## 1 前言

### 1.1 编写目的

本测试报告记录了“豆汁”书籍交流平台白盒测试的情况，方便开发与测试人员掌握项目健康状况。

### 1.2 测试范围

本轮测试主要针对系统中逻辑相对复杂的 service 层源码。由于代码数量较多，测试以判定覆盖为主。

### 1.3 定义

本节为空。

## 2 测试情况

### 2.1 BookServiceImpl

2.1.1 BookServiceImpl.numberPlus

public boolean numberPlus(Integer bookId){

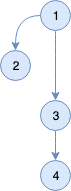
1. if (bookMapper.updateNoteNumberByPrimaryKey\_in(bookId) > 0){

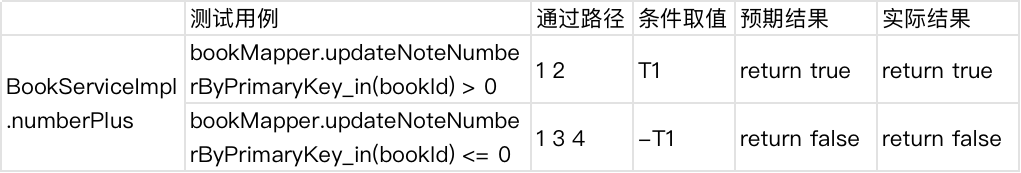
2. return true;

3. }

4. return false;

}





2.1.2 BookServiceImpl.numberMinus

public boolean numberMinus(Integer bookId){

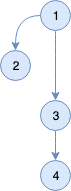
1. if (bookMapper.updateNoteNumberByPrimaryKey\_de(bookId) > 0){

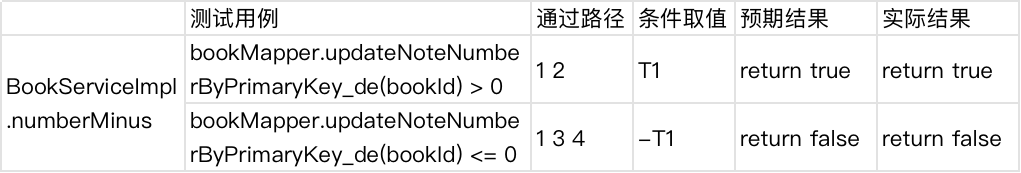
2. return true;

3. }

4. return false;

}





2.1.3 BookServiceImpl.getMyFollowBook

public Page getMyFollowBook(Integer userId,Page page){

1. if (userId == 0){

2. User u = (User) session.getAttribute("user");

3. userId = u.getUserId();

4. }

5. List<FollowBook> followBooks =

followBookMapper.findMyFollowPage(userId,page);

6. List<Book> books = new ArrayList<>();

7. for(FollowBook followBook:followBooks){

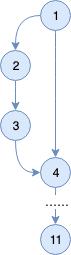
8. books.add(bookMapper.selectByPrimaryKey(followBook.getBook\_id()));

9. }

10. page.setList(books);

11. return page;

}





2.1.4 BookServiceImpl.isfollowBook

public boolean isfollowBook(Integer bookId){

1. try {

2. User u = (User) session.getAttribute("user");

3. FollowBook followBook =

followBookMapper.findFollow(u.getUserId(),bookId);

4. if (followBook == null){

5. return false;

6. }

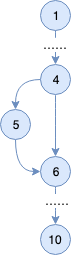
7. }catch (Exception e){

8. e.printStackTrace();

9. }

10. return true;

}





### 2.2 CollectServiceImpl

2.2.1 CollectServiceImpl.getMyCollectFolderByPage

public Page getMyCollectFolderByPage(Integer userId,Page page){

1. if (userId == 0){

2. User u = (User)session.getAttribute("user");

3. userId = u.getUserId();

4. }

5. List<CollectionFolderModel> collectionFolderModels =

collectionFolderMapper.selectByUserPage(userId,page);

6. for (CollectionFolderModel collectionFolderModel:collectionFolderModels){

7. collectionFolderModel.setNumber(collectMapper.

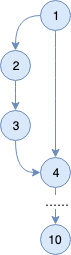
selectCountByFolderId(collectionFolderModel.getFolderId()));

8. }

9. page.setList(collectionFolderModels);

10. return page;

}





### 2.3 NoteBookServiceImpl

2.3.1 NoteBookServiceImpl.userCreateNotebook

public boolean userCreateNotebook(User user,Integer bookId){

1. NoteBook noteBook = new NoteBook();

2. noteBook.setAuthor\_id(user.getUserId());

3. noteBook.setBelongBook\_id(bookId);

4. noteBook.setCreateTime(new Timestamp(new Date().getTime()));

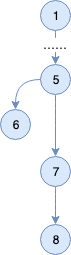
5. if (noteBooksMapper.addNoteBoook(noteBook) > 0){

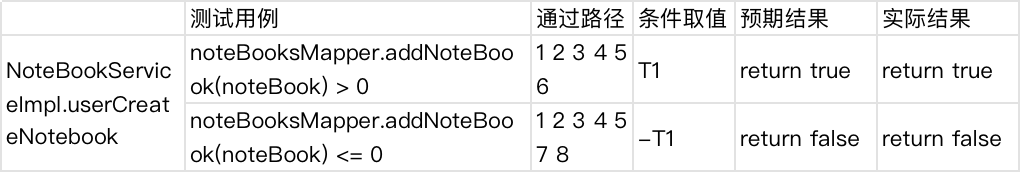
6. return true;

7. }

8. return false;

}





2.3.2 NoteBookServiceImpl.getNoteBookByuserId

public Page getNoteBookByuserId(Integer userId, Page page){

1. if (userId == 0){

2. User u = (User)session.getAttribute("user");

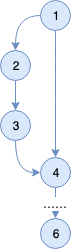
3. userId = u.getUserId();

4. }

5. page.setList(noteBooksMapper.selectByUserIdPage(userId,page));

6. return page;

}





### 2.4 UserServiceImpl

2.4.1 UserServiceImpl.UserSignUp

public boolean UserSignUp(User user){

1. User u = userMapper.SelectByAccount(user.getAccount());

2. if (u == null){

3. userMapper.addUser(user);

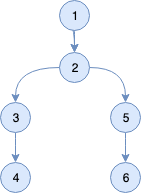
4. return true;

5. }else {

6. return false;

7. }

}





2.4.2 UserServiceImpl.VerifyAdmit

public boolean VerifyAdmit(User user){

1. User u = userMapper.SelectByAccountAndPassword(

user.getAccount(),user.getPassword());

2. if (u != null){

3. session.setAttribute("user",u);

4. User user1 = (User) session.getAttribute("user");

5. System.out.println(user1.getUserId());

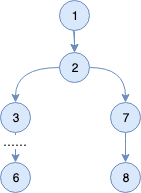
6. return true;

7. }else {

8. return false;

9. }

}





2.4.3 UserServiceImpl.updateAvatar

public String updateAvatar(MultipartFile file){

1. User user = (User) session.getAttribute("user");

2. String[] fileName = file.getOriginalFilename().split("\\.");

3. String suffix = fileName[fileName.length-1];

4. String fileSaveName = user.getUserId()+ "-" +

System.currentTimeMillis() +"."+suffix;

5. try {

6. File outDir =new File(MyFileUtil.getResourcesUrl()+"/static/avatar");

7. if(!outDir.exists())

8. outDir.mkdir();

9. FileUtils.copyInputStreamToFile(

file.getInputStream(),new File(outDir,fileSaveName));

10. } catch (IOException e) {

11. e.printStackTrace();

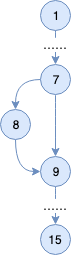
12. }

13. fileSaveName = "/static/avatar/" +fileSaveName;

14. userMapper.updateAvatarByPrimaryKey(user.getUserId(),fileSaveName);

15. return fileSaveName;

}





2.4.4 UserServiceImpl.loadUserInfo

public UserModel loadUserInfo(Integer userId){

1. if (userId == 0){

2. User u = (User)session.getAttribute("user");

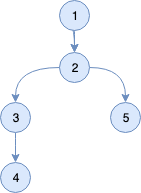
3. return userMapper.selectUserModelByPrimaryKey(u.getUserId());

4. }else {

5. return userMapper.selectUserModelByPrimaryKey(userId);

6. }

}





2.4.5 UserServiceImpl.loadFollowUser

public Page loadFollowUser(Integer userId,Page page){

1. if (userId == 0){

2. User u = (User)session.getAttribute("user");

3. userId = u.getUserId();

4. }

5. List<FollowUser> followUsers =

followUserMapper.findMyFollowPage(userId,page);

6. List<UserModel> userModels = new ArrayList<>();

7. for (FollowUser followUser:followUsers){

8. UserModel userModel =

userMapper.selectUserModelByPrimaryKey(followUser.getIdol\_id());

9. userModel.setFansNumber(followUserMapper.countMyfans(

followUser.getIdol\_id()));

10. userModel.setNoteNumber(noteBooksMapper.countMynoteNumber(

followUser.getIdol\_id()));

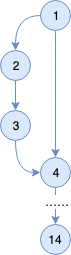
11. userModels.add(userModel);

12. }

13. page.setList(userModels);

14. return page;

}





2.4.6 UserServiceImpl.isFollow

public boolean isFollow(Integer userId){

1. User u = (User) session.getAttribute("user");

2. FollowUser followUser = followUserMapper.isFollow(u.getUserId(),userId);

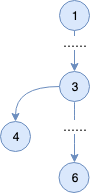
3. if (followUser == null){

4. return false;

5. }

6. return true;

}





2.4.7 UserServiceImpl.getUserInfo

public UserInfo getUserInfo(Integer userId){

1. if (userId == 0){

2. User user = (User) session.getAttribute("user");

3. userId = user.getUserId();

4. }

5. UserInfo userInfo = new UserInfo();

6. userInfo.setCommentNumber(commentBookMapper.

selectUserCommentBookCount(userId));

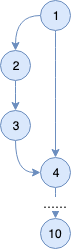
7. userInfo.setFollowed(followUserMapper.countMyidols(userId));

8. userInfo.setFollower(followUserMapper.countMyfans(userId));

9. userInfo.setNoteNumber(noteBooksMapper.countMynoteNumber(userId));

10. return userInfo;

}





## 3 测试结论

本次测试对“豆汁”阅读平台的service层进行了白盒测试，并且达到了判定覆盖。通过充分测试，平台满足相关的逻辑功能需求。