# Part1

*1.1 As a user, I want to search a book in the website so that I can find the information of the book, like the category of the book, the URL to look at it online and the rate from other users.*

My condition of satisfaction is that I give the right book name and click the search button. And there is book related to the name in the API or our database.

*1.2 As a user, I want to create a channel in the website so that I can have a personal place to invite people who also read the book to discuss about the book.*

My condition of satisfaction is that I click the ‘Create a Channel’ button and give the name of the channel.

*1.3 As a host of channel, I can manage the group information and accept joining request to the channel I own so that I can discuss about the book with my friends.*

My condition of satisfaction is that I click the agree button to give others the permission to join in the discussion channel.

*1.4 As a normal user, I can send the joining request to the host of the channel so that the host of channel can know that I want to join his or her channel.*

My condition of satisfaction is that I have already chosen the channel which I want to join in. Then I input some request text and click the ‘send request’ button.

*1.5 As a user, I can rate for the book which I have already searched if I have signed in the web so that I can express my opinion for the book.*

My condition of satisfaction is that I have searched book and signed in web. And then I click the ‘want to rate’ button and input all the rate information, like rate for plot.

*1.6 As a user, I want to get the recommendation about the books which I am maybe interested based on my favorite books in so that I can read new interesting books.*

My condition of satisfaction is that I have logged in the website and have already choose the category I am interested in. Or I have searched some books in the website with an account.

*1.7 As a user, I want to rerate the books which I have rated if I have logged in so that I revise the rating for the books.*

My condition of satisfaction is that I have rated the book and have logged in the web. I click the ‘rerate’ button and input all the rate information, like rate for plot.

*1.8 As a normal user or host of the same channel, I can chat with others who are in the same channel which I have joined in so that group members can chat privately.*

My condition of satisfaction is that I have already joined in the channel. Then I enter the group chatting channel, write what I want to say and click “send”.

# Part2

2.1 *In the ‘Item’ Class, ‘toJsonObject’ method is used to transfer the item object to the Json object.*

Valid equivalence class is normal item object which have all the information required.

Invalid equivalence class is empty item object.

2.2 *In the ‘OpenLibrary’ Class, ‘Search’ method is used to get book information from OpenLibrary API.*

Valid equivalence class is that the input title name is a book name.

Invalid equivalence class is that the input title name is empty String.

2.3 *In the ‘OpenLibrary’ Class, ‘getCategories’ method is used to get the Categories from the Json Object.*

Valid equivalence class is that the input is a normal Json Object which contains categories.

Invalid equivalence class is that the input is a Json Object which don’t have ‘categories’ or have empty ‘categories’.

2.4 *In the ‘OpenLibrary’ Class, ‘getDescribe’ method is used to get the Description from the Json Object.*

Valid equivalence class is that the input is a normal Json Object which contains valid key to get description.

Invalid equivalence class is that the input is a Json Object which don’t have key or have the invalid key which means the book don’t have any description.

2.5 *In the ‘OpenLibrary’ Class, ‘getItemList’ method is used to get all the information of item from the Json Array.*

Valid equivalence class is that the input is a normal Json Array which contains all the information about the book. For boundary conditions, the Array could have 1 or 2 Json Object, because we only take the first Json object.

Invalid equivalence class is that the input is an empty Json Array.

2.6 *In the ‘OpenLibrary’ Class, ‘getAuthor’ method is used to get the Authors from the Json Object.*

Valid equivalence class is that the input is a normal Json Object which contains authors. Since we only take the first author. As for boundary conditions, the author array will contain 1 or 2 authors.

Invalid equivalence class is that the input is a Json Object which don’t have ‘author or have empty ‘author’.

2.7 *In the ‘OpenLibrary’ Class, ‘saveItem’ method is used to save all the information into the database.*

Valid equivalence class is that the input is a normal Item Object which contains all the information of the book.

Invalid equivalence class is that the input is an empty Item Object.

2.8 *In the ‘MysqlTableCreation’ Class, ‘createTables’ method is used to create database tables before the server start.’addFackeData’ method is to add some data for testing.*

There is a main method in this class, which runs before the application starts. So there is no input.

2.9 *In the ‘MysqlRealData’ Class, ‘addRealData’ method is to add some real data searched*

There is a main method in this class, which runs before the application starts. So there is no input.

2.10 *In the ‘GoogleApiLogin’ Class, ‘login’ method cannot be tested independently without frontend. More details of reasons are discussed in part3.*

2.11 *In the ‘BookRecommend’ Class, ‘recommendItems’ method is used to get recommendations to a specific user.*

Valid equivalence class is that the input is a user who has registered in our application.

Invalid equivalence class is that the input is a user who has not registered, so not in our database.

2.12 *In the ‘MysqlConnection’ Class, ‘setFavoriteItems’ method. Associated with ‘testSetFavoriteItems’ method and’testSetFavoriteItemsInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the inputs, userId, is valid, which means the user has logged in.

Invalid equivalence class is that userId is null, which means the user does not log in.

2.13 *In the ‘MysqlConnection’ Class, ‘unsetFavoriteItems’ method. Associated with ‘tesUusetFavoriteItems’ method and’testUnsetFavoriteItemsInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the inputs, userId, is valid, which means the user has logged in.

Invalid equivalence class is that userId is null, which means the user does not log in.

2.14 *In the ‘MysqlConnection’ Class, ‘getFavoriteItemIds’ method. Associated with ‘testGetFavoriteItemIds’ method and’testGetFavoriteItemIdsInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the inputs, userId, is valid, which means the user has logged in.

Invalid equivalence class is that userId is null, which means the user does not log in.

2.15 *In the ‘MysqlConnection’ Class, ‘getFavoriteItems’ method. Associated with ‘testGetFavoriteItems’ method and ’testGetFavoriteItemsInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the inputs, userId, is valid, which means the user has logged in.

Invalid equivalence class is that userId is null, which means the user does not log in.

2.16 *In the ‘MysqlConnection’ Class, ‘getCategories’ method. Associated with ‘testGetCategories’ method and ‘testGetCategoriesInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the input, itemId, is in our database.

Invalid equivalence class is that itemId is not in our database.

2.17 *In the ‘MysqlConnection’ Class, ‘searchItems’ method is to search items in OpenLibarary. Associated with ‘testSaveItem’ method and ‘testSaveItemInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the input, keyword, is a book name.

Invalid equivalence class is that the keyword is an empty string.

2.19 *In the ‘MysqlConnection’ Class, ‘saveItem’ method. Associated with ‘testSaveItem’ method and ‘testSaveItemInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the input is an ‘item’ class object.

Invalid equivalence class is that the input is empty.

2.20 *In the ‘MysqlConnection’ Class, ‘getItemsOnCat’ method. Associated with ‘testGetItemsOnCat’ method and ‘testGetItemsOnCatInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the input is a kind if category that belongs to a book.

Invalid equivalence class is that the input is empty.

2.21 *In the ‘MysqlConnection’ Class, ‘getItemsOnIds’ method. Associated with ‘testGetItemsOnIds’ method and ‘testGetItemsOnIdsInvalid’ method in ‘MysqlConnectionTest’ class.*

Valid equivalence class is that the input is a set of item ids.

Invalid equivalence class is that the input is an empty set.

2.22 *In the ‘MysqlConnection’ Class, ‘createGroup’ method. Associated with ‘testCreateGroup’ method and ‘testCreateGroupInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that all the inputs are not null.

Invalid equivalence class is that any inputs is null.

2.23 *In the ‘MysqlConnection’ Class, ‘joinGroup’ method. Associated with ‘testJoinGroup’ method and ‘testJoinGroupInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the input, group name, exits.

Invalid equivalence class is that the group does not exit.

2.24 *In the ‘MysqlConnection’ Class, ‘getGroupsByHost’ method. Associated with ‘testGetGroupByHost’ method and ‘testGetGroupByHostInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the input, userId is a valid userId.

Invalid equivalence class is that the userId is invalid.

2.25 *In the ‘MysqlConnection’ Class, ‘getGroupsByMember’ method. Associated with ‘testGetGroupByMember’ method and ‘testGetGroupByMemberInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the input, userId is a valid userId.

Invalid equivalence class is that the userId is invalid.

2.26 *In the ‘MysqlConnection’ Class, ‘getJoinMessage’ method. Associated with ‘testGetJoinMessage’ method and ‘testGetJoinMessageInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the input, userId is valid.

Invalid equivalence class is that the userId is invalid.

2.27 *In the ‘MysqlConnection’ Class, ‘ratingBook’ method. Associated with ‘testRatingBook’ method and ‘testRatingBookInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the input, itemId is valid.

Invalid equivalence class is that the itemId is invalid.

2.28 *In the ‘MysqlConnection’ Class, ‘GetRatingAndComments’ method. Associated with ‘testGetRatingAndComments’ method and ‘testGetRatingAndCommentsInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the input, itemId is valid.

Invalid equivalence class is that the itemId is invalid.

2.29 *In the ‘MysqlConnection’ Class, ‘handleJoinRequest’ method. Associated with ‘testHandleJoinRequest’ method and ‘testHandleJoinRequestInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the input, groupName, is valid.

Invalid equivalence class is that the groupName is invalid.

2.30 *In the ‘MysqlConnection’ Class, ‘rejectJoinRequest’ method. Associated with ‘testRejectJoinRequest’ method and ‘testRejectJoinRequestInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the inputs, userId and groupName, are valid.

Invalid equivalence class is that the groupName or groupName are invalid.

2.31 *In the ‘MysqlConnection’ Class, ‘ifRating’ method. Associated with ‘testIfRateing method and ‘testIfRatingInvalid’ methods in ‘MysqlConnectionTest’ class*

Valid equivalence class is that the inputs, bookName and userId, are valid.

Invalid equivalence class is that the bookName or userId are invalid.

Frontend:

Tests for frontend are mainly about check whether the buttons can be clicked and the callback function inside the event listener can work.

Link to test cases:

<https://github.com/pure1017/iDrop/tree/master/iDrop/src/test/java/unit>

Link to frontend test:

<https://github.com/pure1017/iDrop/tree/master/iDrop/src/main/resources/public/assets/js_test>

# Part3

3.1 examlpe1

*We used Google login API in this project. On backend, I need a one-time code generated by google, which includes profile of the user. So when I was testing backend independently, it was hard to mock a one-time code by myself. Thus, I left both the unit test and integration test related to login method and endpoint.*

3.1 examlpe2

*JDBC was used in this project. I used a bunch of catch-exceptions in my code, which is hard to generate. Besides, getters and setters were not tested as well.*