ISSUES WE MET

1. Syntax of private functions in controller

before test

```
function initMortgageAmount() {
    var mortgageAmountService =
        new OnlineMortgageCalculationService({
            onlineMortgage: $scope.onlineMortgage
        });
    $scope.recommendedMortgageAmount =
        mortgageAmountService.getWishedMortgageAmount();

    getMortgageAmount();
}
```

after test

```
this.initMortgageAmount = function() {
    var mortgageAmountService =
        new OnlineMortgageCalculationService(
            {onlineMortgage: $scope.onlineMortgage}
        );
    $scope.recommendedMortgageAmount =
        mortgageAmountService.getWishedMortgageAmount();

    this.getMortgageAmount();
};
```

Call private function in test

```
// Before each unit test, instantiate a new instance of the
// controller
beforeEach(inject(function($rootScope, $controller) {
    scope = $rootScope.$new();
    scope.onlineMortgage = onlineMortgageMock;
    onlineMortgageStrategyCtrl =
    $controller('OnlineMortgage.StrategyCtrl', {
        $scope: scope
    });
}));
```

```
var refinancingOrder = {
    wished_mortgage_amount: 900000,
    is_refinancing: true
};

it('refinancingOrder test', function() {
    scope.onlineMortgage = refinancingOrder;
    onlineMortgageStrategyCtrl.initMortgageAmount();
    expect(scope.onlineMortgage.is_refinancing).toBe(true);
    onlineMortgageStrategyCtrl.getMortgageAmount();
    expect(scope.onlineMortgage.wished_mortgage_amount).toEqual(
        recommendedMortgageAmount
    );
});
```

ISSUES WE MET

2. Mocking service that returns class with extended prototype

```
function OnlineMortgageCalculationService($log) {

    function CalculationClass(options) {
        var defaults = {
            LTV_LIMIT_1: {
                MAIN_RESIDENCE: 0.65,
                SECOND_RESIDENCE: 0.65
            },
            RATE_DELTA: 0.01
        };

        angular.extend(defaults, options);

        this.LTV_LIMIT_1 = defaults.LTV_LIMIT_1;
        this.RATE_DELTA = defaults.RATE_DELTA;
        this.onlineMortgage = defaults.onlineMortgage;
        this.constants = getDjangoParam('constants');
    }

    CalculationClass.prototype.getWishedMortgageAmount = function() {

        if (this.onlineMortgage.is_refinancing) {
            return this.onlineMortgage.current_mortgage_amount;
        }

        var totalAssets = parseInt(this.onlineMortgage.total_assets, 1
            valueOfProperty = parseInt(this.onlineMortgage.value_of_pr

        var mortgageAmount = Math.max(
            valueOfProperty * this.getLTV1Limit(),
            valueOfProperty - totalAssets
        );

        return Math.floor(mortgageAmount);
    };
```

```
this.initMortgageAmount = function() {
    var mortgageAmountService =
        new OnlineMortgageCalculationService(
            {onlineMortgage: $scope.onlineMortgage}
        );
    $scope.recommendedMortgageAmount =
        mortgageAmountService.getWishedMortgageAmount();

    this.getMortgageAmount();
};
```

```
// Before each test create mock for getWishedMortgageAmount method in
// OnlineMortgageCalculationService and return recommendedMortgageAmount
beforeEach(module(function($provide) {
    function mockCalculationClass() {}
    mockCalculationClass.prototype.getWishedMortgageAmount = function() {
        return recommendedMortgageAmount;
    };
    $provide.value(
        'OnlineMortgageCalculationService', mockCalculationClass
    );
}));
```