2017

A Self-Service Ticketing Kiosk System

EBU5304 SOFTWARE ENGINEERING

Group 35

Zhang Boyang, Wang Chao, Xin Boyan, Li Boyu, Li Chengji, Zhao Bowen

Content

Introduction

This report shows the process that we develop the system: Self-Service Ticketing Kiosk System. It is used for customers to purchase film tickets on the day using the kiosk, and admins to load the film information and generate statistic report. Customers can get the information film the system and choose the type of tickets, seats and the number of tickets. At each step, we used what we learned in the Software Engineering.

1. Summary of Responsibility

Asses the course and self-appraise is in *Appendix A*

Group Member	Section of Work	Section of Work	Section of Work
Zhang Boyang (Group Leader/Scrum Master)	Project Management	Analysis	Estimation and Iteration Plan
	Design	Code and Code Review	
Wang Chao (Scrum Team)	Analysis	Design	Estimation and Iteration Plan
	Code and Code Review	Implement	
Xin Boyan (Scrum Team)	Fact-finding Techniques	Write Epics	Estimation and Iteration Plan
	Risk Management	Code	
Li Boyu (Scrum Team)	Fact-finding Techniques	Write Epics	Detailed Functional Requirements in Use Case
	Code		
Li Chengji (Scrum Team)	Paper prototype	Test	Weekly Report Arrangement
	Code		
Zhao Bowen (Scrum Team)	Test	Paper prototype	Code

Zhang Boyang is the **Group Leader** in the group and is the **Scrum Master** in the Project. He is response for the coordinate the whole team. He also motivates the whole group. Arranging daily Wechat stand up meetings, holding conferences for sprint review, sprint retrospective and keeping the communication going with users concerning the sprint. Every time before a meeting, he informs the time and location through Wechat. In addition, He worked with Wang Chao to complete the project's Analysis and Design, Implement the software framework and code review, also many key functions in the software came from his hands. Zhang Boyang is a great leader in the Group and a great Scrum Master in the Project.

Wang Chao is the **Carry** of our Group and is the **leading member** in the Scrum Team. He worked with Zhang Boyang to complete the project's Analysis and Design, Implement the software framework and code review. He is also response for the code design, implement the functions, graphical interface implementation, and component operation the integration build plan. Wang Chao's Code Power is great. He solved a lot of technical difficulties in the project and ensure the quality and progress of the project. Wang Chao is a hardworking person, and his work is perfect. Wang Chao is the best member in the Scrum Team.

Xin Boyan is the Sub-Team2's leader. He worked with LI Boyu to completed the requirements engineering, and he assisted Group Leader in risk management, estimation and Iteration Plan. He is response for Fact-finding Techniques, writing Epics and creating product backlog. Xin Boyan is also a good Coder, many important modules and functions in the project were completed by him. Xin Boyan did his job brilliantly. He is a good member in the Scrum Team.

Li Boyu is the Sub-Team2's member. He worked with Xin Boyan to completed the requirements engineering. He is response for Fact-finding Techniques, writing Epics and detailing Functional Requirements in Use Case. Li Boyu also contributed a lot of code to the project, especially in the beauty of graphical interfaces. Li Boyu did his job brilliantly. He is a good member in the Scrum Team.

Li Chengji is the Sub-Team3's leader. He worked with Zhao Bowen to completed the Testing, and he assisted Group Leader to manage weekly reports and collect everyone's opinion and questions, then record the conference results. He is also response for Paper prototype. Li Chengji also contributed a lot of code to the project, especially in the beauty of graphical interfaces. Li Chengji did his job brilliantly. He is a good member in the Scrum Team.

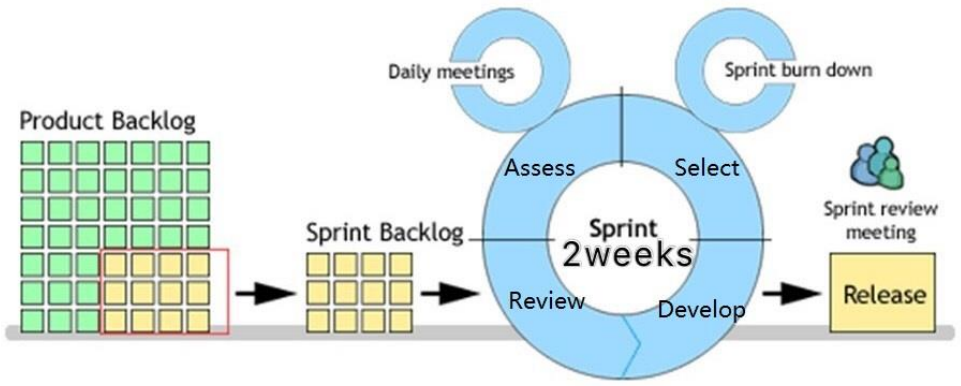
Zhao Bowen is the Sub-Team3's member. He worked with Li Chengji to completed the Testing. He is also response for Paper prototype. Many functions in the project were completed by him. Zhao Bowen did his job brilliantly. He is a good member in the Scrum Team.

2. Project Management

2.1 Introduction

This Software Engineering project is aimed at producing a self-service ticketing kiosk in cinemas using Scrum methods. This system is mainly used for customers to purchase film tickets on the day using the kiosk, and admins to load the film information and generate statistic report. This system needs to meet all requirements mentioned in the coursework handout. And our project team has 6 members, who are divided into 3 sub-teams. We used pair programming. Members in each sub-team would devote themselves in their specific field as the architecture of the system is gradually clarified. In addition, our Group use Coding.net a website like GitHub for version control, team work and project management.

The Scrum Process



- We first created the **Product Backlog.** According Product Backlog, Scrum Team estimation and iteration plan and workload. Our group decided the Sprints are 2 weeks.
- Deciding on the Sprint Backlog by the **Sprint Planning Meeting**.
- **Sprint Backlog** is completed by Scrum Team, and each member is further refined into smaller tasks according to Sprint Backlog
- We use WeChat Meeting and commit and task in Coding.net instead of **Daily Scrum Meeting**.
- When a Sprint Backlog is done, our team will hold **Sprint Review Meeting** and make sure the story in the Sprint Backlog has been fully implemented.
- For our small project, will hold Sprint Review Meeting, Sprint Retrospective Meeting, and next cycle's Sprint Planning Meeting together. Everyone should speak in the meeting, summarize, and discuss improved areas, and put into the next Sprint Cycle.
- Totally, we walked 5 Sprint Cycles.

2.2 People Management

We used pair programming. So, the 6 members are divided into 3 sub-teams with 2 persons in each.

Team	Member	Member	Task
Team1	**Zhang Boyang**	Wang Chao	Analysis, Design, Code and review
Team2	**Xin Boyan**	Li Boyu	Requirement and Code
Team3	**Li Chengji**	Zhao Bowen	Test and Code

2.3 Risk Management

2.3.1 Risk Type

Risk Type	Possible Risks
Organizational	Inefficient allocation of team work may lead to low efficiency. (1) The wrong decision or technique design made by group leader. (2)
Tools	software of different version doesn't work like expected (3) The time to learn a new software may longer than expected (4) Computer crashed during the work. (5)

People	Members of team become ill or non-available in short time (6) Weak communication between team members may lead to low efficiency. (7) Some people may need more time to fit in the unfamiliar tools and environment. (8)
Design and Implementation	Inefficient design may lead to repetition. (9) Low quality of code may lead to extra test and recoding. (10) Wrong design of the project make the time much longer than expected (11) The management with the version of code is confused. (12)
Requirements	The requirements have been identified continue changing. (13) The lack of adequate analysis of requirements in the previous stage. (14)

2.3.2 Risk Analysis

Risk	Likelihood	Consequences	Solutions
Illness of team member (6)	Medium	Leads HR shortage	Adjust the amount of assignments in a timely manner to fit the existing human resources
The schedule not match the practice work needs. (1)	High	Plan delayed	When making plans, consider possible scenarios that are not in line with reality, formulate backup plans, and adjust existing plans in a timely manner after the occurrence of the situation
Difficult to pull various parts of project together due to different IDE. (3)	Medium	The project cannot be integrated efficiently.	Before the system is developed, the compilation environment should be unified to avoid the fact that the code cannot be integrated because of IDE
Computer crashed during the working. (5)	Medium	Some useful data lose, and the plan cannot be completed on time.	When a part of the task is completed, the existing data and data are backed up and stored in time to prevent data loss due to device breakdown
Design error happened due to lack of careful analyze in previous stage. (11)	Low	Part of the function cannot be used.	Redesign the wrong part. And compared with the original design, to ensure that the last time the error does not occur
Critical errors found in the testing stage. (10)	Low	Devote all to solving the problem.	The whole group discussed the errors and put in the effort to test and correct them
The version of the code went off and the collaboration problem was encountered (12)	Medium	The management of the version is confused and the team collaboration is inefficient	Advanced version controllers GIT and coding.net which is a website like GitHub are adopted

2.3.3 Risk planning

1. Make the plan flexible so when some members are not available the project still goes on.
2. Protect and store the data, backup is necessary.

3. Make sure every step in the plan is fully completed and do some analysis before getting into next stage.
4. Make back-up plan to avoid emergences.

2.3.4Risk Monitoring

Monitor the risks throughout the whole project. Through meetings and discussions, compare weekly meetings with plans and evaluate risks at weekly meetings.

According to the current situation to discuss, and then do some reasonable adjustment of the next phase of the plan, avoid the risk in the next stage. When each stage of the task is completed, the overall risk assessment is necessary for us. Through the quantitative assessment of the time, quality of the task which have been completed to monitor risk.

2.3 Work Breakdown-Milestones and Deliverables

Phase	Activity	Star	Duration	Dependencies	Milestone
Prepare	T1. Meeting and sharing	3/19	1		
	T2. Coursework Discussion	3/19	1		
	T3. Fact-finding Techniques	3/18	1	T2	
	T4. Write stories using story cards	3/19-3/20	2	T3	
	T5. Prioritize the stories	3/21	1	T4	
	T6. product backlog	3/21-3/22	2	T5, T4	M1
	T7. Paper prototype user interface	3/22	1		M2
Sprint Cycle, (There were four cycles. The four iterations are the same.)	T8. Deciding on the Sprint Backlog	3/22-3/23	2	T6	
	T9. Estimation and Iteration plan	3/24	1	T8	
	T10. Identify Entity, Boundary and Control classes	3/25	1	T9	
	T11. Identify class relationships	3/35-3/26	2	T10	
	T12. Initial class diagram	3/27-3/29	2	T11	
	T13. Identify attributes for each class	3/29	1	T12	
	T14. Add constraints	3/30	1		
	T15. Develop	3/31-4/1	2	T13, T14	
	T16. Implementation	4/1	1		
	T17. Test and Review	4/1-4/2	2		M3/M4/M5/M6
Final	T18. Beauty of graphical interfaces	5/24-5/25	2	T7	
	T19. Test and Review	5/26	1	T18	M7, Final Software

	T20. Complete the final Report	5/26-5/29	4		M8, Final Report

2.4 Monitoring and Reporting Mechanisms

Weekly reports

Every week the group leader gold weekly meeting to get the rate of the group working progress and weekly meeting summary report reflecting the overall status of the project.

See it detailed in *Appendix B*

3. Requirement

3.1 Fact-finding Techniques

Background Reading

To gain a better understanding of the system, we read the handout of the system carefully. From the handout, we know that our task is to build a Self-Service Ticketing Kiosk System for Cinema. The Self-Service Ticketing Kiosk System will be used to sell film tickets in the Kiosk. Then, the system can product a statistic report of the day and sent it to the administrator's E-mail.

Observation

To learn about the system well, we observe the Cinema's tickets system in operation. It is an effective way for us to understand the system we will build. That is, watching the Cinema Tickets System and the real Kiosk. And, seeing how the users operate with the systems and how the Kiosk work.

With the observation, we can better understand our needs and what we need to do.

3.2 Functional Requirements

The Self-Service Ticketing Kiosk System is a software system used for selling film tickets in the Kiosk, producing statistic report of the day and sending it to the administrator's E-mail.

The user can use the system to view film information and buy film tickets for the day, also they can choose the seat, the ticket's type, and the number of ticket.

The system also able to produce statistic report at the end of each day. The report includes the sales of the day, the total sale of each film, total number of tickets sold and each type of ticket sold etc. The system generates the report automatically and sent to the administrator's E-mail.

In addition, the administrator can use the system to load the film information and film show time on the day to the kiosk in the morning.

3.3 Non-functional requirements

Product requirements

Efficiency requirements: The software needs to be started in 1 second, and the operation response time is within 1 second.

Reliable requirements: The system should be used in simple user language. Reduce the possibility of false appearance. Administrators can log on the system, query sales data, each film information, add information to each film, such as screen, time and useful information.

Flexibility requirement: Each function is required to run independently, and when new functions are added, it does not affect other functions of the system too much.

Organizational requirements

Simple GUI: The system should be developed as a standalone Java application with interfaces coded by Java GUI. The GUI interfaces should maintain color, the overall size of the same, in need of emphasis on the place there are obvious tips.

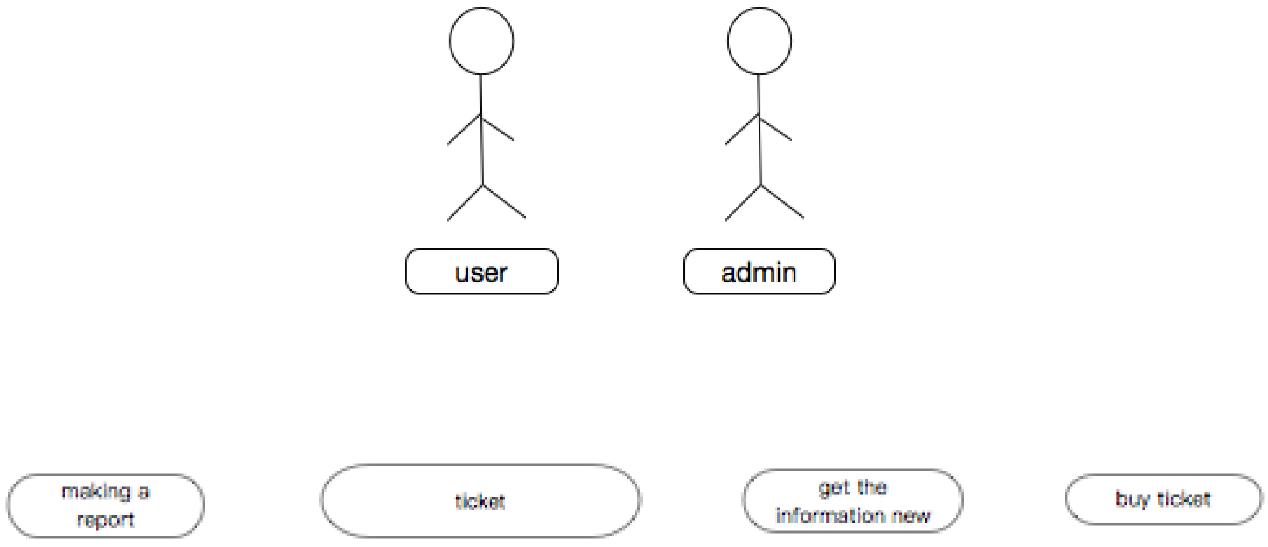
External requirements

Legislative requirements: Every film needs copyright. It's important to protect every film, not to let it out.

Safety requirement: In order to ensure the normal and safe operation of the ticket booking system, firewalls can be added to protect the user's use and data of the cinema.

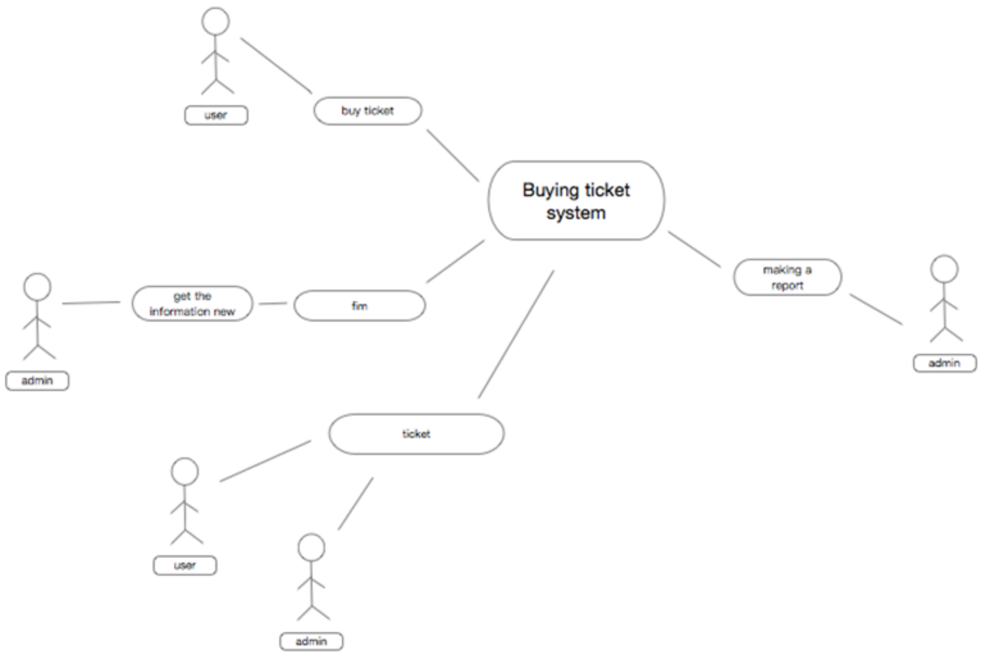
3.4 Detailed Functional Requirements in Use Case

3.4.1 Finding Users and Stories



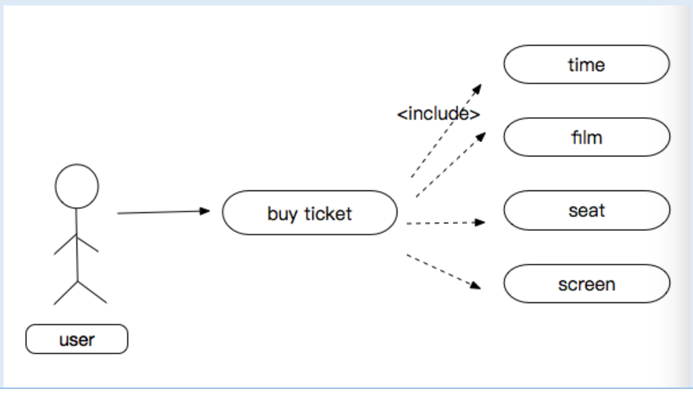
3.4.2 Actors with Description

Name of Actors	A brief Description of The Actor
user	An individual who want to buy tickets of a film.
administrator	An individual who manage the system by controlling the time, screen of the film.

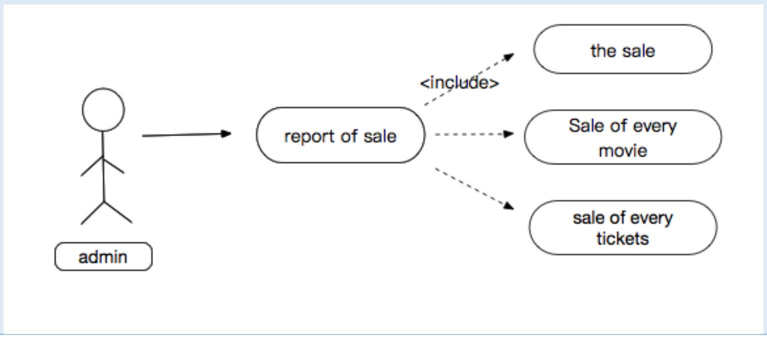


3.4.3 Relationships between Stories

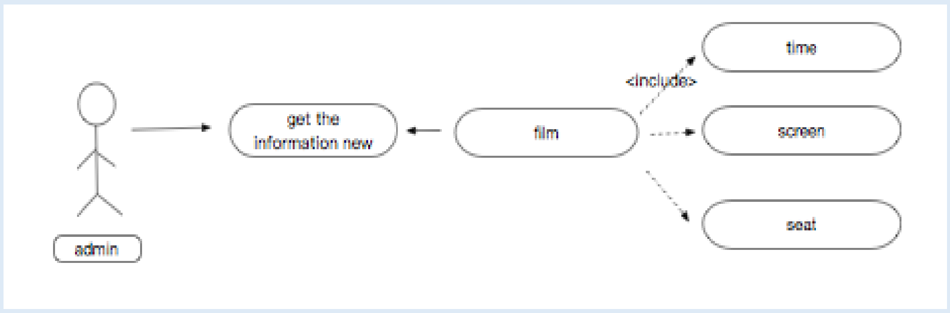
a) Buy ticket; Actors: User, tickets

	Buy ticket; Actors: User, tickets
Use Case Diagram	 time <include> film buy ticket seat screen user
Pre- Conditions	The user already knows which film to watch.
Description	The user gets the ticket system. Search for the film.
Post-Conditions	The user gets to the system successfully and find the film.
Exceptions	If the user is satisfied with the seat and time, they buy the ticket successfully.

b) Report of sale; Actors: admin, report

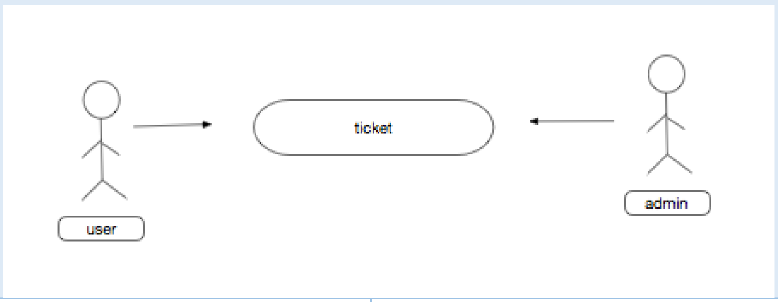
	Report of sale; Actors: admin, report
Use Case Diagram	 the sale <include> report of sale Sale of every movie sale of every tickets admin
Pre- Conditions	The administrator knows the password to login into the system.
Description	The administrator login into the systems. Find the place to for the report.
Post-Conditions	The administrator finds the report.
Exceptions	The administrator sees the report and know the different details of the report. To manage the time and screen of the film will be shown.

c) Get the information new; Actors: admin, film

	Get the information new; Actors: admin, film
Use Case Diagram	 time <include> get the information new film screen seat admin
Pre- Conditions	The administrator knows the password to login the system.
Description	The administrator login into the systems. Find the place to for the report.
Post-Conditions	The administrator finds the information
Exceptions	The administrator sees the information of the film. Then he can manage the time or screen for the film.

d) Ticket; Actors: admin, ticket, user

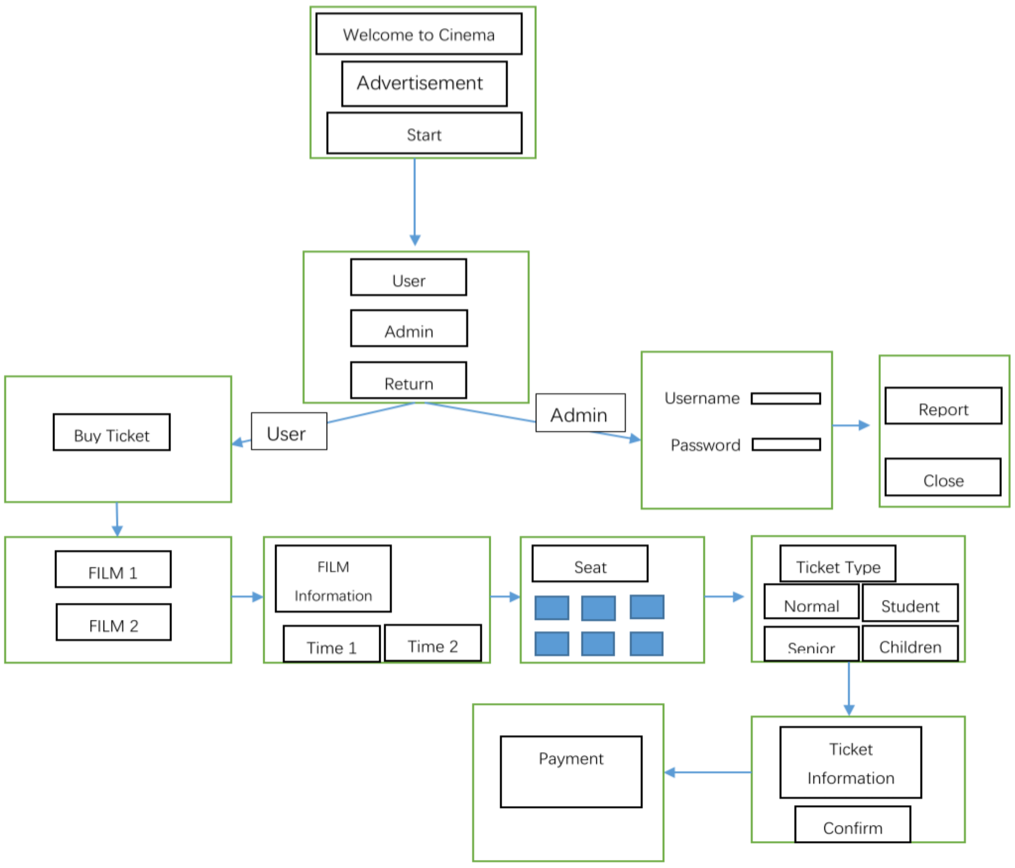
	Ticket; Actors: admin, ticket, user
Use Case Diagram	

		
People	User	Admin
Pre- Conditions	User know they cannot watch the film	They administrator can get to the system.
Description	User gets to the system. Find the returning for the ticket	Longlining into the system. Find the information of the film.
Post-Conditions	User find the information.	Get the returning ticket's seat can be bought.
Exceptions	User get the ticket returned.	Administrator get the seat can be bought

3.5 Difference Between the Product Backlog

We successfully achieve most of the product backlog. However, the administrator cannot change the password as what we said in the product backlog and this is the only difference.

User Interface Prototype



4. Analysis and Design

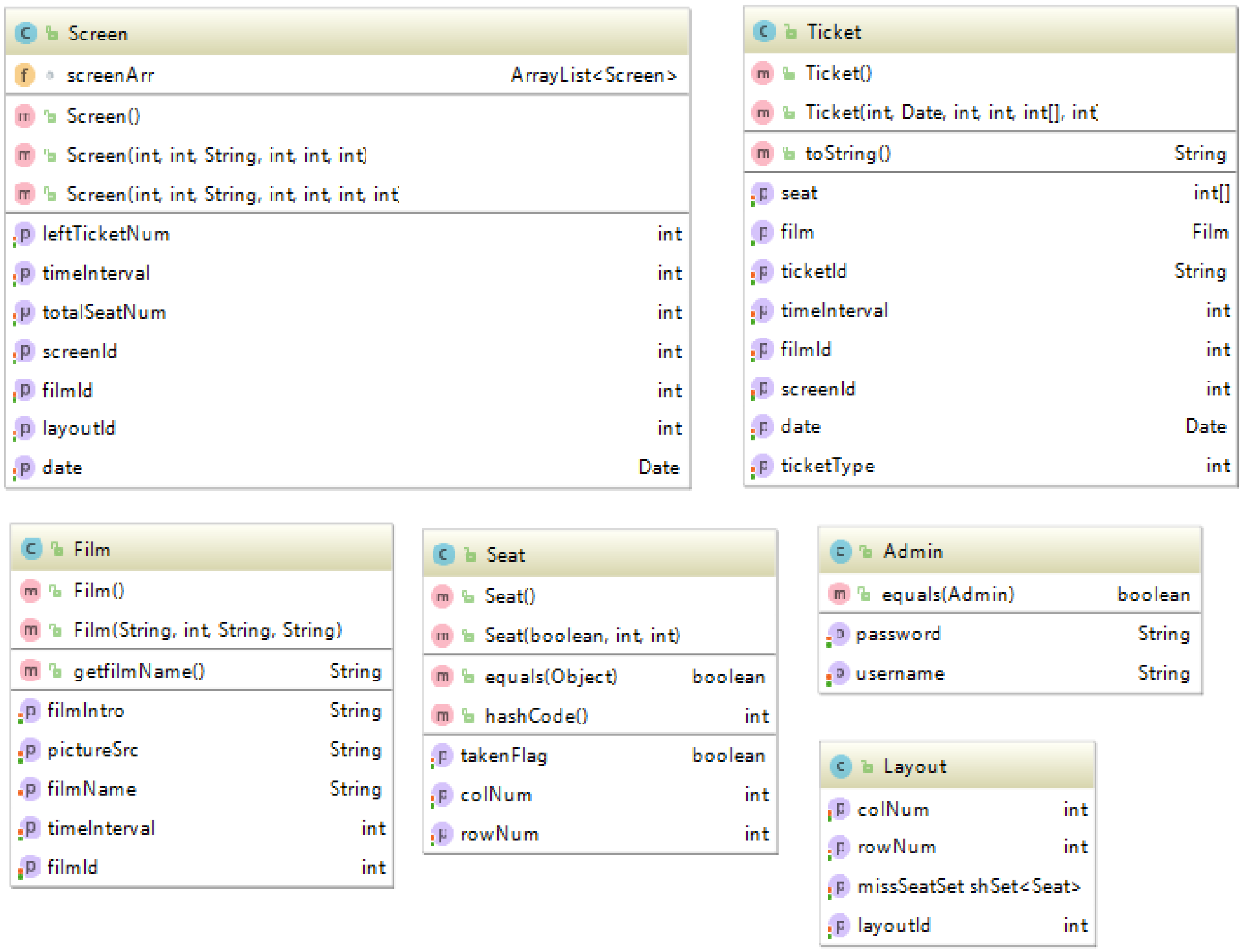
4.1 Identify Entity, Boundary, Control, and Design Class

There are 3 basic stereotypes in our software system, Entity classes, Boundary classes and Control classes.

Entity Classes

Entity Class are the classes which are used to model information that is long-lived and persistent.

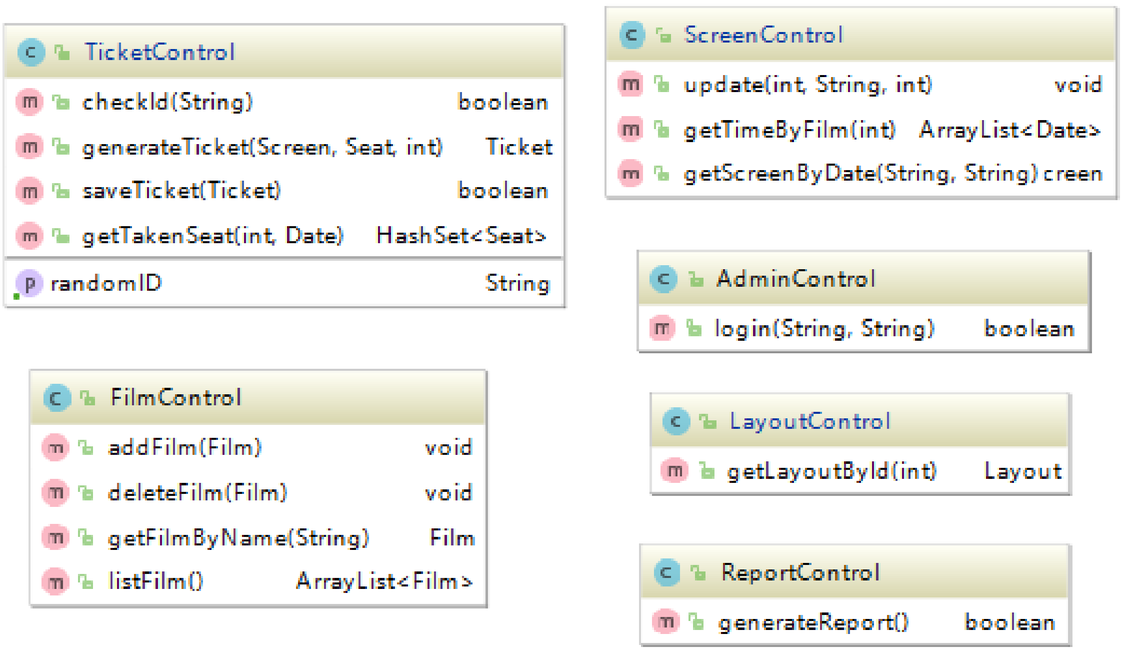
Our system has 6 entity classes. They model the information of Admin, Film, Layout, Seat, Screen, Ticket. When user buy ticket, the system will build an instance of the class.



Control Classes

Control Class are used to encapsulate control and coordination of the main actions and control flows. It represents coordination, sequencing, transactions and control of objects.

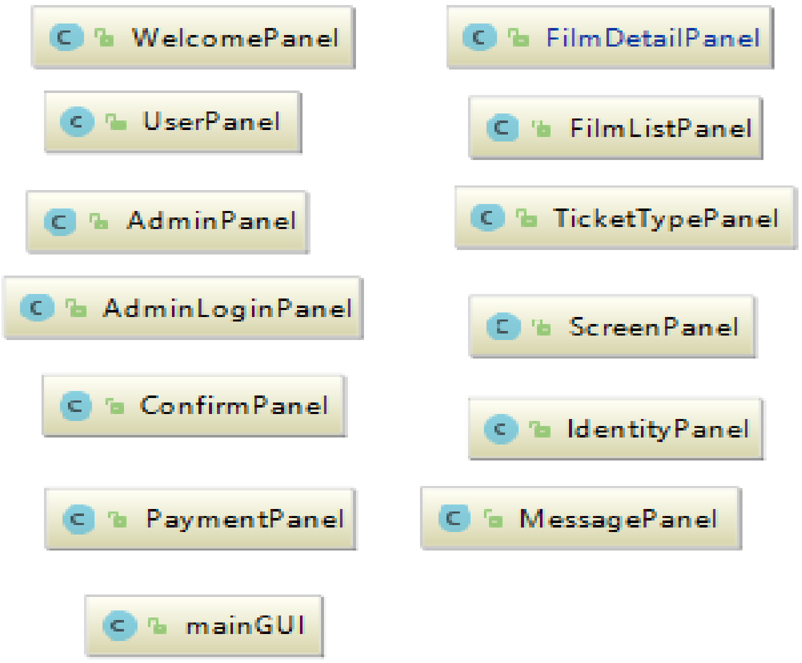
There are six control classes in our system. They can control the entity classes. For example, utilizing *AdminContol.java* to login, and *TicketControl.java* generate Ticket and get taken seat.


TicketControl
checkId(String) boolean
generateTicket(Screen, Seat, int) Ticket
saveTicket(Ticket) boolean
getTakenSeat(int, Date) HashSet<Seat>
randomID String
ScreenControl
update(int, String, int) void
getTimeByFilm(int) ArrayList<Date>
getScreenByDate(String, String) creen
AdminControl
login(String, String) boolean
FilmControl
addFilm(Film) void
deleteFilm(Film) void
getFilmByName(String) Film
listFilm() ArrayList<Film>
LayoutControl
getLayoutById(int) Layout
ReportControl
generateReport() boolean

Boundary Classes

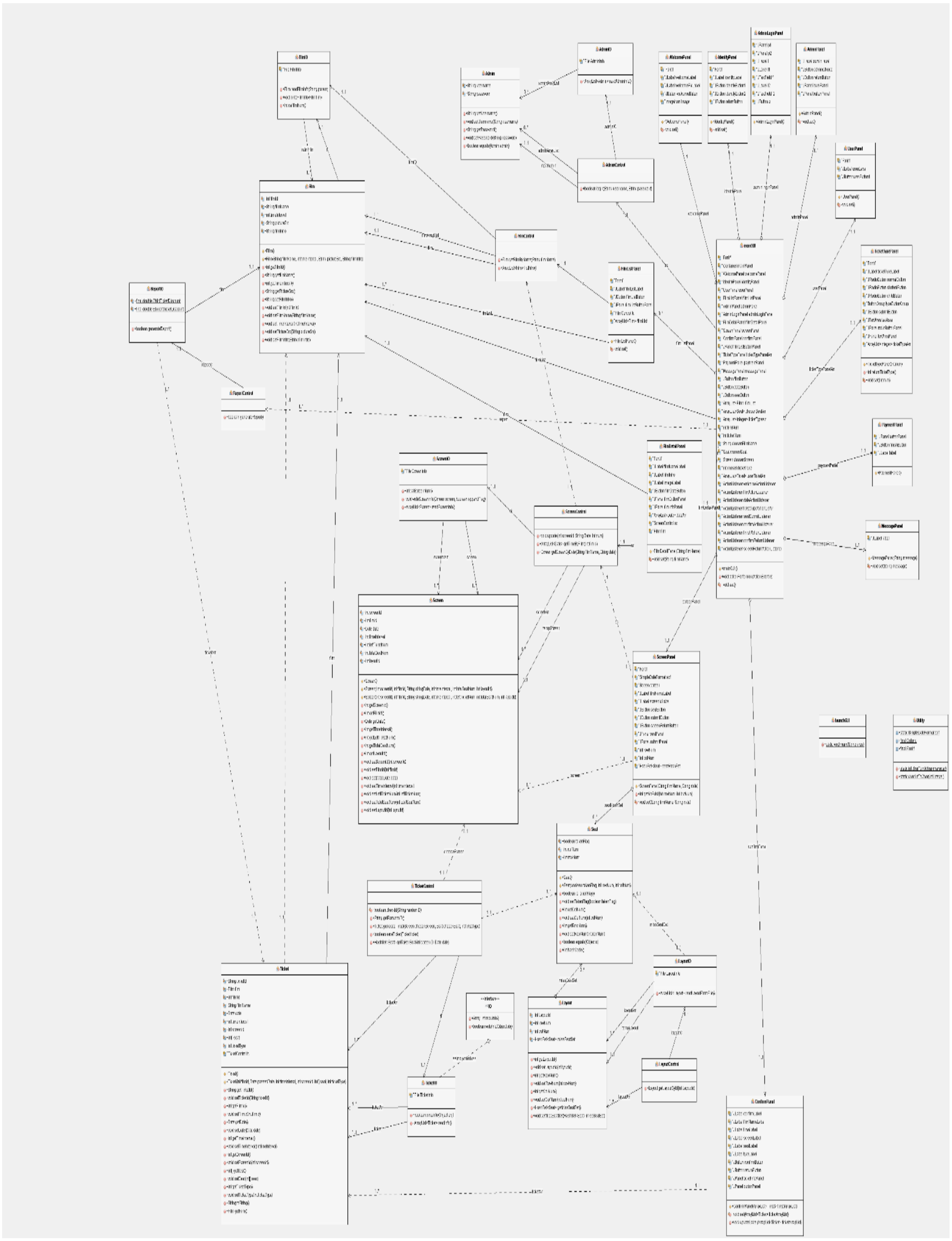
Boundary Class are used to model the interaction. These often involve receiving (presenting) information and

requests from (and to) users and external systems. Normally, Boundary Class represents abstractions of windows, forms, communication interfaces, printer interfaces, sensors, terminals, etc.



4.1 Identify Class Relationships and Draw Class Diagram

We also drew a design class diagram describing the relationship of these classes.



The diagram is too big, and you can click here to view [HD diagram]().

4.2 Identify Attributes for each Entity Class

Admin

Name	Type	Initial value
username	String	N/A
password	String	N/A

Film

Name	Type	Initial value
filmName	String	N/A
pictureSrc	String	N/A
filmIntro	String	N/A
FilmID	Int	N/A
timeInterval	Int	N/A

Layout

Name	Type	Initial value
missSeatSet	HashSet<Seat>	N/A
LayoutId	Int	N/A
RowNum	Int	N/A
ColNum	Int	N/A

Screen

Name	Type	Initial value
date	Date	N/A
screenId	Int	N/A
filmId	Int	N/A
timeInterval	Int	N/A
leftTicketNum	Int	N/A
totalSeatNum	Int	N/A
layoutId	Int	N/A

Seat

Name	Type	Initial value
takenFlag	boolean	false
RowNum	Int	N/A
ColNum	Int	N/A

Ticket

Name	Type	Initial value
ticketId	String	N/A
Film	Film	N/A
filmId	Int	N/A

timeInterval	Int	N/A
filmName	String	N/A
date	Date	N/A
ScreenId	Int	N/A
Seat	Int[ ]	N/A
ticketType	Int	N/A

4.3 Add Constraints

a) Each issued ticket should have a random unique 8-digit ticket ID printed.

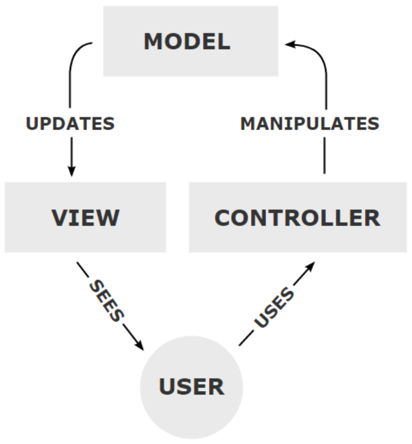
b) The ticket ID: it has exactly 8 digits and the digit can only be 1, 2, 3 or 4. For example, 23212342, 11232421, 33333342 are valid IDs.

c) The customer can only purchase the ticket which show time is AFTER the current time.

d) In a screening room, the film's screen time should not be in conflict

e) In the same film, a seat cannot be sold twice

4.4 Design of the Software



In high-level terms, the MVC pattern means that an MVC application will be split into at least three pieces:

1. Models, which contain or represent the data that users work with. These can be simple view models, which just represent data being transferred between views and controllers; or they can be domain models, which contain the data in a business domain as well as the operations, transformations, and rules for manipulating that data.
2. Views, which are used to render some part of the model as a UI.
3. Controllers, which process incoming requests, perform operations on the model, and select views to render to the user.

As you can see in the Class Diagram, we divided our program into those three parts.

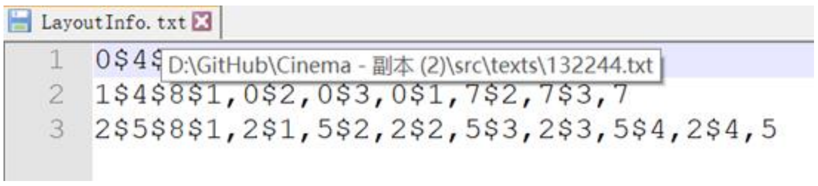
1. Models: Admin.java, AdminIO.java, Ticket.java, TicketIO.java, IO.java, Film.java, FilmIO.java, Layout.java, LayoutIO.java, Screen.java, ScreenIO.java, Seat.java, Utility.java, ReportIO.java
2. Views: AdminLoginPanel.java, AdminPanel.java, TicketTypePanel.java, IdentityPanel.java, launchGUI.java, mainGUI.java, FilmDetailPanel.java, FilmListPanel.java, ScreenPanel.java, ConfirmPanel.java, MessagePanel.java, PaymentPanel.java, UserPanel.java, WelcomePanel.java,
3. Controllers: AdminControl.java, TicketControl.java, FilmControl.java, LayoutControl.java, ScreenControl.java, ReportControl.java,

The structure of the program is distinct by using MVC Pattern. What's more, due to the program is separated into three layers, sometimes only one layer is needed to be changed to meet the new requirement.

4.5 Adaptable to Change

The adaptable change of our system is mainly reflected in the process of loading data from txt to exhibit on the screen and changing static variables to adapt different use circumstances.

All data stored in files are readable, and can be changed to update the information of the system. For example, "LayoutInfo.txt" showed below contains the number of rows, the number of columns and the position of missing seat in this layout. Administrators can easily add another layout or change the existing style of layout by modifying this text.

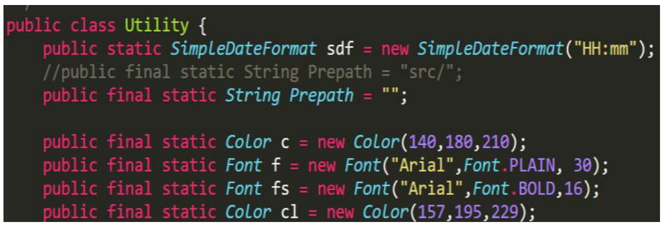


LayoutInfo.txt

D:\GitHub\Cinema - 副本 (2)\src\texts\132244.txt

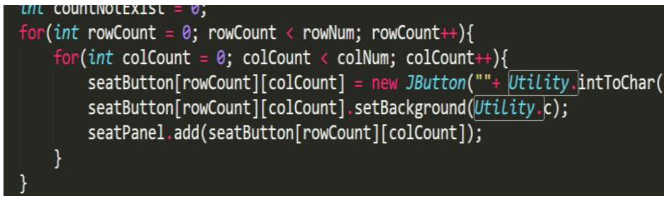
```
0$4$
1$4$8$1,0$2,0$3,0$1,7$2,7$3,7
2$5$8$1,2$1,5$2,2$2,5$3,2$3,5$4,2$4,5
```

Moreover, because static variables such as color, font, date format has been stored in "Utility.java" and all references of these variables is from "Utility.java". So, if the looking of this system can be adjusted easily by modify the java file.



```java
public class Utility {
    public static SimpleDateFormat sdf = new SimpleDateFormat("HH:mm");
    //public final static String Prepath = "src/";
    public final static String Prepath = "";

    public final static Color c = new Color(140,180,210);
    public final static Font f = new Font("Arial",Font.PLAIN, 30);
    public final static Font fs = new Font("Arial",Font.BOLD,16);
    public final static Color cl = new Color(157,195,229);
```

Utility.java



```java
int countNotExist = 0;
for(int rowCount = 0; rowCount < rowNum; rowCount++){
    for(int colCount = 0; colCount < colNum; colCount++){
        seatButton[rowCount][colCount] = new JButton(""+ Utility.intToChar(
        seatButton[rowCount][colCount].setBackground(Utility.c);
        seatPanel.add(seatButton[rowCount][colCount]);
    }
}
```

Code from ScreenPanel.java

4.6 Issue of Re-Usability

In our code, there are many components can be reused when we are going to developed another program. The class AdminIO.java, LayoutIO.java, ReportIO.java, ScreenIO.java and TicketIO.java, and their *readInfo ()* function return *ArrayList<Object>* . It can be reused in the future, they can be used at the other interface.

4.7 Design Principles Applied

Interface Segregation Principles (ISP)

This principle means that it is better to use many specific interfaces than a single overall interface. A class should depend on the other class only though the smallest interface. And a class should not have all the methods in a fat interface. Especially when these methods have no relations between each other. In our code, we applied this principle. There is no method that includes another method. So, when a class want to use another class' s method, it directly invokes the method.

Open-Close Principle (OCP)

Software modules should be open for extension and closed for modification. Our codes are friendly to extend. We design each part separately. The code can be extended to add other class for different requirements.

Single Responsibility Principle (SPR)

For the Single Responsibility Principle, we tried to allocate every single object in the program an exclusive representation in order to give them a lower independency, and we tried to make sure, that all the methods concerning the object are focused on carrying out the single responsibility, which leads our software to highly cohesive. For example: in our software, the class TicketIO.java, with writeInfo (Object obj), readInfo () and genTicketFile (Ticket ticket) method, is just in response for the base I/O operation.

Don't Repeat Yourself Principle (DYP)

We use generalization methods to reuse our existing code rather than copy them. In our code Utility.java determines the path of the file and the color of the GUI. Therefore, we don't need to repeat ourselves.

5. Implementation

We use JAVA to implement our design. The main screen short of the system can be found in Appendix C.

5.1 Components

source code: Admin.java, AdminControl.java, AdminIO.java, AdminLoginPanel.java, AdminPanel.java, ConfirmPanel.java, Film.java, FilmControl.java, FilmDetailPanel.java, FilmIO.java, FilmListPanel.java, IO.java, IdentityPanel.java, Layout.java, LayoutControl.java, LayoutIO.java, ReportControl.java, ReportIO.java, Screen.java, ScreenControl.java, ScreenIO.java, ScreenPanel.java, Seat.java, Ticket.java, TicketControl.java, TicketIO.java, TicketTypePanel.java, UserPanel.java, Utility.java, WelcomePanel.java, launchGUI.java, mainGUI.java

Document: Javadoc

Every component traces the design element it implements.

5.2 Integration build plan

In this session, our team tested each component to determine that these components can work as we planned at the beginning. In addition, we have developed a construction plan from the beginning of the initial design to determine the function of each version of the situation, the degree of realization of each function.

Build version 1

Functionality: The user can login into the system, see the list of films, choose the film.

Use case: buy tickets;

Scenarios: If the user choose the film, it will save in the system.

Components: launchGUI.java, mainGUI.java, Film.java, FilmControl.java, FilmDetailPanel.java, FilmIO.java, FilmListPanel.java, UserPanel.java, WelcomePanel.java, Utility.java

Build version 2

Functionality: The user can choose the seat for their movie.

Use case: Choose seat;

Scenarios: If the user choose the film and choose the seat of the film, the system will remember. And others can see the seat has been already chosen.

Components: launchGUI.java, mainGUI.java, Film.java, FilmControl.java, FilmDetailPanel.java, FilmIO.java, FilmListPanel.java, Layout.java, LayoutControl.java, LayoutIO.java, Screen.java, ScreenControl.java, ScreenIO.java, ScreenPanel.java, Seat.java, UserPanel.java, WelcomePanel.java, Utility.java

Build version 3

Functionality: The administrator can manage the system, see the report, change the password.

Use case: Change password, making a report.

Scenarios: The administrator need to protect safety so that changing the password and decide what film to show by seeing the report.

Components: Admin.java, AdminControl.java, AdminIO.java, AdminLoginPanel.java, AdminPanel.java, Ticket.java, TicketControl.java, TicketIO.java, TicketTypePanel.java, IdentityPanel.java, IO.java, launchGUI.java, mainGUI.java, Film.java, FilmControl.java, FilmDetailPanel.java, FilmIO.java, FilmListPanel.java, Layout.java, LayoutControl.java, LayoutIO.java, Screen.java, ScreenControl.java, ScreenIO.java, ScreenPanel.java, Seat.java, UserPanel.java, WelcomePanel.java, Utility.java, ReportControl.java, ReportIO.java

Build version 4

Functionality: The user can pay for the tickets.

Use case: Buy tickets.

Scenarios: The users have already finished choosing film and need to pay for it for make the seat belonging to them.

Components: Admin.java, AdminControl.java, AdminIO.java, AdminLoginPanel.java, AdminPanel.java, Ticket.java, TicketControl.java, TicketIO.java, TicketTypePanel.java, IdentityPanel.java, IO.java, launchGUI.java, mainGUI.java, Film.java, FilmControl.java, FilmDetailPanel.java, FilmIO.java, FilmListPanel.java, Layout.java, LayoutControl.java, LayoutIO.java, Screen.java, ScreenControl.java, ScreenIO.java, ScreenPanel.java, Seat.java, UserPanel.java, WelcomePanel.java, Utility.java, ConfirmPanel.java, MessagePanel.java, PaymentPanel.java, ReportControl.java, ReportIO.java

6. Testing

In the test stage, we use White-box testing and mainly use Regression testing and Black-box testing. We do test phase during implement period, after implement stage. Different cases for the whole system are tested.

6.1Plan test

- The test of the system is composed of Component testing and System testing.
- Component testing: Login; Change password; Choose tickets; Confirm tickets
- System testing: Update information; statistical report
- Testing Strategy: The White-box testing is used in Component testing, The Black-box testing is used in System testing
- Success criteria: 90% of test cases passed.
- No high priority defects unresolved.

6.2Design test

a) Component Test Cases

- **Login**

Test Cases	Case 1(correct input)	Case 2(incorrect input)	
Input	User name and password are all correct. The identity chosen is the same with what you registered in AdminInfo.txt.	User name does not exist.	User name and password does not match.
Result	Login succeed and enter the function interface.	Wrong user name or password, Enter again.	Wrong user name or password, Enter again.
Conditions	No conditions exist	No conditions exist	

Test procedure

Login	Test description	Test cases	Samples Pass/Fail	No. Of Bugs	Bug#	Comments
N/A	Create two users [username-tom and password-123], [username-wc and password-2345]	setup	—	—	—	Added to system for testing
1.1	Test that user name tom does exist in the system	1.1	P	0	0	Correct behavior observed

1.2	Test that user name wc does exist in the system	1.2	P	0	0	Correct behavior observed
1.3	Test that password 123 is correct for user name tom	1.3	P	0	0	Correct behavior observed
1.4	Test that password 2345 is correct for user name wc	1.4	P	0	0	Correct behavior observed
1.5	Test that password 0000 is incorrect for user name wc	1.5	F	0	0	Incorrect behavior observed

- ## Change password

Test Cases	Case 1(correct input)	Case 2(incorrect input)
Input	Password is changed by changing the file content of AdminInfo.txt	User name and password does not match.
Result	change succeed and enter the function interface.	Wrong user name or password, Enter again.
Conditions	No conditions exist on the test	No conditions exist on the test

Test procedure

Change password	Test description	Test cases	Samples Pass/Fail	No. Of Bugs	Bug#	Comments
N/A	Change [username-tom and password-123] to [username-tom and password-1234]	setup	—	—	—	Added to system for testing
2.1	Test that user name tom does exist in the system	2.1	P	0	0	Correct behavior observed
2.2	Test that password 1234 is correct for user name tom	2.2	P	0	0	Correct behavior observed
2.3	Test that password 123 is incorrect for user name tom	2.3	F	0	0	Correct behavior observed

- ## Choose the tickets

Test case3.1	Choose the film	Test case3.2	Choose the time
input	Press the corresponding button of the film I want to watch	**input**	Press the button showing my favorite time.
result	Switch to the next page——time page	**result**	Switch to the next page——seat selecting page
condition	No conditions exist	**condition**	No conditions exist
procedure	Create the first page——film list page. In the page, all the films available in recent day	**procedure**	All the available time periods of the chosen film are listed in the button form.

	are listed. You choose the films by simply pressing the button of corresponding film.		You can choose your favorite time by just pressing the button.

Test case3.3	**Select the seats**	**Test case3.4**	**return**
input	Press the seats I want to select. The selected seats will turn to black showing I chose them and I shall press 'confirm' button to confirm my operation.	**input**	Press the 'return' button.
result	Succeeded selecting and switch to the next page——tickets type choosing page	**result**	Return to last page
condition	If the seats have been occupied, the corresponding button will be replaced by label witch you are be unable to press.	**condition**	No conditions exist
procedure	Seats choices are listed in the third page. You could select one or more seats and the color of the button turned to black will tell you selected them. If you found yourself no longer want to buy the tickets, there a return button and you can turn back to the welcoming interface.	**procedure**	Press the 'return' button.

- ## Confirm the tickets

Test case4.1	**Choose tickets' type**	**Test case4.2**	**Confirm the payment**
input	Choose the type of the ticket. If it is not normal ticket, you are asked to type in your student ID or child age.	**input**	Press 'confirm' button or 'return' button.
result	Record the ticket type and corresponding ID, then switch to confirm page	**result**	Finish the payment or chancel the deal.
condition	No conditions exist	**condition**	No conditions exist
procedure	Choose the type of the ticket. If it is not normal ticket, you are asked to type in your student ID or child age which should be checked by external system. Then, press the confirm button to	**procedure**	In the confirm page, all the information about the tickets are listed. After checking the information, you should press confirm to finish the payment or return to chancel the deal.

Login	**Test description**	**Test cases**	**Samples Pass/Fail**	**No. Of Bugs**	**Bug#**	**Comments**
3.1	Press the button of film	3.1	P	0	0	Correct switch to next page
3.2	Press the button of time of the film	3.2	P	0	0	Correct switch to next page

3.3	Press the return button	3.3	P	0	0	Successfully return
3.4	Press each button of seats and confirm.	3.4	P	0	0	Correct behavior
4.1	Choose student type of ticket and type in student ID 123.	4.1	P	0	0	Correctly buy ticket showing the student ID 123.

b) System Test Cases

• Update Information

Black-Box Test cases:

- ➢ User requirement: As an administrator, I want to load the film information and film to the kiosk in the morning so that I can update film every day without taking too much effort.
- ➢ Method: Scenario-based testing

Test Cases	Case 1(correct input)	Case 2(incorrect input)
Input	Cinema information is updated by changing the file content of FilmInfo.txt without changing the format.	Input an incorrect file content format
Result	update succeed and enter the information is changed.	Updated failed and appears bug
Conditions	No conditions exist on the test	No conditions exist on the test

Test procedure

Update Information	Test description	Test cases	Samples Pass/Fail	No. Of Bugs	Bug#	Comments
N/A	Change [0$KONG$118$src/pic/kong.jpg$Skull ISLAND] to [0$KINGKONG$118$src/pic/kong.jpg$ISLAND]	setup	—	—	—	Added to system for testing
5.1	Test that the film information is updated	5.1	P	0	0	Correct behavior observed
5.2	Test that the film name is updated	5.2	P	0	0	Correct behavior observed

• Statistical report

Black-Box Test cases:

- ➢ User requirement: As an administrator, I want to receive a statistical report of the daily sales so that I can know the working situation of cinema better.
- ➢ Method: Scenario-based testing

Test Cases	Case (correct input)
Input	receive a statistical report of the daily sales by click the report button after login
Result	The report is printed and updated in file of ReportInfo.txt.
Conditions	No conditions exist on the test

Update Statistical	Test description	Test cases	Samples Pass/Fail	No. Of Bugs	Bug#	Comments
N/A	Report	setup	—	—	—	Added to system for testing
6.1	Test that the statistic information is updated	6.1	P	0	0	Correct behavior observed
6.2	Test the sum of selling is updated correctly	6.2	P	0	0	Correct behavior observed

6.3 Implement Test

In our testing plan, 50% of tests are automated. The program that is used for auto-test is written in JAVA and used the same interface with the system. We replace the tested component with our written test program. To ensure the correctness of testing, all above are tested both in automatic and manual way.

6.4 TDD

TDD is short for Test Driven Development. TDD is a simple, short-cycled mechanism. And Test-Driven-Development means writing tests prior to write the production code.

TDD short-cycled mechanism:

- Write a specification, in code and in the form of a unit test. The test verifies a functional unit of your code.
- Demonstrate test failure.
- Write code to meet the specification.
- Demonstrate test success.
- Refactor the code, to ensure that the system still has an optimally clean code base.

Junit is simple unit-testing framework for supporting TDD. TDD's focus is unit testing, and we used the white-box testing which is shown above. And the Junit part will be shown in the *Appendix D*

Conclusion

After all these phases, we have developed a successful software. It is not that perfect but it will be continuously improved in the future to be more perfect. As we all know, the software engineering is a long-term process. And we learned a lot from this experience. And this course work helps us know the process of developing a software better. Thanks for the teachers who give us instructions, and also the teaching assistants who will spend a lot of time to read this report and give us precious advice. Thank you all.

Reference

- Chapter 3 and 22 – “Software Engineering” textbook by Ian Sommerville
- http://en.wikipedia.org/wiki/Agile_software_development
- Chapter 4 – “Software Engineering” textbook by Ian Sommerville
- User stories by Mike Cohn http://www.mountaingoatsoftware.com/agile/
- Chapter 4, 5 – “Head First Object-Oriented Analysis & Design” textbook by Brett McLaughlin et al
- Chapters 6, 7 – “Software Engineering” textbook by Ian Sommerville
- Chapter 5 – “Head First Object-Oriented Analysis & Design” textbook by Brett McLaughlin et al
- “Head First Object Oriented Analysis & Design” textbook by Brett McLaughlin et al
- Chapters 8 – “Software Engineering” textbook by Ian Sommerville
- “Head First Object Oriented Analysis & Design” textbook by Brett McLaughlin et al
- Agile Java™: Crafting Code with Test-Driven Development, Jeff Langr
- Introduction to Agile by Sondra Ashmore

Appendix A: Individual Responsibility

Summary of Responsibilities and Achievements

Group Members:

Name	QMID	BUPTID
Boyan Zhang	140919581	2014212966
Chao Wang	140922352	2014213241
Boyan Xin	140919709	2014212978
Bowen Zhao	140919086	2014212911
Boyu Li	140920691	2014213075
Chengji Li	140921805	2014213182

Group Leaders Name: Boyang Zhang 140919581 2014212966

Individual Members Contribution and Self-Appraisal of Work Done

I. Individual member: Boyang Zhang 140919581 2014212966

(1). Individual member contribution:

Boyang Zhang as a group leader is responsible to coordinate the whole team and motivates the whole group.

Arranging daily Wechat stand up meetings, holding conferences for sprint review, sprint retrospective and keeping the communication going with users concerning the sprint. Informs the time and location through Wechat.

Worked to complete the project's Analysis and Design, Implement the software framework and code review, also many key functions in the software came from his hands.

(2). The outcome appraisal of the coursework

Boyang Zhang is absolutely a nice leader with excellent talents in organizing as well as communicating. All these things make he become the core of our group. And he has a high standard of responsibility and with a good sense time that he never shows up for meeting. He is not only a good leader, but also a kind person. He always does better on his own work and consider about the whole work process.

(3). Self-appraisal of work done

Through this project, I find it's easier for me to cooperate with other team members by using Agile method. This project enhanced my ability in java programming and one of the most important thing I learned from this project is time management. As a leader of our group, it's my responsibility to participate in whole process and learn more new things well. And it's no doubt that the successful of this project belongs to every member in our team. I learned a lot from the project itself and from the other members in the group.

II. Individual member: Chao Wang 140922352 2014213241

(1). Individual member contribution:

Chao Wang as a powerful coder complete the project's Analysis and Design.

Implement the software framework and code review.

Also, response for the code design, implement the functions, graphical interface implementation, and component operation the integration build plan.

Solved a lot of technical difficulties in the project and ensure the quality and progress of the project.

(2). The outcome appraisal of the coursework

Chao Wang is a person with energy and enthusiasm in our group. In general, responsibilities were well finished by him and don't have an error. He was responsible for a large part of system design from coding to testing. he has nice communication skills and is willing to help others to finish their work. It's no doubt that he is a nice people not only in team working, but in our normal life.

(3). Self-appraisal of work done

In this project, I pay more attention on the design and programming of our project, and I also communicate with other members to ensure my thoughts. I tried something new in java programming, which helped me to enhance my ability of java programming. What's more, I do feel that meeting the design principles are very important in terms of program design, such as the allocation of different 'class', which saved a lot time for us to develop this system. And the most important thing I learned is cooperation makes the work much easier.

III. Individual member: Boyan Xin 140919709 2014212978

(1). Individual member contribution:

Boyan Xin is the Sub-Team2's leader worked with Boyu Li to completed the requirements engineering and assisted Group Leader in risk management, estimation and Iteration Plan.

Fact-finding Techniques, writing Epics and creating product backlog. Boyan Xin is also a good Coder, many important modules and functions in the project were completed by him.

(2). The outcome appraisal of the coursework

Boyan Xin is a person with high standard of responsibility. He actively participated in the discussion and every step of the development of this project. He has a nice time sense and never shows up late for his work. And he is willing to learned new knowledge and cooperation with other members in our group. At the same time, he has the excellent at the knowledge we studied during the class.

(3). Self-appraisal of work done

Weeks of work full of laugh and difficulties. I enjoyed myself for working together with my teammates in order to achieve the same goal and try to do our best to working. I worked for the 'selling report' design, risk analyze etc, which made me felt that a good design and analyze before the develop is very important for a project. When I faced challenge, I always learn more skills to try my best and solve it with my team. And I learned to consider more to ensure there is no errors. Also, I strive to improve my programming knowledge that I finished my task on time. Every teammate in this group have merits worthy to learn for me.

IV. Individual member: Bowen Zhao 140919086 2014212911

(1). Individual member contribution:

Bowen Zhao is the Sub-Team3's member.

He worked with Chengji Li to completed the Testing. He is also response for Paper prototype. Many functions in the project were completed by him.

(2). The outcome appraisal of the coursework

Bowen Zhao is a very kind person and very liking to push out his own opinion. He also has great cohesion with the principle we learned from the class. When facing with problems, he could always act positively and be enthusiastic. It's no doubt that he is a nice team member to operation with him.

(3). Self-appraisal of work done

I really appreciate that I can work with my team members. According to this project I understand that when we encounter difficult, we should face them with courage rather than give up. Participate in the communication and each step in this project improve my programming skills and the ability to consider more on system design and complete. All in all, I not only learned a lot, but also understand the strengths and weakness of myself thanks to the task.

V. Individual member: Boyu Li 140920691 2014213075

(1). Individual member contribution:

Boyu Li is the Sub-Team2's member. He worked with Boyan Xin to completed the requirements engineering.

Working for Fact-finding Techniques, writing Epics and detailing Functional Requirements in Use Case. Also, contributed a lot of code to the project, especially in the beauty of graphical interfaces.

(2). The outcome appraisal of the coursework

As a member in our group, Boyu Li plays a key role in the report design and project analyze. Every group meeting, he can express his view actively, and finish the task as requested on his own. With his contribution, our work runs smooth and efficiently. In short, he is an indispensable member in our team, and we are glad to work with him as a team.

(3). Self-appraisal of work done

During the time working on software engineering coursework, I first got nervous and finally overcome difficult and finished my part. In the entire process, I learned more than coding and designing skills but how to share workloads and new idea with my partner. Team cooperation helps us figure out many problems, and in my opinion the meeting we get every week also very important, we can solve problems together. Through this project, I also have a good cohesion on the principle I have learned in class.

VI. Individual member: Chengji Li 140921805 2014213182

(1). Individual member contribution:

Chengji Li is the Sub-Team3's leader. He worked with Zhao Bowen to completed the Testing, and he assisted Group Leader to manage weekly reports and collect everyone's opinion and questions, then record the conference results.

Working for Paper prototype and contributed a lot of code to the project, especially in the beauty of graphical interfaces.

(2). The outcome appraisal of the coursework

Chengji Li is a positive member in the group who give us a lot of confidence that our work can be completed perfectly. He actively participates in every step and every meeting. The analyze and testing part which he made is perfect. When faced some problems he always insists to solve it and finally finish the distribute work very well. He made a great contribution to the success of this coursework.

(3). Self-appraisal of work done

I'm proud of being a member of this group. Everyone in this team have their strengths and considerate that we could work together to complete this tough project in a pleasant atmosphere and do the job smoothly. Although some difficult may be a challenge for me, I never give up and learn more new things and principles to strengthen myself, sometimes also need to figure out solutions with my team. In addition, to finish my task efficiently, I reviewed the lectures many times and get a good understanding on Software Engineering.

Appendix B: Weekly Reports

Week1

- **Time:** 2017.3.19 Week 3 Sunday 18:30
- **Location:** The third floor of the student dormitory 5
- **Attendance:** All attend
- **Conference content:**

  Take the group photo

  Everyone is familiar with each other

  Understand the advantages and disadvantages of our team

  Select the leader

  Roughly determine the team core

  Agreed to open the group every Wednesday

  Discuss the task before next week
- **Task:**

  Read handout comprehensively

  Draw a large brain map

Week2

- **Time:** 2017.3.22 Thursday Weekend 21:00
- **Location:** The third floor of the student dormitory 5
- **Attendance:** Except Xin Boyan
- **Task Completion**：

  Read handout comprehensively

  Draw the general brain map
- **Conference content:**

  Discuss handout and complete the brain map logic

  Determine the general structure

  Discuss the requirements
- **Task:**

  The Construction of the Main Frame -Wang Chao

  The next group before the requirements of the prototype - lead by Li Boyu

Week3

- **Time:** 2017.3.29 Week 5 Wednesday 18: 00-17: 00
- **Location:** The second floor of the Student Development Center
- **Attendance:** All attend
- **Task Completion**：

  The prototype of the requirement is basically completed

  Continue in the frame
- **Conference content:**

  Carefully studied the report of previous years

  Discuss the whole project in detail

  Have the overall framework of the report

  Discussed the specific details of the software storage

  Based on the reporting framework, the project is divided into phases and the group has made a specific allocation

  Teach us to use git and GitHub.
- **Task:**

  Continue to be familiar with git and GitHub

  Mind map - Zhang Boyang

  Software use process - Li Boyu, Li Chengji

  User case model - Zhao Bowen

  Introduction - Xin Boyan

Week5

- **Time:** 2017.4.28 Week 8 Tuesday 18: 00-17: 00
- **Location:** The second floor of the Student Development Center

- **Attendance:** Except Li Chengji
- **Task Completion**：

  The prototype of the requirement is completed
  The software frame is completed
  The reporting framework continues
  Git is finished
- **Conference content:**

  Talk about part of the lecture slides
  Continue to discuss the report and talk about requirement for sub-groups
- **Task:**

  Read books and realize the system design
  Look at the existing framework and ready to write code

Week6

- **Time:** 2017.5.3 Week 10 Wednesday 19: 00-20: 00
- **Location:** The second floor of the Student Development Center
- **Attendance:** All attend
- **Task Completion**：

  The prototype of the requirement is completed
  The frame is completed
  The reporting framework continues
  Git is finished
- **Conference content:**

  Discuss about part of the lecture slides
  Begin to write the report
  Discuss whether the code needs to be refactored
- **Task:**

  Start to write the requirement part of the report - Li Boyu
  Determine the framework of the MVC structure - Zhang Boyang
  Continue to improve the function of the existing framework - Wang Chao

Week7

- **Time:** 2017.5.10 Week 11 Thursday 19: 00-20: 00
- **Location:** The second floor of the Student Development Center
- **Attendance:** Except Li Boyu
- **Conference content:**

  Explain part of the lecture slides
  The report has completed 10 percent
  Further discussion of previous requirements, add or modify some necessary and unnecessary

parts

- **Task:**
  Requirements - Li Boyu
  Began to write analysis part of the report- Zhang Boyang
  Improve the ticket system - Wang Chao
  Generate daily report - Xin Boyan
  Administer part - Zhao Bowen

Week8

- **Time:** 2017.5.17 Week 12 Wednesday 18: 00-19: 00
- **Location:** The second floor of the Student Development Center
- **Attendance:** All attend
- **Task Completion:**
  Requirement parts of the report is unqualified
  Analysis and design part are not finished
  The ticket system needs to be optimized
  Generate daily report is completed
  Administer part is completed
- **Conference content:**
  Discuss about testing
  The report has completed 50 percent
  Further discussion of previous requirements, add or modify some necessary and unnecessary parts

- **Task:**
  Requirement part of the report - Li Boyu
  Analysis - Zhang Boyang
  Risk- Xin Boyan
  The last payment link- Zhao Bowen

Week9

- **Time:** 2017.5.24 Week 13 Wednesday 18: 00-19: 00
- **Location:** The second floor of the Student Development Center
- **Attendance:** Except Wang Chao
- **Task Completion:**
  Requirement parts of the report is unqualified
  Analysis and design part are not finished
  The ticket system needs to be optimized
  Generate daily report is completed

Administer part is completed

- **Conference content:**

  Discuss about testing

  The report has completed 60 percent

  Further discussion of previous requirements, add or modify some necessary and unnecessary parts

- **Task:**

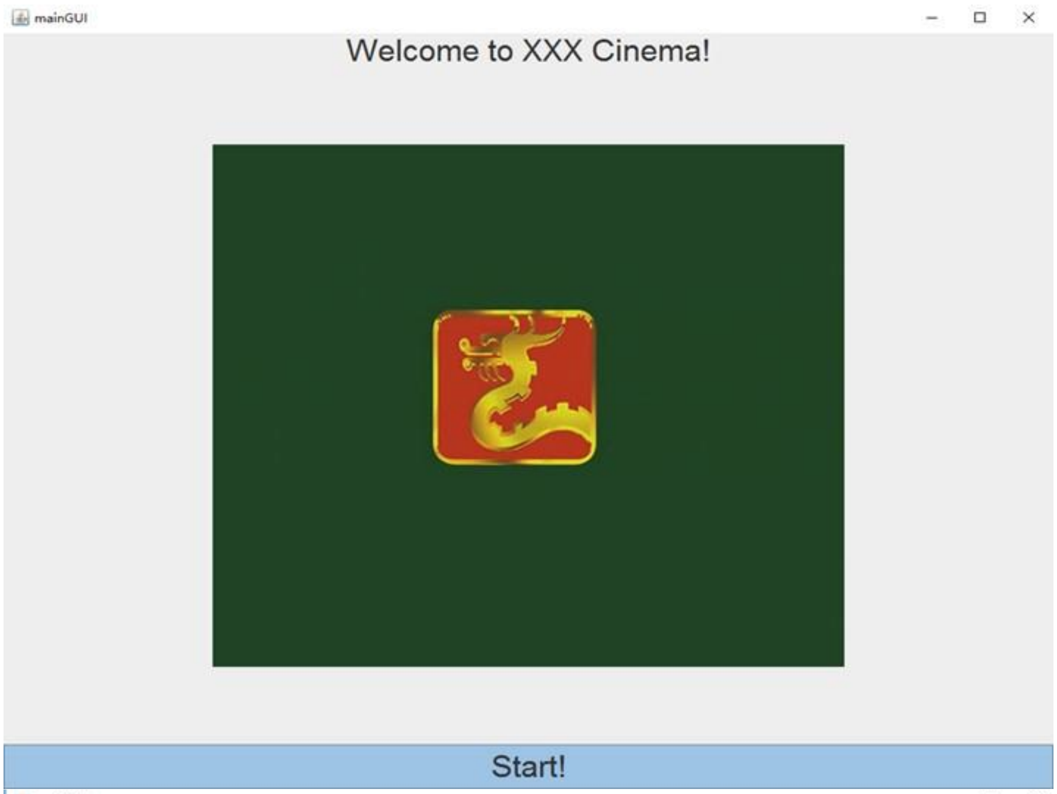
  Requirements part of the report - Li Boyu

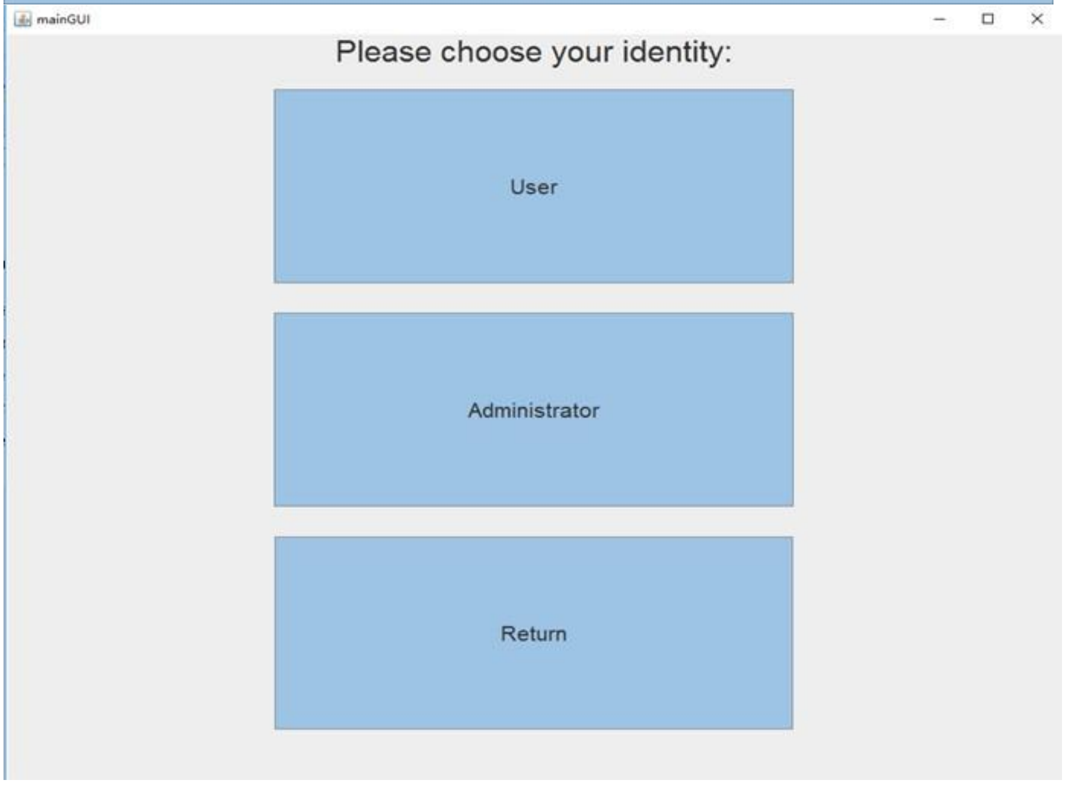
  Analysis and Design- Zhang Boyang

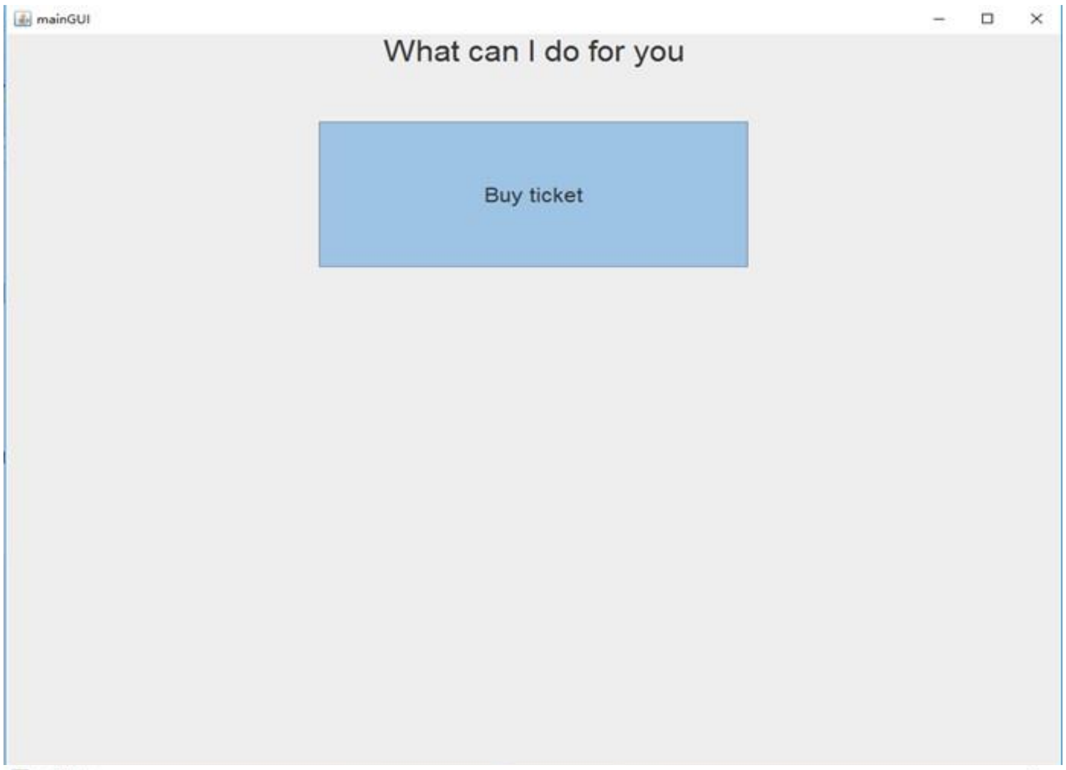
  User manual- Xin Boyan

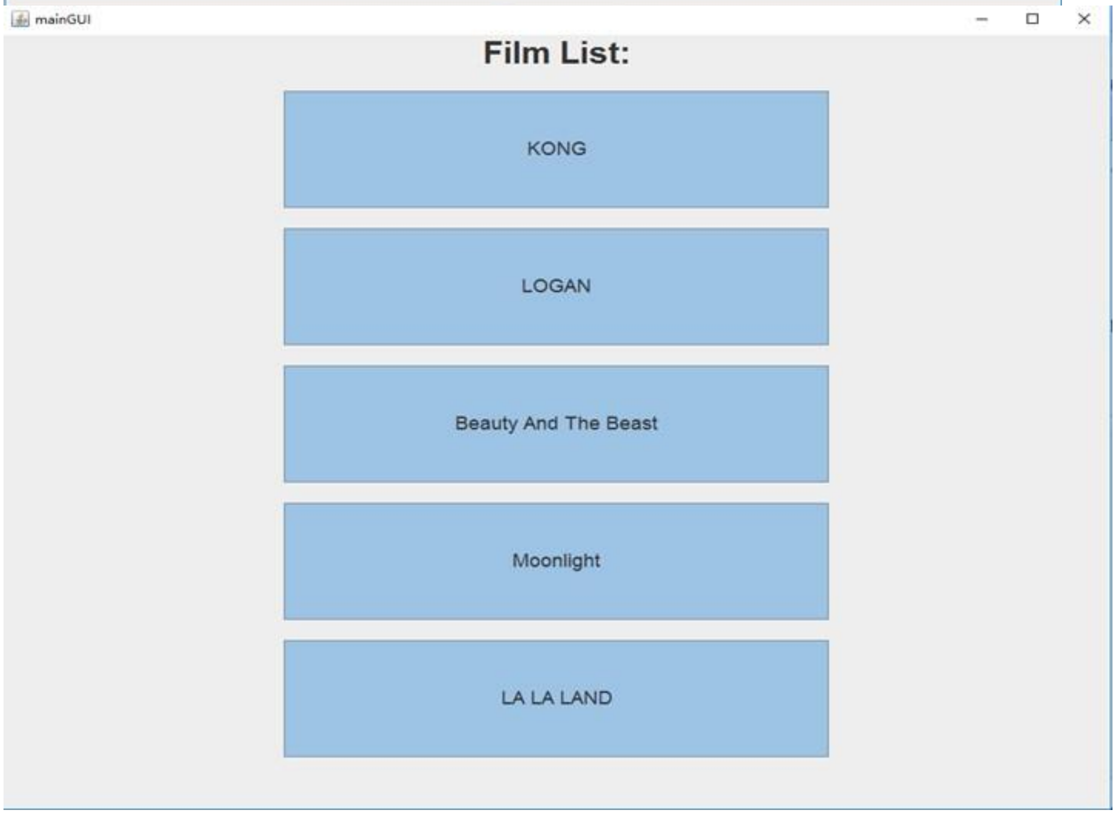
  Test - Li Chengji, Zhao Bowen

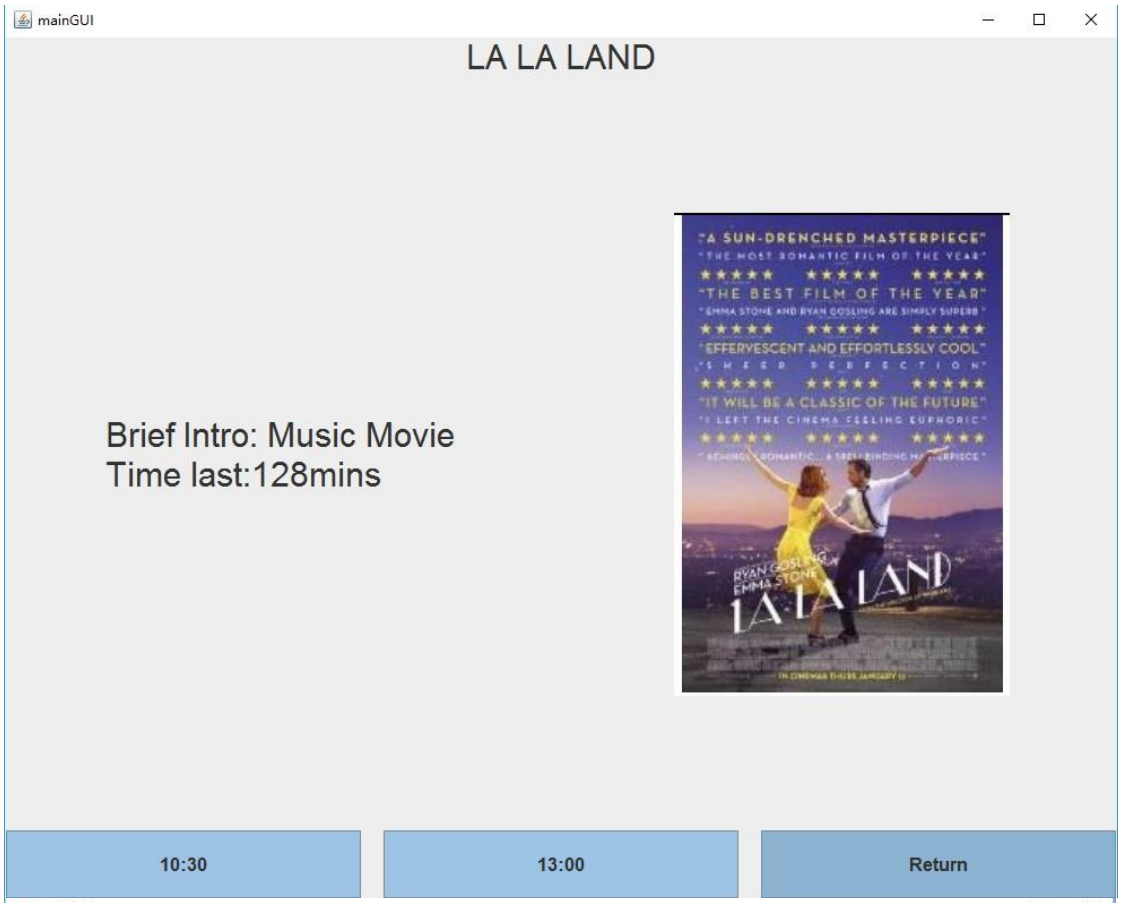
Appendix C: Main screenshots of the system





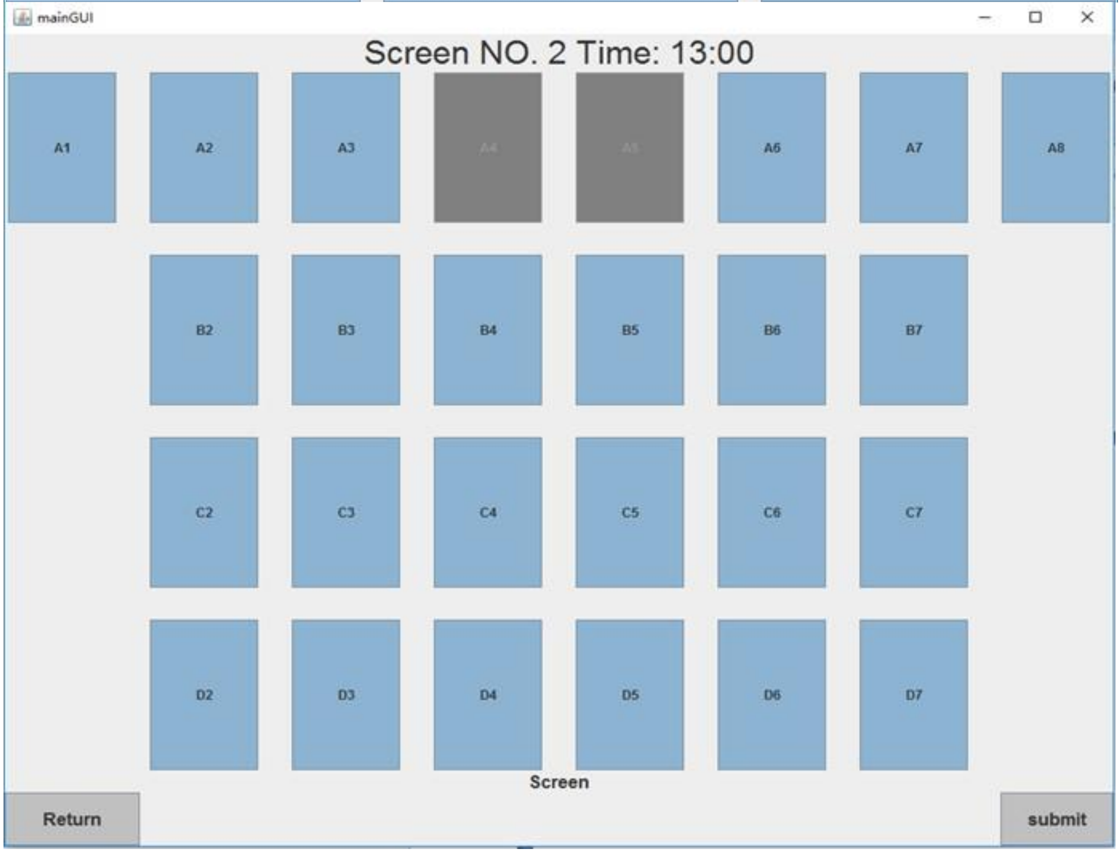

mainGUI
What can I do for you
Buy ticket


mainGUI
Film List:
KONG
LOGAN
Beauty And The Beast
Moonlight
LA LA LAND

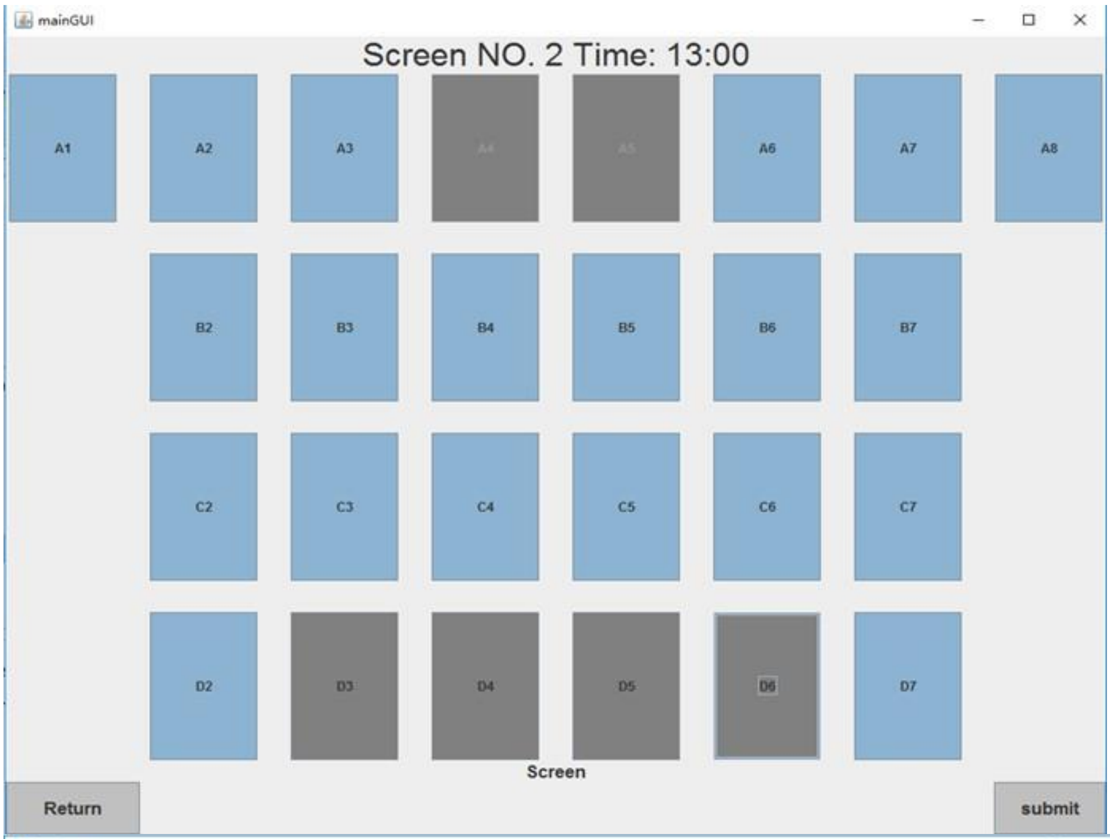

mainGUI

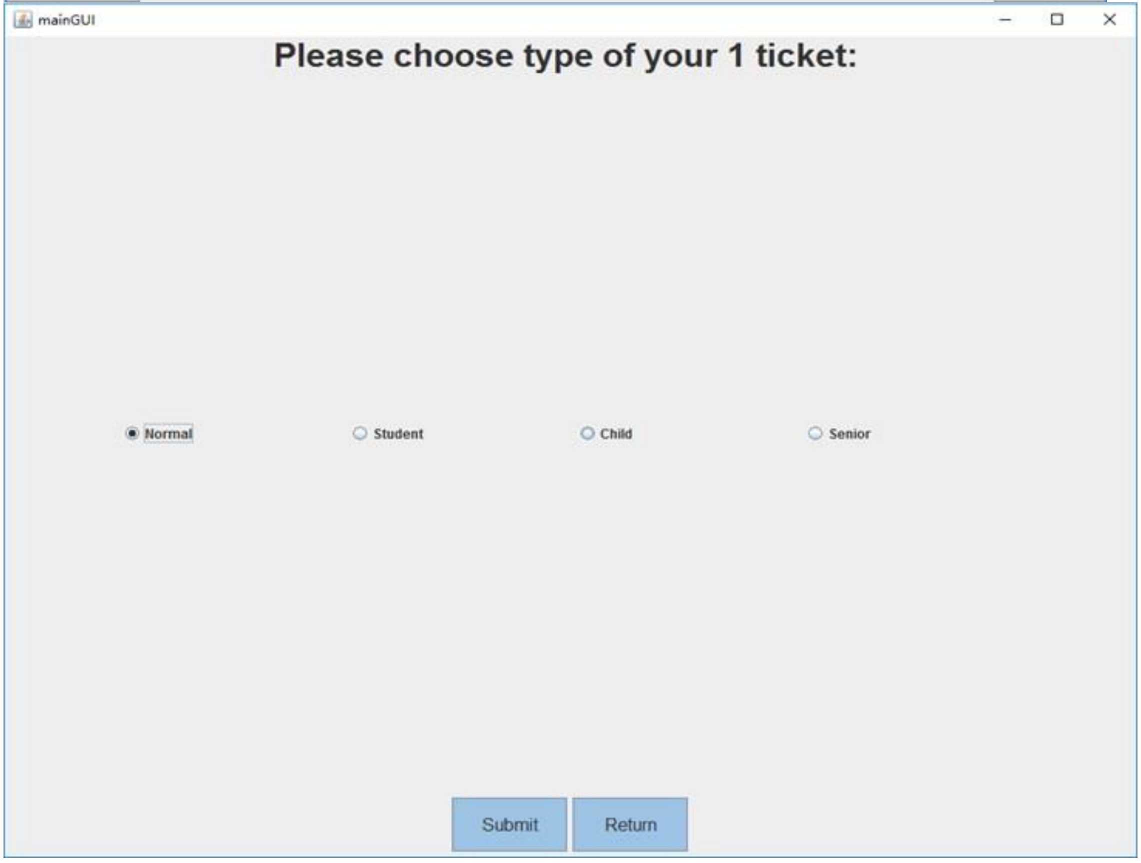
LA LA LAND

Brief Intro: Music Movie
Time last:128mins

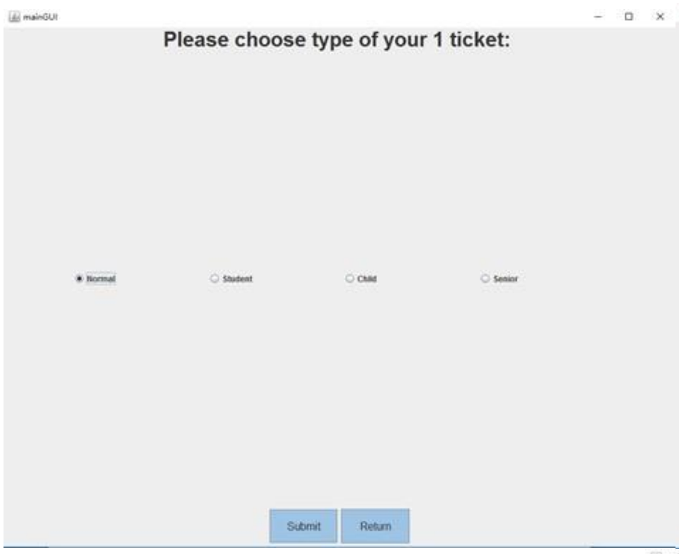

mainGUI

Screen NO. 2 Time: 13:00



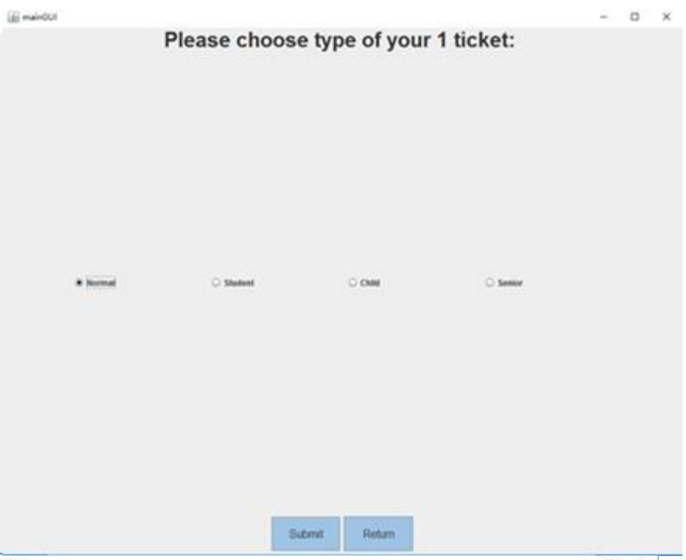


[✓ Normal] Student Child Senior



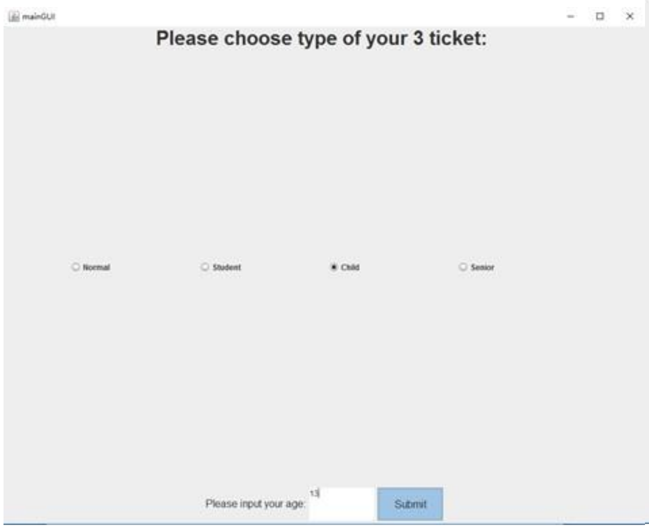
Please choose type of your 1 ticket:

[✓ Normal] Student Child Senior



Please choose type of your 1 ticket:

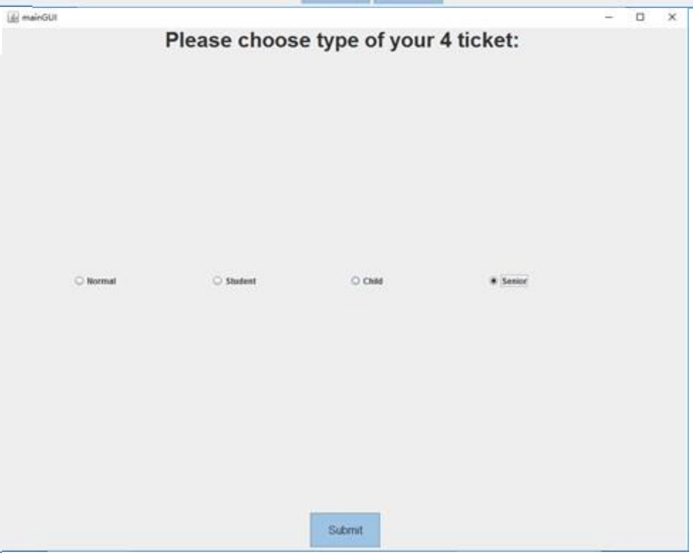
[✓ Normal] Student Child Senior



Please choose type of your 3 ticket:

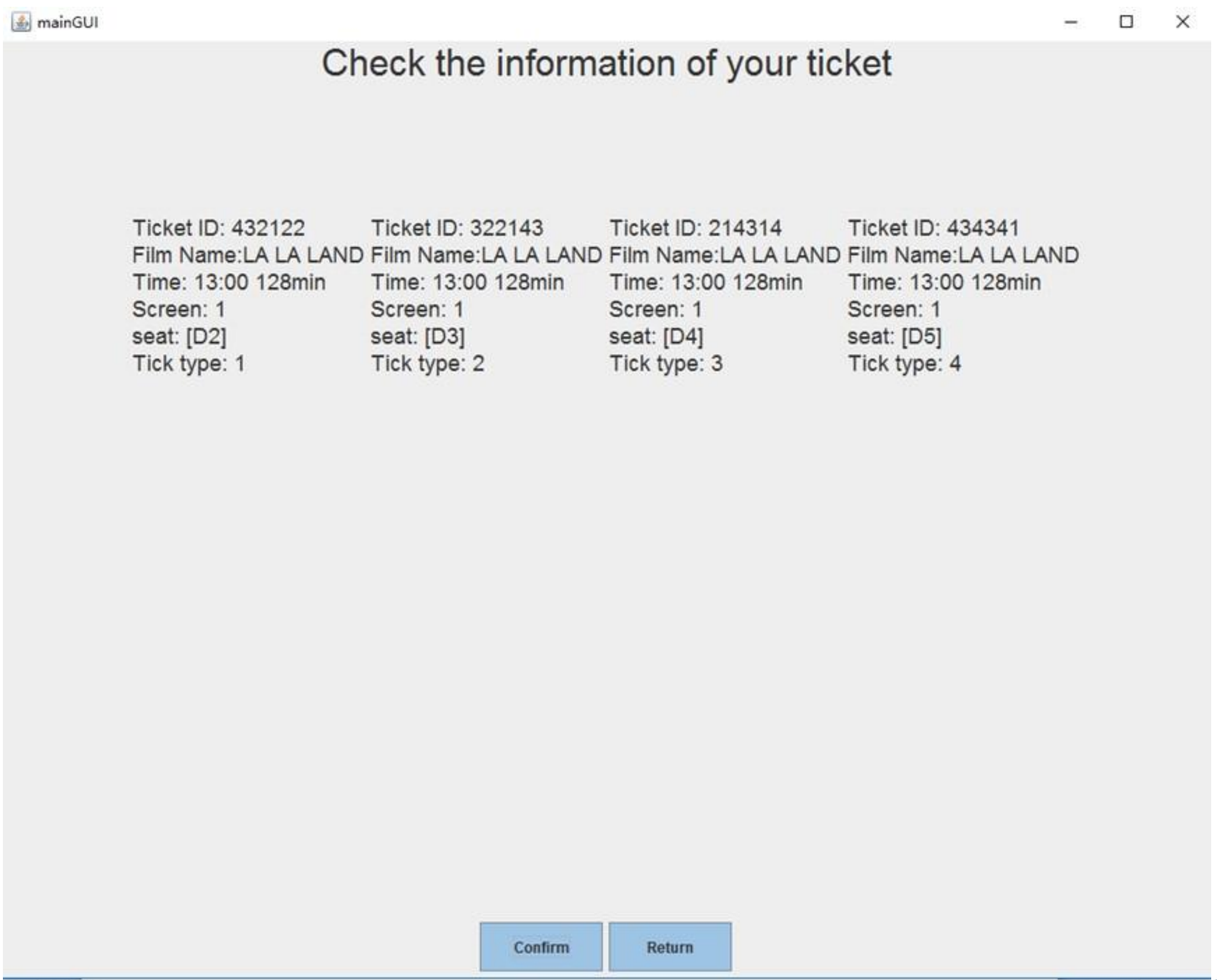
Normal Student [✓ Child] Senior

Please input your age: 13



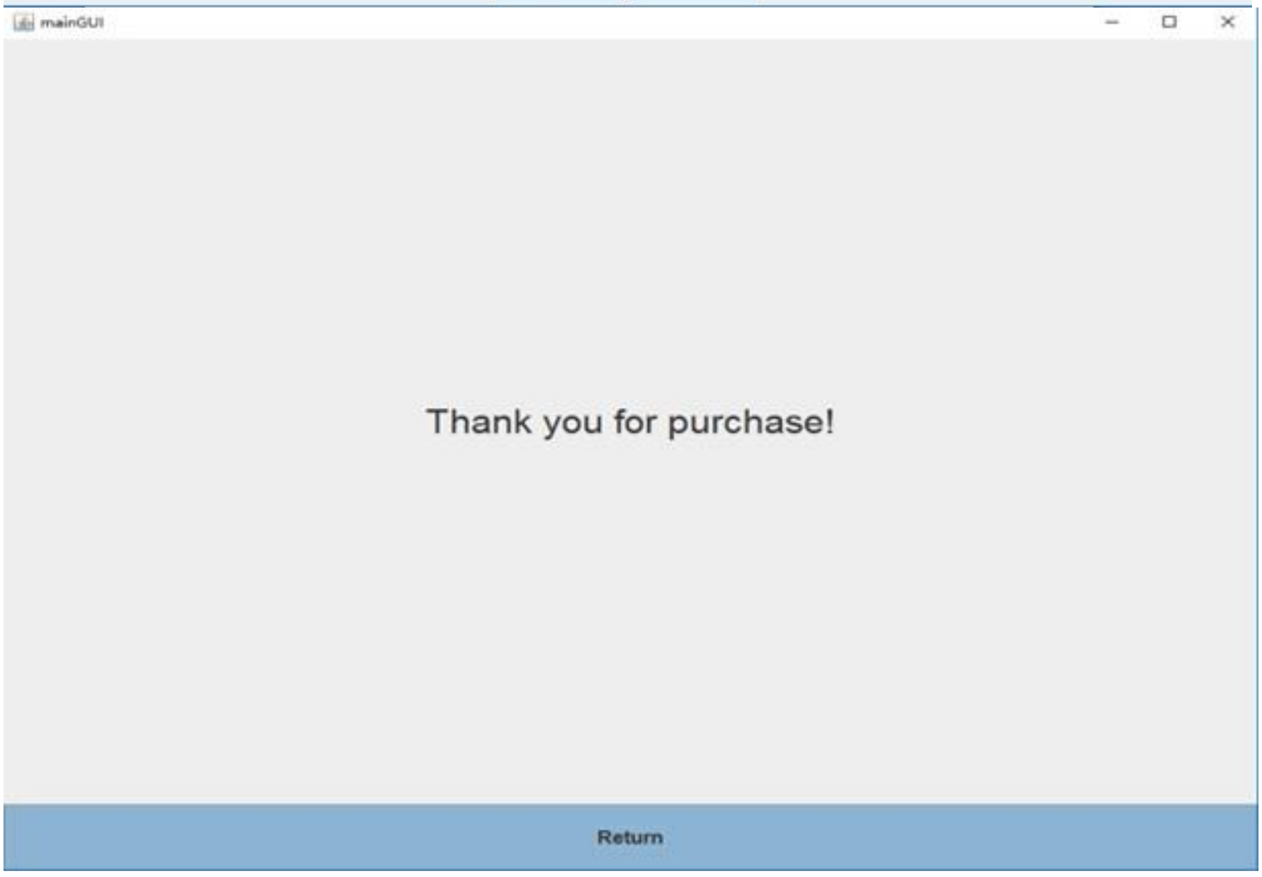
Please choose type of your 4 ticket:

Normal Student Child [✓ Senior]

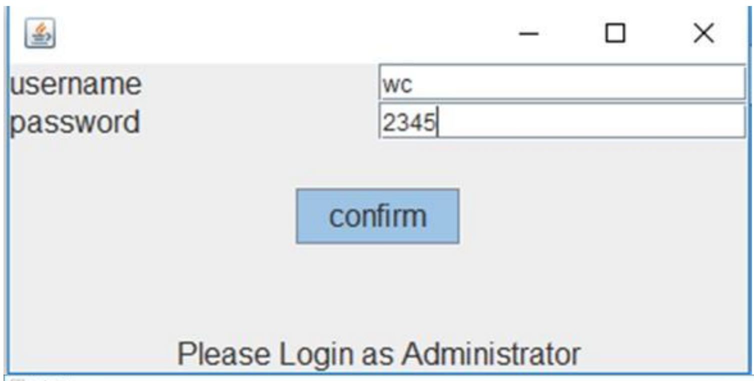


Check the information of your ticket

Ticket ID: 432122	Ticket ID: 322143	Ticket ID: 214314	Ticket ID: 434341
Film Name:LA LA LAND	Film Name:LA LA LAND	Film Name:LA LA LAND	Film Name:LA LA LAND
Time: 13:00 128min	Time: 13:00 128min	Time: 13:00 128min	Time: 13:00 128min
Screen: 1	Screen: 1	Screen: 1	Screen: 1
seat: [D2]	seat: [D3]	seat: [D4]	seat: [D5]
Tick type: 1	Tick type: 2	Tick type: 3	Tick type: 4

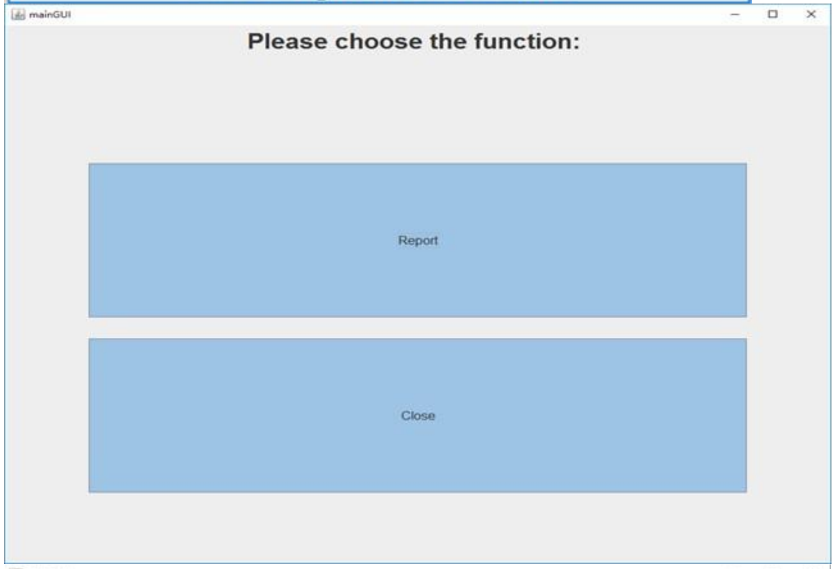


Thank you for purchase!

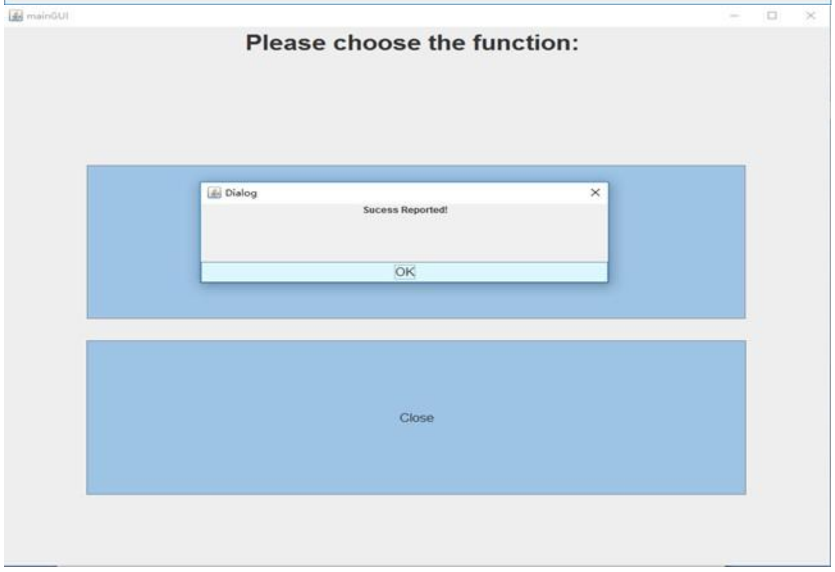


username wc
password 2345

Please Login as Administrator



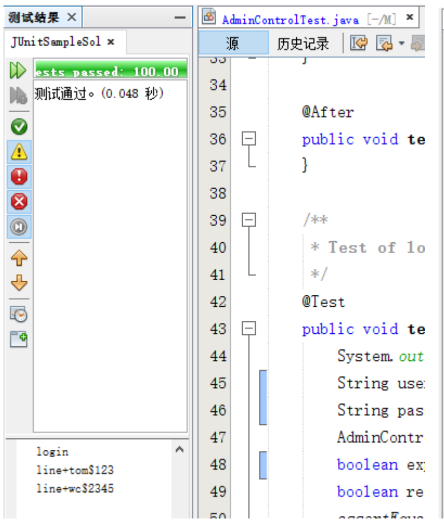
Please choose the function:



Please choose the function:

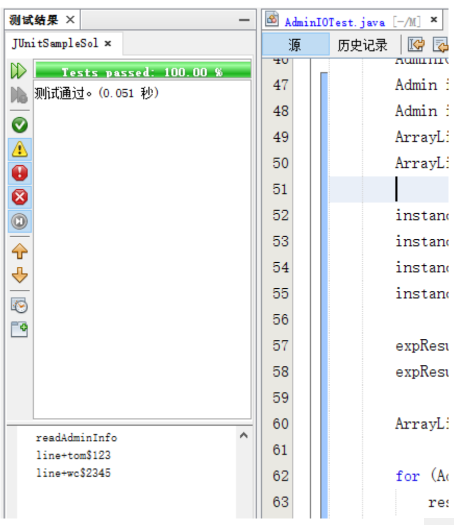
Sucess Reported!

Appendix D: Junit Test



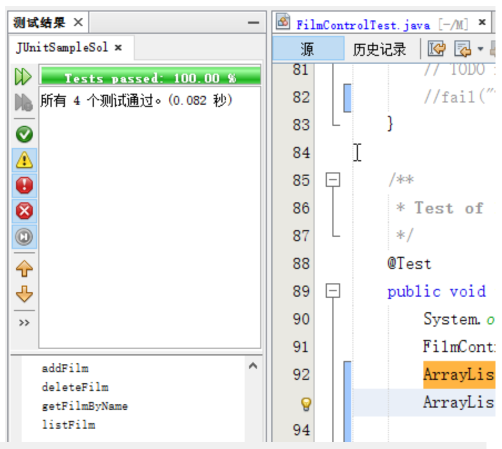
```
测试结果 ×    JUnitSampleSol ×
ests passed: 100.00
测试通过。(0.048 秒)

login
line+tom$123
line+wc$2345
```

```
AdminControlTest.java [-/M]    源  历史记录
    }

    @After
    public void te
    }

    /**
     * Test of lo
     */
    @Test
    public void te
        System.out
        String use
        String pas
        AdminContr
        boolean ex
        boolean re
```



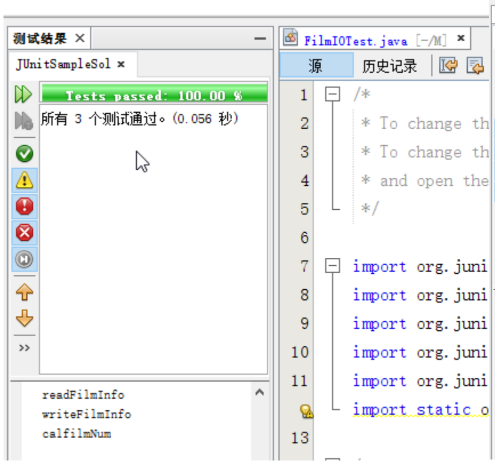
```
测试结果 ×    JUnitSampleSol ×
Tests passed: 100.00 %
测试通过。(0.051 秒)

readAdminInfo
line+tom$123
line+wc$2345
```

```
AdminIOTest.java [-/M]    源  历史记录
        Admin :
        Admin :
        ArrayLi
        ArrayLi

        instanc
        instanc
        instanc
        instanc

        expResu
        expResu

        ArrayLi

        for (Ad
            res
```



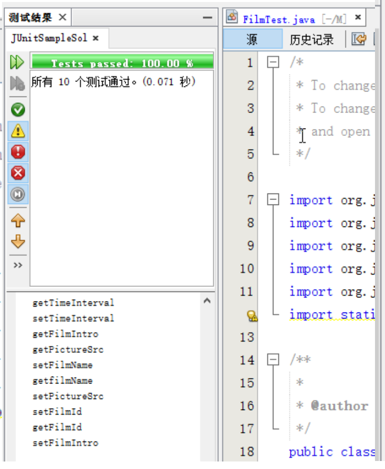
```
测试结果 ×    JUnitSampleSol ×
Tests passed: 100.00 %
所有 4 个测试通过。(0.082 秒)

addFilm
deleteFilm
getFilmByName
listFilm
```

```
FilmControlTest.java [-/M]    源  历史记录
        // TODO
        //fail("
    }

    /**
     * Test of
     */
    @Test
    public void
        System.o
        FilmCont
        ArrayLis
        ArrayLis
```



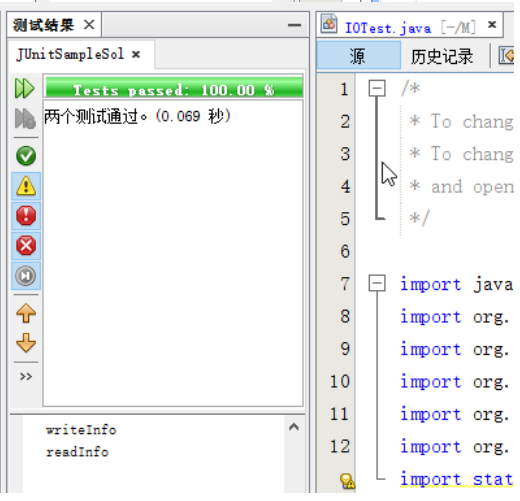
```
测试结果 ×    JUnitSampleSol ×
Tests passed: 100.00 %
所有 3 个测试通过。(0.056 秒)

readFilmInfo
writeFilmInfo
calfilmNum
```

```
FilmIOTest.java [-/M]    源  历史记录
/*
 * To change th
 * To change th
 * and open the
 */

import org.juni
import org.juni
import org.juni
import org.juni
import org.juni
import static o
```



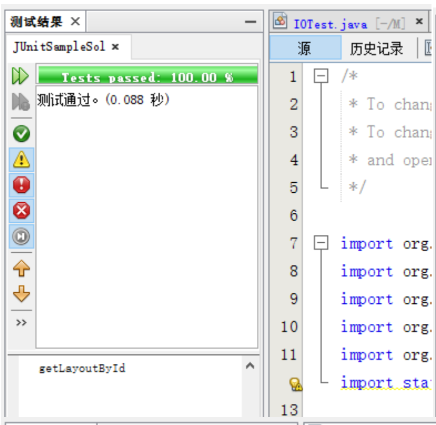
```
测试结果 ×    JUnitSampleSol ×
Tests passed: 100.00 %
所有 10 个测试通过。(0.071 秒)

getTimeInterval
setTimeInterval
getFilmIntro
getPictureSrc
setFilmName
getfilmName
setPictureSrc
setFilmId
getFilmId
setFilmIntro
```

```
FilmTest.java [-/M]    源  历史记录
/*
 * To change
 * To change
 * and open
 */

import org.j
import org.j
import org.j
import org.j
import org.j
import stati

/**
 *
 * @author
 */
public class
```



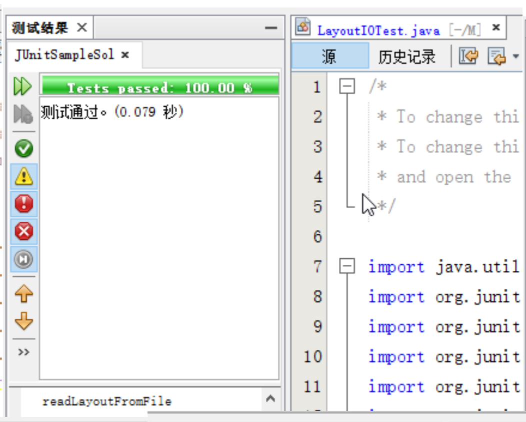
```
测试结果 ×    JUnitSampleSol ×
Tests passed: 100.00 %
两个测试通过。(0.069 秒)

writeInfo
readInfo
```

```
IOTest.java [-/M]    源  历史记录
/*
 * To chang
 * To chang
 * and open
 */

import java
import org.
import org.
import org.
import org.
import org.
import stat
```



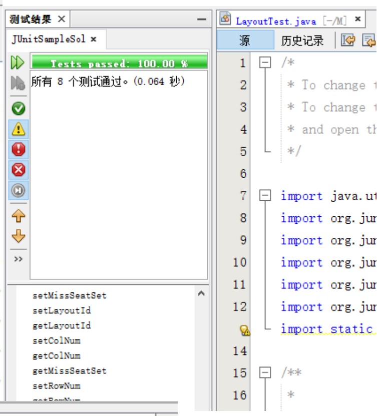
```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
测试通过。(0.088 秒)
getLayoutById
```

```
IOTest.java [-/M] ×
源 历史记录
/*
 * To chan
 * To chan
 * and ope
 */

import org.
import org.
import org.
import org.
import org.
import sta
```



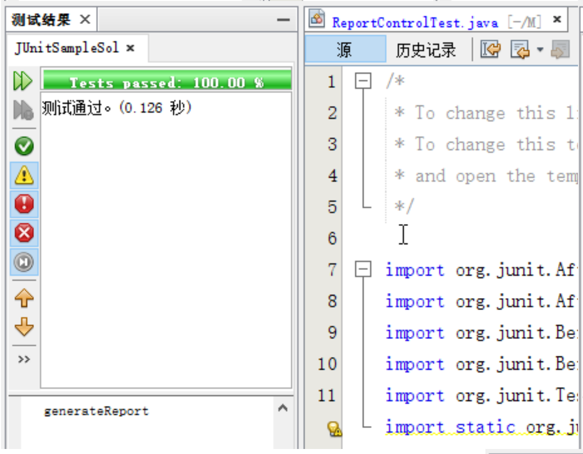
```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
测试通过。(0.079 秒)
readLayoutFromFile
```

```
LayoutIOTest.java [-/M] ×
源 历史记录
/*
 * To change thi
 * To change thi
 * and open the
 */

import java.util
import org.junit
import org.junit
import org.junit
import org.junit
```



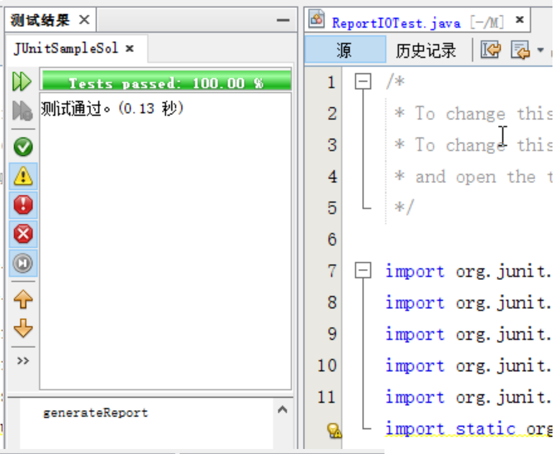
```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
所有 8 个测试通过。(0.064 秒)
setMissSeatSet
setLayoutId
getLayoutId
setColNum
getColNum
getMissSeatSet
setRowNum
```

```
LayoutTest.java [-/M] ×
源 历史记录
/*
 * To change
 * To change
 * and open th
 */

import java.ut
import org.jur
import org.jur
import org.jur
import org.jur
import org.jur
import static

/**
 *
```



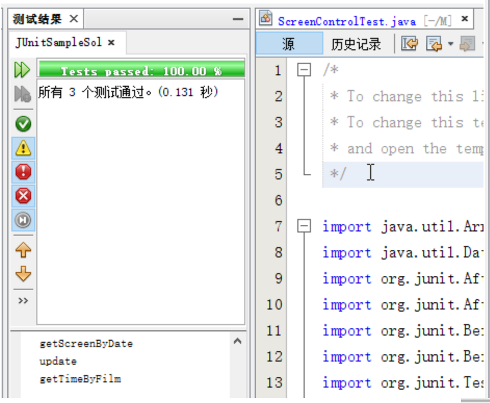
```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
测试通过。(0.126 秒)
generateReport
```

```
ReportControlTest.java [-/M] ×
源 历史记录
/*
 * To change this l
 * To change this t
 * and open the tem
 */

import org.junit.Af
import org.junit.Af
import org.junit.Be
import org.junit.Be
import org.junit.Te
import static org.j
```



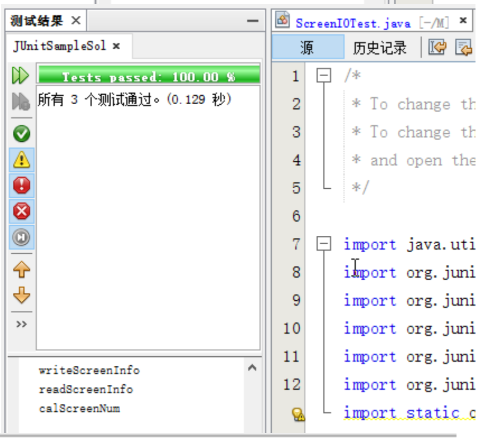
```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
测试通过。(0.13 秒)
generateReport
```

```
ReportIOTest.java [-/M] ×
源 历史记录
/*
 * To change this
 * To change this
 * and open the t
 */

import org.junit.
import org.junit.
import org.junit.
import org.junit.
import org.junit.
import static org
```



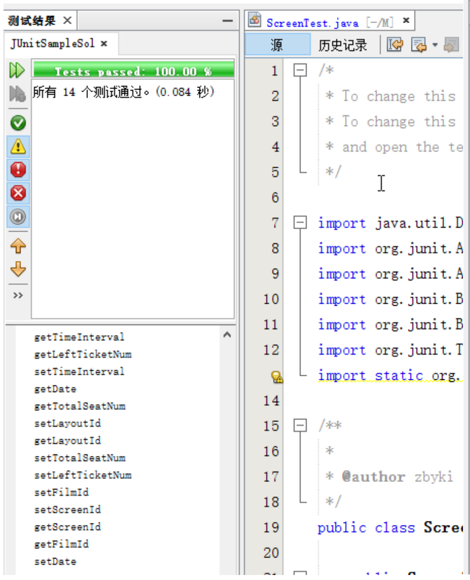
```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
所有 3 个测试通过。(0.131 秒)
getScreenByDate
update
getTimeByFilm
```

```
ScreenControlTest.java [-/M] ×
源 历史记录
/*
 * To change this l
 * To change this t
 * and open the tem
 */

import java.util.Ar
import java.util.Da
import org.junit.Af
import org.junit.Af
import org.junit.Be
import org.junit.Be
import org.junit.Te
```



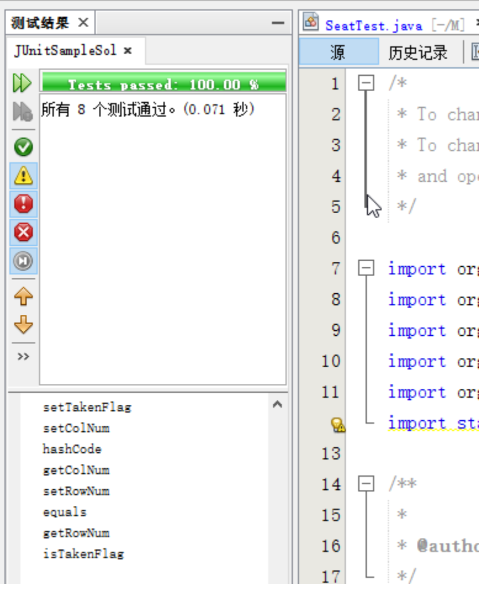
```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
所有 3 个测试通过。(0.129 秒)
writeScreenInfo
readScreenInfo
calScreenNum
```

```
ScreenIOTest.java [-/M] ×
源 历史记录
/*
 * To change th
 * To change th
 * and open the
 */

import java.uti
import org.juni
import org.juni
import org.juni
import org.juni
import org.juni
import static c
```



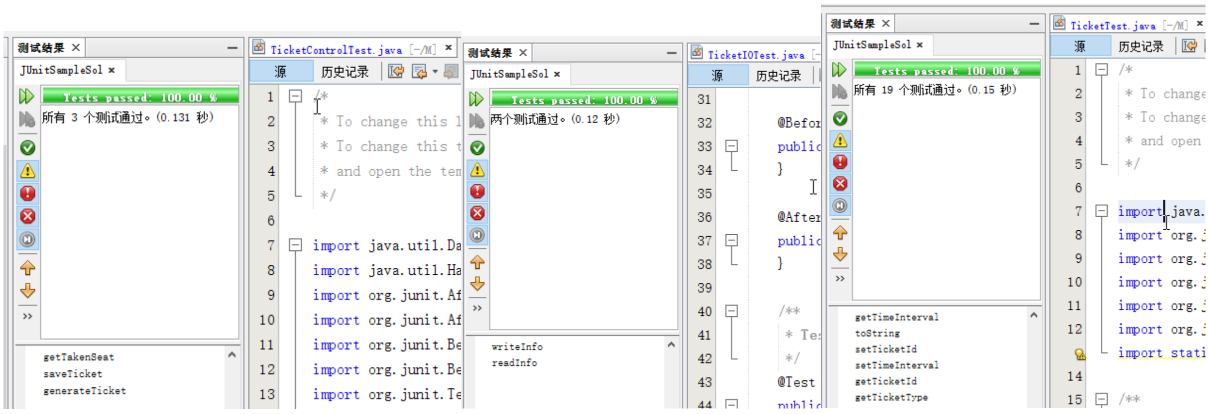
```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
所有 14 个测试通过。(0.084 秒)
getTimeInterval
getLeftTicketNum
setTimeInterval
getDate
getTotalSeatNum
setLayoutId
getLayoutId
setTotalSeatNum
setLeftTicketNum
setFilmId
setScreenId
getScreenId
getFilmId
setDate
```

```
ScreenTest.java [-/M] ×
源 历史记录
/*
 * To change this
 * To change this
 * and open the te
 */

import java.util.D
import org.junit.A
import org.junit.A
import org.junit.B
import org.junit.B
import org.junit.T
import static org.

/**
 *
 * @author zbyki
 */
public class Scre
```



```
测试结果 ×
JUnitSampleSol ×
Tests passed: 100.00 %
所有 8 个测试通过。(0.071 秒)
setTakenFlag
setColNum
hashCode
getColNum
setRowNum
equals
getRowNum
isTakenFlag
```

```
SeatTest.java [-/M] ×
源 历史记录
/*
 * To chan
 * To chan
 * and ope
 */

import or
import or
import or
import or
import or
import st

/**
 *
 * @autho
 */
```



```
Tests passed: 100.00 %
所有 3 个测试通过。(0.131 秒)

getTakenSeat
saveTicket
generateTicket
```

```
/*
 * To change this l
 * To change this t
 * and open the tem
 */

import java.util.Da
import java.util.Ha
import org.junit.Af
import org.junit.Af
import org.junit.Be
import org.junit.Be
import org.junit.Te
```

```
Tests passed: 100.00 %
两个测试通过。(0.12 秒)

writeInfo
readInfo
```

```

    @Befor
    public
    }

    @After
    public
    }

    /**
     * Tes
     */
    @Test
    public
```

```
Tests passed: 100.00 %
所有 19 个测试通过。(0.15 秒)

getTimeInterval
toString
setTicketId
setTimeInterval
getTicketId
getTicketType
```

```
/*
 * To change
 * To change
 * and open
 */

import java.
import org.j
import org.j
import org.j
import org.j
import org.j
import stati

/**
```