# Flashcardenator 3000

# Iteration 2 Report:

Ryan Harmon
Hua Chuang
Yicun Liu

# Table of Contents

# Requirements

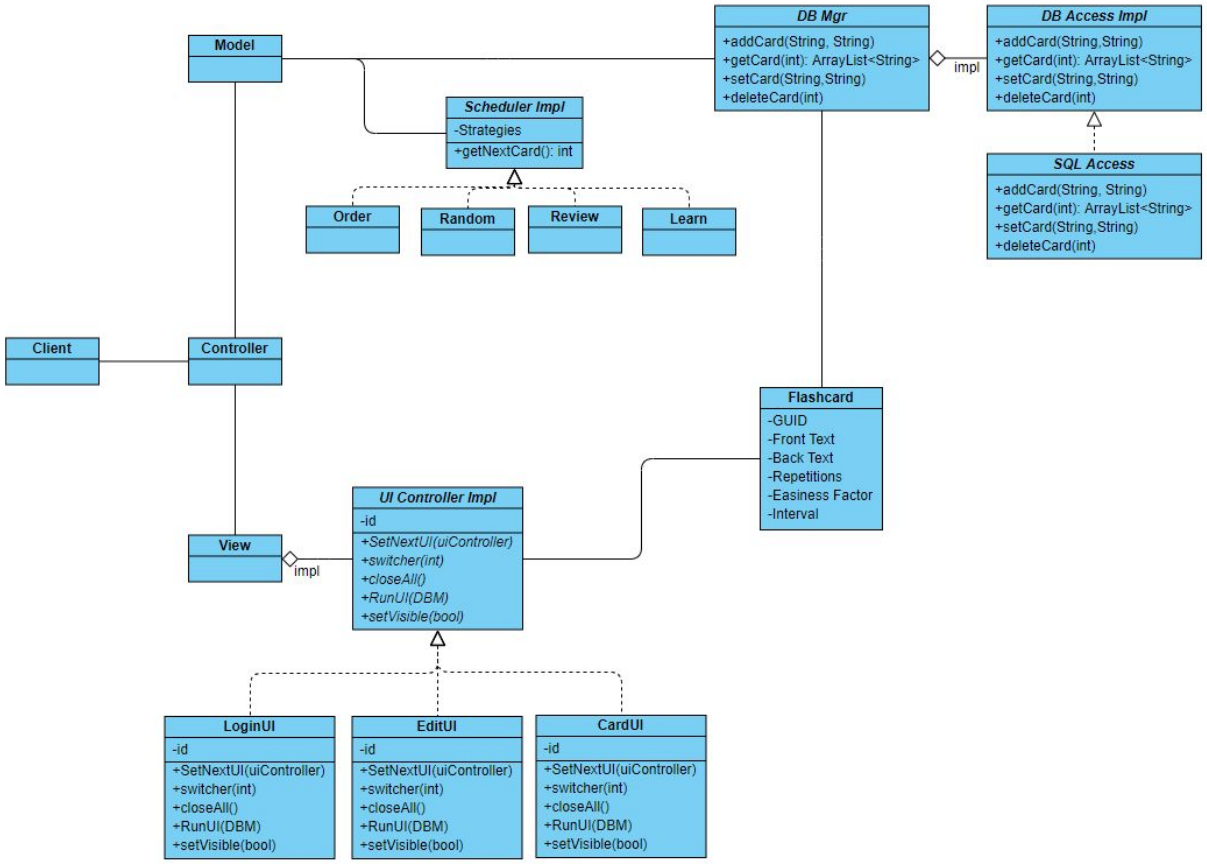
## Functional Requirements (FREQ)

- FREQ-1: The system shall allow the user to login to an existing account
- FREQ-2: The system shall allow the user to register a new account
- FREQ-3: The system shall allow the user to logout of an account
- FREQ-4: The system shall associate and save a database of flashcards, as well as all pertinent scheduling information with a user account
- FREQ-5: Upon login, the system shall present the user with a set of flashcards from their database to review, per the SM2 algorithm
- FREQ-6: The system shall allow users to create flashcards
- FREQ-7: The system shall allow users to edit existing flashcards
- FREQ-8: The system shall allow users to delete existing flashcards
- FREQ-9: The system shall allow users to select a qualitative score of 0-4 in response to a flashcard review, which is used in the SM2 algorithm
- FREQ-10: The system shall allow users to view a list of cards in the database
- FREQ-11: The system shall track user performance on flashcard reviews, and present the user with statistics on their progress
- FREQ-12: The system shall allow the user to review flashcards in learning mode
- FREQ-13: The system shall allow the user to review all flashcards in a random order
- FREQ-14: The system shall allow the user to review all flashcards in alphabetical order

## Non-functional Requirements (NREQ)

- NREQ-1: The system shall use the SM2 algorithm for spaced repetition scheduling
- NREQ-2: The system shall support international character sets in flashcards
- NREQ-3: The login system shall prevent SQL Injection.
- NREQ-4: The login system shall require the user to use a secure password per NIST guidelines

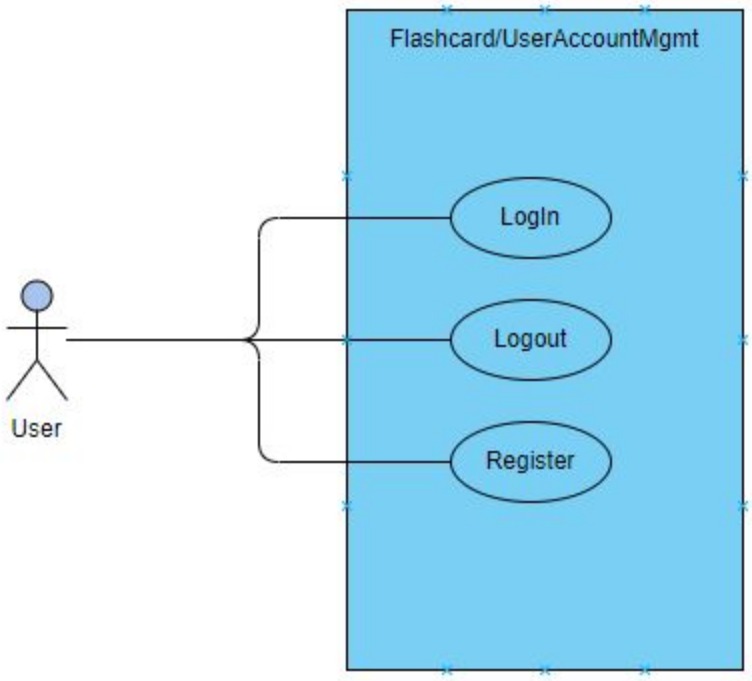
# Design Class Diagram

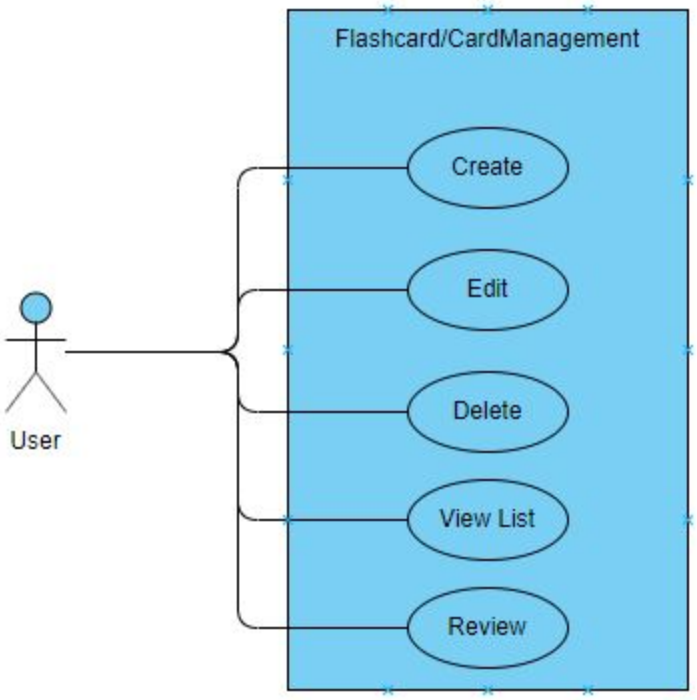



Implementation notes: The login UI currently interfaces directly with the database manager to create a user and setup user specific data. This will be moved in the next iteration.

# Use Case Diagrams

Since many of the use case broadly fall into the same grouping, we have logically broken our diagrams into two separate view. The first is related to account management, and the second is related to editing or reviewing flashcards.




Flashcard/CardManagement
Create
Edit
Delete
View List
Review
User

# 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<br>2. The user is not currently logged in to any account |
| **Postconditions** | 1. The user is logged in to the system<br>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<br>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<br>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<br>2. The system presents the user with the front side of a flash card<br>3. The user selects "show backside"<br>4. The system presents the user with the back of the flashcard<br>5. The user selects a difficulty/ease response button<br>6. The system updates the card scheduling information in the database and removes it from the review set<br>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<br>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<br>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<br>2. The user selects "Confirm"<br>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<br>2. The user enters their information in the boxes<br>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.

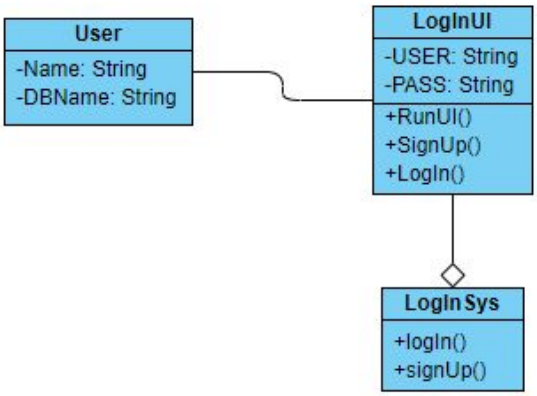
| | |
|---|---|
| **ID** | UC-7 |
| **Name** | Logout of an account |
| **Actors** | User |
| **Description** | The user will be able to log out of an account |
| **Preconditions** | 1. The user has already logged into the system |
| **Postconditions** | 1. The user is no longer logged into the system |
| **TUCBW** | The user selects Log out |
| **Normal Flow** | 1. The user selects logout<br>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 |

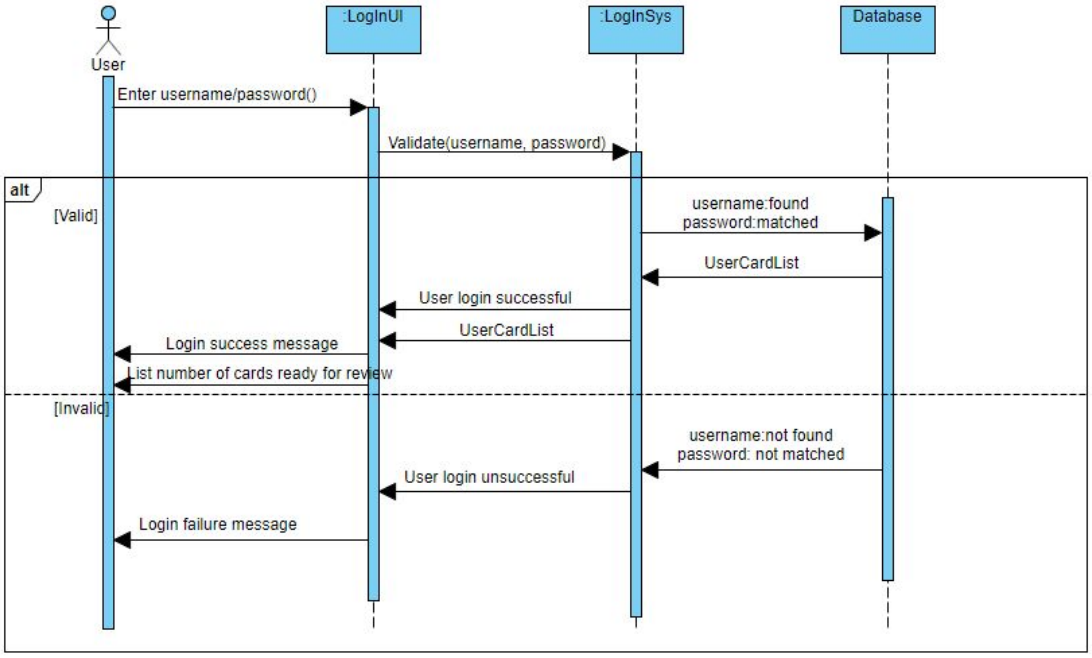
# Requirements Traceability

| | UC-1 | UC-2 | UC-3 | UC-4 | UC-5 | UC-6 | UC-7 |
|---|---|---|---|---|---|---|---|
| FREQ-1 | X | | | | | | |
| FREQ-2 | | | | | | X | |
| FREQ-3 | | | | | | | X |
| FREQ-4 | X | | | | | | |
| FREQ-5 | | | X | | | | |
| FREQ-6 | | X | | | | | |
| FREQ-7 | | | | X | | | |
| FREQ-8 | | | | | X | | |
| FREQ-9 | | | | | | | |
| FREQ-10 | | | | | | | |
| FREQ-11 | | | | | | | |
| FREQ-12 | | | | | | | |
| FREQ-13 | | | | | | | |
| FREQ-14 | | | | | | | |
| NREQ-1 | | | | | | | |
| NREQ-2 | | | | | | | |
| NREQ-3 | | | | | | | |
| NREQ-4 | | | | | | | |

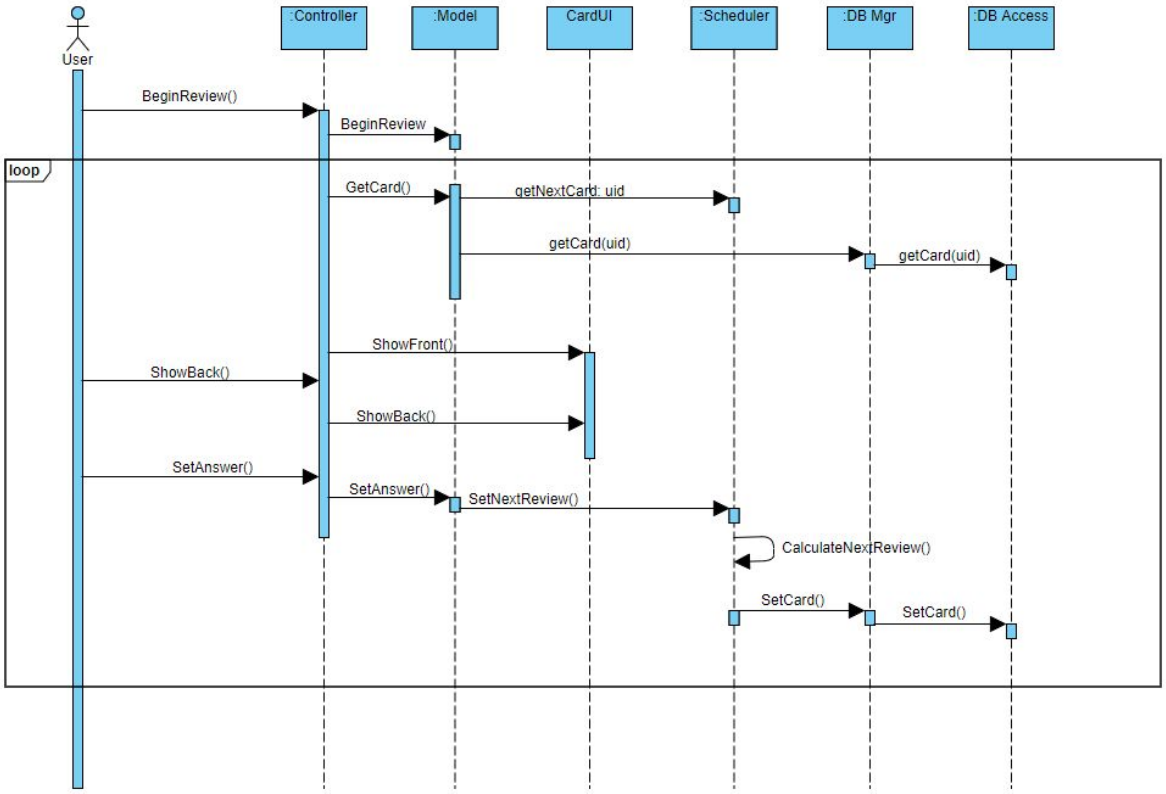
# Sequence Diagrams

## User Login

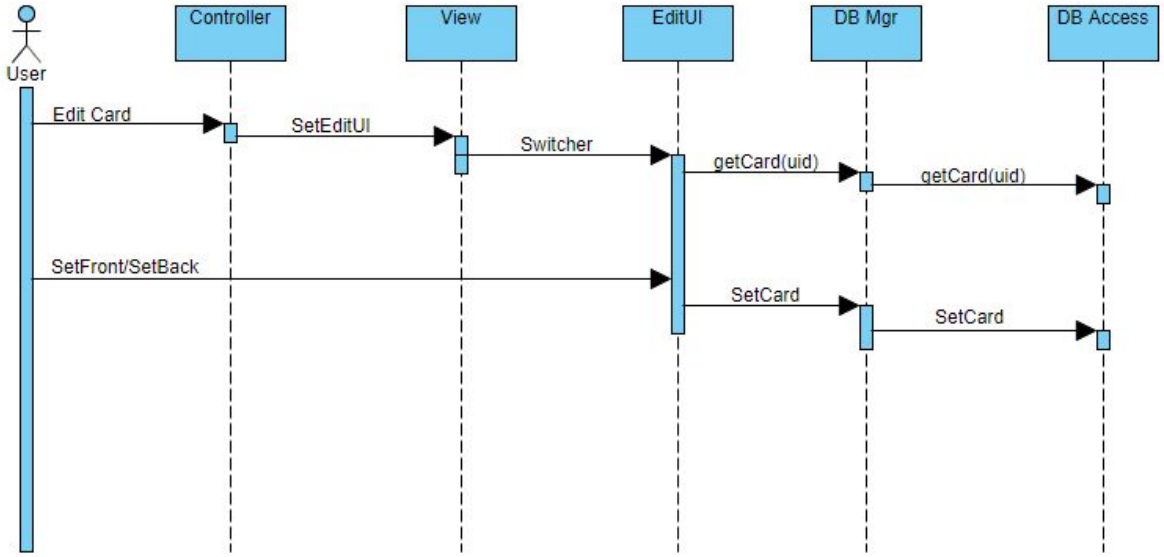


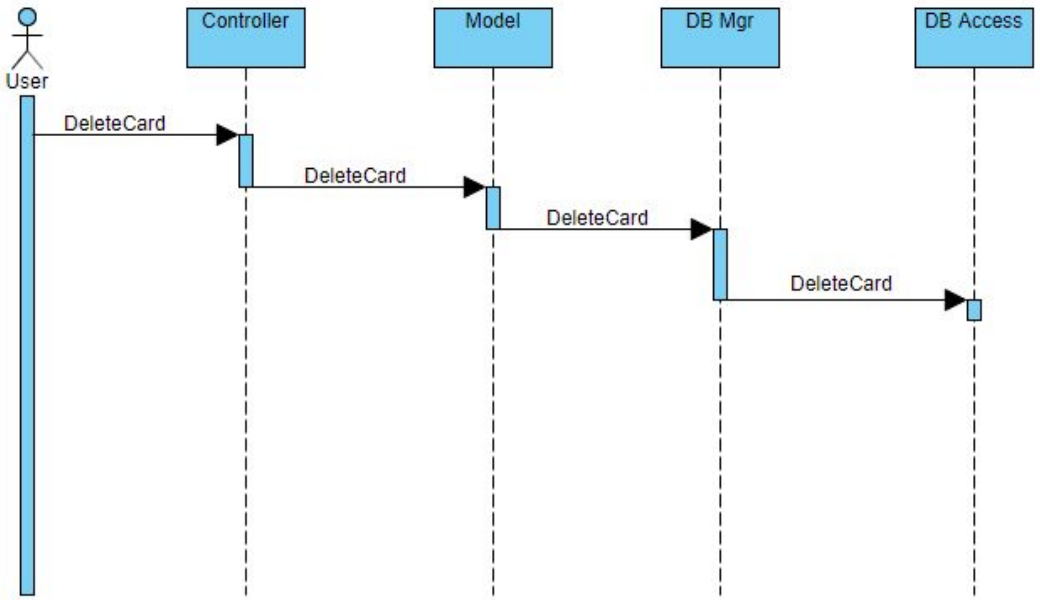
## Review Card




## Edit/Create Card



Delete Card


User
Controller
Model
DB Mgr
DB Access
DeleteCard
DeleteCard
DeleteCard
DeleteCard

# Project Planning

All planning for this iteration was done using github project management. Please see the project “Iteration 2” that we created for a detailed list of tasks and assignees.