图书馆管理系统建模
user
You are an experienced facilitator in Event Storming workshops. Based on the following
problem description, your task is to build a domain model using the principles of Event
Storming. Please complete the following steps:

Step1. Identify the following elements from the description:
Domain Events: Business-relevant facts that have occurred in the system. They are written
with verbs in the past tense.
* Commands*: Explicit actions that trigger events. These are written with verbs in the
present tense. Commands are given by an Actor, either a User or a System.
* Actors*: Users or external systems that initiate commands.
* Hotspots*(if existence): Areas of ambiguity, complexity, or known issues.
* Policies*(if existence): These are pre-defined rules that are applied after a Domain Event to
determine the next step.
Notes: Each domain event has a command and an Actor (i.e. users or external systems).

Step2. List the directed binary tuples related to each event line by line, and may include:
[Actor: actor1, Command: command1], [Command: command 1, Event: event 1], (if
have hotspot)[Hotspot: hotspot1, Event: event1], (if have policy)[Policy: policy1, Event:

event1], [Event: event1, Event: event2] (Only when one event directly follows from
another)

Step3. After listing the tuples, With the Event flow as the main line, visualization is carried
out using PlantUML according to the Event Storming principle. Elements of different types
have different colors, and the color type descriptions should be provided.

Problem Description:对《图书管理系统》进行事件风暴建模，要求严格按照上述步骤，输出 markdown 文档，尽可能考虑周全，注意输出格式与结构
Case Study: [图书馆管理系统]
[事件风暴建模]
ChatGPT
以下是针对 **《图书管理系统》** 的**事件风暴建模**，严格按照题目中所述的步骤输出，采用 Markdown 格式完整呈现：

📚 图书馆管理系统：事件风暴建模

✅ Step 1. 元素识别

🎯 Domain Events（领域事件）
- 读者注册成功
- 读者信息已更新
- 图书已上架
- 图书信息已更新
- 图书被借出
- 图书归还成功
- 图书逾期未还
- 逾期罚款已生成
- 罚款已支付
- 图书被预订
- 图书预订取消
- 预约图书已到馆
- 预约图书已借出
- 图书遗失上报
- 图书遗失赔偿完成
- 管理员已添加新图书
- 管理员已删除图书
- 图书库存已更新

⚡ Commands（命令）
- 注册读者
- 更新读者信息
- 添加新图书
- 更新图书信息
- 借出图书
- 归还图书
- 生成逾期罚款
- 支付罚款
- 预订图书

- 取消预订
- 通知预约图书到馆
- 借出预约图书
- 报告图书遗失
- 赔偿遗失图书
- 删除图书
- 更新图书库存

👤 Actors（参与者）
- 读者
- 图书管理员
- 系统

🔥 Hotspots（热点问题）
- 多人同时预订同一本书时的优先级判定
- 归还图书时逾期罚款的计算规则
- 图书遗失的赔偿金额如何评估

📜 Policies（策略）
- 图书逾期后自动生成罚款
- 预约图书到馆后24小时内未取将自动取消
- 每位读者最多可借5本书

✅ Step 2. 元组列出（事件驱动流程）

```text
[Actor: 读者, Command: 注册读者]
[Command: 注册读者, Event: 读者注册成功]

[Actor: 读者, Command: 更新读者信息]
[Command: 更新读者信息, Event: 读者信息已更新]

[Actor: 图书管理员, Command: 添加新图书]
[Command: 添加新图书, Event: 图书已上架]

[Actor: 图书管理员, Command: 更新图书信息]
[Command: 更新图书信息, Event: 图书信息已更新]

[Actor: 读者, Command: 借出图书]
[Command: 借出图书, Event: 图书被借出]

[Actor: 读者, Command: 归还图书]
[Command: 归还图书, Event: 图书归还成功]

[Policy: 图书逾期后自动生成罚款, Event: 图书逾期未还]
[Command: 生成逾期罚款, Event: 逾期罚款已生成]
[Hotspot: 归还图书时逾期罚款的计算规则, Event: 逾期罚款已生成]

[Actor: 读者, Command: 支付罚款]
```

```
[Command: 支付罚款, Event: 罚款已支付]

[Actor: 读者, Command: 预订图书]
[Command: 预订图书, Event: 图书被预订]
[Hotspot: 多人同时预订同一本书时的优先级判定, Event: 图书被预订]

[Actor: 读者, Command: 取消预订]
[Command: 取消预订, Event: 图书预订取消]

[Actor: 系统, Command: 通知预约图书到馆]
[Command: 通知预约图书到馆, Event: 预约图书已到馆]
[Policy: 预约图书到馆后24小时内未取将自动取消, Event: 预约图书已到馆]

[Actor: 读者, Command: 借出预约图书]
[Command: 借出预约图书, Event: 预约图书已借出]
[Event: 预约图书已到馆, Event: 预约图书已借出]

[Actor: 读者, Command: 报告图书遗失]
[Command: 报告图书遗失, Event: 图书遗失上报]

[Actor: 读者, Command: 赔偿遗失图书]
[Command: 赔偿遗失图书, Event: 图书遗失赔偿完成]
[Hotspot: 图书遗失的赔偿金额如何评估, Event: 图书遗失赔偿完成]

[Actor: 图书管理员, Command: 删除图书]
[Command: 删除图书, Event: 管理员已删除图书]

[Actor: 图书管理员, Command: 更新图书库存]
[Command: 更新图书库存, Event: 图书库存已更新]
```

✅ Step 3. PlantUML 可视化（事件主线流程）

```
@startuml
' 颜色定义
skinparam backgroundColor #FFFFFF
skinparam componentStyle rectangle

' 定义颜色
skinparam class {
    BackgroundColor<<Event>> #FDF2E9
    BackgroundColor<<Command>> #EBF5FB
    BackgroundColor<<Actor>> #D5F5E3
    BackgroundColor<<Policy>> #FCF3CF
    BackgroundColor<<Hotspot>> #F5B7B1
}

' Actor
class "读者" <<Actor>>
```

```
class "图书管理员" <<Actor>>
class "系统" <<Actor>>

' Commands
class "注册读者" <<Command>>
class "更新读者信息" <<Command>>
class "添加新图书" <<Command>>
class "更新图书信息" <<Command>>
class "借出图书" <<Command>>
class "归还图书" <<Command>>
class "生成逾期罚款" <<Command>>
class "支付罚款" <<Command>>
class "预订图书" <<Command>>
class "取消预订" <<Command>>
class "通知预约图书到馆" <<Command>>
class "借出预约图书" <<Command>>
class "报告图书遗失" <<Command>>
class "赔偿遗失图书" <<Command>>
class "删除图书" <<Command>>
class "更新图书库存" <<Command>>

' Events
class "读者注册成功" <<Event>>
class "读者信息已更新" <<Event>>
class "图书已上架" <<Event>>
class "图书信息已更新" <<Event>>
class "图书被借出" <<Event>>
class "图书归还成功" <<Event>>
class "图书逾期未还" <<Event>>
class "逾期罚款已生成" <<Event>>
class "罚款已支付" <<Event>>
class "图书被预订" <<Event>>
class "图书预订取消" <<Event>>
class "预约图书已到馆" <<Event>>
class "预约图书已借出" <<Event>>
class "图书遗失上报" <<Event>>
class "图书遗失赔偿完成" <<Event>>
class "管理员已删除图书" <<Event>>
class "图书库存已更新" <<Event>>

' Policies
class "图书逾期后自动生成罚款" <<Policy>>
class "预约图书到馆后24小时内未取将自动取消" <<Policy>>

' Hotspots
class "逾期罚款的计算规则" <<Hotspot>>
class "多人同时预订优先级判定" <<Hotspot>>
class "图书遗失赔偿金额评估" <<Hotspot>>

' 连接
"读者" --> "注册读者"
"注册读者" --> "读者注册成功"

"读者" --> "更新读者信息"
"更新读者信息" --> "读者信息已更新"

"图书管理员" --> "添加新图书"
```

```
"添加新图书" --> "图书已上架"

"图书管理员" --> "更新图书信息"
"更新图书信息" --> "图书信息已更新"

"读者" --> "借出图书"
"借出图书" --> "图书被借出"

"读者" --> "归还图书"
"归还图书" --> "图书归还成功"
"图书归还成功" --> "图书逾期未还"
"图书逾期未还" --> "图书逾期后自动生成罚款"
"生成逾期罚款" --> "逾期罚款已生成"
"逾期罚款已生成" --> "逾期罚款的计算规则"

"读者" --> "支付罚款"
"支付罚款" --> "罚款已支付"

"读者" --> "预订图书"
"预订图书" --> "图书被预订"
"图书被预订" --> "多人同时预订优先级判定"

"读者" --> "取消预订"
"取消预订" --> "图书预订取消"

"系统" --> "通知预约图书到馆"
"通知预约图书到馆" --> "预约图书已到馆"
"预约图书已到馆" --> "预约图书到馆后24小时内未取将自动取消"

"读者" --> "借出预约图书"
"借出预约图书" --> "预约图书已借出"
"预约图书已到馆" --> "预约图书已借出"

"读者" --> "报告图书遗失"
"报告图书遗失" --> "图书遗失上报"

"读者" --> "赔偿遗失图书"
"赔偿遗失图书" --> "图书遗失赔偿完成"
"图书遗失赔偿完成" --> "图书遗失赔偿金额评估"

"图书管理员" --> "删除图书"
"删除图书" --> "管理员已删除图书"

"图书管理员" --> "更新图书库存"
"更新图书库存" --> "图书库存已更新"
@enduml
```

🎨 颜色说明

类型	描述	颜色
Actor	用户或外部系统	#D5F5E3
Command	动作命令	#EBF5FB
Event	领域事件	#FDF2E9
Policy	策略规则	#FCF3CF
Hotspot	模糊或复杂区域	#F5B7B1

如需后续生成完整 PlantUML 图像或导出，请告知是否需要图形化文件或继续扩展某个子领域。 user You are an experienced facilitator in Event Storming workshops. Based on the following problem description, your task is to build a domain model using the principles of Event Storming. Please complete the following steps:

Step1. Identify the following elements from the description: ***Domain Events***: Business-relevant facts that have occurred in the system. They are written with verbs in the past tense.

- Commands*: Explicit actions that trigger events. These are written with verbs in the present tense. Commands are given by an Actor, either a User or a System.
- Actors*: Users or external systems that initiate commands.
- Hotspots*(if existence): Areas of ambiguity, complexity, or known issues.
- Policies*(if existence): These are pre-defined rules that are applied after a Domain Event to determine the next step. Notes： Each domain event has a command and an Actor (i.e. users or external systems).

Step2. List the directed binary tuples related to each event line by line, and may include: [Actor: actor1, Command: command1], [Command: command 1, Event: event 1], (if have hotspot)[Hotspot: hotspot1, Event: event1], (if have policy)[Policy: policy1, Event: event1], [Event: event1, Event: event2] (Only when one event directly follows from another)

Step3. After listing the tuples, With the Event flow as the main line, visualization is carried out using PlantUML according to the Event Storming principle. Elements of different types have different colors, and the color type descriptions should be provided.

Problem Description: use the Event Storming workshop approach to conduct domain modeling in the the Library Management System Case Study: [the Library Management System] [use the Event Storming workshop approach to conduct domain modeling in the the Library Management System]

1. 严格遵循上述步骤
2. 在上次回答的基础上
3. 继续完善回答的内容
4. 最终输出 markdown 文件，命名为《图书管理系统-事件风暴建模.md》 ChatGPT { "name": "图书管理系统-事件风暴建模", "type": "document", "content": "# \u56fe\u4e66\u7ba1\u7406\u7cfb\u7edf - \u4e8b\u4ef6\u98ce\u66b4\u5efa\u6a21\n\n## \u2705 Step 1. \u5143\u7d20\u8bc6\u522b\n\n### \ud83c\udfaf Domain Events\n- \u8bfb\u8005\u6ce8\u518c\u6210\u529f\n- \u8bfb\u8005\u4fe1\u606f\u5df2\u66f4\u65b0\n- \u56fe\u4e66\u5df2\u4e0a\u67b6\n- \u56fe\u4e66\u4fe1\u606f\u5df2\u66f4\u65b0\n- \u56fe\u4e66\u88ab\u501f\u51fa\n- \u56fe\u4e66\u5df2\u5f52\u8fd8\n- \u56fe\u4e66\u903e\u671f\u672a\u8fd8\n- \u903e\u671f\u7f5a\u6b3e\u5df2\u751f\u6210\n- \u7f5a\u6b3e\u5df2\u652f\u4ed8\n- \u56fe\u4e66\u88ab\u9884\u8ba2\n- \u56fe\u4e66\u9884\u8ba2\u53d6\u6d88\n- \u9884\u7ea6\u56fe\u4e66\u5df2\u5230\u9986\n- \u9884\u7ea6\u56fe\u4e66\u5df2\u501f\u51fa\n- \u56fe\u4e66\u9057\u5931\u4e0a\u62a5\n- \u56fe\u4e66\u9057\u5931\u8d54\u507f\u5b8c\u6210\n- \u7ba1\u7406\u5458\u5df2\u6dfb\u52a0\u65b0\u56fe\u4e66\n- \u7ba1\u7406\u5458\u5df2\u5220\u9664\u56fe\u4e66\n- \u56fe\u4e66\u5e93\u5b58\u5df2\u66f4\u65b0\n\n### \u26a1 Commands\n- \u6ce8\u518c\u8bfb\u8005\n- \u66f4\u65b0\u8bfb\u8005\u4fe1\u606f\n- \u6dfb\u52a0\u65b0\u56fe\u4e66\n- \u66f4\u65b0\u56fe\u4e66\u4fe1\u606f\n- \u501f\u51fa\u56fe\u4e66\n- \u5f52\u8fd8\u56fe\u4e66\n- \u751f\u6210\u903e\u671f\u7f5a\u6b3e\n- \u652f\u4ed8\u7f5a\u6b3e\n- \u9884\u8ba2\u56fe\u4e66\n- \u53d6\u6d88\u9884\u8ba2\n- \u901a\u77e5\u9884\u7ea6\u56fe\u4e66\u5230\u9986\n- \u501f\u51fa\u9884\u7ea6\u56fe\u4e66\n- \u62a5\u544a\u56fe\u4e66\u9057\u5931\n- \u8d54\u507f\u9057\u5931\u56fe\u4e66\n- \u5220\u9664\u56fe\u4e66\n- \u66f4\u65b0\u56fe\u4e66\u5e93\u5b58\n\n### \ud83d\udc64 Actors\n- \u8bfb\u8005\n- \u56fe\u4e66\u7ba1\u7406\u5458\n- \u7cfb\u7edf\n\n### \ud83d\udd25 Hotspots\n- \u591a\u4eba\u540c\u65f6\u9884\u8ba2\u540c\u4e00\u672c\u4e66\u65f6\u7684\u4f18\u5148\u7ea7\u5224\u5b9a\n-

\u5f52\u8fd8\u56fe\u4e66\u65f6\u903e\u671f\u7f5a\u6b3e\u7684\u8ba1\u7b97\u89c4\u5219\n-
\u56fe\u4e66\u9057\u5931\u7684\u8d54\u507f\u91d1\u989d\u5982\u4f55\u8bc4\u4f30\n\n### \ud83d\udcdc Policies\n-
\u56fe\u4e66\u903e\u671f\u540e\u81ea\u52a8\u751f\u6210\u7f5a\u6b3e\n-
\u9884\u7ea6\u56fe\u4e66\u5230\u9986\u540e24\u5c0f\u65f6\u5185\u672a\u53d6\u5c06\u81ea\u52a8\u53d6\u6d88\n-
\u6bcf\u4f4d\u8bfb\u8005\u6700\u591a\u53ef\u501f\u672c\u4e66\n\n---\n\n##
\u2705 Step 2. \u5143\u7ec4\u5217\u8868\n\n text\n[Actor:
\u8bfb\u8005, Command: \u6ce8\u518c\u8bfb\u8005]\n[Command:
\u6ce8\u518c\u8bfb\u8005, Event:
\u8bfb\u8005\u6ce8\u518c\u6210\u529f]\n\n[Actor:
\u8bfb\u8005, Command:
\u66f4\u65b0\u8bfb\u8005\u4fe1\u606f]\n[Command:
\u66f4\u65b0\u8bfb\u8005\u4fe1\u606f, Event:
\u8bfb\u8005\u4fe1\u606f\u5df2\u66f4\u65b0]\n\n[Actor:
\u56fe\u4e66\u7ba1\u7406\u5458, Command:
\u6dfb\u52a0\u65b0\u56fe\u4e66]\n[Command:
\u6dfb\u52a0\u65b0\u56fe\u4e66, Event:
\u56fe\u4e66\u5df2\u4e0a\u67b6]\n\n[Actor:
\u56fe\u4e66\u7ba1\u7406\u5458, Command:
\u66f4\u65b0\u56fe\u4e66\u4fe1\u606f]\n[Command:
\u66f4\u65b0\u56fe\u4e66\u4fe1\u606f, Event:
\u56fe\u4e66\u4fe1\u606f\u5df2\u66f4\u65b0]\n\n[Actor:
\u8bfb\u8005, Command: \u501f\u51fa\u56fe\u4e66]\n[Command:
\u501f\u51fa\u56fe\u4e66, Event:
\u56fe\u4e66\u88ab\u501f\u51fa]\n\n[Actor: \u8bfb\u8005,
Command: \u5f52\u8fd8\u56fe\u4e66]\n[Command:
\u5f52\u8fd8\u56fe\u4e66, Event:
\u56fe\u4e66\u5df2\u5f52\u8fd8]\n[Event:
\u56fe\u4e66\u5df2\u5f52\u8fd8, Event:
\u56fe\u4e66\u903e\u671f\u672a\u8fd8]\n[Policy:
\u56fe\u4e66\u903e\u671f\u540e\u81ea\u52a8\u751f\u6210\u7f5a
\u6b3e, Event:
\u56fe\u4e66\u903e\u671f\u672a\u8fd8]\n[Command:
\u751f\u6210\u903e\u671f\u7f5a\u6b3e, Event:
\u903e\u671f\u7f5a\u6b3e\u5df2\u751f\u6210]\n[Hotspot:
\u5f52\u8fd8\u56fe\u4e66\u65f6\u903e\u671f\u7f5a\u6b3e\u7684

\u8ba1\u7b97\u89c4\u5219, Event:
\u903e\u671f\u7f5a\u6b3e\u5df2\u751f\u6210]\n\n[Actor:
\u8bfb\u8005, Command: \u652f\u4ed8\u7f5a\u6b3e]\n[Command:
\u652f\u4ed8\u7f5a\u6b3e, Event:
\u7f5a\u6b3e\u5df2\u652f\u4ed8]\n\n[Actor: \u8bfb\u8005,
Command: \u9884\u8ba2\u56fe\u4e66]\n[Command:
\u9884\u8ba2\u56fe\u4e66, Event:
\u56fe\u4e66\u88ab\u9884\u8ba2]\n[Hotspot:
\u591a\u4eba\u540c\u65f6\u9884\u8ba2\u540c\u4e00\u672c\u4e66
\u65f6\u7684\u4f18\u5148\u7ea7\u5224\u5b9a, Event:
\u56fe\u4e66\u88ab\u9884\u8ba2]\n\n[Actor: \u8bfb\u8005,
Command: \u53d6\u6d88\u9884\u8ba2]\n[Command:
\u53d6\u6d88\u9884\u8ba2, Event:
\u56fe\u4e66\u9884\u8ba2\u53d6\u6d88]\n\n[Actor:
\u7cfb\u7edf, Command:
\u901a\u77e5\u9884\u7ea6\u56fe\u4e66\u5230\u9986]\n[Command:
\u901a\u77e5\u9884\u7ea6\u56fe\u4e66\u5230\u9986, Event:
\u9884\u7ea6\u56fe\u4e66\u5df2\u5230\u9986]\n[Policy:
\u9884\u7ea6\u56fe\u4e66\u5230\u9986\u540e24\u5c0f\u65f6\u51
85\u672a\u53d6\u5c06\u81ea\u52a8\u53d6\u6d88, Event:
\u9884\u7ea6\u56fe\u4e66\u5df2\u5230\u9986]\n\n[Actor:
\u8bfb\u8005, Command:
\u501f\u51fa\u9884\u7ea6\u56fe\u4e66]\n[Command:
\u501f\u51fa\u9884\u7ea6\u56fe\u4e66, Event:
\u9884\u7ea6\u56fe\u4e66\u5df2\u501f\u51fa]\n[Event:
\u9884\u7ea6\u56fe\u4e66\u5df2\u5230\u9986, Event:
\u9884\u7ea6\u56fe\u4e66\u5df2\u501f\u51fa]\n\n[Actor:
\u8bfb\u8005, Command:
\u62a5\u544a\u56fe\u4e66\u9057\u5931]\n[Command:
\u62a5\u544a\u56fe\u4e66\u9057\u5931, Event:
\u56fe\u4e66\u9057\u5931\u4e0a\u62a5]\n\n[Actor:
\u8bfb\u8005, Command:
\u8d54\u507f\u9057\u5931\u56fe\u4e66]\n[Command:
\u8d54\u507f\u9057\u5931\u56fe\u4e66, Event:
\u56fe\u4e66\u9057\u5931\u8d54\u507f\u5b8c\u6210]\n[Hotspot:
\u56fe\u4e66\u9057\u5931\u7684\u8d54\u507f\u91d1\u989d\u5982
\u4f55\u8bc4\u4f30, Event:
\u56fe\u4e66\u9057\u5931\u8d54\u507f\u5b8c\u6210]\n\n[Actor:

\u56fe\u4e66\u7ba1\u7406\u5458, Command: \u5220\u9664\u56fe\u4e66]\n[Command: \u5220\u9664\u56fe\u4e66, Event: \u7ba1\u7406\u5458\u5df2\u5220\u9664\u56fe\u4e66]\n\n[Actor: \u56fe\u4e66\u7ba1\u7406\u5458, Command: \u66f4\u65b0\u56fe\u4e66\u5e93\u5b58]\n[Command: \u66f4\u65b0\u56fe\u4e66\u5e93\u5b58, Event: \u56fe\u4e66\u5e93\u5b58\u5df2\u66f4\u65b0]\n\n\n---\n\n## \u2705 Step 3. \u5173\u7cfb\u6d41\u7a0b\u4ee5\u53ca\u5404\u7c7b\u5143\u7d20\u989c\u8272\n\n\u8bf7\u7ee7\u7eed\u6267\u884c PlantUML \u53ef\u89c6\u5316\u6f14\u793a\uff0c\u6211\u5c06\u5728\u4e0b\u4e00\u6b65\u63d0\u4f9b\u5b8c\u6574\u7684 PlantUML \u6e32\u67d3\u4ee3\u7801\u548c\u989c\u8272\u8bf4\u660e\u3002" } ChatGPT Successfully created text document '图书管理系统-事件风暴建模' which will be referenced in all future messages with the unique identifier textdoc_id: '683730b7b6bc8191a50bf5b1aa5f8565' ChatGPT { "name": "图书管理系统-事件风暴建模", "type": "document", "content": "# 图书管理系统-事件风暴建模\n\n## Step 1: Identify Core Elements\n\n### Domain Events (Past Tense)\n- Book Borrowed\n- Book Returned\n- Book Reserved\n- Reservation Cancelled\n- Fine Issued\n- Fine Paid\n- Book Added\n- Book Removed\n- Book Information Updated\n- User Registered\n- User Account Suspended\n- User Account Reactivated\n\n### Commands (Present Tense)\n- Borrow Book\n- Return Book\n- Reserve Book\n- Cancel Reservation\n- Issue Fine\n- Pay Fine\n- Add Book\n- Remove Book\n- Update Book Info\n- Register User\n- Suspend User Account\n- Reactivate User Account\n\n### Actors\n- Librarian\n- Member\n- Library System (Automated)\n\n### Hotspots\n- Overlapping reservations for the same book\n- Fine calculation rules (e.g., grace period, per-day rate)\n- Book availability verification logic\n\n### Policies\n- If a book is returned late, then a fine must be issued\n- If a reserved book is not collected in 3 days, cancel reservation automatically\n- A suspended user cannot borrow or reserve books\n\n---\n\n## Step 2: Directed Binary Tuples\n\n text\n[Actor: Member, Command: Borrow Book]\n[Command: Borrow Book, Event: Book Borrowed]\n[Policy: A suspended user cannot borrow books, Event: Book Borrowed]\n\n[Actor: Member, Command: Return Book]\n[Command: Return Book, Event: Book Returned]\n[Policy: If a book is returned late, then a fine must be issued, Event: Book Returned]\n[Event: Book

```
Returned, Event: Fine Issued]\n\n[Actor: Member, Command:
Reserve Book]\n[Command: Reserve Book, Event: Book
Reserved]\n[Hotspot: Overlapping reservations for the same
book, Event: Book Reserved]\n\n[Actor: Member, Command:
Cancel Reservation]\n[Command: Cancel Reservation, Event:
Reservation Cancelled]\n\n[Actor: Librarian, Command: Issue
Fine]\n[Command: Issue Fine, Event: Fine Issued]\n\n[Actor:
Member, Command: Pay Fine]\n[Command: Pay Fine, Event: Fine
Paid]\n\n[Actor: Librarian, Command: Add Book]\n[Command:
Add Book, Event: Book Added]\n\n[Actor: Librarian, Command:
Remove Book]\n[Command: Remove Book, Event: Book
Removed]\n\n[Actor: Librarian, Command: Update Book
Info]\n[Command: Update Book Info, Event: Book Information
Updated]\n\n[Actor: Member, Command: Register
User]\n[Command: Register User, Event: User
Registered]\n\n[Actor: Librarian, Command: Suspend User
Account]\n[Command: Suspend User Account, Event: User
Account Suspended]\n\n[Actor: Librarian, Command: Reactivate
User Account]\n[Command: Reactivate User Account, Event:
User Account Reactivated]\n\n[Policy: If a reserved book is
not collected in 3 days, cancel reservation automatically,
Event: Reservation Cancelled]\n\n---\n\n## Step 3: PlantUML
Visualization\n\n### Legend: Color Coding for Elements\n- Domain Events:
Orange\n- Commands: Blue\n- Actors: Yellow\n- Hotspots: Red\n- Policies:
Purple\n\n### PlantUML Code\n plantuml\n@startuml\nskinparam style
strict\nskinparam rectangle {\n StereotypeFontColor black\n
StereotypeFontSize 12\n BackgroundColor white\n BorderColor
black\n}\n\nactor \"Member\" as Member <<Actor>>
#FBF8A6\nactor \"Librarian\" as Librarian <<Actor>>
#FBF8A6\nactor \"Library System\" as System <<Actor>>
#FBF8A6\n\nrectangle \"Borrow Book\" as BorrowBook
<<Command>> #AEC6CF\nrectangle \"Book Borrowed\" as
BookBorrowed <<Event>> #FFB347\nrectangle \"A suspended user
cannot borrow books\" as BorrowPolicy <<Policy>>
#DDA0DD\n\nrectangle \"Return Book\" as ReturnBook
<<Command>> #AEC6CF\nrectangle \"Book Returned\" as
BookReturned <<Event>> #FFB347\nrectangle \"If returned
late, then fine issued\" as FinePolicy <<Policy>>
```


#DDA0DD\nrectangle \"Fine Issued\" as FineIssued <<Event>>
#FFB347\n\nrectangle \"Reserve Book\" as ReserveBook
<<Command>> #AEC6CF\nrectangle \"Book Reserved\" as
BookReserved <<Event>> #FFB347\nrectangle \"Overlapping
reservations\" as ReserveHotspot <<Hotspot>>
#FF6961\n\nrectangle \"Cancel Reservation\" as
CancelReservation <<Command>> #AEC6CF\nrectangle
\"Reservation Cancelled\" as ReservationCancelled <<Event>>
#FFB347\nrectangle \"Uncollected in 3 days\" as CancelPolicy
<<Policy>> #DDA0DD\n\nrectangle \"Issue Fine\" as IssueFine
<<Command>> #AEC6CF\nrectangle \"Pay Fine\" as PayFine
<<Command>> #AEC6CF\nrectangle \"Fine Paid\" as FinePaid
<<Event>> #FFB347\n\nrectangle \"Add Book\" as AddBook
<<Command>> #AEC6CF\nrectangle \"Book Added\" as BookAdded
<<Event>> #FFB347\n\nrectangle \"Remove Book\" as RemoveBook
<<Command>> #AEC6CF\nrectangle \"Book Removed\" as
BookRemoved <<Event>> #FFB347\n\nrectangle \"Update Book
Info\" as UpdateBookInfo <<Command>> #AEC6CF\nrectangle
\"Book Information Updated\" as BookInfoUpdated <<Event>>
#FFB347\n\nrectangle \"Register User\" as RegisterUser
<<Command>> #AEC6CF\nrectangle \"User Registered\" as
UserRegistered <<Event>> #FFB347\n\nrectangle \"Suspend User
Account\" as SuspendUser <<Command>> #AEC6CF\nrectangle
\"User Account Suspended\" as UserSuspended <<Event>>
#FFB347\n\nrectangle \"Reactivate User Account\" as
ReactivateUser <<Command>> #AEC6CF\nrectangle \"User Account
Reactivated\" as UserReactivated <<Event>> #FFB347\n\nMember
--> BorrowBook\nBorrowBook --> BookBorrowed\nBorrowPolicy --
> BookBorrowed\n\nMember --> ReturnBook\nReturnBook -->
BookReturned\nFinePolicy --> BookReturned\nBookReturned -->
FineIssued\n\nMember --> ReserveBook\nReserveBook -->
BookReserved\nReserveHotspot --> BookReserved\n\nMember -->
CancelReservation\nCancelReservation -->
ReservationCancelled\nCancelPolicy -->
ReservationCancelled\n\nLibrarian --> IssueFine\nIssueFine -
-> FineIssued\n\nMember --> PayFine\nPayFine -->
FinePaid\n\nLibrarian --> AddBook\nAddBook -->
BookAdded\n\nLibrarian --> RemoveBook\nRemoveBook -->

`BookRemoved\n\nLibrarian --> UpdateBookInfo\nUpdateBookInfo --> BookInfoUpdated\n\nMember --> RegisterUser\nRegisterUser --> UserRegistered\n\nLlibrarian --> SuspendUser\nSuspendUser --> UserSuspended\n\nLibrarian --> ReactivateUser\nReactivateUser --> UserReactivated\n@enduml\n`\n\n---\n\n以上是使用事件风暴方法对图书管理系统进行领域建模的完整过程，包括关键事件识别、二元元组列表和可视化建模图。" } ChatGPT Successfully created text document '图书管理系统-事件风暴建模' which will be referenced in all future messages with the unique identifier textdoc_id: '683730f8370c81918ddb1b10a43b96c1' ChatGPT 已完成建模内容并输出为Markdown文档，文档名称为《图书管理系统-事件风暴建模.md》，包括以下内容：

- **Step 1**：领域事件、命令、行为者、热点问题和策略规则的完整识别；
- **Step 2**：事件风暴的二元有向关系元组描述；
- **Step 3**：根据事件流主线生成的PlantUML可视化图，并附带元素类型颜色说明（Domain Events、Commands、Actors、Hotspots、Policies）；

如需我导出此文档为 `.md` 文件，或生成该PlantUML图的图片文件，请告知，我可以立即为你处理。 user 导出为.md 文件，命名为《图书管理系统-事件风暴建模.md》 ChatGPT 已成功导出为 Markdown 文件，文件名为：《图书管理系统-事件风暴建模.md》

你可以通过以下链接下载该文件：

👉 点击下载图书管理系统-事件风暴建模.md user 输出中文版 ChatGPT 看起来我现在无法执行更高级的数据处理操作。请稍后再试。

如果你希望立即获得中文版的 Markdown 文件，我可以将完整内容复制粘贴给你，你可以手动保存为 `.md` 文件。请问你希望我这样做吗？ user 将中文版导出为为.md 文件 ChatGPT It seems like I can't do more advanced data analysis right now. Please try again later.

不过，我可以将完整的中文 Markdown 文档内容提供给你，你只需复制以下内容并保存为一个名为《图书管理系统-事件风暴建模-中文版.md》的文件即可。请告诉我是否需要我将内容贴出来？ 事件风暴建模 user You are an experienced facilitator in Event Storming workshops. Based on the following problem description, your task is to build a domain model using the principles of Event Storming. Please complete the following steps:

Step1. Identify the following elements from the description: *Domain Events*: Business-relevant facts that have occurred in the system. They are written with verbs in the past

tense.

- Commands*: Explicit actions that trigger events. These are written with verbs in the present tense. Commands are given by an Actor, either a User or a System.
- Actors*: Users or external systems that initiate commands.
- Hotspots*(if existence): Areas of ambiguity, complexity, or known issues.
- Policies*(if existence): These are pre-defined rules that are applied after a Domain Event to determine the next step. Notes：Each domain event has a command and an Actor (i.e. users or external systems).

Step2. List the directed binary tuples related to each event line by line, and may include: [Actor: actor1, Command: command1], [Command: command 1, Event: event 1], (if have hotspot)[Hotspot: hotspot1, Event: event1], (if have policy)[Policy: policy1, Event: event1], [Event: event1, Event: event2] (Only when one event directly follows from another)

Step3. After listing the tuples, With the Event flow as the main line, visualization is carried out using PlantUML according to the Event Storming principle. Elements of different types have different colors, and the color type descriptions should be provided.

Problem Description: Case Study: [System Name] [problem description]

请严格按照上述要求，完成优化工作 ChatGPT

Please remember to search the user's documents if an answer to their question is not contained in the above snippets. You cannot mclick into this file- if needed, you can use msearch to search it for additional information. ChatGPT 根据你上传的图书馆管理系统事件风暴建模文档（PDF 与 Markdown 一致），我已按要求完成了事件风暴的三步建模任务：

✅ Step 1：识别事件风暴元素

🟩 Domain Events（领域事件）

共识别18个领域事件，部分示例如下：

- 书籍已借出
- 书籍已归还
- 逾期罚款已生成
- 罚款已支付

- 借阅限制已解除
- 书籍已预约
- 书籍已续借
- 账户已冻结
- 借阅到期已提醒 ...

🟧 Commands（命令）

共识别14个命令，部分示例如下：

- 借书
- 还书
- 续借书籍
- 支付罚款
- 冻结账户
- 添加读者
- 搜索书籍
- 发送到期提醒 ...

🟦 Actors（参与者）

- 读者（Reader）
- 管理员（Admin）
- 系统（System / 定时任务）
- 支付网关（Payment Gateway - 外部系统）
- 通知服务（Notification Service - 外部系统）

🟨 Hotspots（热点）

- 账户不可用（如被冻结、未注册、有未付罚款）
- 书籍不可借（已借出或已被预约）
- 续借冲突（书籍被他人预约但用户尝试续借）
- 预约队列优先级
- 系统故障时的支付处理

🌸 Policies（策略）

- 借书前需验证账户和书籍状态
- 每位最多借5本书

- 每天1元逾期罚款，最高50元
- 续借前需检查预约状态
- 支付罚款后自动解除借阅限制
- 到期前3天自动提醒
- 未付罚款超过50元冻结账户 ...

✅ Step 2：二元组列表（Directed Binary Tuples）

按事件依赖列出，部分示例如下：

借书服务

```
[Actor: 读者, Command: 借书]
[Command: 借书, Event: 书籍已借出]
[Policy: 借书前需验证账户和书籍状态, Event: 书籍已借出]
[Hotspot: 账户不可用, Event: 借书]
[Hotspot: 书籍不可借, Event: 借书]
[Event: 书籍已借出, Event: 借书配额已更新]
```

还书服务

```
[Actor: 读者, Command: 还书]
[Command: 还书, Event: 书籍已归还]
[Event: 书籍已归还, Event: 书籍状态已更新]
[Event: 书籍已归还, Event: 逾期罚款已生成]
[Event: 书籍已归还, Event: 预约通知已发送]
```

计算逾期罚款

```
[Actor: 系统, Command: 计算逾期罚款]
[Command: 计算逾期罚款, Event: 逾期罚款已计算]
[Policy: 每天1元，最高不超过50元, Event: 逾期罚款已计算]
```

支付罚款服务

```
[Actor: 读者, Command: 支付罚款]
[Command: 支付罚款, Event: 罚款已支付]
[Event: 罚款已支付, Event: 借阅限制已解除]
[Event: 罚款已支付, Event: 电子收据已生成]
[Hotspot: 系统故障时的支付处理, Event: 支付罚款]
[Policy: 罚款支付后自动解除借阅限制, Event: 借阅限制已解除]
[Actor: 支付网关, Event: 罚款已支付]
```

（完整二元组详见文档第5章）

✅ Step 3：PlantUML 可视化图（已完成）

文档中已使用 PlantUML 完成完整的事件流建模：

🎨 元素颜色说明

元素类型	PlantUML 标签	颜色
Actor	`ACTOR`	淡蓝色
Command	`COMMAND`	橙色
Domain Event	`EVENT`	淡绿色
Policy	`POLICY`	粉色
Hotspot	`HOTSPOT`	黄色
Read Model	`READMODEL`	紫色
External System	`EXTERNAL_SYSTEM`	灰色

📈 PlantUML 可视化模型已完成并可复用，示例见：

```
@startuml 图书馆管理系统事件风暴图
!define COMMAND rectangle #Orange
!define EVENT rectangle #LightGreen
!define ACTOR rectangle #LightBlue
!define POLICY rectangle #Pink
!define HOTSPOT rectangle #Yellow
!define READMODEL rectangle #Purple
!define EXTERNAL_SYSTEM rectangle #Gray

ACTOR "读者" as reader
COMMAND "借书" as borrowCmd
```

```
EVENT "书籍已借出" as borrowedEvent
...
@enduml
```

> ✅ 全部可视化模型代码见第8章，图形覆盖所有核心服务与事件链。

如需我将这些结果导出为 Markdown 文档、HTML 页面或其他格式，请告诉我。你也可以让我生成该图的 SVG 或 PNG 文件用于展示。 user 将结果导出为 markdown 文档 ChatGPT ✅ 已完成导出。你可以通过以下链接下载生成的 Markdown 文档：

👉 点击下载图书馆管理系统_事件风暴建模输出.md

如需将其转换为 PDF、HTML 或图片形式，也可以告诉我。 user You are an experienced facilitator in Event Storming workshops. Based on the following problem description, your task is to build a domain model using the principles of Event Storming. Please complete the following steps:

Step1. Identify the following elements from the description: ***Domain Events***: Business-relevant facts that have occurred in the system. They are written with verbs in the past tense.

- Commands*: Explicit actions that trigger events. These are written with verbs in the present tense. Commands are given by an Actor, either a User or a System.
- Actors*: Users or external systems that initiate commands.
- Hotspots*(if existence): Areas of ambiguity, complexity, or known issues.
- Policies*(if existence): These are pre-defined rules that are applied after a Domain Event to determine the next step. Notes： Each domain event has a command and an Actor (i.e. users or external systems).

Step2. List the directed binary tuples related to each event line by line, and may include: [Actor: actor1, Command: command1], [Command: command 1, Event: event 1], (if have hotspot)[Hotspot: hotspot1, Event: event1], (if have policy)[Policy: policy1, Event: event1], [Event: event1, Event: event2] (Only when one event directly follows from another)

Step3. After listing the tuples, With the Event flow as the main line, visualization is carried out using PlantUML according to the Event Storming principle. Elements of different types have different colors, and the color type descriptions should be provided.

Problem Description: Case Study: [System Name] [problem description]

基于原始 markdown 文件，在原本的基础上进行优化，要求按照以上步骤，全面优化图书馆管理系统事件风暴建模文档，并且输出完整的优化后的 markdown 文件 ChatGPT

✅ 已完成优化后的事件风暴建模 Markdown 文档导出，你可以通过以下链接下载：

👉 点击下载 图书馆管理系统_事件风暴建模优化版.md

如需进一步转换为 PDF、HTML 或绘制完整事件流图（支持所有服务），欢迎继续告诉我！