Flashcard 3000

Hua Chuang, Yicun Liu, Ryan Harmon

CS6359.002 Team2

Contents

Introduction

Design Pattern

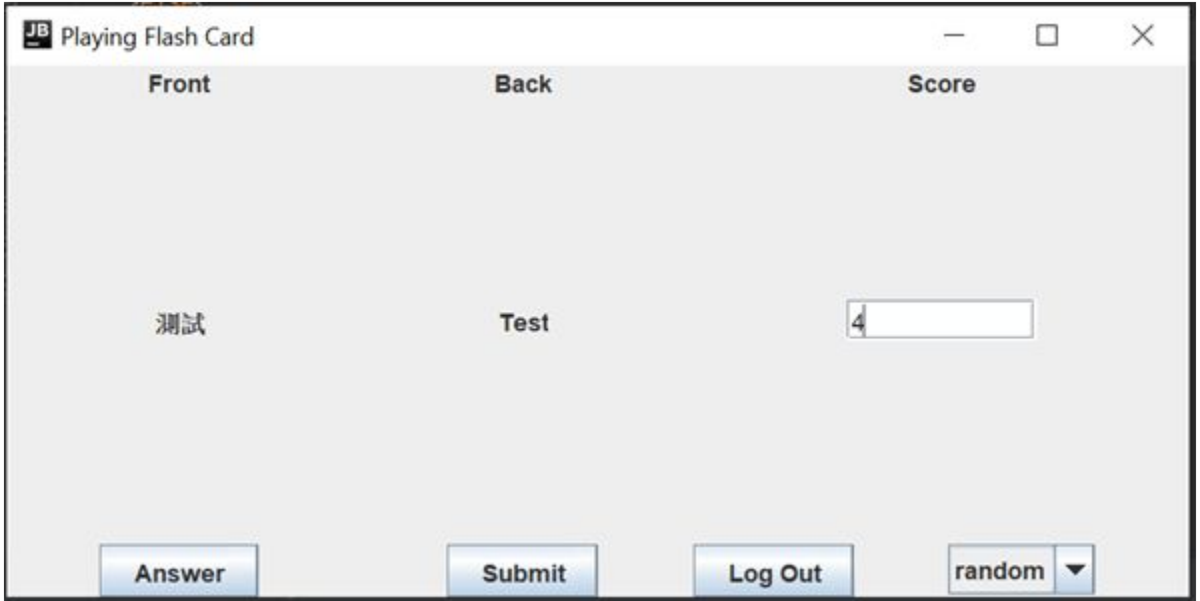
User Interface

Use Cases and User Stories

Conclusion

Introduction

One of the most effective ways of remembering information is through the combination of flashcards and a spaced repetition reviewing system. While it is possible to create and review flashcards by hand, they are time-consuming to make, prone to get lost, easy to mix up, and very difficult to systematically review in an efficient manner. In this project, we make a simple Java app that can be used to create, review and manage generic flash cards.



Design Pattern

MVC for System

- We use MVC pattern decouples these major components allowing for code reuse and parallel development.
- Reuse: DBM(database), strategies.
- Parallel development: UI design (Log In UI, Card Editor UI and Playing Card UI) and model design (DBM, Strategies)

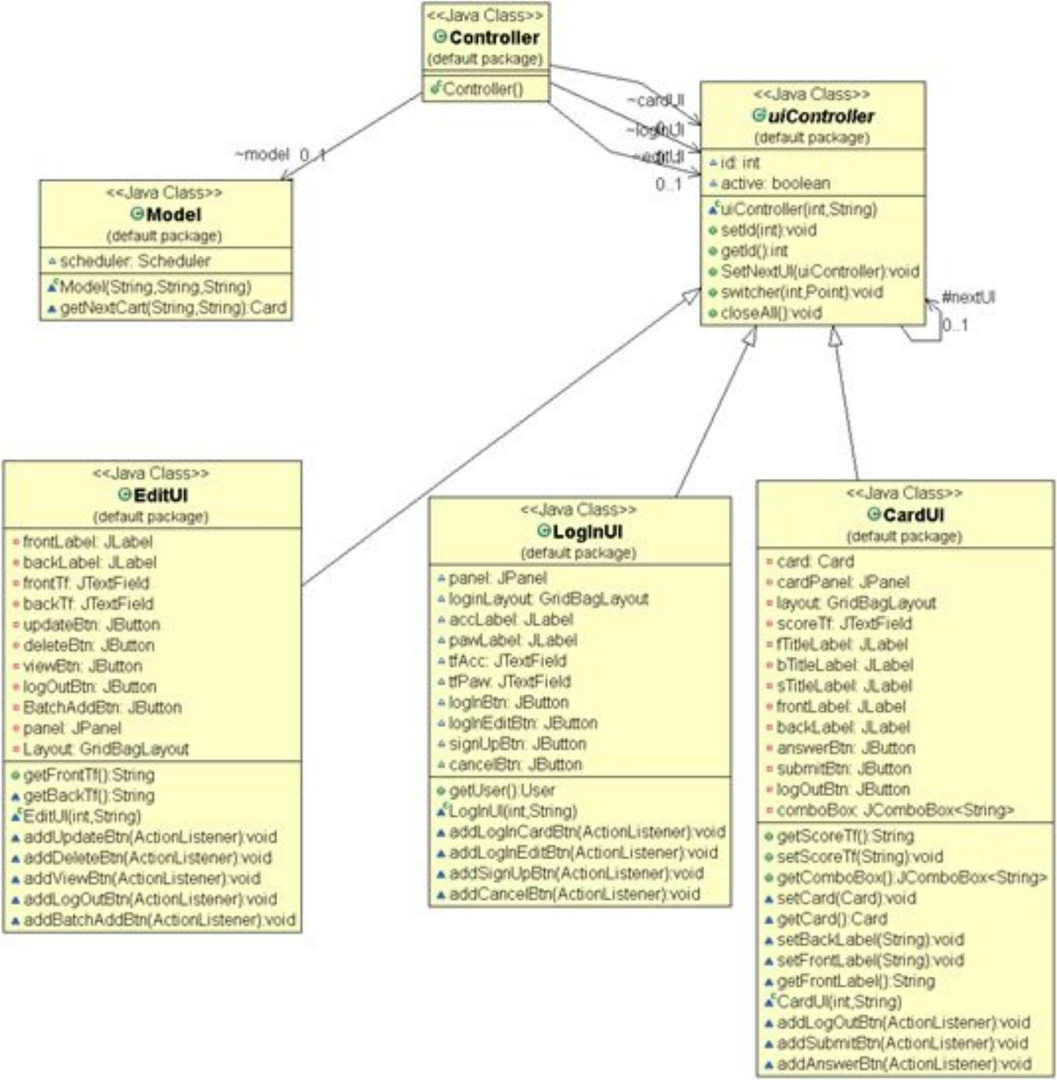
Strategies for choosing learning mode

- Strategies pattern selects one strategy from the pool for use during runtime. This pattern allows the user to change the learning mode arbitrarily.

Chain of Responsibility for control UI

- This system use chain of responsibility to switch the UI display.
- The pattern can also guarantee the all UI object will be destroyed when the user exits the system.

MVC and Chain of Responsibility



<<Java Class>>
Controller
(default package)

Controller()

<<Java Class>>
uiController
(default package)

id: int
active: boolean

uiController(int,String)
setId(int):void
getId():int
SetNextUI(uiController):void
switcher(int,Point):void
closeAll():void

<<Java Class>>
Model
(default package)

scheduler: Scheduler

Model(String,String,String)
getNextCart(String,String):Card

<<Java Class>>
EditUI
(default package)

frontLabel: JLabel
backLabel: JLabel
frontTf: JTextField
backTf: JTextField
updateBtn: JButton
deleteBtn: JButton
viewBtn: JButton
logOutBtn: JButton
BatchAddBtn: JButton
panel: JPanel
Layout: GridBagLayout

getFrontTf():String
getBackTf():String
EditUI(int,String)
addUpdateBtn(ActionListener):void
addDeleteBtn(ActionListener):void
addViewBtn(ActionListener):void
addLogOutBtn(ActionListener):void
addBatchAddBtn(ActionListener):void

<<Java Class>>
LogInUI
(default package)

panel: JPanel
loginLayout: GridBagLayout
accLabel: JLabel
pawLabel: JLabel
tfAcc: JTextField
tfPaw: JTextField
logInBtn: JButton
logInEditBtn: JButton
signUpBtn: JButton
cancelBtn: JButton

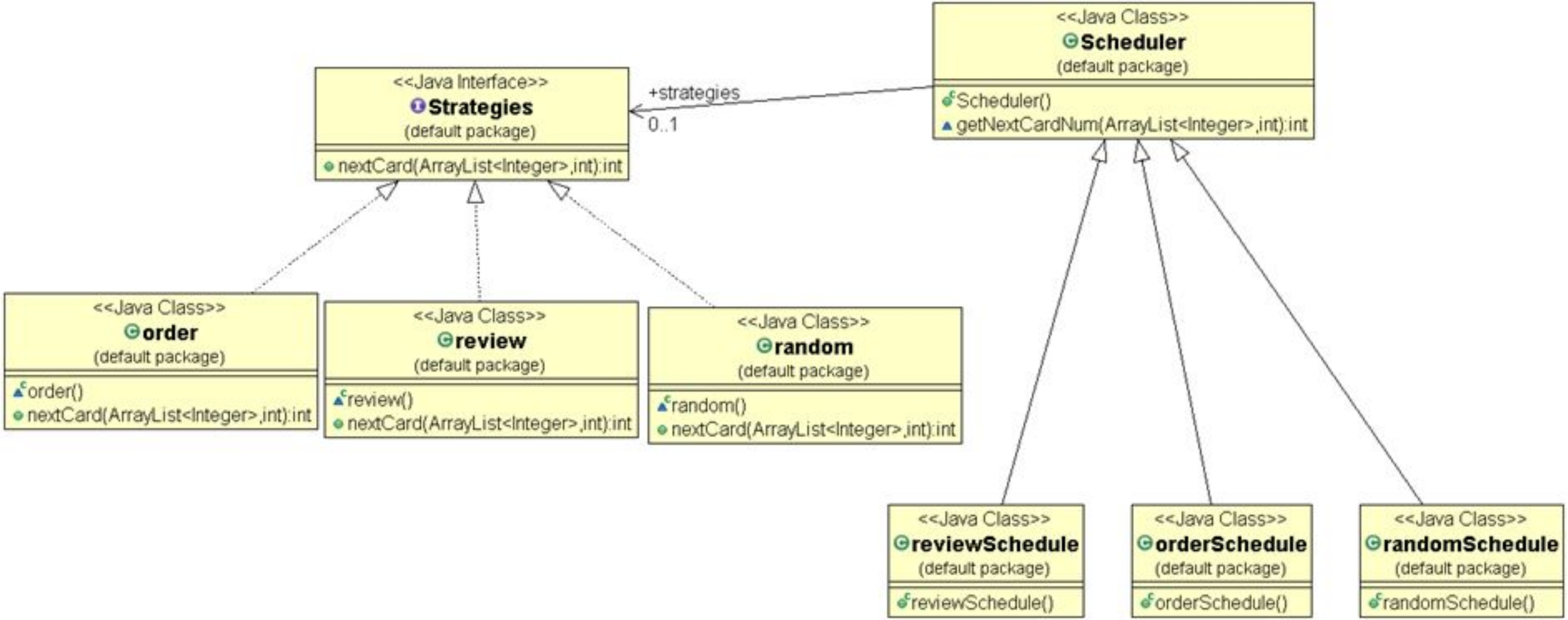
getUser():User
LogInUI(int,String)
addLogInCardBtn(ActionListener):void
addLogInEditBtn(ActionListener):void
addSignUpBtn(ActionListener):void
addCancelBtn(ActionListener):void

<<Java Class>>
CardUI
(default package)

card: Card
cardPanel: JPanel
layout: GridBagLayout
scoreTf: JTextField
fTitleLabel: JLabel
bTitleLabel: JLabel
sTitleLabel: JLabel
frontLabel: JLabel
backLabel: JLabel
answerBtn: JButton
submitBtn: JButton
logOutBtn: JButton
comboBox: JComboBox<String>

getScoreTf():String
setScoreTf(String):void
getComboBox():JComboBox<String>
setCard(Card):void
getCard():Card
setBackLabel(String):void
setFrontLabel(String):void
getFrontLabel():String
CardUI(int,String)
addLogOutBtn(ActionListener):void
addSubmitBtn(ActionListener):void
addAnswerBtn(ActionListener):void

Strategies

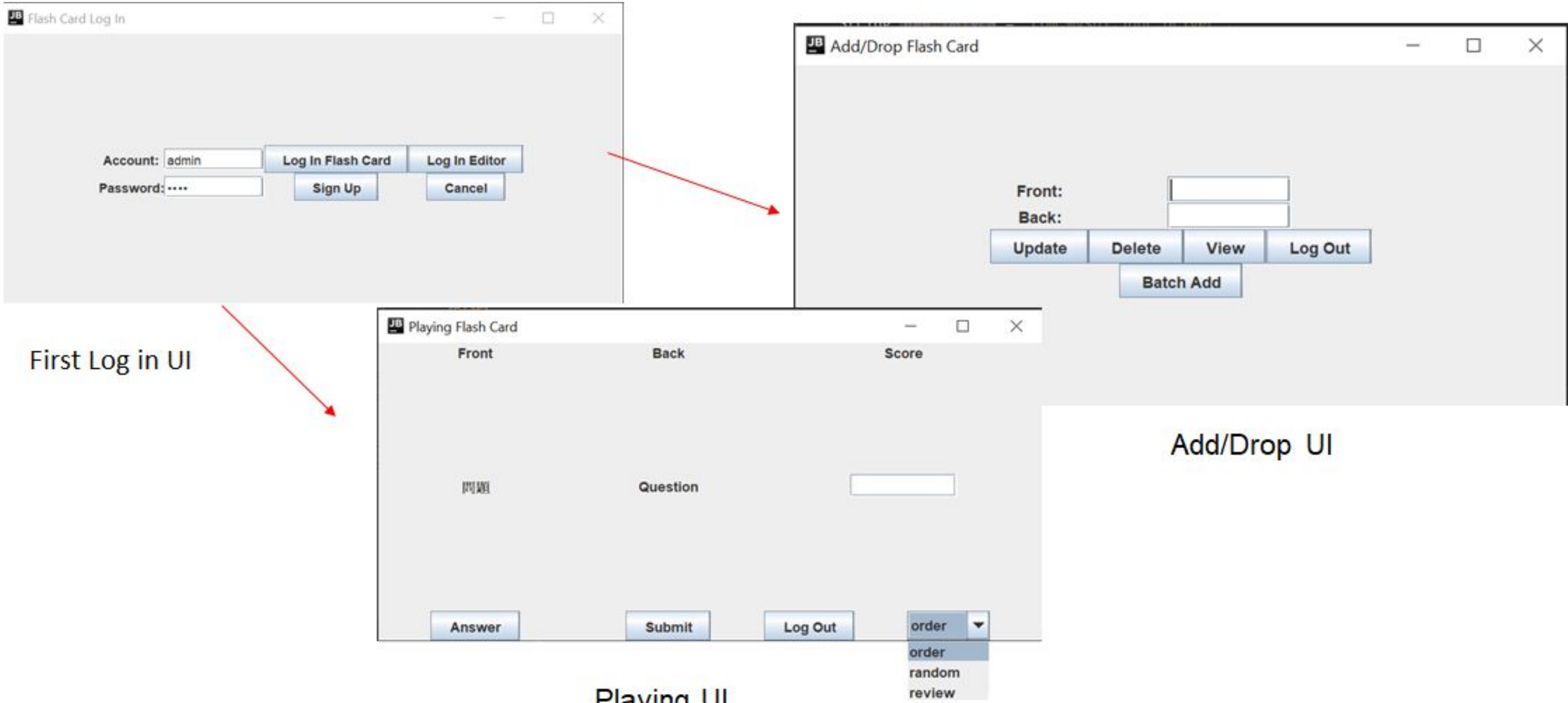

<<Java Interface>>
Strategies
(default package)
nextCard(ArrayList<Integer>,int):int
+strategies
0..1
<<Java Class>>
Scheduler
(default package)
Scheduler()
getNextCardNum(ArrayList<Integer>,int):int
<<Java Class>>
order
(default package)
order()
nextCard(ArrayList<Integer>,int):int
<<Java Class>>
review
(default package)
review()
nextCard(ArrayList<Integer>,int):int
<<Java Class>>
random
(default package)
random()
nextCard(ArrayList<Integer>,int):int
<<Java Class>>
reviewSchedule
(default package)
reviewSchedule()
<<Java Class>>
orderSchedule
(default package)
orderSchedule()
<<Java Class>>
randomSchedule
(default package)
randomSchedule()

Features

- Interesting features:
- Three Strategies have different learning ways.
- Order: learn words from easy to hard.
- Random: learn words randomly(more challenge).
- Review: review the unfamiliar words.

- Useful features:
- Everyone has own card set.
- Easily to add/drop/delete/update the card.
- Batch-add is support in our system.

User Interface



Flash Card Log In

Account: admin

Password: ····

First Log in UI

Add/Drop Flash Card

Front:

Back:

Add/Drop UI

Playing Flash Card

Front: 問題

Back: Question

Score:

order: order

Playing UI

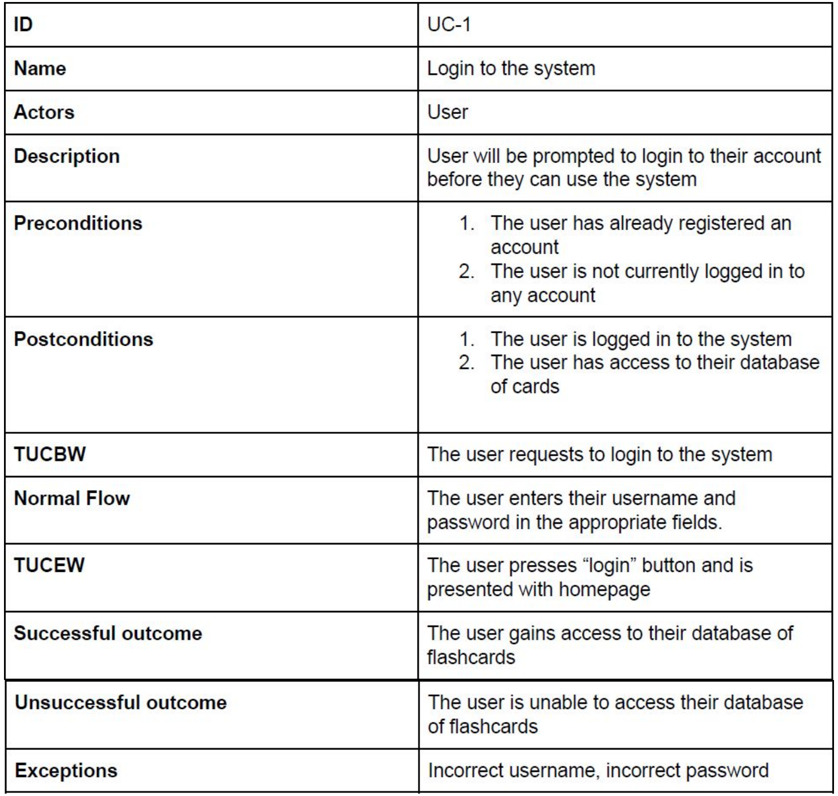
Use Cases and User Stories

Below are 7 detailed use cases from the preceding diagrams, as well as an associated user story for each. The use cases are:

1. Login to the system
2. Create a flashcard
3. Review a set of flashcards
4. Edit an existing flashcard
5. Delete an existing flashcard
6. Register an account
7. Logout of system

Use Case #1

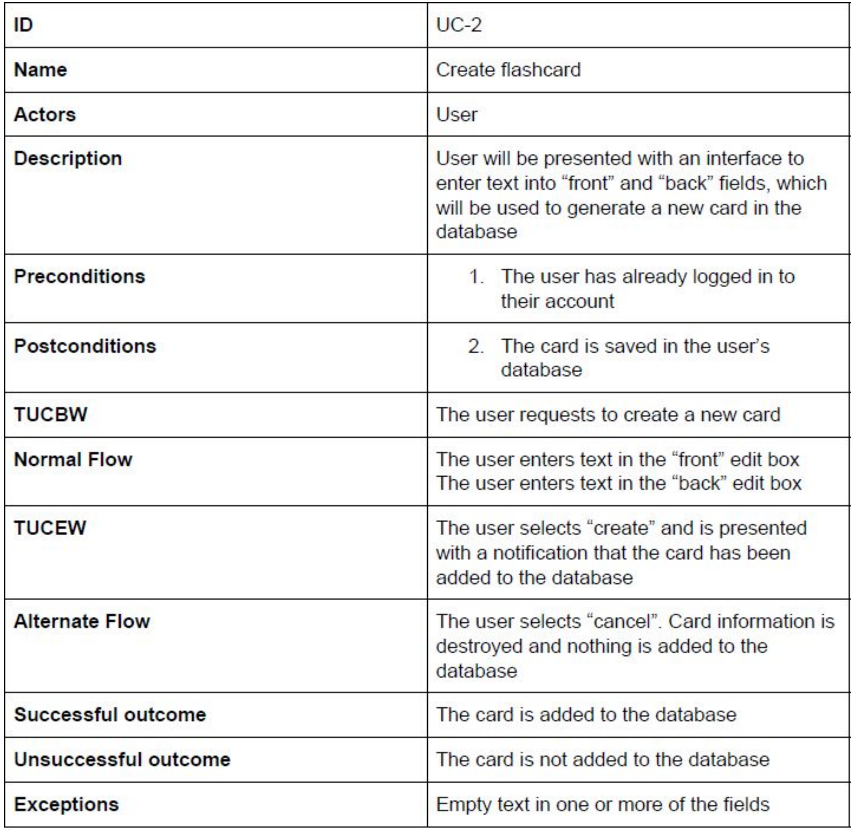
User Story: As a user, I can login to the system so that I can access my saved set of flashcards.



ID	UC-1
Name	Login to the system
Actors	User
Description	User will be prompted to login to their account before they can use the system
Preconditions	1. The user has already registered an account 2. The user is not currently logged in to any account
Postconditions	1. The user is logged in to the system 2. The user has access to their database of cards
TUCBW	The user requests to login to the system
Normal Flow	The user enters their username and password in the appropriate fields.
TUCEW	The user presses “login” button and is presented with homepage
Successful outcome	The user gains access to their database of flashcards
Unsuccessful outcome	The user is unable to access their database of flashcards
Exceptions	Incorrect username, incorrect password

Use Case #2

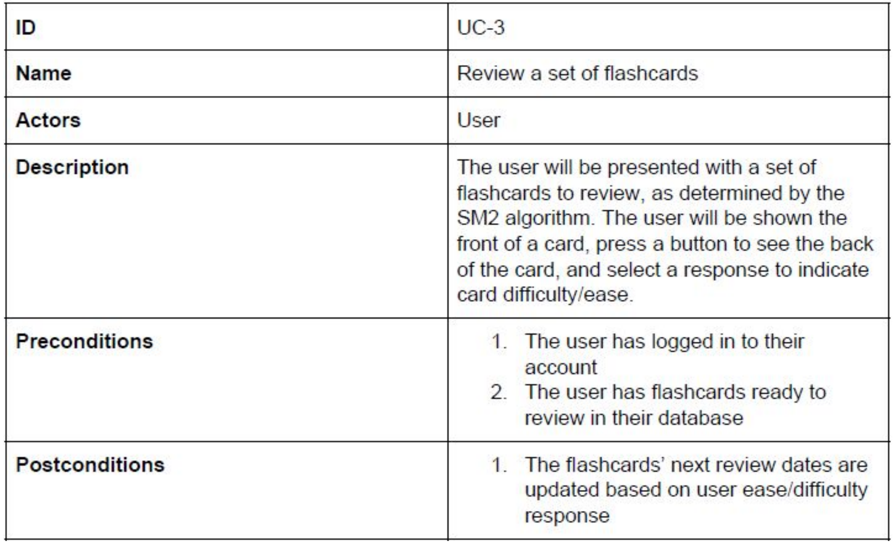
User Story: As a user, I can create a new flashcard so that I can review it at a later date.



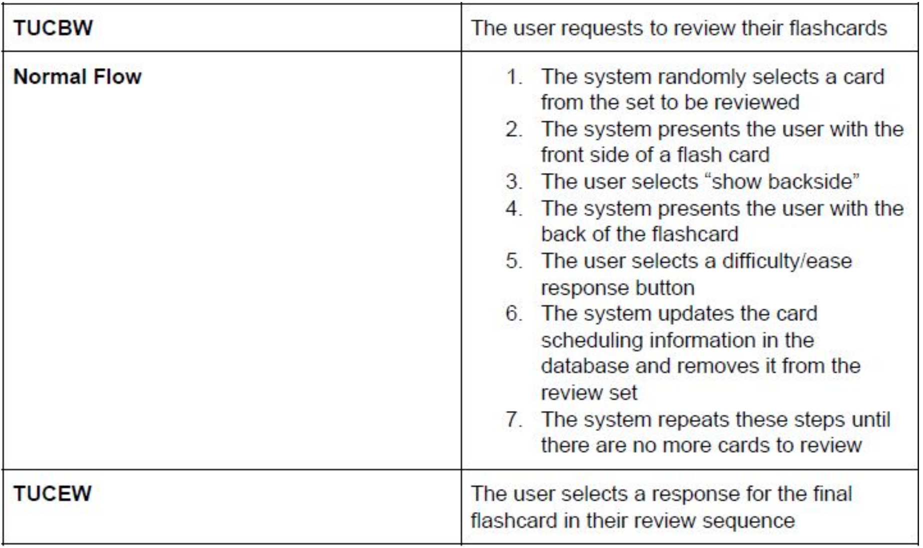
ID	UC-2
Name	Create flashcard
Actors	User
Description	User will be presented with an interface to enter text into “front” and “back” fields, which will be used to generate a new card in the database
Preconditions	1. The user has already logged in to their account
Postconditions	2. The card is saved in the user’s database
TUCBW	The user requests to create a new card
Normal Flow	The user enters text in the “front” edit box The user enters text in the “back” edit box
TUCEW	The user selects “create” and is presented with a notification that the card has been added to the database
Alternate Flow	The user selects “cancel”. Card information is destroyed and nothing is added to the database
Successful outcome	The card is added to the database
Unsuccessful outcome	The card is not added to the database
Exceptions	Empty text in one or more of the fields

Use Case #3

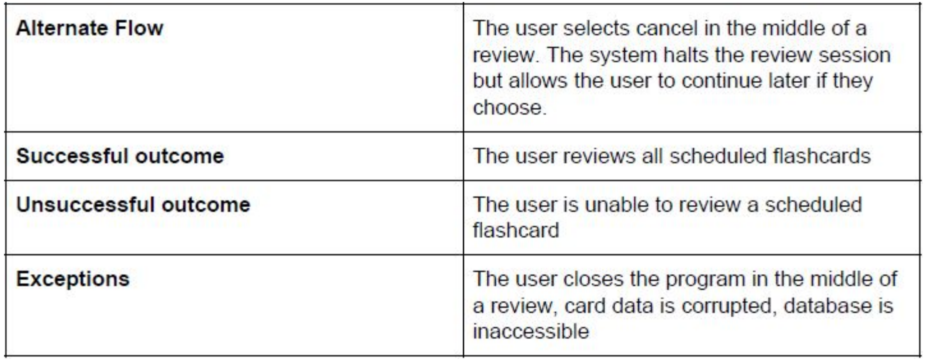
User Story: As a user, I can review a set of flashcards so that I can better remember the information contained in them.



ID	UC-3
Name	Review a set of flashcards
Actors	User
Description	The user will be presented with a set of flashcards to review, as determined by the SM2 algorithm. The user will be shown the front of a card, press a button to see the back of the card, and select a response to indicate card difficulty/ease.
Preconditions	1. The user has logged in to their account 2. The user has flashcards ready to review in their database
Postconditions	1. The flashcards' next review dates are updated based on user ease/difficulty response



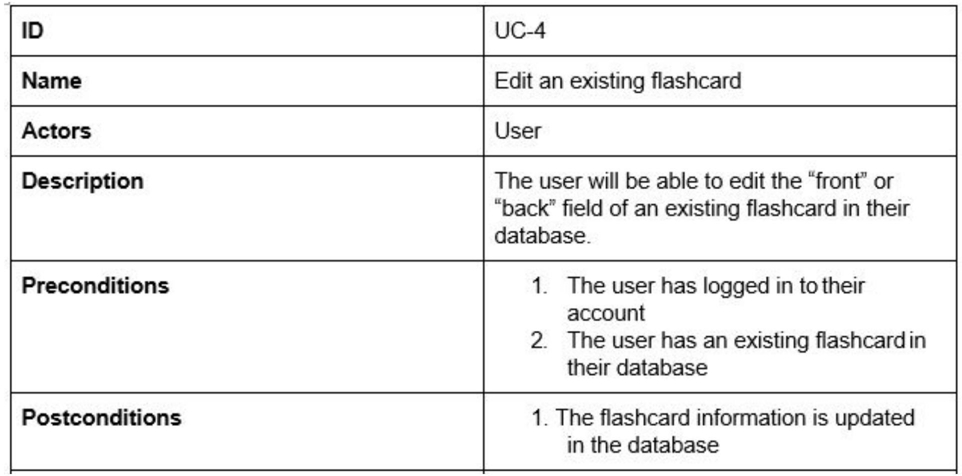
TUCBW	The user requests to review their flashcards
Normal Flow	1. The system randomly selects a card from the set to be reviewed 2. The system presents the user with the front side of a flash card 3. The user selects "show backside" 4. The system presents the user with the back of the flashcard 5. The user selects a difficulty/ease response button 6. The system updates the card scheduling information in the database and removes it from the review set 7. The system repeats these steps until there are no more cards to review
TUCEW	The user selects a response for the final flashcard in their review sequence



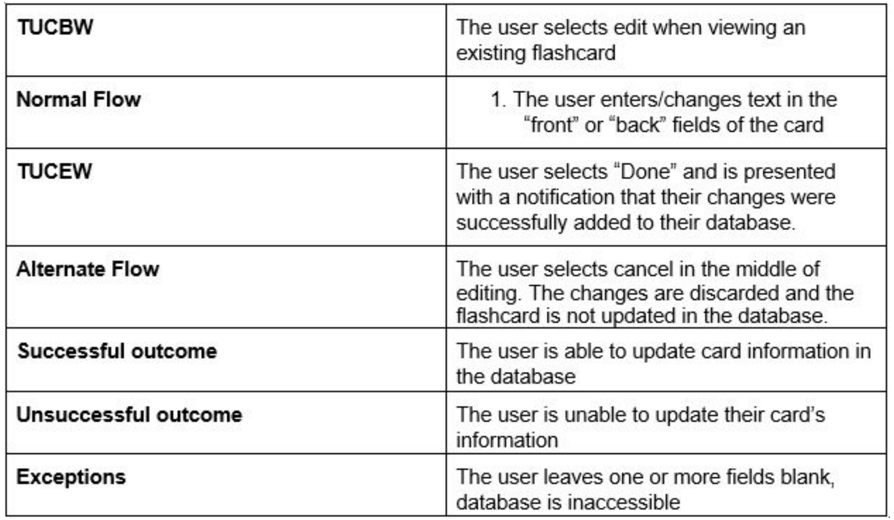
Alternate Flow	The user selects cancel in the middle of a review. The system halts the review session but allows the user to continue later if they choose.
Successful outcome	The user reviews all scheduled flashcards
Unsuccessful outcome	The user is unable to review a scheduled flashcard
Exceptions	The user closes the program in the middle of a review, card data is corrupted, database is inaccessible

Use Case #4

User Story: As a user, I can edit the text of an existing flashcard so that I can clarify or change the information contained within.



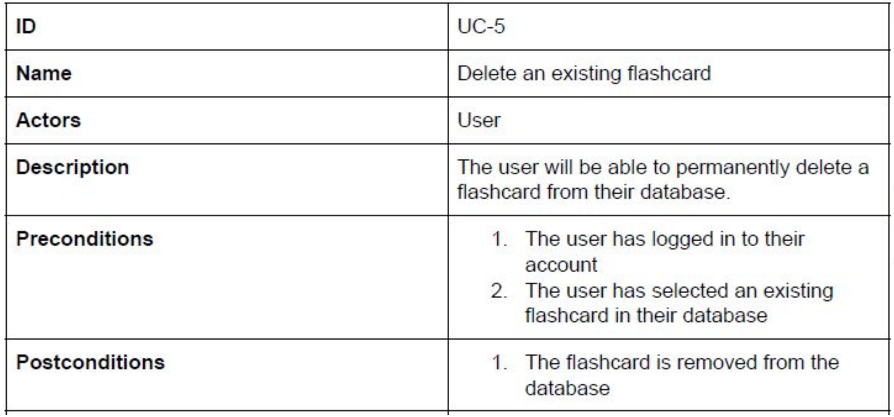
ID	UC-4
Name	Edit an existing flashcard
Actors	User
Description	The user will be able to edit the “front” or “back” field of an existing flashcard in their database.
Preconditions	1. The user has logged in to their account 2. The user has an existing flashcard in their database
Postconditions	1. The flashcard information is updated in the database



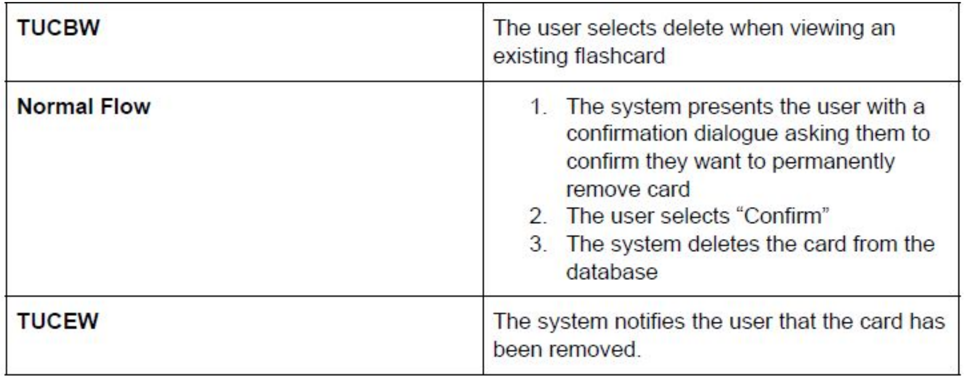
TUCBW	The user selects edit when viewing an existing flashcard
Normal Flow	1. The user enters/changes text in the “front” or “back” fields of the card
TUCEW	The user selects “Done” and is presented with a notification that their changes were successfully added to their database.
Alternate Flow	The user selects cancel in the middle of editing. The changes are discarded and the flashcard is not updated in the database.
Successful outcome	The user is able to update card information in the database
Unsuccessful outcome	The user is unable to update their card’s information
Exceptions	The user leaves one or more fields blank, database is inaccessible

Use Case #5

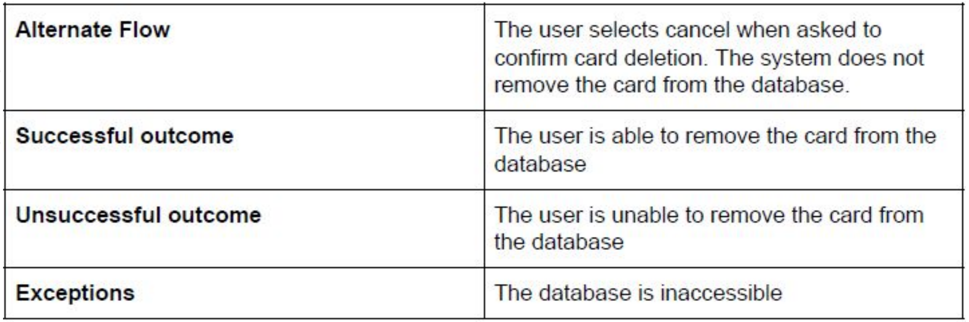
User Story: As a user, I can delete a flashcard from the database so that I will no longer see it in reviews.



ID	UC-5
Name	Delete an existing flashcard
Actors	User
Description	The user will be able to permanently delete a flashcard from their database.
Preconditions	1. The user has logged in to their account 2. The user has selected an existing flashcard in their database
Postconditions	1. The flashcard is removed from the database



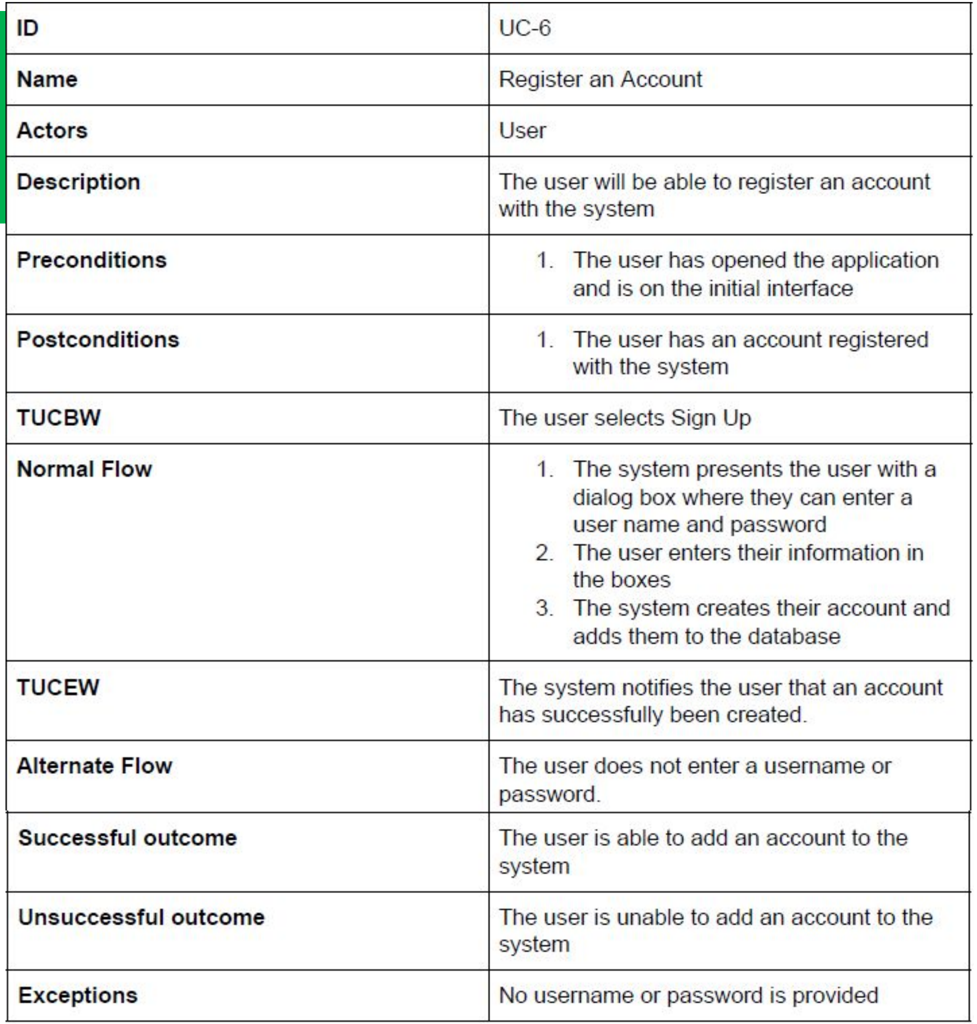
TUCBW	The user selects delete when viewing an existing flashcard
Normal Flow	1. The system presents the user with a confirmation dialogue asking them to confirm they want to permanently remove card 2. The user selects "Confirm" 3. The system deletes the card from the database
TUCEW	The system notifies the user that the card has been removed.



Alternate Flow	The user selects cancel when asked to confirm card deletion. The system does not remove the card from the database.
Successful outcome	The user is able to remove the card from the database
Unsuccessful outcome	The user is unable to remove the card from the database
Exceptions	The database is inaccessible

Use Case #6

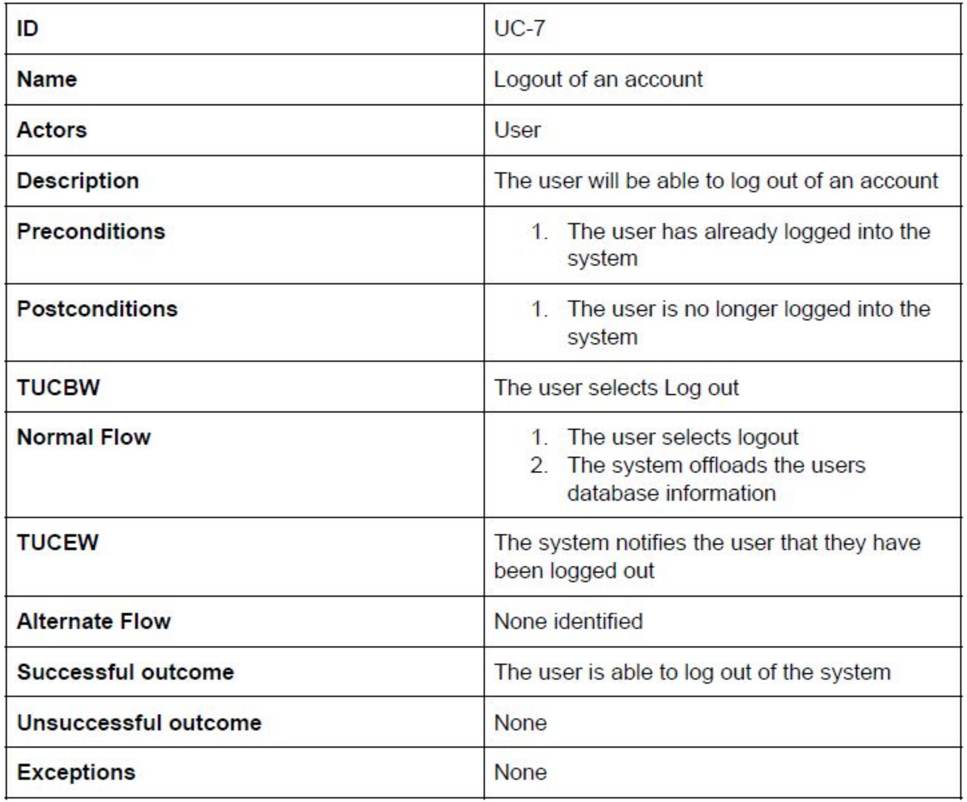
User Story: As a user, I can register an account so that I can keep track of and manage my flashcards.



ID	UC-6
Name	Register an Account
Actors	User
Description	The user will be able to register an account with the system
Preconditions	1. The user has opened the application and is on the initial interface
Postconditions	1. The user has an account registered with the system
TUCBW	The user selects Sign Up
Normal Flow	1. The system presents the user with a dialog box where they can enter a user name and password 2. The user enters their information in the boxes 3. The system creates their account and adds them to the database
TUCEW	The system notifies the user that an account has successfully been created.
Alternate Flow	The user does not enter a username or password.
Successful outcome	The user is able to add an account to the system
Unsuccessful outcome	The user is unable to add an account to the system
Exceptions	No username or password is provided

Use Case #7

User Story: As a user, I can logout of an account so that another user may login to the system.



ID	UC-7
Name	Logout of an account
Actors	User
Description	The user will be able to log out of an account
Preconditions	1. The user has already logged into the system
Postconditions	1. The user is no longer logged into the system
TUCBW	The user selects Log out
Normal Flow	1. The user selects logout 2. The system offloads the users database information
TUCEW	The system notifies the user that they have been logged out
Alternate Flow	None identified
Successful outcome	The user is able to log out of the system
Unsuccessful outcome	None
Exceptions	None

Conclusion

•From this project, we learned most phases of software developing. From having an idea and making a plan, to finishing coding and testing the flashcard application. It's a precious experience in software design. And the most interesting part must be that we know some vocabulary of Chinese, English, and Spanish when creating the database.

•Additionally, we learn lots of Design Patterns. Before this project, we all suffer from how to construct the program. Sometimes, the program is not scalable when it down, so we need to redesign the program. After this project, we learn how to analyst the program and how to choose the pattern. Instead of jumping into the code, we analyst and design first, then build the robust program.