	Tcover1.4.2.16	Tcover1.4.2.14	
State	request = Request("/login", ... containers.Map(["userId", "password","clientKey"], ... [string(uid), "114514","123456"]), ... "app", 0);	request = Request("/login", ... containers.Map(["userId", "password"], ... ["3495524663390897", "114514"]), ... "app", 0);	
Expected Output	res.body("msg") == "Success"	res.body("msg") == "密码错误."	
Test Result	Passed	Passed	

- Test Coverage: 12/16 = 75%

1.4.3 Do SLogin

```
function res = doSLogin(obj, req)
            % Check if the request is valid and do the admin login
            % sequences.

            % Parameters
            % ----------
            % request: instance of Request.

            if (~isKey(req.body, "adminId")) || (~isKey(req.body,
"password"))
                res = obj.makeErrorMsg("请求参数不足."); % Branch
Tcover1.4.3.1
                return
            end
            aid = req.body("adminId");
            password = req.body("password");
            [res, err] = obj.adminManager.checkLoged(aid, req.device);
```

```
        if err ~= "nil" % Branch Tcover1.4.3.2
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.3.3
            res = obj.makeErrorMsg(res);
            return
            % Admin Has Login
        end
        [res, err] = obj.database.has("admin", aid);
        if err ~= "nil" % Branch Tcover1.4.3.4
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.3.5
            res = obj.makeErrorMsg(res);
            return
            % Admin Not Exist.
        end
        [answer, err] = obj.database.get("admin", aid);
        if err ~= "nil" % Branch Tcover1.4.3.6
            res = obj.makeErrorMsg(err);
            return
        end
        answer = string(answer{2});
        if answer ~= password % Branch Tcover1.4.3.7
            res = obj.makeErrorMsg("密码错误.");
            return
            % Wrong Password.
        end
        [sid, err] = obj.adminManager.createSession(aid, req.device);
        if err ~= "nil" % Branch Tcover1.4.3.8
            res = obj.makeErrorMsg(err);
            return
        end
        res = Response("/", ... % Branch Tcover1.4.3.9
            containers.Map("msg", "Success"), ...
            "server", sid, 200 ...
        );
        % 200 For Success.
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoSLogin

- Test Case

	Test Case T1.4.3.1	Test Case T1.4.3.2	Test Case T1.4.3.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.3.1	Tcover1.4.3.9	Tcover1.4.2.2
State	request = Request("/slogin", ... containers.Map(), ... "admin", 0);	request = Request("/slogin", ... containers.Map(["adminId", "password"], ... ["1926081719260817", "123456"]), ... "admin", 0);	request = Request("/slogin", ... containers.Map(["adminId", "password"], ... ["1926081719260817", "123456"]), ... "admin", 0); adminManager.mutex = true;
Expected Output	res.body("msg") == "请求参数不足."	res.body("msg") == "Success"	res.body("msg") == "服务器响应超时."
Test Result	Passed	Passed	Passed
	Test Case T1.4.3.4	Test Case T1.4.3.5	Test Case T1.4.3.6
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.3.3	Tcover1.4.3.4	Tcover1.4.3.5
State	Admin login. request = Request("/slogin", ... containers.Map(["adminId", "password"], ... ["1926081719260817", "123456"]), ... "admin", 0);	request = Request("/slogin", ... containers.Map(["adminId", "password"], ... ["1926081719260817", "123456"]), ... "admin", 0); database.mutex = true;	request = Request("/slogin", ... containers.Map(["adminId", "password"], ... ["114514", "123456"]), ... "admin", 0);
Expected Output	res.body("msg") == "用户已登录."	res.body("msg") == "数据库相应超时."	res.body("msg") == "用户不存在或银行卡无效"
Test Result	Passed	Passed	Passed
	Test Case T1.4.3.7		
Input	(request)		
Coverage Item	Tcover1.4.3.7		
State	request = Request("/slogin", ... containers.Map(["adminId", "password"], ... ["1926081719260817", "114514"]), ... "admin", 0);		
Expected Output	res.body("msg") == "密码错误."		
Test Result	Passed		

- Test Coverage: 7/9 = 77%

1.4.4 Do Logout

```
function res = doLogout(obj, req)
        % Check if the request is valid and do the user logout
        % sequences.

        % Parameters
        % ----------
        % request: instance of Request.

        % Test function:
        % tests/unitTestBackend/TestServer.m/testDoLogout

        sid = req.sessionId;
        [res, err] = obj.userManager.checkSessionId(sid, req.device);
        if err ~= "nil" % Branch Tcover1.4.4.1
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.4.2
            res = obj.makeErrorMsg(res);
            return
        end
        err = obj.userManager.destroySession(sid);
        if err ~= "nil" % Branch Tcover1.4.4.3 % notcover
            res = obj.makeErrorMsg(err);
            return
        end
        res = Response("/", ... % Branch Tcover1.4.4.4
            containers.Map("msg", "Success"), ...
            "server", 0, 200 ...
        );
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoLogout
- Test Case

	Test Case T1.4.4.1	Test Case T1.4.4.2	Test Case T1.4.4.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.4.1	Tcover1.4.4.2	Tcover1.4.4.4

State	User login. request = Request("/logout", containers.Map(), "atm", 114514);	User login. request = Request("/logout", containers.Map(), "atm", sessionId); userManager.mutex = true;	User login. request = Request("/logout", containers.Map(), "atm", sessionId);
Expected Output	res.body("msg") == "登陆状态无效."	res.body("msg") == "服务器响应超时."	res.body("msg") == "Success."
Test Result	Passed	Passed	Passed

- Test Coverage: 3/4 = 75%

1.4.5 Do SLogout

```
function res = doSLogout(obj, req)
        % Check if the request is valid and do the admin logout
        % sequences.

        % Parameters
        % ----------
        % request: instance of Request.

        % Test function:
        % tests/unitTestBackend/TestServer.m/testDoSLogout


        sid = req.sessionId;
        [res, err] = obj.adminManager.checkSessionId(sid, req.device);
        if err ~= "nil" % Branch Tcover1.4.5.1
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.5.2
            res = obj.makeErrorMsg(res);
            return
        end
        err = obj.adminManager.destroySession(sid);
        if err ~= "nil" % Branch Tcover1.4.5.3
            res = obj.makeErrorMsg(err);
            return
        end
        res = Response("/", ... % 1.4.5.4
```

```
            containers.Map("msg", "Success"), ...
            "server", 0, 200 ...
        );
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoSLogout
- Test Case

	Test Case T1.4.5.1	Test Case T1.4.5.2	Test Case T1.4.5.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.5.1	Tcover1.4.5.2	Tcover1.4.5.4

State	Admin login.
	request = Request("/slogout", containers.Map(), "admin", 114514);
		Admin login.
		request = Request("/slogout", containers.Map(), "admin", sessionId);
		adminManager.mutex = true;
			Admin login.
			request = Request("/slogout", containers.Map(), "admin", sessionId);

Expected Output	res.body("msg") == "登陆状态无效."	res.body("msg") == "服务器响应超时."	res.body("msg") == "Success."
Test Result	Passed	Passed	Passed

- Test Coverage: 3/4 = 75%

1.4.6 Do Health Check

```
function res = doHealthCheck(obj, req)
        % Manually run health check of the database for admin.

        % Parameters
        % ----------
        % request: instance of Request.

        % Test function:
        % tests/unitTestBackend/TestServer.m/testDoHealthCheck

        sid = req.sessionId;
        [res, err] = obj.adminManager.checkSessionId(sid, req.device);
        if err ~= "nil" % Branch Tcover1.4.6.1
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.6.2
```

```
            res = obj.makeErrorMsg(res);
            return
        end
        verifier = Notice(0, Error(0, 0, 0));
        [res, err] = verifier.healthCheck(obj.database);
        if err ~= "nil" % Branch Tcover1.4.6.3
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.6.4
            res = obj.makeErrorMsg(res);
            return
        end
        res = Response("/", ...
            containers.Map("msg", "Success"), ...
            "server", 0, 200 ...
        );
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoHealthCheck
- Test Case

	Test Case T1.4.6.1	Test Case T1.4.6.2	Test Case T1.4.6.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.6.1	Tcover1.4.6.2	Tcover1.4.6.4
State	request = Request("/healthCheck", containers.Map(), "admin", sessionId); adminManager.mutex = true;	request = Request("/healthCheck", containers.Map(), "admin", 114514);	request = Request("/healthCheck", containers.Map(), "admin", sid); database.mutex = true;
Expected Output	res.body("msg") == "服务器响应超时."	res.body("msg") == "登陆状态无效."	res.body("msg") == "数据库响应超时."
Test Result	Passed	Passed	Passed
	Test Case T1.4.6.4		
Input	(request)		
Coverage Item	Tcover1.4.6.4		
State	request = Request("/healthCheck", containers.Map(), "admin", res.sessionId);		
Expected Output	res.body("msg") == "Savings Error Detected."		

Test Result	Passed		

- Test Coverage: 4/4 = 100%

1.4.7 Do Get Data

```
function res = doGetData(obj, req)
        % Check if the request is valid and get data for the
        % user.

        % Parameters
        % ----------
        % request: instance of Request.

        % Test function:
        % tests/unitTestBackend/TestServer.m/testDoGetData

        sid = req.sessionId;
        [res, err] = obj.userManager.checkSessionId(sid, req.device);
        if err ~= "nil" % Branch Tcover1.4.7.1
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.7.2
            res = obj.makeErrorMsg(res);
            return
        end
        [session, err] = obj.userManager.getSession(sid);
        if err ~= "nil" % Branch Tcover1.4.7.3 %notcover
            res = obj.makeErrorMsg(err);
            return
        end
        uid = session.userId;
        handler = GetData(uid);
        [res, err] = handler.verifyUser(obj.database);
        if err ~= "nil" % Branch Tcover1.4.7.4
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.7.5 %notcover
            res = obj.makeErrorMsg(res);
            return
        end
        [bill, err] = handler.getData(obj.database);
```

```
        if err ~= "nil" % Branch Tcover1.4.7.6 %notcover
            res = obj.makeErrorMsg(err);
            return
        end
        res = Response("/", ...
            containers.Map(["msg", "bill"], {"Success", bill}), ...
            "server", 0, 200 ...
        );
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoGetData
- Test Case

	Test Case T1.4.7.1	Test Case T1.4.7.2	Test Case T1.4.7.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.7.1	Tcover1.4.7.2	Tcover1.4.7.4

State	User login.
	request = Request("/getData", ... containers.Map(), "atm", sid);
	userManager.mutex = true;
		User login.
		request = Request("/getData", ... containers.Map(), "atm", 114514);
			User login.
			request = Request("/getData", ... containers.Map(), "atm", sid);
			database.mutex = true;

Expected Output	res.body("msg") == "服务器响应超时."	res.body("msg") == "登陆状态无效."	res.body("msg") == "数据库响应超时."
Test Result	Passed	Passed	Passed

- Test Coverage: 3/6 = 50%

1.4.8 Do Update Data

```
function res = doUpdateData(obj, req)
        % Check if the request is valid and update data for the
        % user.

        % Parameters
        % ----------
        % request: instance of Request.

        sid = req.sessionId;
        [res, err] = obj.userManager.checkSessionId(sid, req.device);
        if err ~= "nil" % Branch Tcover1.4.8.1
            res = obj.makeErrorMsg(err);
```

```
            return
        end
        if res ~= "nil" % Branch Tcover1.4.8.2
            res = obj.makeErrorMsg(res);
            return
        end
        [session, err] = obj.userManager.getSession(sid);
        if err ~= "nil" % Branch Tcover1.4.8.3 %notcover
            res = obj.makeErrorMsg(err);
            return
        end
        uid = session.userId;
        handler = UpdateData(uid);
        [res, err] = handler.verifyUser(obj.database);
        if err ~= "nil" % Branch Tcover1.4.8.4
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.8.5 %notcover
            res = obj.makeErrorMsg(res);
            return
        end % Branch Tcover1.4.8.6
        if (~isKey(req.body, "action")) || (~isKey(req.body, "content"))
            res = obj.makeErrorMsg("请求参数不足.");
            return
        end
        action = req.body("action");
        content = req.body("content");
        err = handler.updateData(obj.database, action, content);
        if err ~= "nil" % Branch Tcover1.4.8.7 %notcover
            res = obj.makeErrorMsg(err);
            return
        end
        res = Response("/", ...
            containers.Map("msg", "Success"), ...
            "server", 0, 200 ...
        );
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoUpdateData
- Test Case

	Test Case T1.4.8.1	Test Case T1.4.8.2	Test Case T1.4.8.3
Input	(request)	(request)	(request)

Coverage Item	Tcover1.4.8.1	Tcover1.4.8.2	Tcover1.4.8.4

State	User login.
	request = Request("/updateData", containers.Map (["action", "content"], ... {"withdraw", containers.Map ("value", 12350)}), ... "atm", sid);
	userManager.mutex = true;
		User login.
		request = Request("/updateData", containers.Ma p(["action", "content"], ... {"withdraw", containers.Map ("value", 12350)}), ... "atm", 114514);
			User login.
			request = Request("/updateData", containers.Map ([“action", "content"], ... {"withdraw", containers.Map ("value", 12350)}), ... "atm", sid);
			database.mutex = true;

Expected Output	res.body("msg") == "服务器响应超时."	res.body("msg") == "登陆状态无效."	res.body("msg") == "数据库响应超时."
Test Result	Passed	Passed	Passed
	Test Case T1.4.8.4		
Input	(request)		
Coverage Item	Tcover1.4.8.6		
State	User login. request = Request("/updateData", containers.Map ([“content"], ... {containers.Map("value", 12350)}), ... "atm", sid);		
Expected Output	res.body("msg") == "请求参数不足."		
Test Result	Passed		

- Test Coverage: 4/7 = 57%

1.4.9 Do Create Account

```
function res = doCreateAccount(obj, req)
        % Check if the request is valid and create a new user
        % for admin.

        % Parameters
        % ----------
        % request: instance of Request.

        sid = req.sessionId;
```

```
[res, err] = obj.adminManager.checkSessionId(sid, req.device);
if err ~= "nil" % Branch Tcover1.4.9.1
    res = obj.makeErrorMsg(err);
    return
end
if res ~= "nil" % Branch Tcover1.4.9.2
    res = obj.makeErrorMsg(res);
    return
end
uid = [randi([1, 9], 1, 1), randi([0,9], 1, 15)];
uid = num2str(uid);
uid = strrep(uid, " ", "");
[has, err] = obj.database.has("user", uid);
if err ~= "nil" % Branch Tcover1.4.9.3
    res = obj.makeErrorMsg(err);
    return
end
while has == "nil"
    uid = [randi([1, 9], 1, 1), randi([0,9], 1, 15)];
    uid = num2str(uid);
    uid = strrep(uid, " ", "");
    [has, err] = obj.database.has("user", uid);
    if err ~= "nil" % Branch Tcover1.4.9.4 %notcover
        res = obj.makeErrorMsg(err);
        return
    end
end
if ~isKey(req.body, "password") % Branch Tcover1.4.9.5
    res = obj.makeErrorMsg("请求参数不足.");
    return
end
password = req.body("password");
user = uid + ", " + password + ", " + "0, 1";
bill = (uid + ", " + "'init:" + uid + "', 0, 0, " + ...
    "'" + datestr(now, 31) + "'" ...
);
err = obj.database.createUser(user, bill);
if err ~= "nil" % Branch Tcover1.4.9.6 %notcover
    res = obj.makeErrorMsg(err);
    return
end
res = Response("/", ...
    containers.Map(["msg", "userId"], ["Success", uid]), ...
    "server", 0, 200 ...
```

```
        );
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoCreateAccount
- Test Case

	Test Case T1.4.9.1	Test Case T1.4.9.2	Test Case T1.4.9.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.9.1	Tcover1.4.9.2	Tcover1.4.9.3

State	Admin login
	request = Request("/deactiveAccount", ...
	containers.Map("userId", "1236171647798361"), "admin", sid);
	adminManager.mutex = true;
		Admin login
		request = Request("/deactiveAccount", ...
		containers.Map("userId", "1236171647798361"), "admin", 114514);
			Admin login
			request = Request("/deactiveAccount", ...
			containers.Map("userId", "1236171647798361"), "admin", sid);
			database.mutex = true;

Expected Output	res.body("msg") == "服务器响应超时."	res.body("msg") == "登陆状态无效."	res.body("msg") == "数据库响应超时."
Test Result	Passed	Passed	Passed
	Test Case T1.4.9.4		
Input	(request)		
Coverage Item	Tcover1.4.9.4		
State	Admin login request = Request("/deactiveAccount", ... containers.Map(["content"], ... {containers.Map("value", 12350)}), ... "atm", sid);		
Expected Output	res.body("msg") == "请求参数不足."		
Test Result	Passed		

- Test Coverage: 4/6 = 66%

1.4.10 Do Deactive Account

```
function res = doDeactiveAccount(obj, req)
    % Check if the request is valid and deactive the account
```

```
        % of certain user.

        % Parameters
        % ----------
        % request: instance of Request.

        sid = req.sessionId;
        [res, err] = obj.adminManager.checkSessionId(sid, req.device);
        if err ~= "nil" % Branch Tcover1.4.10.1
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.10.2
            res = obj.makeErrorMsg(res);
            return
        end
        if ~isKey(req.body, "userId") % Branch Tcover1.4.10.3
            res = obj.makeErrorMsg("请求参数不足.");
            return
        end
        uid = req.body("userId");
        [res, err] = obj.database.has("user", uid);
        if err ~= "nil" % Branch Tcover1.4.10.4
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.10.5
            res = obj.makeErrorMsg(res);
            return
        end
        err = obj.database.deactiveUser(uid);
        if err ~= "nil" % Branch Tcover1.4.10.6 %notcover
            res = obj.makeErrorMsg(err);
            return
        end
        res = Response("/", ...
            containers.Map("msg", "Success"), ...
            "server", 0, 200 ...
        );
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoDeactiveAccount
- Test Case

	Test Case T1.4.10.1	Test Case T1.4.10.2	Test Case T1.4.10.3
Input	(request)	(request)	(request)
Coverage Item	Tcover1.4.10.1	Tcover1.4.10.2	Tcover1.4.10.3

State	Admin login
	request = Request("/deactiveAccount", ...
	containers.Map("userId", "1236171647798361"), "admin", sid);
	adminManager.mutex = true;
		Admin login
		request = Request("/deactiveAccount", ...
		containers.Map("userId", "1236171647798361"), "admin", 114514);
			Admin login
			Request("/deactiveAccount", ...
			containers.Map("userId", "1236171647798361"), "admin", sid);

Expected Output	res.body("msg") == "服务器响应超时."	res.body("msg") == "登陆状态无效."	res.body("msg") == "请求参数不足." "
Test Result	Passed	Passed	Passed
	Test Case T1.4.10.4	Test Case T1.4.10.5	
Input	(request)	(request)	
Coverage Item	Tcover1.4.10.4	Tcover1.4.10.5	
State	Admin login request = Request("/deactiveAccount", ... containers.Map(["content"], ... {containers.Map("value", 12350)}), ... "atm", sid); database.mutex = true;	Admin login request = Request("/deactiveAccount", ... containers.Map("userId", "114514"), "admin", sid);	
Expected Output	res.body("msg") == "数据库响应超时.	res.body("msg") == “用户不存在或银行卡无效”	
Test Result	Passed	Passed	

- Test Coverage: 5/6 = 83%

1.4.11 Do Active Account

```
function res = doActiveAccount(obj, req)
    % Check if the request is valid and activate the account
    % of certain user.
```

```
        % Parameters
        % ----------
        % request: instance of Request.

        sid = req.sessionId;
        [res, err] = obj.adminManager.checkSessionId(sid, req.device);
        if err ~= "nil" % Branch Tcover1.4.11.1
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.11.2
            res = obj.makeErrorMsg(res);
            return
        end
        if ~isKey(req.body, "userId")
            res = obj.makeErrorMsg("请求参数不足.");
            return
        end
        uid = req.body("userId");
        [res, err] = obj.database.has("user", uid);
        if err ~= "nil" % Branch Tcover1.4.11.3
            res = obj.makeErrorMsg(err);
            return
        end
        if res ~= "nil" % Branch Tcover1.4.11.4
            res = obj.makeErrorMsg(res);
            return
        end
        err = obj.database.activeUser(uid);
        if err ~= "nil" % Branch Tcover1.4.11.5 %notcover
            res = obj.makeErrorMsg(err);
            return
        end
        res = Response("/", ...
            containers.Map("msg", "Success"), ...
            "server", 0, 200 ...
        );
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testDoActiveAccount
- Test Case

	Test Case T1.4.11.1	Test Case T1.4.11.2	Test Case T1.4.11.3
Input	(request)	(request)	(request)

Coverage Item	Tcover1.4.11.1	Tcover1.4.11.2	Tcover1.4.11.3

State	Admin login
	request = Request("/activeAccount", containers.Map("userId", "1236171647798361"), "admin", sid);
	adminManager.mutex = true;
		Admin login
		request = Request("/activeAccount", containers.Map("userId", "1236171647798361"), "admin", 114514)
			Admin login
			Request("/activeAccount", containers.Map(), "admin", sid);

Expected Output	res.body("msg") == "服务器响应超时."	res.body("msg") == "登陆状态无效."	res.body("msg") == "请求参数不足." "
Test Result	Passed	Passed	Passed
	Test Case T1.4.11.4	Test Case T1.4.11.5	
Input	(request)	(request)	
Coverage Item	Tcover1.4.11.4	Tcover1.4.11.5	

State	Admin login
	request = Request("/activeAccount", containers.Map("userId", "1236171647798361"), "admin", sid);
	database.mutex = true;
		Admin login
		request = Request("/activeAccount", containers.Map("userId", "114514"), "admin", sid);	

Expected Output	res.body("msg") == "数据库响应超时.	res.body("msg") == “用户不存在或银行卡无效”	
Test Result	Passed	Passed	

- Test Coverage: 5/6 = 83%

1.4.12 Verify Session

```
function res = verifySession(obj)
    % Update the session’s status.

     % Test function:
    % tests/unitTestBackend/testServer.m/testVerifySession

    % Statement Tcover1.4.12.1
    obj.checkExpire();
    res = obj.isActive;
end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestBackend/testServer.m/testVerifySession
- Test Case

	Test Case T1.4.12.1
Input	-----
Coverage Item	Tcover1.4.12.1
State	uid = "114514"; sid = 1; dev = "atm";
Expected Output	res == true
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.4.13 Check Expire

```
function checkExpire(obj)
      % Check if the session is expired, if so then deactivate
      % it.

      % Test function:
      % tests/unitTestBackend/testServer.m/testCheckExpire

      if datetime('now') > obj.maxLifetime

          % Statement Tcover1.4.13.1
          obj.deactive();
      end
  end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestBackend/testServer.m/testCheckExpire
- Test Case

	Test Case T1.4.13.1
Input	-----
Coverage Item	Tcover1.4.13.1
State	uid = "114514"; sid = 1; dev = "atm"; maxLifetime > datatime(“now”)
Expected Output	isActive == false
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.4.14 Deactive

```
function deactive(obj)
      % Deactivate the session.

      % Test function:
      % tests/unitTestBackend/testServer.m/testDeactive

      % Statement Tcover1.4.14.1
      obj.isActive = false;
  end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestBackend/testServer.m/testDeactive
- Test Case

	Test Case T1.4.14.1
Input	-----
Coverage Item	Tcover1.4.14.1
State	uid = "114514"; sid = 1; dev = "atm";
Expected Output	isActive == false
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.4.15 Update Status

```
function err = updateStatus(obj)
      % Update all the sessions'' status in the pool.

      % Test function:
      % tests/unitTestBackend/testServer.m/testUpdateStatus

      err = obj.checkMutex();
      if err ~= "nil" % Branch Tcover1.4.15.1
          return
      end
      obj.mutex = true;
      for i = 1: obj.count
```

```
            if ~obj.pool(i).isActive % Branch Tcover1.4.15.2
                continue
            end
            obj.pool(i).checkExpire()
        end
        err = "nil";
        obj.mutex = false;
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testUpdateStatus
- Test Case

	Test Case T1.4.15.1	Test Cast T1.4.15.2
Input	-----	-----
Coverage Item	Tcover1.4.15.1	Tcover1.4.15.2
State	SessionManager.mutex = true;	SessionManager.mutex = false
Expected Output	err == “服务器响应超时.“	err == “nil”,
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.4.16 Check SessionId

```
function [res, err] = checkSessionId(obj, sid, dev)
    % Check if a session is valid using given data.

    % Parameters
    % ----------
    % sid: int64, session's id.
    % dev: string, session's device tag.

    % Test function:
    % tests/unitTestBackend/testServer.m/testCheckSessionId

    res = "nil";
    err = obj.checkMutex();
    if err ~= "nil" % Branch Tcover1.4.16.1
        return
    end
    obj.mutex = true;
    if sid <= 0 || sid > obj.count % Branch Tcover1.4.16.2
        res = "登录状态无效.";
```

```
        elseif ~obj.pool(sid).isActive % Branch Tcover1.4.16.3
            res = "登录状态已过期.";
        elseif obj.pool(sid).device ~= dev % Branch Tcover1.4.16.4
            res = "请求设备非法.";
        else % Branch Tcover1.4.16.5
            res = "nil";
        end
        err = "nil";
        obj.mutex = false;
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testCheckSessionId
- Test Case

	Test Case T1.4.16.1	Test Case T1.4.16.2	Test Case T1.4.16.3
Input	(1, "atm")	(-1, "atm")	(sessionId, "atm")
Coverage Item	Tcover1.4.16.1	Tcover1.4.16.2	Tcover1.4.16.4
State	SessionManager.mutex = true;		User login time out
Expected Output	err == “服务器响应超时.	res == "登陆状态无效."	res == “登陆状态已过期."
Test Result	Passed	Passed	Passed
	Test Case T1.4.16.4	Test Case T1.4.16.5	
Input	(sessionId, "InvalidATM")	(sessionId, "atm")	
Coverage Item	Tcover1.4.16.4	Tcover1.4.16.5	
State			
Expected Output	res == "请求设备非法.."	res == "nil."	
Test Result	Passed	Passed	

- Test Coverage: 5/5 = 100%

1.4.17 Check Loged

```
function [res, err] = checkLoged(obj, uid, dev)
        % Check if a user has logged using give data.

        % Parameters
        % ----------
        % uid: string, user's id.
        % dev: string, device tag of user's login request.

        % Test function:
```

```
        % tests/unitTestBackend/testServer.m/testCheckLoged

        res = "nil";
        err = obj.checkMutex();
        if err ~= "nil" % Branch Tcover1.4.17.1
            return
        end
        obj.mutex = true;
        for i = 1:obj.count
            session = obj.pool(i);
            if ~session.isActive % Branch Tcover1.4.17.2
                continue;
            end
            if strcmp(session.userId, uid) && session.device == dev

                % Branch Tcover1.4.17.3
                res = "用户已登录.";
            end
        end
        err = "nil";
        obj.mutex = false;
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestBackend/testServer.m/testCheckLoged
- Test Case

	Test Case T1.4.17.1	Test Case T1.4.17.2	Test Case T1.4.17.3
Input	(“114514”, "atm")	(“114514”, "atm")	(“114514”, "atm")
Coverage Item	Tcover1.4.17.1	Tcover1.4.17.2	Tcover1.4.17.4
State	SessionManager.mutex = true;		User login
Expected Output	err == “服务器响应超时.	res == "nil."	res == “用户已登录."
Test Result	Passed	Passed	Passed

- Test Coverage: 3/3 = 100%

1.4.18 Get Session

```
function [res, err] = getSession(obj, sid)
        % Get a session from the pool using given data.
```

```
        % Parameters
        % ----------
        % sid: int64, session’s id.

        % Test function:
        % tests/unitTestBackend/testServer.m/testGetSession

        res = "nil";
        err = obj.checkMutex();
        if err ~= "nil" % Statement Tcover1.4.18.1
            return
        end
        obj.mutex = true;
        res = obj.pool(sid);
        err = "nil";
        obj.mutex = false;
    end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestBackend/testServer.m/testGetSession
- Test Case

	Test Case T1.4.18.1
Input	(sessionId)
Coverage Item	Tcover1.4.18.1
State	SessionManager.mutex = true
Expected Output	err == “服务器相应超时.”
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.4.19 Create Session

```
    function [res, err] = createSession(obj, uid, dev)
        % Create a session using given data.

        % Parameters
        % ----------
        % uid: string, user's id.
        % dev: string, device tag of user's login request.

        % Test function:
        % tests/unitTestBackend/testServer.m/testCreateSession
```

```
        res = "nil";
        err = obj.checkMutex();
        if err ~= "nil" % Statement Tcover1.4.19.1
            return
        end
        obj.mutex = true;
        obj.count = obj.count + 1;
        session = Session(uid, obj.count, dev);
        obj.pool(end + 1) = session;
        res = session.sessionId;
        err = "nil";
        obj.mutex = false;
    end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestBackend/testServer.m/testCreateSession
- Test Case

	Test Case T1.4.19.1
Input	(“114514”,”atm”)
Coverage Item	Tcover1.4.19.1
State	SessionManager.mutex = true
Expected Output	err == “服务器相应超时.”
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.4.20 Destroy Session

```
function err = destroySession(obj, sid)
        % Destroy a session using given data.

        % Parameters
        % ----------
        % sid: string, session’s id.

        % Test function:
        % tests/unitTestBackend/testServer.m/testDestroySession

        err = obj.checkMutex();
        if err ~= "nil" % Statement Tcover1.4.20.1
            return
        end
        obj.mutex = true;
```

```
            obj.pool(sid).deactive();
            err = "nil";
            obj.mutex = false;
        end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestBackend/testServer.m/testDestroySession
- Test Case

	Test Case T1.4.20.1
Input	(sessionId)
Coverage Item	Tcover1.4.20.1
State	SessionManager.mutex = true
Expected Output	err == “服务器相应超时.”
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.5 Frontend Matview Test

1.5.1 Matview Router Register

```
        function register(obj, path, component)
            % Add component to routes. Invoke onCreated to set template and
            % link callbacks. Then set template to invisible.

            % Test function:
            % tests/testFrontend/testMatview.m/testRouterRegister

            % Statement Tcover1.5.1.1
            component.onCreated();
            component.template.Visible = "off";
            component.template.Enable = "off";
            obj.routes(path) = component;
        end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testMatview.m/testRouterRegister
- Test Case

	Test Case T1.5.1.1
Input	“/test”, APPTransfer()
Coverage Item	Tcover1.5.1.1
State	-----
Expected Output	The component is registered with “/test” key.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.5.2 Matview Router Destroy

```
function destroy(obj)
    % Set component on current location to invisible, invoke
    % onDestroy hook.

    % Test function:
    % tests/testFrontend/testMatview.m/testRouterDestroy

    if isKey(obj.routes, obj.location)
        % Branch Tcover1.5.2.1
        component = obj.routes(obj.location);
        component.template.Visible = "off";
        component.template.Enable = "off";
        component.onDestroy();
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testMatview.m/testRouterDestroy
- Test Case

	Test Case T1.5.2.1
Input	-----
Coverage Item	Tcover1.5.2.1
State	A component as APPTransfer is registered with key “/test”
Expected Output	The component is unmounted.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.5.3 Matview Router Mount

```
function mount(obj, args)
    % Set component on current location to visible, invoke script
    % and render to initialize the component. Then invoke onMounted
    % hook.

    % Parameters
    % ----------
    % args: instance of containers.Map, will be passed as parameters
    %    when invoking script.
```

```
        % Test function:
        % tests/unitTestFrontend/testMatview.m/testRouterDestroy

        if isKey(obj.routes, obj.location)
            % Branch Tcover1.5.3.1
            component = obj.routes(obj.location);
            component.script(args);
            component.template.Visible = "on";
            component.template.Enable = "on";
            component.render();
            component.onMounted();
        end
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testMatview.m/testRouterMount
- Test Case

	Test Case T1.5.3.1
Input	containers.Map()
Coverage Item	Tcover1.5.3.1
State	A component as APPTransfer is registered with key “/test”
Expected Output	The component is mounted.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.5.4 Matview Router Push

```
    function push(obj, path, args)
        % Change location of the matview application.

        % Parameters
        % ----------
        % path: string, usually starts with '/'.
        % args: instance of containers.Map, will be passed as
        %   parameters when invoking mount.

        % Test function:
        % tests/unitTestFrontend/testMatview.m/testRouterPush

        if isKey(obj.routes, path)
            % Branch Tcover1.5.4.1
            obj.destroy();
            if ~exist('args', 'var')
                % Branch Tcover1.5.4.2
```

```
                args = containers.Map();
            end
            obj.location = path;
            obj.mount(args);
        end
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testMatview.m/testRouterMount
- Test Case

	Test Case T1.5.4.1
Input	"/test"
Coverage Item	Tcover1.5.4.1, Tcover1.5.4.2
State	A component as APPTransfer is registered with key "/test"
Expected Output	The component is mounted.
Test Result	Passed

- Test Coverage: 2/2 = 100%

1.5.5 Matview Core Use

```
    function use(app, routes)
        % Register routes in large scale.

        % Parameters
        % ----------
        % routes: n x 2 cell, in the order of {path, component}
        % path: string, usually starts with '/'.
        % component: function handle for subclasses of
        %     MatviewComponent.

        % Test function:
        % tests/unitTestFrontend/testMatview.m/testCoreUse

        % Statement Tcover1.5.5.1
        for i = 1:length(routes)
            path = routes{i}{1};
            component = routes{i}{2};
            app.router.register(path, component(app));
        end
    end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testMatview.m/testCoreUse
- Test Case

	Test Case T1.5.5.1

Input	{{"/test", @APPTransfer}}
Coverage Item	Tcover1.5.5.1
State	-----
Expected Output	The component is registered
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.5.6 Matview Core Callback

```
function res = callback(app, cbk)
    % Add MATLAB APP wrapper to raw callback functions.

    % Parameters
    % ----------
    % cbk: function handle.

    % Statement Tcover1.5.6.1
    res = createCallbackFcn(app.window, cbk, true);
end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testMatview.m/testCoreUse
- Test Case

	Test Case T1.5.6.1
Input	@()NaN
Coverage Item	Tcover1.5.6.1
State	-----
Expected Output	The function is turned into a callback.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.6 Frontend Hardware Test

1.6.1 Hardware In

```
function in(obj)
    % Set to input mode, and clear buffer.

    % Test function:
    % tests/unitTestFrontend/testHardware.m/testHardwareIn

    % Statement Tcover1.6.1.1
```

```
            obj.resume();
            obj.buffer = "";
        end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testHardwareIn
- Test Case

	Test Case T1.6.1.1
Input	-----
Coverage Item	Tcover1.6.1.1
State	-----
Expected Output	The hardware is set to input mode.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.6.2 Hardware Pending

```
        function pending(obj)
            % Set to pending mode.

            % Test function:
            % tests/unitTestFrontend/testHardware.m/testHardwarePending

            % Statement Tcover1.6.2.1
            obj.status = "pending";
            obj.device.FontColor = [0 0 0];
        end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testHardwarePending
- Test Case

	Test Case T1.6.2.1
Input	-----
Coverage Item	Tcover1.6.2.1
State	-----
Expected Output	The hardware is set to pending mode.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.6.3 Hardware Out

```
        function out(obj)
            % Set to output mode.
```

```
        % Test function:
        % tests/unitTestFrontend/testHardware.m/testHardwareOut

        % Statement Tcover1.6.3.1
        obj.status = "off";
        obj.device.FontColor = [1 0 0];
    end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testHardwareOut
- Test Case

	Test Case T1.6.3.1
Input	-----
Coverage Item	Tcover1.6.3.1
State	-----
Expected Output	The hardware is set to output mode.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.6.4 Hardware Resume

```
    function resume(obj)
        % Set to resume mode, but don't clear buffer.

        % Test function:
        % tests/unitTestFrontend/testHardware.m/testHardwareResume

        % Statement Tcover1.6.4.1

        obj.status = "on";
        obj.device.FontColor = [0.3922 0.8314 0.0745];
    end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testHardwareResume
- Test Case

	Test Case T1.6.4.1
Input	-----
Coverage Item	Tcover1.6.4.1
State	-----
Expected Output	The hardware is set to resume mode.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.6.5 Card Slot Inject

```
function inject(obj)
    % react to user's input when the card slot is in on
    % mode, will accept input from a msg box and filt
    % invalid inputs.

    % Test function:
    % tests/unitTestFrontend/testHardware.m/testCardSlotInject

    if obj.status == "on"
        % Branch Tcover1.6.5.1
        clearCharts("InputChart")
        box = inputChart();
        box.init("请输入卡号:", @(answer)asyncOnIn(obj, answer));
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testCardSlotInject
- Test Case

	Test Case T1.6.5.1
Input	-----
Coverage Item	Tcover1.6.5.1
State	-----
Expected Output	An input chart is created.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.6.6 Card Slot Eject

```
function eject(obj)
    % react to user's input when the card slot is in off
    % mode, will pop out a msg to inform the user.

    % Test function:
    % tests/unitTestFrontend/testHardware.m/testCardSlotEject

    if obj.status == "off"
        % Branch Tcover1.6.6.1
        box = msgChart();
        box.init("弹出卡成功.");
        obj.in();
```

```
            obj.onOut();
        end
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testCardSlotEject
- Test Case

	Test Case T1.6.6.1
Input	-----
Coverage Item	Tcover1.6.6.1
State	-----
Expected Output	An message chart is created.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.6.7 Cash Draw Deposit

```
    function deposit(obj)
        % react to user's input when the cash draw is in on
        % mode, will accept input from a msgbox and will filt
        % invalid inputs.

        % Test function:
        % tests/unitTestFrontend/testHardware.m/testCashDrawDeposit

        if obj.status == "on"
            % Branch Tcover1.6.7.1
            clearCharts("InputChart");
            box = inputChart();
            box.init("请输入金额:", @(answer)asyncOnIn(obj, answer));
        end

    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testCashDrawDeposit
- Test Case

	Test Case T1.6.7.1
Input	-----
Coverage Item	Tcover1.6.7.1
State	-----
Expected Output	An input chart is created.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.6.8 Cash Draw Withdraw

```
function withdraw(obj)
    % react to user's input when the cash draw is in off
    % mode, will pop out a msgbox to inform the user.

    % Test function:
    % tests/unitTestFrontend/testHardware.m/testCashDrawWithdraw

    if obj.status == "off"
        % Branch Tcover1.6.8.1
        if obj.buffer ~= ""
            % Branch Tcover1.6.8.2
            output = sprintf("%.2f", double(obj.buffer));
            box = msgChart();
            box.init("取出现金" + output + "元.");
        end
        obj.pending();
        obj.onOut();
    end
end

end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testCashDrawWithdraw
- Test Case

	Test Case T1.6.8.1
Input	-----
Coverage Item	Tcover1.6.8.1, Tcover1.6.8.2
State	-----
Expected Output	An message chart is created.
Test Result	Passed

- Test Coverage: 2/2 = 100%

1.6.9 Keyboard Press

```
function press(obj, str)
    % React to user's input when the keyboard is in on mode,
    % '*' for backspace and '#' for confirm.

    % Test function:
    % tests/unitTestFrontend/testHardware.m/testKeyboardPress
```

```
        if obj.status == "on"
            % Branch Tcover1.6.9.1
            if str == "*"
                % Branch Tcover1.6.9.2
                if strlength(obj.buffer) >= 1
                    % Branch Tcover1.6.9.3
                    len = strlength(obj.buffer);
                    obj.buffer = char(obj.buffer);
                    obj.buffer = obj.buffer(1:len - 1);
                    obj.buffer = string(obj.buffer);
                end
            elseif str ~= "#"
                % Branch Tcover1.6.9.4
                obj.buffer = obj.buffer + str;
            end
            obj.onSync();
            if str == "#"
                % Branch Tcover1.6.9.5
                obj.pending();
                obj.onIn();
            end
        end
    end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testHardware.m/testKeyboardPress
- Test Case

	Test Case T1.6.9.1	Test Case T1.6.9.2	Test Case T1.6.9.3
Input	“*”	“1”	“#”
Coverage Item	Tcover1.6.9.1, Tcover1.6.9.2, Tcover1.6.9.3	Tcover1.6.9.1, Tcover1.6.9.4	Tcover1.6.9.1, Tcover1.6.9.5
State	Keyboard buffer == “123”	Keyboard buffer == “123”	Keyboard buffer == “123”
Expected Output	The buffer is backspaced to “12”.	The buffer is added to “1231”	The keyboard is in pending mode.
Test Result	Passed	Passed	Passed

- Test Coverage: 5/5 = 100%

1.7 Frontend Composable Test

1.7.1 Clear Charts

```
function clearCharts(name)
    % Clear figures with given name.

    % Parameters
    % ----------
    % tag: string, used as the name of the figure.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testClearCharts

    if ~exist('name', 'var') % Branch Tcover1.7.1.1
        name = "";
    end
    figs = findall(groot,'Type','figure');
    for i = 1:length(figs)
        if figs(i).Name == name || name == ""
            % Branch Tcover1.7.1.2
            delete(figs(i))
        end
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testClearCharts
- Test Case

	Test Case T1.7.1.1
Input	-----
Coverage Item	Tcover1.7.1.1, Tcover1.7.1.2
State	A msgChart is created.
Expected Output	All the chats are closed.
Test Result	Passed

- Test Coverage: 2/2 = 100%

1.7.2 Create Timer

```
function createTimer(tag, handler)
    % Create a Timer using given data.
```

```
    % Parameters
    % ----------
    % tag: string, used as the tag of the timer.
    % handler: function handle, used as the callback of the timer.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testCreateTimer

    clock = timer("Tag", tag, "ExecutionMode", "fixedDelay", "BusyMode",
"queue" ...
            , "Period", 1, "TimerFcn", handler ...
        ); % Statement Tcover1.7.2.1
    start(clock);
end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testCreateTimer
- Test Case

	Test Case T1.7.2.1
Input	“test”, @(~, ~, ~)NaN
Coverage Item	Tcover1.7.2.1
State	-----
Expected Output	A timer with “test” tag is created
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.7.3 Do Health Check

```
function err = doHealthCheck(obj)
    % Check the health status of the system.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testDoHealthCheck

    try
        server = evalin("base", 'server');
    catch
        err = "服务器连接错误.";
        return
    end
```

```
    err = "nil";
    if isKey(obj.controller.state, "stock") && isKey(obj.controller.state,
"space")
        % Branch Tcover1.7.3.1
        if obj.controller.state("stock") <= 0 &&
obj.controller.state("space") <= 0
            % Branch Tcover1.7.3.2
            err = "客户端已暂停服务.";
        end
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testDoHealthCheck
- Test Case

	Test Case T1.7.3.1
Input	-----
Coverage Item	Tcover1.7.3.1, Tcover1.7.3.2
State	ATM is run out of withdrawal money and deposition space.
Expected Output	"客户端已暂停服务."
Test Result	Passed

- Test Coverage: 2/2 = 100%

1.7.4 Do Logout

```
function doLogout(obj)
    % Do the logout sequences.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testDoLogout

    if isKey(obj.controller.state, "status") && isKey(obj.controller.state,
"sessionId")
        % Branch Tcover1.7.4.1
        if obj.controller.state("status")== "on"
            % Branch Tcover1.7.4.2
            obj.controller.post("/logout",
obj.controller.state("sessionId"));
            obj.controller.state("sessionId") = 0;
            obj.controller.state("status") = "off";
```

```
            if isprop(obj.controller, 'cardSlot')
                % Branch Tcover1.7.4.3
                obj.controller.cardSlot.out();
            end
            obj.controller.router.push("/entry");
        end
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testDoLogout
- Test Case

	Test Case T1.7.4.1
Input	-----
Coverage Item	Tcover1.7.4.1, Tcover1.7.4.2, Tcover1.7.4.3
State	ATM is logged in by users.
Expected Output	ATM is logged out.
Test Result	Passed

- Test Coverage: 3/3 = 100%

1.7.5 Do Parse Bill

```
function billTable = doParseBill(content, condition)
    % Generate bill table from cell-like bill content.

    % Parameters
    % ----------
    % content: n x 5 cell, raw content of the bill.
    % condition: string, condition for filt use.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testDoParseBill

    if ~exist('condition', 'var')
        % Branch Tcover1.7.5.1
        condition = "";
    end

    timeList = {};
    changeList = {};
    savingsList = {};
    actionList = {};
    len = size(content);
    len = len(1);
```

```
    row = 0;
    for i = len:-1:1
        behavior = content{i, 2};
        behavior = split(behavior, ":");
        action = behavior(1);
        target = behavior(2);
        change = content{i, 3};
        date = content{i, 5};
        savings = content{i, 4};
        if condition == "" || condition == action
            % Branch Tcover1.7.5.2
            row = row + 1;
            timeList{row} = datestr(datevec(date), "YYYY-mm-dd HH:MM");
            changeList{row} = "￥" + sprintf("%.2f", double(change) / 100);
            savingsList{row} = "￥" + sprintf("%.2f", double(savings) /
100);
            if action == "transfer"
                % Branch Tcover1.7.5.3
                actionList{row} = "向" + target + "转账";
            elseif action == "receive"
                % Branch Tcover1.7.5.4
                actionList{row} =  "从" + target + "收款";
            elseif action == "return"
                % Branch Tcover1.7.5.5
                actionList{row} =  "转账失败系统自动退款";
            elseif action == "init"
                % Branch Tcover1.7.5.6
                actionList{row} = "创建账号";
            elseif action == "deposit"
                % Branch Tcover1.7.5.7
                actionList{row} = "从 ATM 机存款";
            elseif action == "withdraw"
                % Branch Tcover1.7.5.8
                actionList{row} = "从 ATM 机取款";
            end
        end
    end
    billTable = table(timeList', actionList', changeList', savingsList' ...
            , 'VariableNames', ["时间", "操作", "金额", "存款"]);
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testDoParseBill
- Test Case

	Test Case T1.7.5.1

Input	{
	'2124099443931732', 'init:2124099443931732', 0, 0, '2022-06-03 10:14:12';
	'2124099443931732', 'deposit:2124099443931732', 100000, 100000, '2022-06-03 10:19:35';
	'2124099443931732', 'transfer:2124099443931732', 10000, 90000, '2022-06-03 10:20:32';
	'2124099443931732', 'withdraw:2124099443931732', 10000, 80000, '2022-06-03 10:23:04';
	'2124099443931732', 'receive:2124099443931732', 80000, 0, '2022-06-03 10:23:55';
	'2124099443931732', 'return:2124099443931732', 20000, 20000, '2022-06-03 19:05:40'
	}

Coverage Item	Tcover1.7.5.1, Tcover1.7.5.2, Tcover1.7.5.3, Tcover1.7.5.4, Tcover1.7.5.5, Tcover1.7.5.6, Tcover1.7.5.7, Tcover1.7.5.8
State	-----

Expected Output	target = {
	{'2022-06-03 19:05'} {["转账失败系统自动退款"]} {["￥200.00"]} {["￥200.00"]}
	{'2022-06-03 10:23'} {["从 2124099443931732 收款"]} {["￥800.00"]} {["￥0.00"]}
	{'2022-06-03 10:23'} {["从 ATM 机取款"]} {["￥100.00"]} {["￥800.00"]}
	{'2022-06-03 10:20'} {["向 2124099443931732 转账"]} {["￥100.00"]} {["￥900.00"]}
	{'2022-06-03 10:19'} {["从 ATM 机存款"]} {["￥1000.00"]} {["￥1000.00"]}
	{'2022-06-03 10:14'} {["创建账号"]} {["￥0.00"]} {["￥0.00"]}
	};
	target = cell2table(target, "VariableNames", ["时间", "操作", "金额", "存款"]);

Test Result	Passed

- Test Coverage: 8/8= 100%

1.7.6 Do Reparse Bill

```
function doReparseBill(obj)
    % Filt the bill content from the original bill table.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent or its subclasses.
```

```
    % Test function:
    % tests/unitTestFrontend/testComposable.m/testDoReparseBill

    if ~isKey(obj.data, "bill") % Branch Tcover1.7.6.1
        return
    end
    action = obj.controller.window.BillDropdownAction.Value;
    bill = obj.data("bill");
    if action == "初始化" % Branch Tcover1.7.6.2
        billTable = doParseBill(bill, "init");
    elseif action == "取款" % Branch Tcover1.7.6.3
        billTable = doParseBill(bill, "withdraw");
    elseif action == "存款" % Branch Tcover1.7.6.4
        billTable = doParseBill(bill, "deposit");
    elseif action == "转账" % Branch Tcover1.7.6.5
        billTable = doParseBill(bill, "transfer");
    elseif action == "收款" % Branch Tcover1.7.6.6
        billTable = doParseBill(bill, "receive");
    elseif action == "退款" % Branch Tcover1.7.6.7
        billTable = doParseBill(bill, "return");
    else % Branch Tcover1.7.6.8
        billTable = doParseBill(bill);
    end
    order = obj.controller.window.BillDropdownOrder.Value;
    if order == "从新到旧" % Branch Tcover1.7.6.9
        obj.controller.window.BillTableBill.Data = ...
            sortrows(billTable, "时间", 'descend');
    else % Branch Tcover1.7.6.10
        obj.controller.window.BillTableBill.Data = ...
            sortrows(billTable, "时间", 'ascend');
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testDoReparseBill
- Test Case

	Test Case T1.7.6.1	Test Case T1.7.6.2	Test Case T1.7.6.3
Input	-----	-----	-----
Coverage Item	Tcover1.7.6.1	Tcover1.7.6.2, Tcover1.7.6.9	Tcover1.7.6.3, Tcover1.7.6.10

State	obj = ATMBill();
	obj.data has no key named “bill”;
		obj = ATMBill();
		obj.data("bill") = {
		'2124099443931732',
		'init:2124099443931732',
		0, 0, '2022-06-03
			obj = ATMBill();
			obj.data("bill") = {
			'2124099443931732',
			'init:2124099443931732', 0,
			0, '2022-06-03 10:14:12';

		10:14:12'; '2124099443931732', 'deposit:21240994439317 32', 100000, 100000, '2022-06-03 10:19:35'; '2124099443931732', 'transfer:21240994439317 32', 10000, 90000, '2022- 06-03 10:20:32'; '2124099443931732', 'withdraw:2124099443931 732', 10000, 80000, '2022- 06-03 10:23:04'; '2124099443931732', 'receive:212409944393173 2', 80000, 0, '2022-06-03 10:23:55'; '2124099443931732', 'return:212409944393173 2', 20000, 20000, '2022- 06-03 19:05:40'};	'2124099443931732', 'deposit:212409944393173 2', 100000, 100000, '2022- 06-03 10:19:35'; '2124099443931732', 'transfer:212409944393173 2', 10000, 90000, '2022- 06-03 10:20:32'; '2124099443931732', 'withdraw:21240994439317 32', 10000, 80000, '2022- 06-03 10:23:04'; '2124099443931732', 'receive:212409944393173 2', 80000, 0, '2022-06-03 10:23:55'; '2124099443931732', 'return:2124099443931732' , 20000, 20000, '2022-06- 03 19:05:40'};
Expected Output	Displayed bill is empty.	Displayed bill only has init, sorted from new to old.	Displayed bill only has withdraw, sorted from old to new.
Test Result	Passed	Passed	Passed
	Test Case T1.7.6.4	Test Case T1.7.6.5	Test Case T1.7.6.6
Input	-----	-----	-----
Coverage Item	Tcover1.7.6.4, Tcover1.7.6.10	Tcover1.7.6.5, Tcover1.7.6.10	Tcover1.7.6.6, Tcover1.7.6.10

State	obj = ATMBill();
	obj.data("bill") = {
	'2124099443931732',
	'init:2124099443931732',
	0, 0, '2022-06-03
	10:14:12';
	'2124099443931732',
	'deposit:21240994439317
	32', 100000, 100000,
	'2022-06-03 10:19:35';
	'2124099443931732',
	'transfer:21240994439317
	32', 10000, 90000, '2022-
	06-03 10:20:32';
	'2124099443931732',
		obj = ATMBill();
		obj.data("bill") = {
		'2124099443931732',
		'init:2124099443931732',
		0, 0, '2022-06-03
		10:14:12';
		'2124099443931732',
		'deposit:21240994439317
		32', 100000, 100000,
		'2022-06-03 10:19:35';
		'2124099443931732',
		'transfer:21240994439317
		32', 10000, 90000, '2022-
		06-03 10:20:32';
		'2124099443931732',
			obj = ATMBill();
			obj.data("bill") = {
			'2124099443931732',
			'init:2124099443931732', 0,
			0, '2022-06-03 10:14:12';
			'2124099443931732',
			'deposit:212409944393173
			2', 100000, 100000, '2022-
			06-03 10:19:35';
			'2124099443931732',
			'transfer:212409944393173
			2', 10000, 90000, '2022-
			06-03 10:20:32';
			'2124099443931732',
			'withdraw:21240994439317

	'withdraw:2124099443931732', 10000, 80000, '2022-06-03 10:23:04'; '2124099443931732', 'receive:2124099443931732', 80000, 0, '2022-06-03 10:23:55'; '2124099443931732', 'return:2124099443931732', 20000, 20000, '2022-06-03 19:05:40'};	'withdraw:2124099443931732', 10000, 80000, '2022-06-03 10:23:04'; '2124099443931732', 'receive:2124099443931732', 80000, 0, '2022-06-03 10:23:55'; '2124099443931732', 'return:2124099443931732', 20000, 20000, '2022-06-03 19:05:40'};	32', 10000, 80000, '2022-06-03 10:23:04'; '2124099443931732', 'receive:2124099443931732', 80000, 0, '2022-06-03 10:23:55'; '2124099443931732', 'return:2124099443931732', 20000, 20000, '2022-06-03 19:05:40'};
Expected Output	Displayed bill only has deposit, sorted from old to new.	Displayed bill only has transfer, sorted from new to old.	Displayed bill only has receive, sorted from old to new.
Test Result	Passed	Passed	Passed
	Test Case T1.7.6.7	Test Case T1.7.6.8	
Input	-----	-----	
Coverage Item	Tcover1.7.6.7, Tcover1.7.6.10	Tcover1.7.6.8, Tcover1.7.6.10	
State	obj = ATMBill(); obj.data("bill") = { '2124099443931732', 'init:2124099443931732', 0, 0, '2022-06-03 10:14:12'; '2124099443931732', 'deposit:2124099443931732', 100000, 100000, '2022-06-03 10:19:35'; '2124099443931732', 'transfer:2124099443931732', 10000, 90000, '2022-06-03 10:20:32'; '2124099443931732', 'withdraw:2124099443931732', 10000, 80000, '2022-06-03 10:23:04'; '2124099443931732', 'receive:2124099443931732', 80000, 0, '2022-06-03 10:23:55'; '2124099443931732', 'return:2124099443931732', 20000, 20000, '2022-	obj = ATMBill(); obj.data("bill") = { '2124099443931732', 'init:2124099443931732', 0, 0, '2022-06-03 10:14:12'; '2124099443931732', 'deposit:2124099443931732', 100000, 100000, '2022-06-03 10:19:35'; '2124099443931732', 'transfer:2124099443931732', 10000, 90000, '2022-06-03 10:20:32'; '2124099443931732', 'withdraw:2124099443931732', 10000, 80000, '2022-06-03 10:23:04'; '2124099443931732', 'receive:2124099443931732', 80000, 0, '2022-06-03 10:23:55'; '2124099443931732', 'return:2124099443931732', 20000, 20000, '2022-	

	06-03 19:05:40'};	06-03 19:05:40'};	
Expected Output	Displayed bill only has return, sorted from old to new.	Displayed bill has all the data, sorted from new to old.	
Test Result	Passed	Passed	

- Test Coverage: 10/10= 100%

1.7.7 Do Resort Bill

```
function doResortBill(obj)
    % Resort the bill order from a bill table.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent or its subclasses.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testDoResortBill

    if ~isKey(obj.data, "bill") % Branch Tcover1.7.7.1
        return
    end
    order = obj.controller.window.BillDropdownOrder.Value;
    if order == "从新到旧"
        % Branch Tcover1.7.7.2
        obj.controller.window.BillTableBill.Data = ...
            sortrows(obj.controller.window.BillTableBill.Data, "时间",
'descend');
    else % Branch Tcover1.7.7.3
        obj.controller.window.BillTableBill.Data = ...
            sortrows(obj.controller.window.BillTableBill.Data, "时间",
'ascend');
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testDoResortBill
- Test Case

	Test Case T1.7.7.1	Test Case T1.7.7.2	Test Case T1.7.7.3
Input	-----	-----	-----
Coverage Item	Tcover1.7.7.1	Tcover1.7.7.2,	Tcover1.7.7.3,

State	obj = ATMBill();
	obj.data has no key named “bill”;
		obj = ATMBill();
		obj.data("bill") = {
		'2124099443931732',
			obj = ATMBill();
			obj.data("bill") = {
			'2124099443931732',

		'init:2124099443931732', 0, 0, '2022-06-03 10:14:12'};	'init:2124099443931732', 0, 0, '2022-06-03 10:14:12'; '2124099443931732'};
Expected Output	Displayed bill is empty.	Displayed bill is sorted from new to old.	Displayed bill is sorted from old to new.
Test Result	Passed	Passed	Passed

- Test Coverage: 3/3 = 100%

1.7.8 Make Error Log

```
function makeErrorLog(position, level, content)
    % make error log on console using given data.

    % Parameters
    % ----------
    % position: string, where the error occurred.
    % level: int64, danger level of the error.
    % content: string | exception, content of the error.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testMakeErrorLog

    % Statement Tcover1.7.8.1
    fprintf("An Error has occurred at " + position);
    fprintf("\nCode: 400\n");
    fprintf("Level: " + string(level));
    fprintf("\nContent: ");
    fprintf(content);
    fprintf("\nTime: " + datestr(datetime('now')));
    fprintf("\n");
end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testMakeErrorLog
- Test Case

	Test Case T1.7.8.1
Input	-----
Coverage Item	Tcover1.7.8.1
State	-----
Expected Output	An error log is printed to the console.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.7.9 Make Post

```
function res = makePost(dev, head, sid, keySet, valueSet)
    % make post to the backend using given data.

    % Parameters
    % ----------
    % dev: string, device property of the post.
    % head: string, head property of the post.
    % sid: int64, sessionId property of the post.
    % keySet: cell | array, a set of keys as the key of data property
    %   of the post.
    % valueSet: cell | array with the same size to keySet, a set of
    %   values as the value of data property of the post.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testMakePost

    try
        server = evalin("base", 'server');
    catch
        res = Response("/", containers.Map("msg", "服务器连接错误."),
"server", 0, 400);
        return;
    end
    if exist('keySet', 'var') && exist('valueSet', 'var')
        % Branch Tcover1.7.9.1
        body = containers.Map(keySet, valueSet);
    else % Branch Tcover1.7.9.2
        body = containers.Map();
    end
    req = Request(head, body, dev, sid);
    res = server.post(req);
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testMakePost
- Test Case

	Test Case T1.7.9.1	Test Case T1.7.9.2
Input	"atm", "/123", 0, "haha", "hawhaw"	"atm", "/123", 0
Coverage Item	Tcover1.7.9.1	Tcover1.7.9.2
State	-----	-----
Expected Output	A response with code 400.	A response with code 400.
Test Result	Passed	Passed

- Test Coverage: 2/2 = 100%

1.7.10 Remove Timer

```
function removeTimer(tag)
    % Remove timers with given tag.

    % Parameters
    % ----------
    % tag: string, used as the tag of the timer to be deleted.

    % Test function:
    % tests/unitTestFrontend/testComposable.m/testRemoveTimer

    if ~exist('tag', 'var') || tag == "" % Branch Tcover1.7.10.1
        clocks = timerfind();
    else % Branch Tcover1.7.10.2
        clocks = timerfind("Tag", tag);
    end
    if ~isempty(clocks) % Branch Tcover1.7.10.3
        stop(clocks);
        delete(clocks);
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testRemoveTimer
- Test Case

	Test Case T1.7.10.1	Test Case T1.7.10.2
Input	-----	“test”
Coverage Item	Tcover1.7.10.1, Tcover1.7.10.3	Tcover1.7.10.2, Tcover1.7.10.3
State	A timer with tag “test” is created.	A timer with tag “test” is created.
Expected Output	All timers are removed.	The timer is removed.
Test Result	Passed	Passed

- Test Coverage: 3/3 = 100%

1.8 Frontend ATM Test

1.8.1 Do Login

```
function doLogin(obj)
    % Validate user's input of the password, then start the login
    % sequences.
```

```
    % Parameters
    % ----------
    % obj: instance of MatviewComponent.

    % Test function:
    % tests/unitTestFrontend/testATM.m/testDoLogin
    obj.controller.window.LoginLabelHint.Visible = "off";
    password = obj.data("password");
    userId = obj.data("userId");
    if strlength(password) < 6 %Branch Tcover1.8.1.1
        obj.controller.window.LoginLabelHint.Visible = "on";
        obj.controller.window.LoginLabelHint.Text = "密码至少六位.";
        obj.controller.keyboard.resume();
        return;
    end
    res = obj.controller.post("/login", 0, {'userId', 'password'}, {userId,
password});
    if res.code == 400 %Branch Tcover1.8.1.2
        if res.body("msg") == "密码错误." %Branch Tcover1.8.1.3
            obj.controller.window.LoginLabelHint.Visible = "on";
            obj.controller.window.LoginLabelHint.Text = res.body("msg");
            obj.controller.keyboard.resume();
        else                %Branch Tcover1.8.1.4
            obj.controller.router.push('/error', containers.Map("notice",
res.body("msg")));
        end
    else %Branch Tcover1.8.1.5
        obj.controller.state("sessionId") = res.sessionId;
        obj.controller.state("status") = "on";
        obj.controller.router.push("/main");
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testATM.m/testDoLogin
- Test Case

	Test Case T1.8.1.1	Test Case T1.8.1.2	Test Case T1.8.1.3
Input	-----	-----	-----
Coverage Item	Tcover1.8.1.1	Tcover1.8.1.2 Tcover1.8.1.3	Tcover1.8.1.5
State	userId = "3495524663390897" password = “”	userId = "3495524663390897" password = “1222222”	userId = "3495524663390897" password = “654321”
Expected Output	ATM give error “密码至	ATM give error "密码错	ATM login successfully

	少六位."	误."	
Test Result	Passed	Passed	Passed

- Test Coverage: 4/5 = 80%

1.8.2 Do Deposit

```
function doDeposit(obj)
    % Validate user's input of the cash, then start the login
    % sequences.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent.

    % Test function:
    % tests/unitTestFrontend/testATM.m/testDoDeposit
    obj.controller.window.DepositLabelHint.Visible = "off";
    cash = obj.data("cash"); % 1
    if cash == ""   %Branch Tcover 1.8.2.1
        obj.controller.window.DepositLabelHint.Visible = "on";
        obj.controller.window.DepositLabelHint.Text = "存入金额不能为零.";
        obj.controller.cashDraw.resume();
        return
    end
    if cash == ""%Branch Tcover 1.8.2.2
        money = 0;
    else        %Branch Tcover 1.8.2.3
        money = double(cash) * 100;   % 0.01
    end
    if mod(money, 10000) %Branch Tcover 1.8.2.4
        obj.controller.cashDraw.out();
        obj.controller.cashDraw.buffer = cash;
        obj.controller.cashDraw.onOut = @()obj.controller.cashDraw.resume();
        obj.controller.window.DepositLabelHint.Visible = "on";
        obj.controller.window.DepositLabelHint.Text = "存款金额只能是整百.";
        return
    end
    if money > obj.controller.state("space") %Branch Tcover 1.8.2.5
        obj.controller.window.DepositLabelHint.Visible = "on";
        obj.controller.window.DepositLabelHint.Text = "客户端空间不足.";
        obj.controller.cashDraw.out();
        obj.controller.cashDraw.buffer = cash;
        obj.controller.cashDraw.onOut = @()obj.controller.cashDraw.resume();
```

```
        return;
    end
    maxDepositSpaceOnce = 10000000; % 10w max deposit per time.
    if money > maxDepositSpaceOnce %Branch Tcover 1.8.2.6
        obj.controller.window.DepositLabelHint.Visible = "on";
        obj.controller.window.DepositLabelHint.Text = "单次存款不能超过十万.";
        obj.controller.cashDraw.out();
        obj.controller.cashDraw.buffer = cash;
        obj.controller.cashDraw.onOut = @()obj.controller.cashDraw.resume();
        return
    end
    res = obj.controller.post("/updateData",
obj.controller.state("sessionId"), ...
        ["action", "content"], {"deposit", containers.Map("value", money)});
    if res.code == 400 %Branch Tcover 1.8.2.7
        if res.body("msg") == "登录状态无效." || res.body("msg") == "登录状态
已过期." %Branch Tcover 1.8.2.8
            doLogout(obj);
        end
        obj.controller.router.push('/error', containers.Map("notice",
res.body("msg")));
        obj.controller.cashDraw.buffer = cash;
        obj.controller.cashDraw.onOut = @()NaN;
        obj.controller.cashDraw.out();
        return
    end
    bill = obj.data("bill");
    obj.controller.state("space") = obj.controller.state("space") - money;
        if obj.controller.state("space") <= 0 %Branch Tcover 1.8.2.9
            obj.controller.logError("No Enough Cash Space in ATM.");
        end
    obj.controller.cashDraw.pending();
    obj.controller.window.DepositLabelHint.Visible = "on";
    obj.controller.window.DepositLabelHint.Text = "存款成功.";
    obj.controller.router.push("/result", ...
            containers.Map(["action", "savings"], ...
            ["存款", sprintf("%.2f", double(bill.savings + money) /
100)]) ...
        );
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testATM.m/testDoDeposit
- Test Case

	Test Case T1.8.2.1	Test Case T1.8.2.2	Test Case T1.8.2.3

Input	-----	-----	-----
Coverage Item	Tcover1.8.2.1	Tcover1.8.2.3 Tcover1.8.2.4	Tcover1.8.2.3 Tcover1.8.2.5
State	cash=""	cash="123"	cash=" 1000000000000"
Expected Output	ATM give error "存入金额不能为零."	ATM give error "存款金额只能是整百."	ATM give an error "客户端空间不足."
Test Result	Passed	Passed	Passed
	Test Case T1.8.2.4	Test Case T1.8.2.5	Test Case T1.8.2.6
Input	-----	-----	-----
Coverage Item	Tcover1.8.2.3 Tcover1.8.2.6	Tcover1.8.2.3 Tcover1.8.2.7 Tcover1.8.2.9	Tcover1.8.2.3 Tcover1.8.2.9
State	cash="200000"	cash="200"	cash=1000 ATM has Login
Expected Output	ATM give error "单次存款不能超过十万."	ATM give error and logout	ATM deposit successfully and print log "No Enough Cash Space in ATM."
Test Result	Passed	Passed	Passed

- Test Coverage: 8/9 = 89%

1.8.3 Do Withdraw

```
function doWithdraw(obj)
    % Validate user's input of the value, then start the withdraw
    % sequences.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent.

    % Test function:
    % tests/unitTestFrontend/testATM.m/testDoWithdraw
    obj.controller.window.WithdrawLabelWithdrawHint.Visible = "off";
    cash = obj.data("cash");
    if cash == "" %Branch Tcover1.8.3.1
        obj.controller.window.WithdrawLabelWithdrawHint.Visible = "on";
        obj.controller.window.WithdrawLabelWithdrawHint.Text = "取款金额不能
为零.";
        obj.controller.keyboard.resume();
        return
    end
```

```
    if cash == ""%Branch Tcover1.8.3.2
        money = 0;
    else %Branch Tcover1.8.3.3
        money = double(cash) * 100;   % 0.01
    end
    if mod(money, 10000)%Branch Tcover1.8.3.4
        obj.controller.window.WithdrawLabelWithdrawHint.Visible = "on";
        obj.controller.window.WithdrawLabelWithdrawHint.Text = "取款金额只能
是整百.";
        obj.controller.keyboard.resume();
        return
    end
    bill = obj.data("bill");
    if money > bill.savings%Branch Tcover1.8.3.5
        obj.controller.window.WithdrawLabelWithdrawHint.Visible = "on";
        obj.controller.window.WithdrawLabelWithdrawHint.Text = "金额超过存款
总额.";
        obj.controller.keyboard.resume();
        return
    elseif money > bill.withdrew%Branch Tcover1.8.3.6
        obj.controller.window.WithdrawLabelWithdrawHint.Visible = "on";
        obj.controller.window.WithdrawLabelWithdrawHint.Text = "金额超过当日
上限.";
        obj.controller.keyboard.resume();
        return
    elseif money > obj.controller.state("stock")%Branch Tcover1.8.3.7
        obj.controller.window.WithdrawLabelWithdrawHint.Visible = "on";
        obj.controller.window.WithdrawLabelWithdrawHint.Text = "客户端现金不
足.";
        obj.controller.keyboard.resume();
        return
    else %Branch Tcover1.8.3.8
        res = obj.controller.post("/updateData",
obj.controller.state("sessionId"), ...
            ["action", "content"], {"withdraw", containers.Map("value",
money)});
        if res.code == 400 %Branch Tcover1.8.3.9
            if res.body("msg") == "登录状态无效." || res.body("msg") == "登录
状态已过期."%Branch Tcover1.8.3.10
                doLogout(obj);
            end
            obj.controller.router.push('/error', containers.Map("notice",
res.body("msg")));
            return
```

```
        end
        obj.controller.cashDraw.out();
        obj.controller.cashDraw.buffer = string(double(cash));
        obj.controller.state("stock") = obj.controller.state("stock") -
money;
        if obj.controller.state("stock") <= 0 %Branch Tcover1.8.3.11
            obj.controller.logError("No Enough Cash Stock in ATM.");
        end
        obj.controller.window.WithdrawLabelWithdrawHint.Visible = "on";
        obj.controller.window.WithdrawLabelWithdrawHint.Text = "取款成功.";
        obj.controller.router.push("/result", ...
                containers.Map(["action", "savings"], ...
                ["取款", sprintf("%.2f", double(bill.savings - money) /
100)]) ...
            );
    end
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testATM.m/testDoWithdraw
- Test Case

	Test Case T1.8.3.1	Test Case T1.8.3.2	Test Case T1.8.3.3
Input	-----	-----	-----
Coverage Item	Tcover1.8.3.1	Tcover1.8.3.3 Tcover1.8.3.4	Tcover1.8.3.3 Tcover1.8.3.5
State	cash=""	cash="123"	cash=" 100" bill.savings=1000
Expected Output	ATM give error "取款金额不能为零."	ATM give error "取款金额只能是整百."	ATM give an error "金额超过存款总额"
Test Result	Passed	Passed	Passed
	Test Case T1.8.3.4	Test Case T1.8.3.5	Test Case T1.8.3.6
Input	-----	-----	-----
Coverage Item	Tcover1.8.3.3 Tcover1.8.3.6	Tcover1.8.3.3 Tcover1.8.3.7	Tcover1.8.3.3 Tcover1.8.3.8 Tcover1.8.3.9 Tcover1.8.3.10
State	cash="100" bill.savings=10000000000 bill.withdrew=1000	cash="100" bill.withdrew=10000000000 bill.savings=10000000000 stock=100	cash=1000 stock=100000000
Expected Output	ATM give error "金额超过当日上限."	ATM give error "客户端现金不足.	ATM give an error and logout

		”	
Test Result	Passed	Passed	Passed
	Test Case T1.8.3.7		
Input	-----		
Coverage Item	Tcover1.8.3.3 Tcover1.8.3.8 Tcover1.8.3.11		
State	cash=”1000” stock=100000 ATM has login		
Expected Output	ATM withdraw successfully and give an log “No Enough Cash Stock in ATM.”		
Test Result	Passed		

- Test Coverage: 10/11 = 91%

1.8.4 Do Transfer

```
function doTransfer(obj)
    % Validate user's input of the target and value, then start the
    % transfer sequences.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent.

    % Test function:
    % tests/unitTestFrontend/testATM.m/testDoTransfer
    obj.controller.window.TransferLabelTransferHint.Visible = "off";
    bill = obj.data("bill");
    target = obj.data("target");
    if strlength(target) ~= 16 %Branch Tcover1.8.4.1
        obj.controller.window.TransferLabelTransferHint.Visible = "on";
        obj.controller.window.TransferLabelTransferHint.Text = "无效的收款账
户.";
        obj.controller.keyboard.resume();
        return;
    end
    uid = bill.content{1,1};
    if uid == target  %Branch Tcover1.8.4.2
        obj.controller.window.TransferLabelTransferHint.Visible = "on";
```

```
        obj.controller.window.TransferLabelTransferHint.Text = "收款账户不能为自己.";
        obj.controller.keyboard.resume();
        return;
    end
    res = obj.controller.post("/login", 0, {'userId', 'password'}, {target, ""});
    if res.code == 400 && res.body("msg") ~= "密码错误." && res.body("msg") ~= "用户已登录." %Branch Tcover1.8.4.3
        if res.body("msg") ~= "用户不存在或银行卡无效." && res.body("msg") ~= "用户已被冻结." %Branch Tcover1.8.4.4
            obj.controller.router.push('/error', containers.Map("notice", res.body("msg")));
            return;
        else %Branch Tcover1.8.4.5
            obj.controller.window.TransferLabelTransferHint.Visible = "on";
            obj.controller.window.TransferLabelTransferHint.Text = "收款账户不存在或已被冻结.";
            obj.controller.keyboard.resume();
            return;
        end
    end
    cash = obj.data("cash");
    if cash == "" || cash == "0" || cash == "0." || cash == "0.0" || cash == "0.00" %Branch Tcover1.8.4.6
        obj.controller.window.TransferLabelTransferHint.Visible = "on";
        obj.controller.window.TransferLabelTransferHint.Text = "转账金额不能为零.";
        obj.controller.keyboard.resume();
        return
    end
    if isempty(regexp(cash, "^\d+(\.\d{0,2})?$", "match")) %Branch Tcover1.8.4.7
        obj.controller.window.TransferLabelTransferHint.Visible = "on";
        obj.controller.window.TransferLabelTransferHint.Text = "无效的转账金额.";
        return;
    end
    money = double(cash) * 100;   % 0.01
    if money > bill.savings %Branch Tcover1.8.4.8
        obj.controller.window.TransferLabelTransferHint.Visible = "on";
        obj.controller.window.TransferLabelTransferHint.Text = "金额超过存款总额.";
        obj.controller.keyboard.resume();
```

```
        return
    end
    if money > bill.transferred %Branch Tcover1.8.4.9
        obj.controller.window.TransferLabelTransferHint.Visible = "on";
        obj.controller.window.TransferLabelTransferHint.Text = "金额超过当日
上限.";
        obj.controller.keyboard.resume();
        return
    end
    res = obj.controller.post("/updateData",
obj.controller.state("sessionId"), ...
        ["action", "content"], {"transfer", containers.Map(["value",
"target"], {money, target})});
    if res.code == 400 %Branch Tcover1.8.4.10
        if res.body("msg") == "登录状态无效." || res.body("msg") == "登录状态
已过期." %Branch Tcover1.8.4.11
            doLogout(obj);
        end
        obj.controller.router.push('/error', containers.Map("notice",
res.body("msg")));
        return
    end
    obj.controller.window.TransferLabelTransferHint.Visible = "on";
    obj.controller.window.TransferLabelTransferHint.Text = "转账成功.";
    obj.controller.router.push("/result", ...
            containers.Map(["action", "savings"], ...
            ["转账", sprintf("%.2f", double(bill.savings - money) /
100)]) ...
        );
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testATM.m/testDoTransfer
- Test Case

	Test Case T1.8.4.1	Test Case T1.8.4.2	Test Case T1.8.4.3
Input	-----	-----	-----
Coverage Item	Tcover1.8.4.1	Tcover1.8.4.2	Tcover1.8.4.5
State	target= '1232421412'	target= '5927735005791540' userId='5927735005791540'	target= '5927735005791540' userId='5927735005791541'
Expected Output	ATM give error “无效的收款账户. ”	ATM give error "收款账户不能为自己."	ATM give an error “收款账户不存在或已被冻结.”

Test Result	Passed	Passed	Passed
	Test Case T1.8.4.4	Test Case T1.8.4.5	Test Case T1.8.4.6
Input	-----	-----	-----
Coverage Item	Tcover1.8.4.6	Tcover1.8.4.7	Tcover1.8.4.8
State	target="1754966885128041" cash=""	cash="as113"	cash="100" bill.savings="1000"
Expected Output	ATM give error "转账金额不能为零."	ATM give error "无效的转账金额."	ATM give error "金额超过存款总额."
Test Result	Passed	Passed	Passed
	Test Case T1.8.4.7	Test Case T1.8.4.8	
Input	-----		
Coverage Item	Tcover1.8.4.9	Tcover1.8.4.10 Tcover1.8.4.11	
State	cash="10000" savings=10000000 transferred=1000	cash="1000" savings=10000000 transferred=100000	
Expected Output	ATM give error "金额超过当日上限."	ATM give an error and logout	
Test Result	Passed	Passed	

- Test Coverage: 9/11 = 82%

1.9 Frontend APP Test

1.9.1 Do Save Bill

```
function doSaveBill(obj)
    % Save the bill to local file system, will pop up a msgbox to
    % inform the user.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent.

    % Test function:
    % tests/unitTestFrontend/testAPP.m/testDoSaveBill

    % Statement Tcover1.9.1.1
    billTable = obj.controller.window.BillTableBill.Data;
    try
```

```
        writetable(billTable, "bill.xlsx");
        box = msgChart();
        box.init("下载账单成功.");
    catch
        box = msgChart();
        box.init("下载账单失败.");
    end
end
```

- Coverage Criteria: Statement coverage
- Test Function: tests/unitTestFrontend/testAPP.m/testDoSaveBill
- Test Case

	Test Case T1.9.1.1
Input	-----
Coverage Item	Tcover1.9.1.1
State	A bill is displayed on the ATM.
Expected Output	A file named bill.xlsx is downloaded to local file system.
Test Result	Passed

- Test Coverage: 1/1 = 100%

1.9.2 Do Login

```
function doLogin(obj)
    % Validate user's input of the account, password and captcha,
    % then start the login sequences.

    % Parameters
    % ----------
    % obj: instance of MatviewComponent.

    % Test function:
    % tests/unitTestFrontend/testAPP.m/testDoLogin

    uid = obj.data("userId");
    password = obj.data("password");
    captcha = obj.data("captcha");
    if uid == "" || password == "" || ...
      (obj.controller.window.LoginInputLoginCaptcha.Enable == "on" &&
captcha == "")
        % Branch Tcover1.9.2.1
        obj.controller.window.LoginLabelHintLogin.Visible = "on";
        obj.controller.window.LoginLabelHintLogin.Text = "请输入完整的登录信
息.";
        return;
```

```
    end
    if strlength(uid) ~= 16 || isempty(regexp(uid, "^\d+$", 'match'))
        % Branch Tcover1.9.2.2
        obj.controller.window.LoginLabelHintLogin.Visible = "on";
        obj.controller.window.LoginLabelHintLogin.Text = "无效的银行卡号.";
        return;
    end
    if isempty(regexp(password, "^\d+$", 'match'))
        % Branch Tcover1.9.2.3
        obj.controller.window.LoginLabelHintLogin.Visible = "on";
        obj.controller.window.LoginLabelHintLogin.Text = "密码应仅含有数字.";
        return;
    end
    if strlength(password) < 6
        % Branch Tcover1.9.2.4
        obj.controller.window.LoginLabelHintLogin.Visible = "on";
        obj.controller.window.LoginLabelHintLogin.Text = "密码至少六位.";
        return;
    end
    res = obj.controller.post("/login", 0, {'userId', 'password'}, {uid,
""});
    if res.code == 400 && res.body("msg") ~= "密码错误." && res.body("msg")
~= "APP 登录未被激活."
        % Branch Tcover1.9.2.5
        if res.body("msg") == "用户不存在或银行卡无效." || res.body("msg") ==
"用户已被冻结."
            % Branch Tcover1.9.2.6
            obj.controller.window.LoginLabelHintLogin.Visible = "on";
            obj.controller.window.LoginLabelHintLogin.Text =
res.body("msg");
            return;
        else % Branch Tcover1.9.2.7
            obj.controller.router.push("/error", containers.Map("notice",
res.body("msg")))
            return;
        end
    end
    if obj.controller.window.LoginInputLoginCaptcha.Enable == "on"
        % Branch Tcover1.9.2.8
        if isempty(regexp(captcha, "^\d+$", 'match'))
            % Branch Tcover1.9.2.9
            obj.controller.window.LoginLabelHintCaptcha.Visible = "on";
            obj.controller.window.LoginLabelHintCaptcha.Text = "激活码应仅含
有数字.";
```

```
            return;
        end
        if strlength(captcha) < 6
            % Branch Tcover1.9.2.10
            obj.controller.window.LoginLabelHintCaptcha.Visible = "on";
            obj.controller.window.LoginLabelHintCaptcha.Text = "激活码至少六
位.";
            return;
        end
        res = obj.controller.post("/login", 0, ...
            {'userId', 'password', 'clientKey'}, {uid, password, captcha});
        if res.code == 400 % Branch Tcover1.9.2.11
            if res.body("msg") == "密码错误."
                % Branch Tcover1.9.2.12
                obj.controller.window.LoginLabelHintLogin.Visible = "on";
                obj.controller.window.LoginLabelHintLogin.Text =
res.body("msg");
                return;
            end
            if res.body("msg") == "激活码错误."
                % Branch Tcover1.9.2.13
                obj.controller.window.LoginLabelHintCaptcha.Visible = "on";
                obj.controller.window.LoginLabelHintCaptcha.Text =
res.body("msg");
                return;
            else % Branch Tcover1.9.2.14
                obj.controller.router.push('/error',
containers.Map("notice", res.body("msg")));
                return;
            end
        end
        obj.controller.window.LoginLabelHintCaptcha.Visible = "on";
        obj.controller.window.LoginLabelHintCaptcha.Text = "APP 激活成功.";
    else % Branch Tcover1.9.2.15
        res = obj.controller.post("/login", 0, {'userId', 'password'}, {uid,
password});
        if res.code == 400 % Branch Tcover1.9.2.16
            if res.body("msg") == "密码错误."
                % Branch Tcover1.9.2.17
                obj.controller.window.LoginLabelHintLogin.Visible = "on";
                obj.controller.window.LoginLabelHintLogin.Text =
res.body("msg");
                return;
            elseif res.body("msg") == "APP 登录未被激活."
```

```
                % Branch Tcover1.9.2.18
                obj.controller.window.LoginLabelHintLogin.Visible = "on";
                obj.controller.window.LoginLabelHintLogin.Text =
res.body("msg");
                obj.controller.window.LoginLabelLoginCaptcha.Visible = "on";
                obj.controller.window.LoginInputLoginCaptcha.Visible = "on";
                obj.controller.window.LoginInputLoginCaptcha.Enable = "on";
                return;
            else % Branch Tcover1.9.2.19
                obj.controller.router.push('/error',
containers.Map("notice", res.body("msg")));
                return;
            end
        end
    end
    obj.controller.window.LoginLabelHintLogin.Visible = "on";
    obj.controller.window.LoginLabelHintLogin.Text = "登陆成功.";
    obj.controller.state("sessionId") = res.sessionId;
    obj.controller.state("status") = "on";
    obj.controller.router.push("/main");
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testDoLogin
- Test Case

	Test Case T1.9.2.1	Test Case T1.9.2.2	Test Case T1.9.2.3
Input	-----	-----	-----
Coverage Item	Tcover1.9.2.1	Tcover1.9.2.2	Tcover1.9.2.3
State	obj = APPLogin(); obj.data(“userId”) = “”; obj.data(“password”) = “”; obj.data(“captcha”) = “”;	obj = APPLogin(); obj.data(“userId”) = “123”; obj.data(“password”) = “a”; obj.data(“captcha”) = “a”;	obj = APPLogin(); obj.data(“userId”) = “1754966885128041”; obj.data(“password”) = “a”; obj.data(“captcha”) = “a”;
Expected Output	This login is rejected.	This login is rejected.	This login is rejected.
Test Result	Passed	Passed	Passed
	Test Case T1.9.2.4	Test Case T1.9.2.5	Test Case T1.9.2.6
Input	-----	-----	-----
Coverage Item	Tcover1.9.2.4	Tcover1.9.2.5, Tcover1.9.2.6	Tcover1.9.2.8, Tcover1.9.2.9
State	obj = APPLogin(); obj.data(“userId”) = “1754966885128041”; obj.data(“password”) =	obj = APPLogin(); obj.data(“userId”) = “1754966885128042”; obj.data(“password”) =	obj = APPLogin(); obj.data(“userId”) = “1754966885128041”; obj.data(“password”) =

	“123”; obj.data(“captcha”) = “a”;	“654321”; obj.data(“captcha”) = “a”; captcha is required.	“654321”; obj.data(“captcha”) = “a”; captcha is required.
Expected Output	This login is rejected.	This login is rejected.	This login is rejected.
Test Result	Passed	Passed	Passed
	Test Case T1.9.2.7	Test Case T1.9.2.8	Test Case T1.9.2.9
Input	-----	-----	-----
Coverage Item	Tcover1.9.2.8, Tcover1.9.2.10	Tcover1.9.2.8, Tcover1.9.2.11, Tcover1.9.2.12	Tcover1.9.2.8, Tcover1.9.2.11, Tcover1.9.2.13
State	obj = APPLogin(); obj.data(“userId”) = “1754966885128041”; obj.data(“password”) = “654321”; obj.data(“captcha”) = “123”; captcha is required.	obj = APPLogin(); obj.data(“userId”) = “1754966885128042”; obj.data(“password”) = “654322”; obj.data(“captcha”) = “123456”; captcha is required.	obj = APPLogin(); obj.data(“userId”) = “5927735005791540”; obj.data(“password”) = “654321”; obj.data(“captcha”) = “123455”; captcha is required.
Expected Output	This login is rejected.	This login is rejected.	This login is rejected.
Test Result	Passed	Passed	Passed
	Test Case T1.9.2.10	Test Case T1.9.2.11	
Input	-----	-----	
Coverage Item	Tcover1.9.2.15, Tcover1.9.2.16, Tcover1.9.2.17	Tcover1.9.2.15, Tcover1.9.2.16, Tcover1.9.2.18	
State	obj = APPLogin(); obj.data(“userId”) = “1754966885128041”; obj.data(“password”) = “654322”; obj.data(“captcha”) = “123455”;	obj = APPLogin(); obj.data(“userId”) = “5927735005791540”; obj.data(“password”) = “654322”; obj.data(“captcha”) = “123455”;	
Expected Output	This login is rejected.	This login is rejected.	
Test Result	Passed	Passed	

- Test Coverage: 17/19 = 89%

1.9.3 Do Transfer

```
function doTransfer(obj)
    % Validate user's input of the target and value, then start the
```

```
% transfer sequences.

% Parameters
% ----------
% obj: instance of MatviewComponent.

% Test function:
% tests/testFrontend/testAPP.m/testDoTransfer

target = obj.data("target");
cash = obj.data("cash");
if target == "" || cash == ""
    % Branch Tcover1.9.3.1
    obj.controller.window.TransferLabelHint.Visible = "on";
    obj.controller.window.TransferLabelHint.Text = ...
        "请输入完整的转账信息.";
    return;
end
if cash == "0" || cash == "0." ...
        || cash == "0.0" || cash == "0.00"
    % Branch Tcover1.9.3.2
    obj.controller.window.TransferLabelHint.Visible = "on";
    obj.controller.window.TransferLabelHint.Text = "转账金额不能为零.";
    return
end
if strlength(target) ~= 16 || ...
        isempty(regexp(target, "^\d+$", "match"))
    % Branch Tcover1.9.3.3
    obj.controller.window.TransferLabelHint.Visible = "on";
    obj.controller.window.TransferLabelHint.Text = "无效的收款账号.";
    return;
end
bill = obj.data("bill");
uid = bill.content{1, 1};
if uid == target
    % Branch Tcover1.9.3.4
    obj.controller.window.TransferLabelHint.Visible = "on";
    obj.controller.window.TransferLabelHint.Text = ...
        "收款账户不能为自己.";
    return;
end
res = obj.controller.post("/login", 0, ...
    {'userId', 'password'}, {target, ""});
if res.code == 400 && res.body("msg") ~= "密码错误." ...
```

```
        && res.body("msg") ~= "用户已登录." && ...
        res.body("msg") ~= "APP 登录未被激活."
    % Branch Tcover1.9.3.5
    if res.body("msg") ~= "用户不存在或银行卡无效." ...
            && res.body("msg") ~= "用户已被冻结."
        % Branch Tcover1.9.3.6
        obj.controller.router.push('/error', ...
            containers.Map("notice", res.body("msg")));
        return;
    else % Branch Tcover1.9.3.7
        obj.controller.window.TransferLabelHint.Visible = "on";
        obj.controller.window.TransferLabelHint.Text = ...
            "收款账户不存在或已被冻结.";
        return;
    end
end
if isempty(regexp(cash, "^\d+(\.\d{0,2})?$", "match"))
    % Branch Tcover1.9.3.8
    obj.controller.window.TransferLabelHint.Visible = "on";
    obj.controller.window.TransferLabelHint.Text = "无效的转账金额.";
    return;
end
money = double(cash) * 100;
if money > bill.savings % Branch Tcover1.9.3.9
    obj.controller.window.TransferLabelHint.Visible = "on";
    obj.controller.window.TransferLabelHint.Text = "金额超过存款总额.";
    return;
end
if money > bill.transferred % Branch Tcover1.9.3.10
    obj.controller.window.TransferLabelHint.Visible = "on";
    obj.controller.window.TransferLabelHint.Text = "金额超过当日上限.";
    return;
end
res = obj.controller.post("/updateData", ...
    obj.controller.state("sessionId"), ...
    ["action", "content"], {"transfer", ...
    containers.Map(["value", "target"], {money, target})});
if res.code == 400 % Branch Tcover1.9.3.11
    if res.body("msg") == "登录状态无效." ...
        || res.body("msg") == "登录状态已过期."
        % Branch Tcover1.9.3.12
        doLogout(obj);
    end
    obj.controller.router.push('/error', ...
```

```
            containers.Map("notice", res.body("msg")));
        return
    end
    obj.controller.window.TransferLabelHint.Visible = "on";
    obj.controller.window.TransferLabelHint.Text = "转账成功.";
    obj.controller.router.push("/result", ...
            containers.Map(["action", "savings"], ...
            ["转账", sprintf("%.2f", double(bill.savings - money) /
100)]) ...
        );
end
```

- Coverage Criteria: Branch coverage
- Test Function: tests/unitTestFrontend/testComposable.m/testDoTransfer
- Test Case

	Test Case T1.9.3.1	Test Case T1.9.3.2	Test Case T1.9.3.3
Input	-----	-----	-----
Coverage Item	Tcover1.9.3.1	Tcover1.9.3.2	Tcover1.9.3.3
State	obj = APPTransfer(); obj.data(“target”) = “”; obj.data(“cash”) = “”;	obj = APPTransfer(); obj.data(“target”) = “1”; obj.data(“cash”) = “0”;	obj = APPTransfer(); obj.data(“target”) = “1”; obj.data(“cash”) = “1”;
Expected Output	This transfer is rejected.	This transfer is rejected.	This transfer is rejected.
Test Result	Passed	Passed	Passed
	Test Case T1.9.3.4	Test Case T1.9.3.5	Test Case T1.9.3.6
Input	-----	-----	-----
Coverage Item	Tcover1.9.3.4	Tcover1.9.3.5, Tcover1.9.3.7	Tcover1.9.3.8

State	obj = APPTransfer();
	obj.data(“target”) = “5927735005791540”;
	obj.data(“cash”) = “1”;
	uid = 5927735005791540;
		obj = APPTransfer();
		obj.data(“target”) = “5927735005791541”;
		obj.data(“cash”) = “1”;
		uid = 5927735005791540;
			obj = APPTransfer();
			obj.data(“target”) = “1754966885128041”;
			obj.data(“cash”) = “123.456”;
			uid = 5927735005791540;

Expected Output	This transfer is rejected.	This transfer is rejected.	This transfer is rejected.
Test Result	Passed	Passed	Passed
	Test Case T1.9.3.7	Test Case T1.9.3.8	Test Case T1.9.3.9
Input	-----	-----	-----
Coverage Item	Tcover1.9.3.9	Tcover1.9.3.10	Tcover1.9.3.11, Tcover1.9.3.12
State	obj = APPTransfer(); obj.data(“target”) = “1754966885128041”; obj.data(“cash”) = “123.45”;	obj = APPTransfer(); obj.data(“target”) = “1754966885128041”; obj.data(“cash”) = “123.45”;	obj = APPTransfer(); obj.data(“target”) = “1754966885128041”; obj.data(“cash”) = “123.45”;

	savings = 0;
	transferred = 10000;
	uid = 5927735005791540;
		savings = 100000;
		transferred = 0;
		uid = 5927735005791540;
			savings = 100000;
			transferred = 10000;
			uid = 5927735005791540;
			user is not logged in.

Expected Output	This transfer is rejected.	This transfer is rejected.	This transfer is rejected, and user is logged out.
Test Result	Passed	Passed	Passed

- Test Coverage: 11/12 = 91%

2. Integration Test

2.1 ATM and Server

- Test function: tests/integrationTest/bankIntergrationTest.m/testATM
- Test Case

	Test Case T2.1
Operation	ATM login. Check bill and return. Withdraw 100RMB and return to the main page. Deposit 200RMB and check bill. Transfer 100RMB to 3495524663390897 and exit. Logout
Coverage Item	Tcover1.1.1.1, Tcover1.1.1.4 Tcover1.1.2.1, Tcover1.1.3.1, Tcover1.1.3.3, Tcover1.1.3.4, Tcover1.1.3.5, Tcover1.1.3.6, Tcover1.1.3.8, Tcover1.2.2.2, Tcover1.2.2.4, Tcover1.2.4.2, Tcover1.4.1.1, Tcover1.4.1.3, Tcover1.4.1.6, Tcover1.4.1.7, Tcover1.4.2.2,

	Tcover1.4.2.16, Tcover1.4.4.4, Tcover1.4.12.1, Tcover1.4.15.2, Tcover1.4.16.5, Tcover1.4.17.2,
Expected Output	User tried everything ATM can do, and ATM should return the successful result every time.
Test Result	Passed

- Test Coverage: 23/23 = 100%

2.2 APP and Server

- Test function: tests/integrationTest/bankIntergrationTest.m/testAPP
- Test Case

	Test Case T2.2
Operation	App login. Transfer 1.32RMB to 3495524663390897. Check bill. Check account. Logout and exit.
Coverage Item	Tcover1.1.1.1, Tcover1.1.1.4 Tcover1.1.2.1, Tcover1.1.3.5, Tcover1.1.3.6, Tcover1.1.3.8, Tcover1.2.2.2, Tcover1.2.2.4, Tcover1.2.4.2, Tcover1.4.1.1, Tcover1.4.1.3, Tcover1.4.1.6, Tcover1.4.1.7, Tcover1.4.2.2, Tcover1.4.2.16, Tcover1.4.4.4, Tcover1.4.12.1, Tcover1.4.15.2, Tcover1.4.16.5, Tcover1.4.17.2,

Expected Output	User tried everything APP can do, and APP should return the successful result every time.
Test Result	Passed

- Test Coverage: 19/19 = 100%

2.3 ATM, APP and Server

- Test function: tests/integrationTest/bankIntergrationTest.m/testAPPAndATM
- Test Case

	Test Case T2.3
Operation	APP login and check bill. ATM login. ATM withdraw, deposit money. APP check bill. APP and ATM logout.
Coverage Item	Tcover1.1.1.1, Tcover1.1.1.4 Tcover1.1.2.1, Tcover1.1.3.1, Tcover1.1.3.3, Tcover1.1.3.4, Tcover1.2.2.2, Tcover1.2.2.4, Tcover1.2.4.2, Tcover1.4.1.1, Tcover1.4.1.3, Tcover1.4.1.6, Tcover1.4.1.7, Tcover1.4.2.2, Tcover1.4.2.16, Tcover1.4.4.4, Tcover1.4.12.1, Tcover1.4.15.2, Tcover1.4.16.5, Tcover1.4.17.2,
Expected Output	Login APP and ATM, then use ATM to transfer money, then use the APP to check if the bill is the newest.
Test Result	Passed

- Test Coverage: 20/20 = 100%

2.4 ATM, APP without Server

- Test function:
  tests/integrationTest/ bankIntergrationNoBackendTest.m/testNoBackend
- Test Case

	Test Case T2.4
Operation	APP login. ATM login.
Coverage Item	Tcover1.4.2.3,
Expected Output	APP login failed. ATM login failed.
Test Result	Passed

- Test Coverage: 1/1 = 100%

3. Functionality Test

3.1 Use Case “ATM Check Bill”

3.1.1 Test “ATM Check Bill”

- Test Function: tests/functionalityTest/ bankATMFunctionalityTest.m/testCheckBill
- Test Case

	Test Case T3.1.1
Operation	Login through the ATM and check bill
State	
Expected Behavior	Display recent bill
Test Result	Passed

3.2 Use Case “ATM Withdraw”

3.2.1 Test “ATM Withdraw Success”

- Test Function: tests/functionalityTest/bankATMFunctionalityTest.m/testWithdraw
- Test Case

	Test Case T3.2.1

Operation	Login through the ATM and withdraw some money
State	
Expected Behavior	Withdraw success.
Test Result	Passed

3.2.2 Test “ATM Withdraw Over Amount”

- Test Function: tests/functionalityTest/bankATMFunctionalityTest.m/testWithdraw
- Test Case

	Test Case T3.2.2
Operation	Login through the ATM and withdraw too much money
State	
Expected Behavior	Display withdraw too much.
Test Result	Passed

3.2.3 Test “ATM Withdraw Invalid Amount”

- Test Function: tests/functionalityTest/bankATMFunctionalityTest.m/testWithdraw
- Test Case

	Test Case T3.2.3
Operation	Login through the ATM and withdraw invalid amount of money
State	
Expected Behavior	Display withdraw invalid amount.
Test Result	Passed

3.3 Use Case “ATM Deposit”

3.3.1 Test “ATM Deposit Success”

- Test Function: tests/functionalityTest/bankATMFunctionalityTest.m/testDeposit
- Test Case

	Test Case T3.3.1
Operation	Login through the ATM and deposit some money
State	
Expected Behavior	Deposit success.
Test Result	Passed

3.3.2 Test "ATM Deposit Fake Money"

- Test Function: tests/functionalityTest/bankATMFunctionalityTest.m/testDeposit
- Test Case

	Test Case T3.3.2
Operation	Login through the ATM and deposit fake money
State	
Expected Behavior	Display please put in money
Test Result	Passed

3.3.3 Test "ATM Deposit Invalid Amount"

- Test Function: tests/functionalityTest/bankATMFunctionalityTest.m/testDeposit
- Test Case

	Test Case T3.3.3
Operation	Login through the ATM and deposit unacceptable money
State	
Expected Behavior	Display please put valid amount of money
Test Result	Passed

3.4 Use Case "ATM Transfer"

3.4.1 Test "ATM Transfer Success"

- Test Function: tests/functionalityTest/bankATMFunctionalityTest.m/testTransfer
- Test Case

	Test Case T3.4.1
Operation	Login through the ATM and transfer some money.
State	
Expected Behavior	Transfer success
Test Result	Passed

3.4.2 Test "ATM Transfer Over Amount"

- Test Function: tests/functionalityTest/bankATMFunctionalityTest.m/testTransfer
- Test Case

	Test Case T3.4.2

Operation	Login through the ATM and transfer over amount money
State	
Expected Behavior	Display transfer over amount.
Test Result	Passed

3.5 Use Case “APP Check Bill”

3.5.1 Test “APP Check Bill”

- Test Function: tests/functionalityTest/ bankAPPFunctionalityTest.m/testCheckBill
- Test Case

	Test Case T3.6.1
Operation	Login through the APP and check bill
State	
Expected Behavior	Display recent bill
Test Result	Passed

3.6 Use Case “APP Transfer”

3.6.1 Test “APP Transfer Success”

- Test Function: tests/functionalityTest/bankAPPFunctionalityTest.m/testTransfer
- Test Case

	Test Case T3.6.1
Operation	Login through the ATM and transfer some money
State	
Expected Behavior	Transfer sussess
Test Result	Passed

3.6.2 Test “APP Transfer Over Amount”

- Test Function: tests/functionalityTest/bankAPPFunctionalityTest.m/testTransfer
- Test Case

	Test Case T3.6.2
Operation	Login through the ATM and transfer over amount money
State	
Expected Behavior	Display transfer over amount money

Test Result	Passed

3.7 Use Case “APP Check Account”

3.7.1 Test “APP Check Account”

- Test Function:
  tests/functionalityTest/ bankAPPFunctionalityTest.m/testCheckAccount
- Test Case

	Test Case T3.7.1
Operation	Login through the APP and check bill
State	
Expected Behavior	Display account information.
Test Result	Passed

4. Model Checking

This section provides an abstract model built in UPPAAL for model checking purposes. You can find the source files in tests/modelChecking and run it locally using an UPPAAL application (version>=4.1.26).

4.1 Introduction

In our model, we assume that the user can either choose to use the frontend applications: ATM and APP, or exit and cannot stay in the initialize state for any long. In the ATM, users can choose to check their bills, withdraw, deposit and transfer money. And in the APP, users can choose to check their bills, view their profiles, and make transference.
When the user posted a request to the system, the system will response, success or failure implies whether this is a legal request, also we considered the situation when the system was down, the ATM and the App will be able to handle the situation. In such a situation, we assume it will receive an error message, and then it forces the frontend to enter the error state.

4.2 Bank System model

4.2.1 Backend

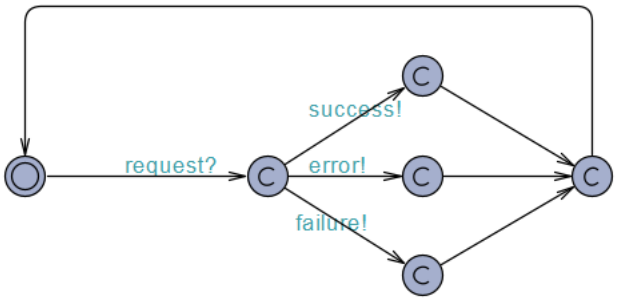

Fig 4.1. Backend

The backend will wait for a request from the frontend applications. When a request is posted to it, it will give one of the following responses: Success, which means this post is valid and system will handle the request. Failure, which means there's something wrong with this request, mainly because the information users typed in are invalid. And Error, which means there's something wrong with the backend, usually it is because the backend is shut down. We added this in order to simulate some unexpected accidents in the process.

4.2.2 ATM

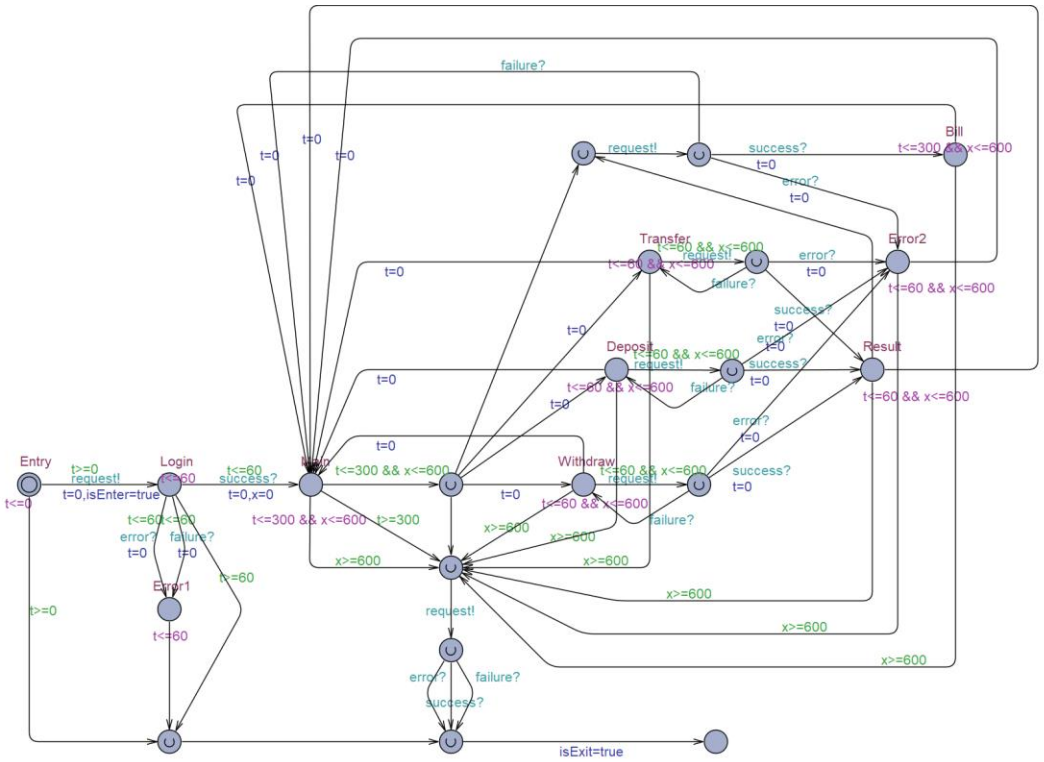

Fig 4.2. ATM

The ATM model consists of all the states a user can do with this machine as well as the initialize state and exit state. Users can directly exit or use the ATM. When using the ATM, they will first enter the machine and try to login. If any failure or error happens, the system will exit. If they successfully login, they will go to the main menu. Notice that users cannot stay in one state for more than 60s except the menu or the bill page, which is 300s. And they cannot stay in the system more than 600s since they login. After entering the main menu, they can choose to check the bill, which will first send a request to the backend. If it was successfully operated, they will be able to see the bill. Otherwise, the ATM will go to the error state or the main menu. Also, they can choose to deposit, withdraw, or transfer money. Before a deposition, withdrawal or transference is made, a request will be posted to the backend, which will check the request, update database, and give a response. If the response is a success, the ATM will go to the result state, and stay at maximum 60 seconds then return to the main menu. If the response is a failure, the ATM will allow user to change their inputs and send a request again until the response is success, or time runs out. And if the response is an error, the ATM will go to error state, and stay at maximum 60 seconds then return to the main menu. At any state, if users' 600 seconds session is overdue, the ATM will force users to logout and enter exit state.

4.2.3 App

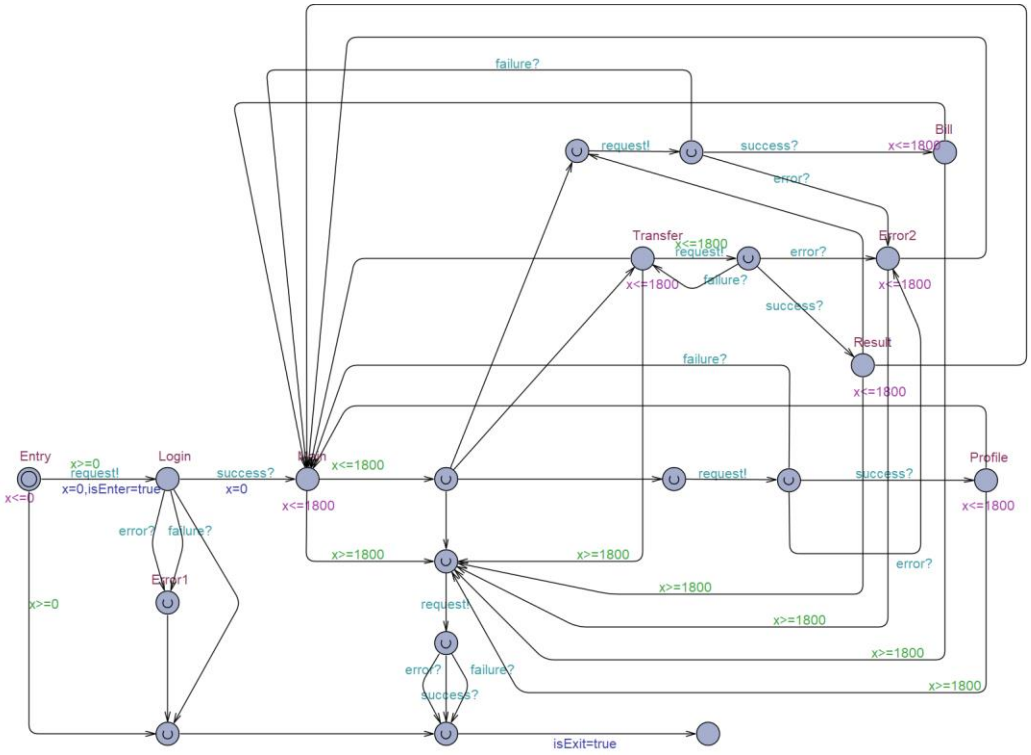

Fig 4.3. App

The APP is similar to the ATM, but it has some differences compare to the ATM. Unlike the ATM, it does not have a time limit on each page, but still have a total time limit. Meanwhile, users can only make transference on the APP and cannot make depositions or withdrawal. Except from these, the rest of APP is the same to the ATM.

4.2.4 Check Properties

4.2.4.1 ATM total time limit.

Property	A[] forall(i: id_t) ATM(i).isEnter imply (ATM(i).x <= 600 || ATM(i).isExit)
Description	The ATM machine has a time limit and will not allow user to use the machine for too long time, so since the user had login, he has only 600s to do his operation, or he will be logout.
Result	Passed

4.2.4.2 ATM can exit anyway and will not deadlock.

Property	A<> forall(i: id_t) ATM(i).isEnter imply ATM(i).isExit
Description	No matter what the user do, there will be a way to exit the system.
Result	Passed

4.2.4.3 ATM 300s page time limit.

Property	A[] forall(i: id_t) ATM(i).isEnter imply (ATM(i).t <= 300 || ATM(i).isExit)
Description	Some page has a limitation of 300s, if the user did not finish his operation, he will be logout. So he will not stay at a page for more than 300s.
Result	Passed

4.2.4.4 APP total time limit.

Property	A[] forall(i: id_t) APP(i).isEnter imply (APP(i).x <= 1800 || APP(i).isExit)
Description	The APP has a time limit and will not allow user to use the APPs for too long time, so since the user had login, he has only 600s to do his operation, or he will be logout.
Result	Passed

4.2.4.5 APP can exit anyway and will not deadlock.

Property	A<> forall(i: id_t) APP(i).isEnter imply APP(i).isExit
Description	No matter what the user do, there will be a way to exit the system.
Result	Passed