Farm City Application Use Case Realization

G2-T08

Chee Hong Kun
Nguyen Phuong Thao
Jeremy Lee Ting Kok

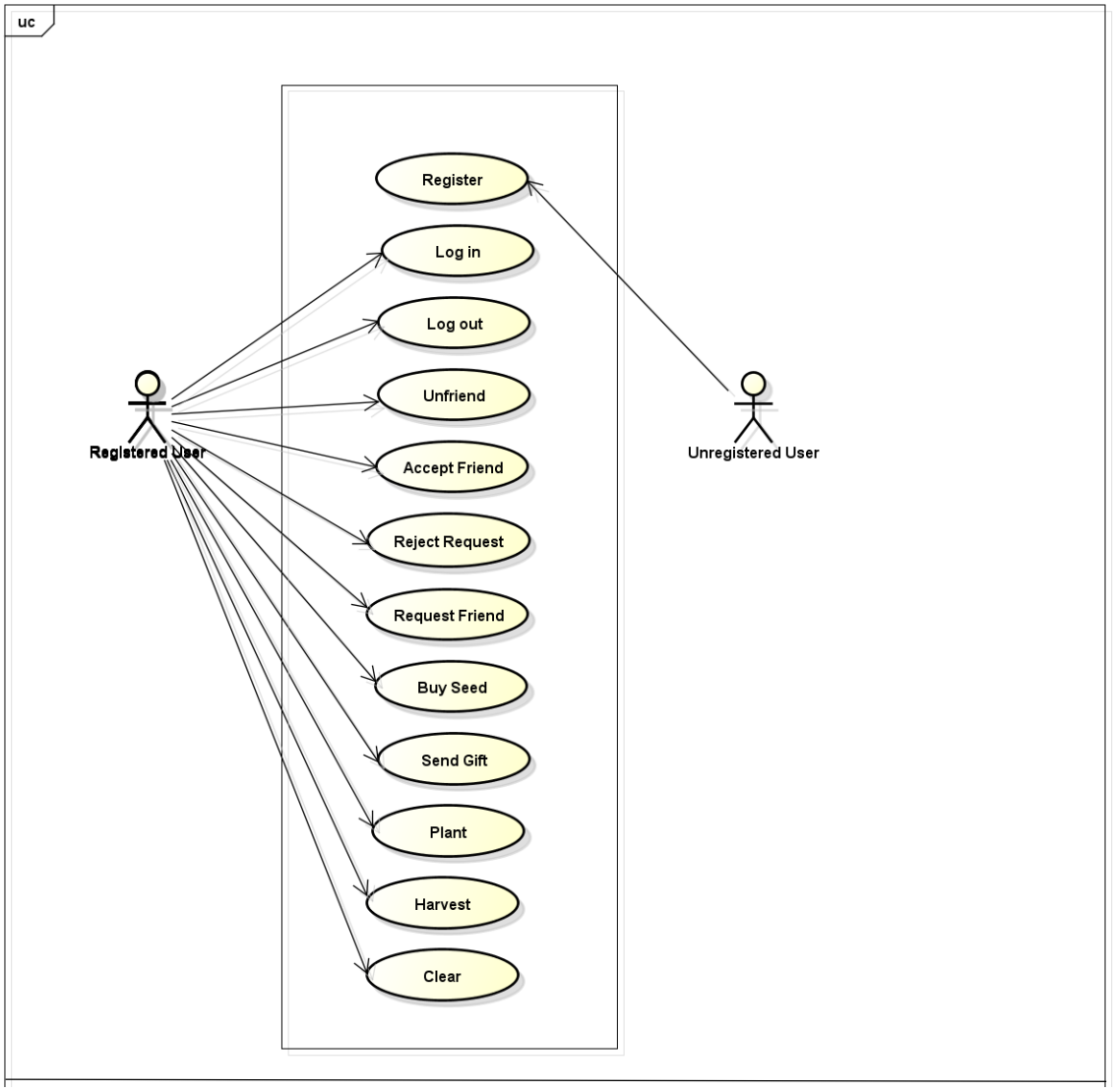
Table of Contents

1. USE CASE 4
2. DOMAIN DIAGRAM 5
3. CLASS DIAGRAM 6
4. USE CASE SCENARIOS 7
4.1 Register 7
4.2 Login 7
4.3 Logout 8
4.4 Unfriend 8
4.5 Request friend 8
4.6 Accept request 9
4.7 Reject request 9
4.8 Plant crop 9
4.9 Harvest crop 10
4.10 Clear 10
4.11 Buy seed 10
4.12 Gift seed 11
5. SYSTEM SEQUENCE DIAGRAM 12
5.1 Register 12
5.1.1 SSD 12
5.2 Login 12
5.2.1 SSD 12
5.3 Logout 13
5.3.1 SSD 13
5.4 Unfriend 13
5.4.1 SSD 13
5.5 Request friend 14
5.5.1 SSD 14
5.6 Accept request 15
5.6.1 SSD 15
5.7 Reject request 15
5.7.1 SSD 15
5.8 Plant crop 16
5.8.1 SSD 16
5.9 Harvest crop 16
5.9.1 SSD 16

5.10 Clear 17
5.10.1 SSD 17
5.11 Buy seed 17
5.11.1 SSD 17
5.12 Gift seed 18
5.12.1 SSD 18
6. SEQUENCE DIAGRAM 18
6.1 Register 18
6.1.1 SD 18
6.2 Login 19
6.2.1 SD 19
6.3 Logout 20
6.3.1 SD 20
6.4 Unfriend 21
6.4.1 SD 21
6.5 Request friend 21
6.5.1 SD 21
6.6 Accept request 22
6.6.1 SD 22
6.7 Reject request 22
6.7.1 SD 22
6.8 Plant crop 23
6.8.1 SD 23
6.9 Harvest crop 24
6.9.1 SD 24
6.10 Clear crops 25
6.10.1 SD 25
6.11 Buy seed 26
6.11.1 SD 26
6.12 Gift seed 27
6.12.1 SD 27
7. OBJECT-ORIENTED DESIGN CONSIDERATIONS 28
8. MISCELLANEOUS 29
9. USABILITY AND SCREENSHOTS 30
9.1 Welcome Menu 30
9.1.1 Screenshot 30

9.2 Register 30
9.2.1 Screenshot 30
9.3 Login 31
9.3.1 Screenshot 31
9.4 Unfriend 31
9.4.1 Screenshot 31
9.5 Request friend 32
9.5.1 Screenshot 32
9.6 Accept request 32
9.6.1 Screenshot 32
9.7 Reject request 33
9.7.1 Screenshot 33
9.8 Plant crop 33
9.8.1 Screenshot 33
9.9 Harvest crop 34
9.9.1 Screenshot 34
9.10 Clear 34
9.10.1 Screenshot 34
9.11 Buy seed 35
9.11.1 Screenshot 35
9.12 Gift seed 36
9.12.1 Screenshot 36

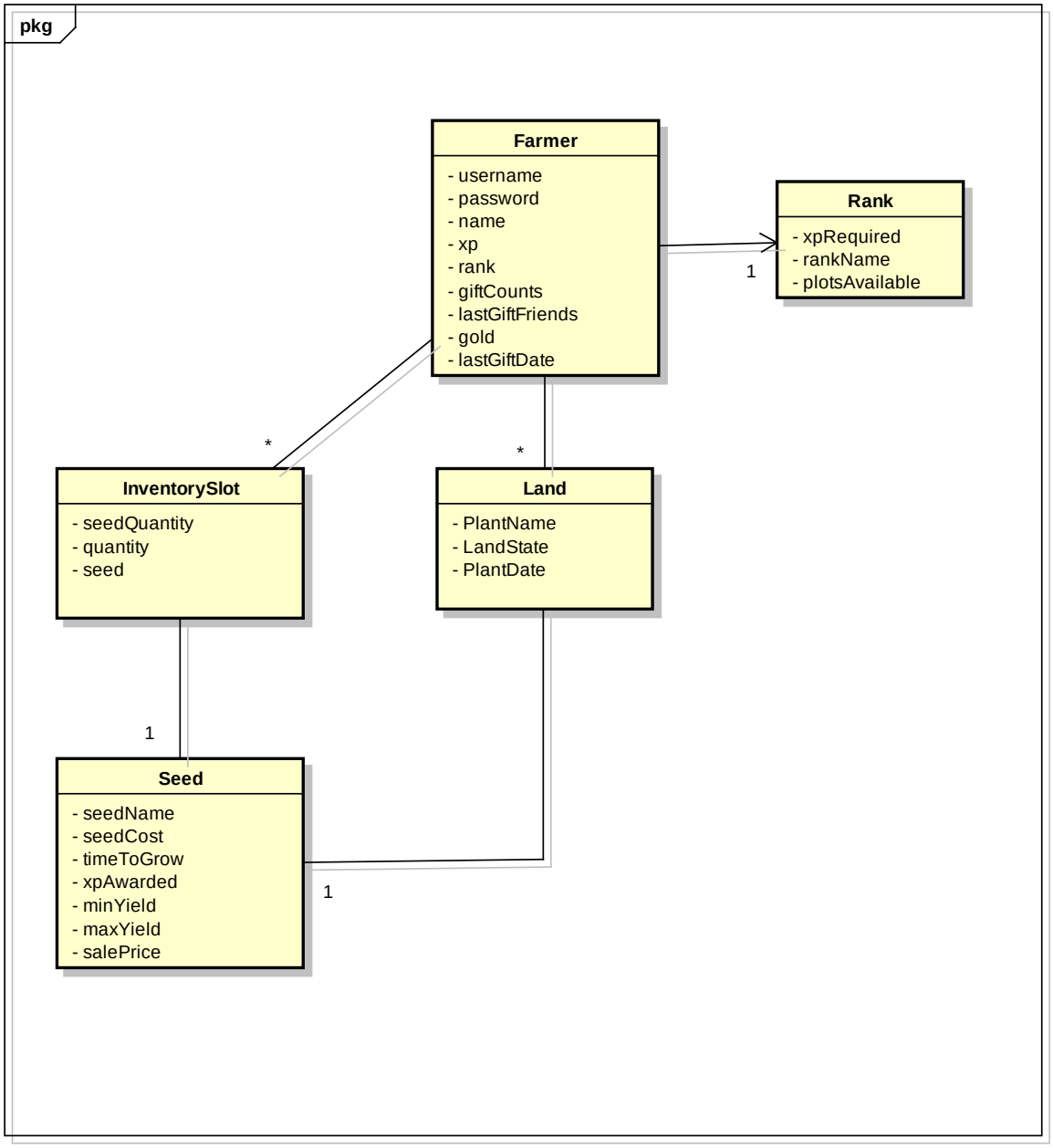
1. USE CASE



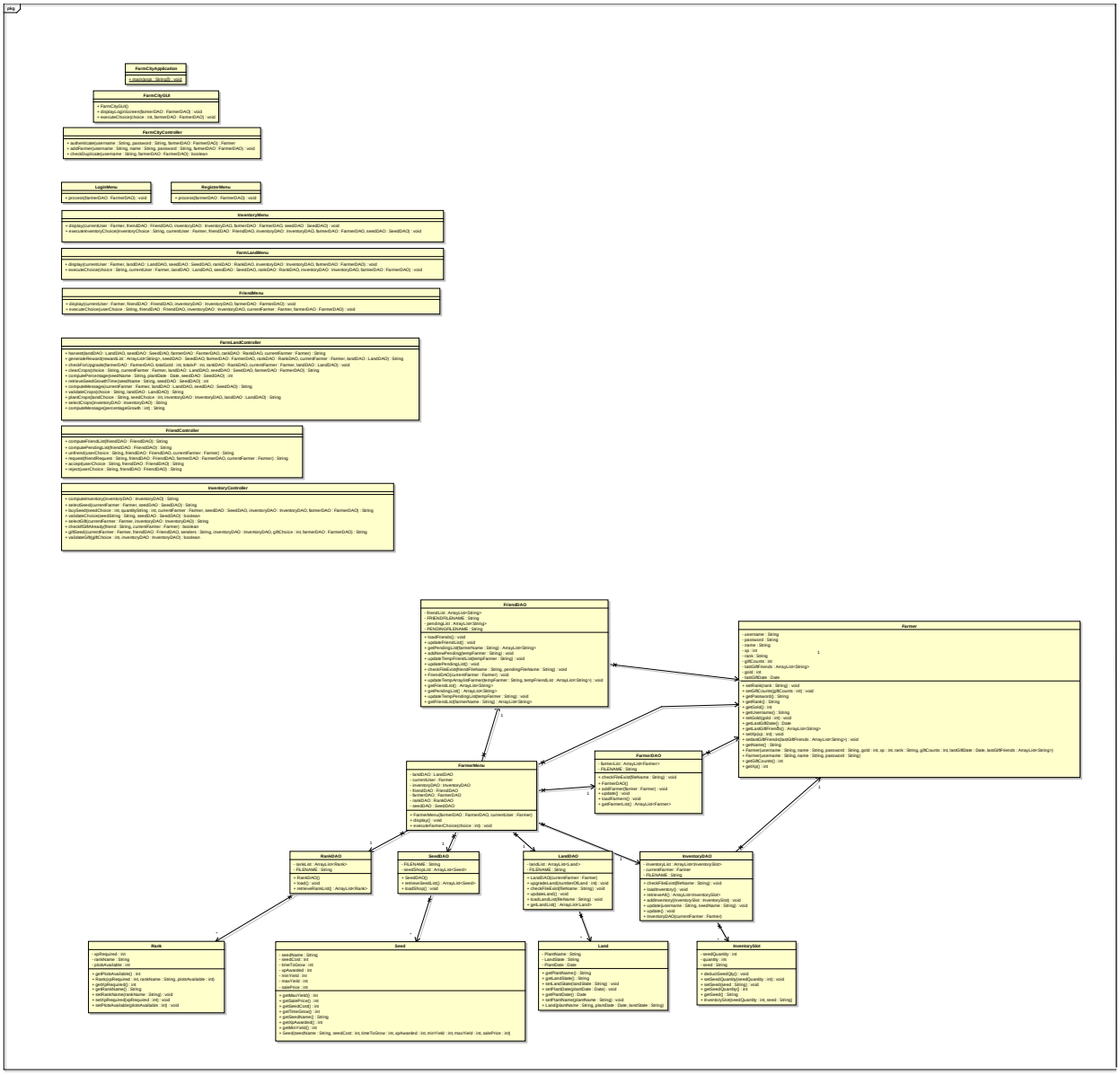
Use Case	Description
Register	This use case allows unregistered player to register into the game.
Login	This use case allows registered player to login.
Logout	This use case allows registered player to logout
Unfriend	This use case allows registered player to unfriend his friend.
Accept Friend	This use case allows registered player to accept friend request(s).
Reject Request	This use case allows registered player to reject friend request(s).
Request Friend	This use case allows registered player to send Friend Request to another user(s).
Plant	This use case allows registered player to plant seed on his land.
Harvest	This use case allows registered player to harvest his crop(s).
Clear	This use case allows registered player to clear his wilted crops(s).
Buy Seed	This use case allows librarian to buy seed(s) from Seed Shop.
Send Gift	This use case allows registered player to send gift(s) to his friends.

Assume game management is out of scope for this system.

2. DOMAIN DIAGRAM



3. CLASS DIAGRAM



4. USE CASE SCENARIOS

4.1 Register

Actor: Unregistered user

Precondition: None

Main flow of events

1. This use case begins when the unregistered user wants to register.
2. The user provides his registration details – username and password.
3. The system validates registration details and outputs registration message.
4. Use case ends.

Alternate Flow(s):

Invalid registration characters provided.

i. The system informs the user that registration credentials are invalid.
ii. Use case ends.

Username already registered.

i. The system informs the user that his registration credentials is taken.
ii. Use case ends.

4.2 Login

Actor: Registered user

Precondition: None

Main flow of events

1. This use case begins when the registered user wants to login.
2. The user provides his login credentials – username and password.
3. The system verifies his login credentials and directs the user to the welcome screen
4. Use case ends.

Alternate Flow(s):

Login credentials are incorrect.

i. The system informs the user that login credentials are incorrect.
ii. Use case ends.

4.3 Logout

Actor: Registered user

Precondition: The user is logged in.

Main flow of events

1. This use case begins when the user wants to logout.
2. The system directs the user to the logout screen
3. Use case ends.

4.4 Unfriend

Actor: Registered user

Precondition: User has successfully logged in and is in the friend menu.

Main flow of Events

1. The use case begins when the user chooses to unfriend an existing friend.
2. The user selects a friend to unfriend.
3. The system displays friend list.
4. The use case ends.

Alternate Flow(s):

User has no friends

i) The system displays to the user that the friend list is empty.
ii) The use case ends.

4.5 Request friend

Actor: Registered user

Precondition: User has successfully logged in and is in the friend menu.

Main flow of Events

1. The use case begins when the user chooses to send a friend request.
2. The user inputs the username of other registered user.
3. The system displays a send request message.
4. The use case ends.

Alternate Flow(s):

Request username not registered

iii) The system displays to the user an invalid request message.
iv) The use case ends.

4.6 Accept request

Actor: Registered user

Precondition: User has successfully logged in and is in the friend menu.

Main flow of Events

1. The use case begins when the user chooses to accept a friend request.
2. The user selects a friend to accept.
3. The system displays friend added message.
4. The use case ends.

Alternate Flow(s):

User has no friend request

i. The system displays an empty friend request list.
ii. The use case ends.

4.7 Reject request

Actor: Registered user

Precondition: User has successfully logged in and is in the friend menu.

Main flow of Events

1. The use case begins when the user chooses to reject an existing friend request.
2. The user selects a friend request to reject.
3. The system displays friend rejected message and friend menu.
4. The use case ends.

Alternate Flow(s):

User has no friend request

v) The system displays to the user that the friend request list is empty.
vi) The use case ends.

4.8 Plant crop

Actor: Registered User

Precondition: User has successfully logged in and is in the farm menu.

Main flow of Events

1. The use case begins when the farmer chooses a piece of land to grow a crop.
2. The user selects a piece of land to grow a crop.
3. The system displays all the available crops that can be grown.
4. The user selects the crop to be grown.
5. The system displays the new crops planted on the plot of land.

6. The use case ends.

Alternate Flow(s):
User has no available seeds to be grown
i) The system displays to the user that his inventory store is empty.
ii) The use case ends.

User has no available plot of land
i) The system displays to the user that the current plot of land is occupied
ii) The use case ends.

4.9 Harvest crop

Actor: Registered User

Precondition: Farmer has successfully logged in and is in the farm menu.

Main flow of Events
1. The use case begins when the farmer wants to harvest a crop.2. The system displays to the farmer the total amount of units, xp and gold that have been earned from harvesting fully grown crops.
3. The use case ends.

Alternate Flow(s):
2a: Farmer has no available crops to be harvest
i) The system displays to the farmer that there is no crop available to be harvested.
ii) The use case ends.

4.10 Clear crops

Actor: Registered User

Precondition: User has successfully logged in and is in the farm menu.

Main flow of Events
1. The use case begins when the user wants to clear a piece of crop.
2. The user selects the plot of land which has withered to be cleared away.
3. The system displays that the plot of land has been cleared.
4. The use case ends.

Alternate Flow(s):
2a: User has no withered plot of land
i) The system displays to the user that the user has no withered plot of land
ii) The use case ends

4.11 Buy seed

Actor: Registered User

Precondition: User has successfully logged in and is in the Inventory menu.

Main flow of Events

1. The use case begins when the user wants to buy Seed(s) for his Inventory.
2. The user selects the Seed that he wants to buy and keys in the quantity.
3. The system displays that the seed has been added into his Inventory.
4. The use case ends.

Alternate Flow(s):

2a: User keys in an invalid choice of seed.

i) The system displays to the user that his choice of seed is invalid. His choice might be invalid due to invalid seed's name or invalid quantity.

2b: User doesn't have enough Gold to buy seed.

i) The system displays to the user that the user doesn't have enough Gold to buy seed.

ii) The use case ends

4.12 Gift seed

Actor: Registered User

Precondition: User has successfully logged in and is in the farm menu.

Main flow of Events

1. The use case begins when the user wants to send gifts to his friend(s).
2. The user selects the seed that he wants to send and keys in his friends' names.
3. The system displays that the gifts have been sent successfully.
4. The use case ends.

Alternate Flow(s):

2a: His friend has been sent a gift within the specific day.

i) The system displays to the user that his friend has received a gift for today and User has to key in another name since one friend can only receive one gift per day.

2b: User doesn't have enough seed to send to his friends.

i) The system displays to the user that he doesn't have enough seed in his inventory to send to his friends. Seed sent as gift is taken out from Farmer's own inventory.

2c: User requests to send more than 5 Gifts per day.

i) The system displays to the user that he has reached his limit. Each farmer can only send out 5 gifts per day.

2d: User keys in an invalid Gift choice

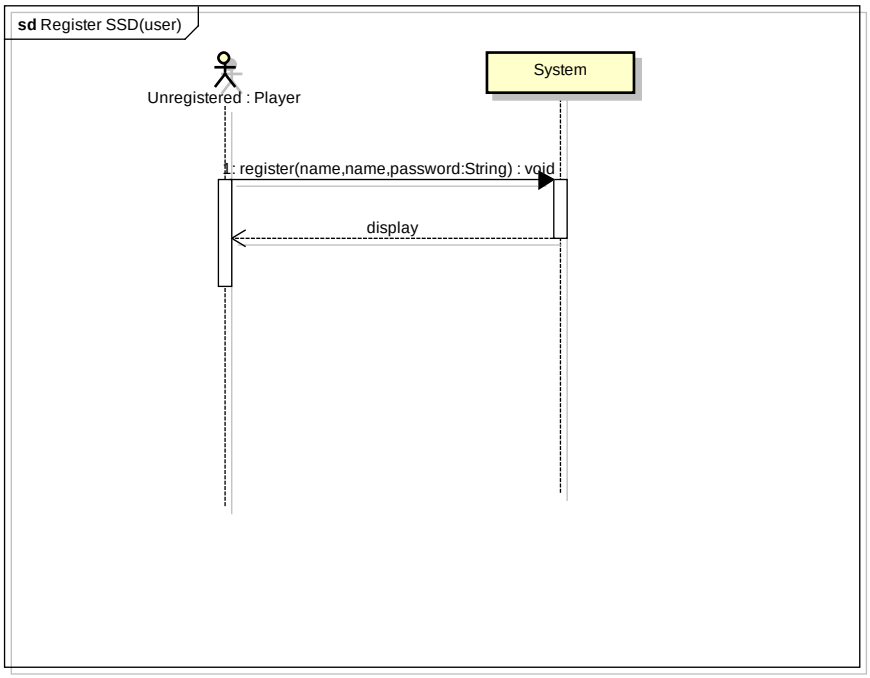
i) The system displays to the user that his Gift choice is invalid and he is requested to key in another valid choice.

ii) The use case ends

5. SYSTEM SEQUENCE DIAGRAM

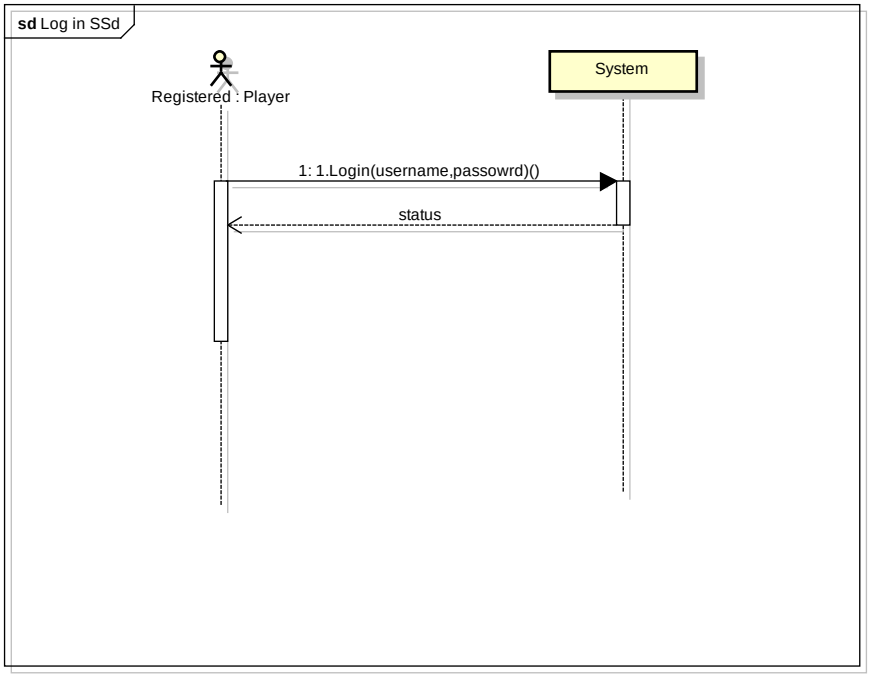
5.1 Register

5.1.1 SSD



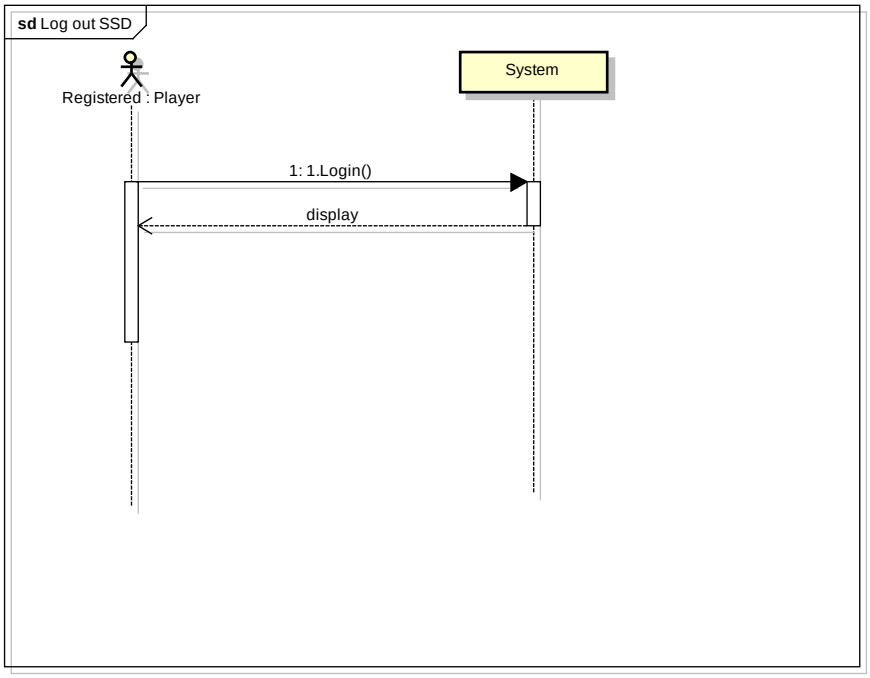
5.2 Login

5.2.1 SSD



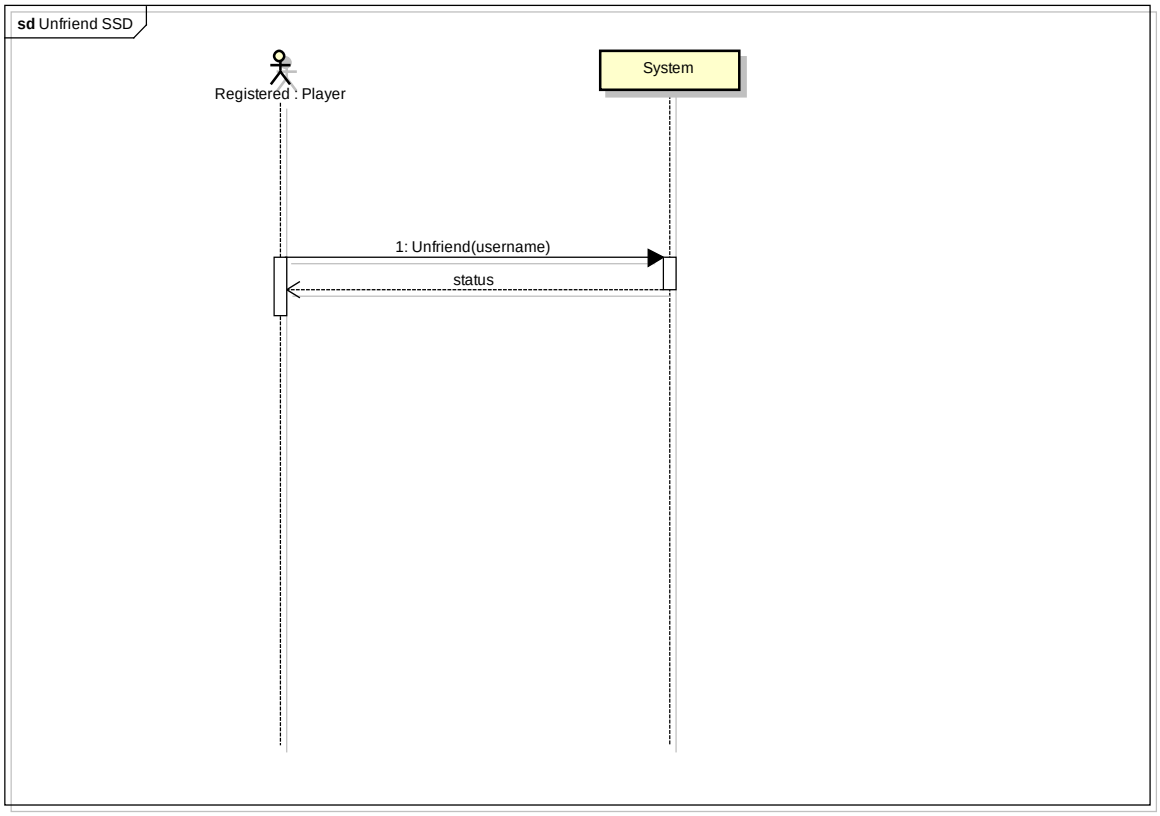
5.3 Logout

5.3.1 SSD



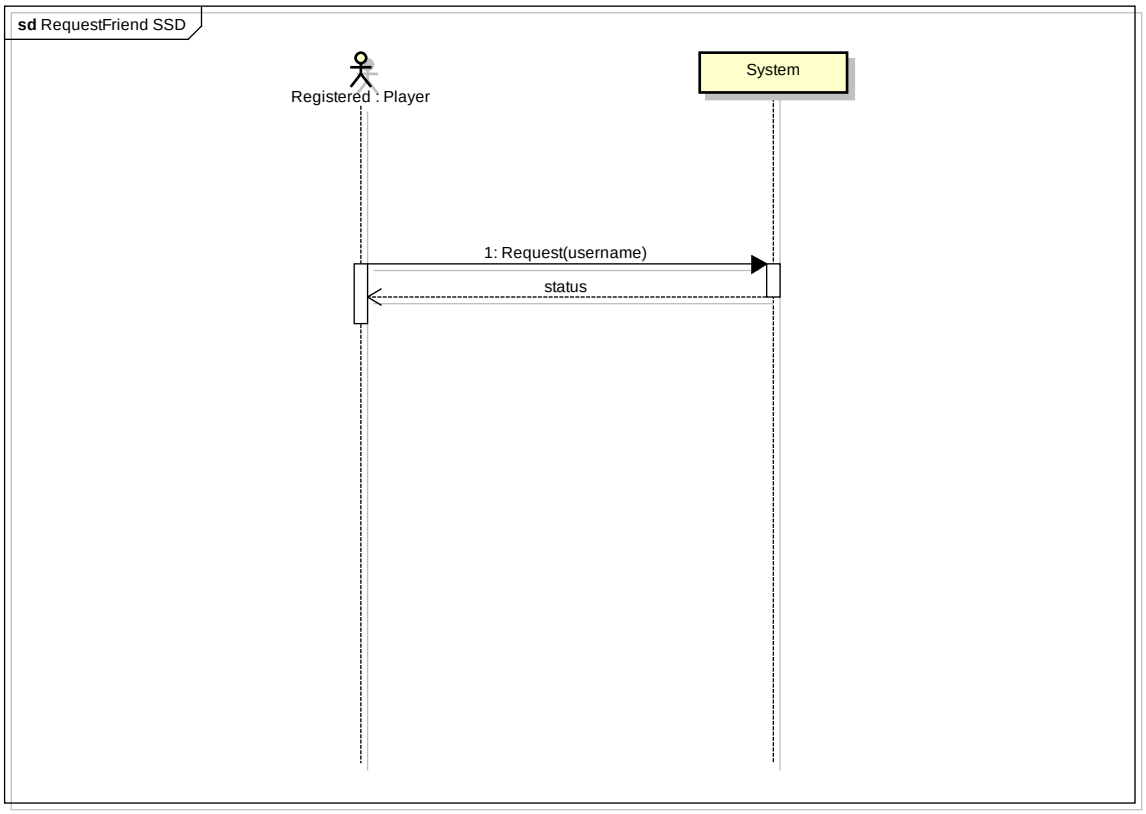
5.4 Unfriend

5.4.1 SSD



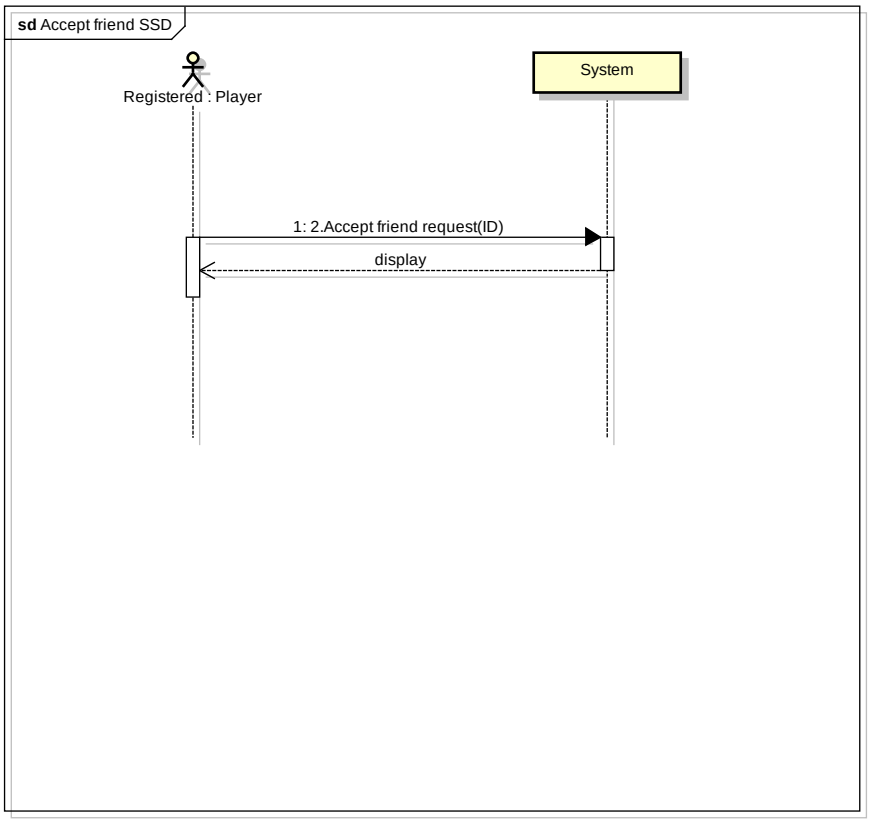
5.5 Request friend

5.5.1 SSD



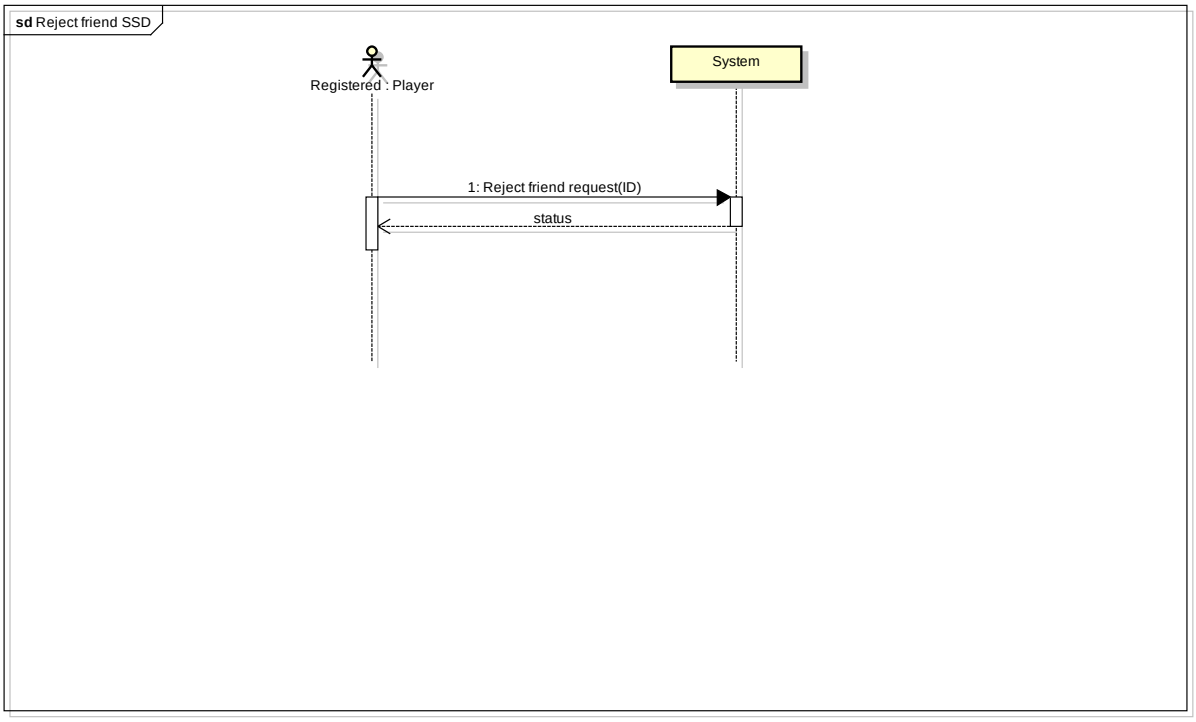
5.6 Accept request

5.6.1 SSD



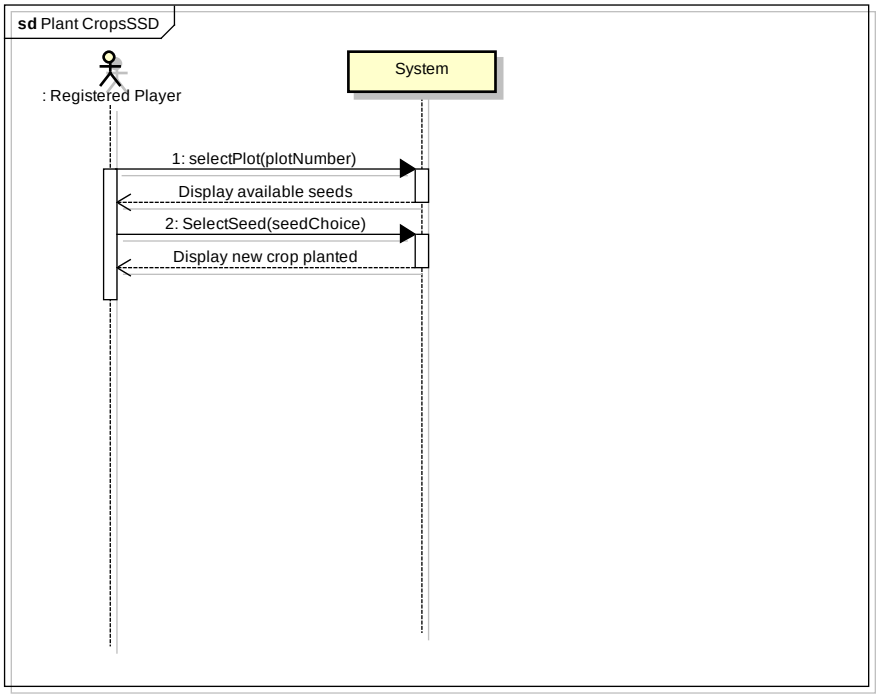
5.7 Reject request

5.7.1 SSD



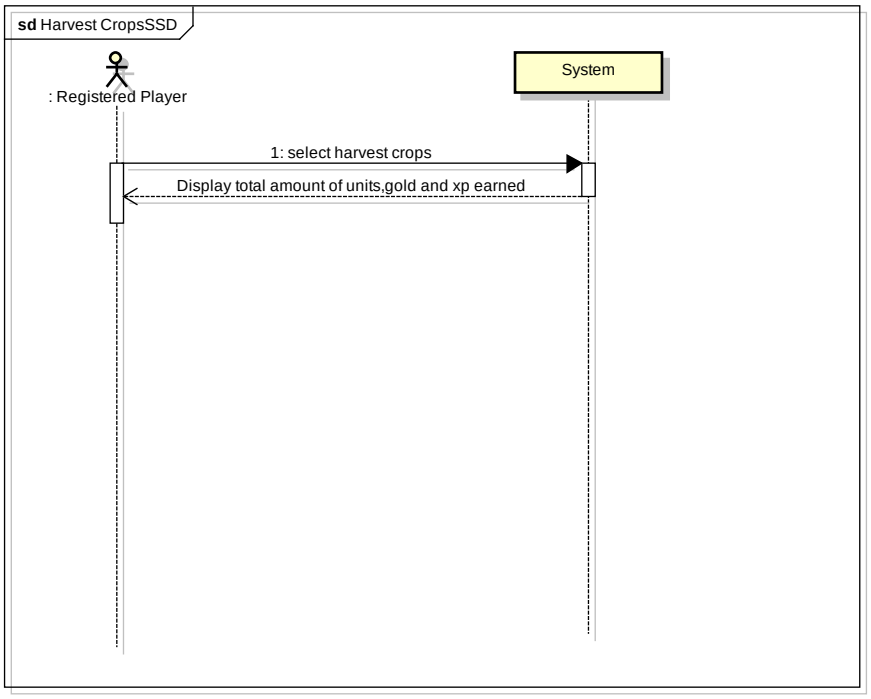
5.8 Plant crop

5.8.1 SSD



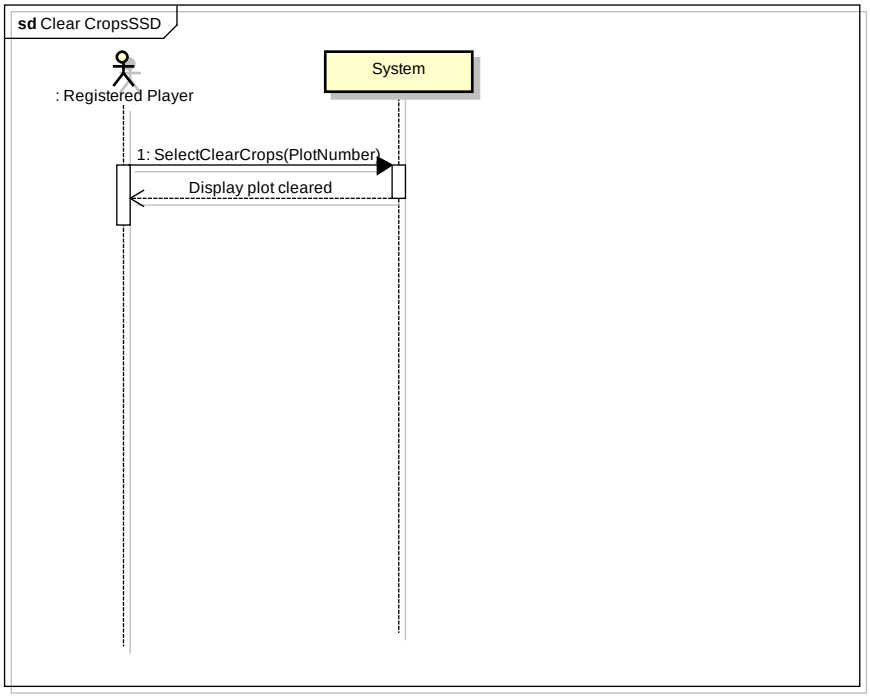
5.9 Harvest crop

5.9.1 SSD



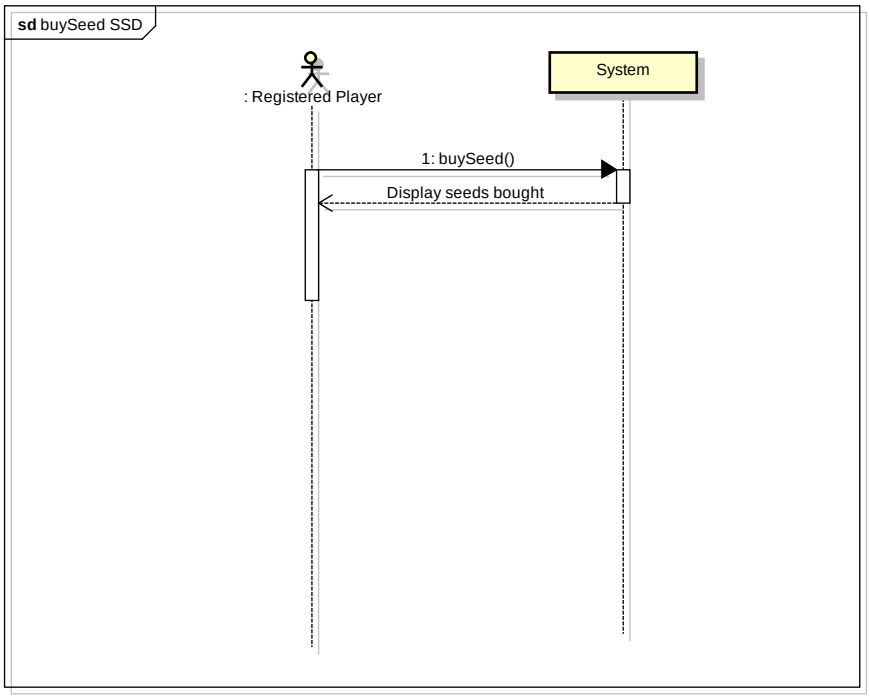
5.10 Clear crops

5.10.1 SSD



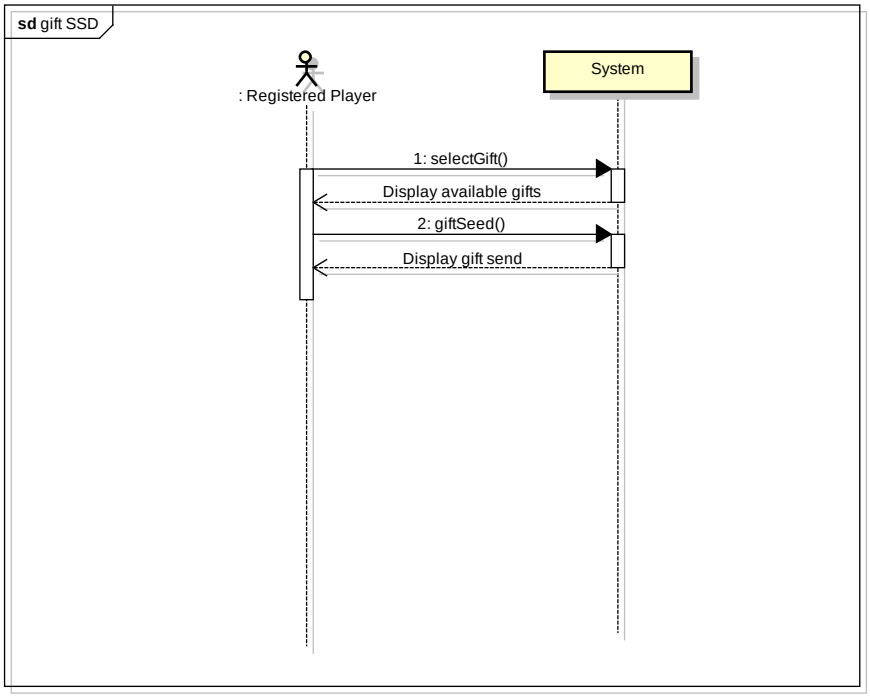
5.11 Buy seed

5.11.1 SSD



5.12 Gift seed

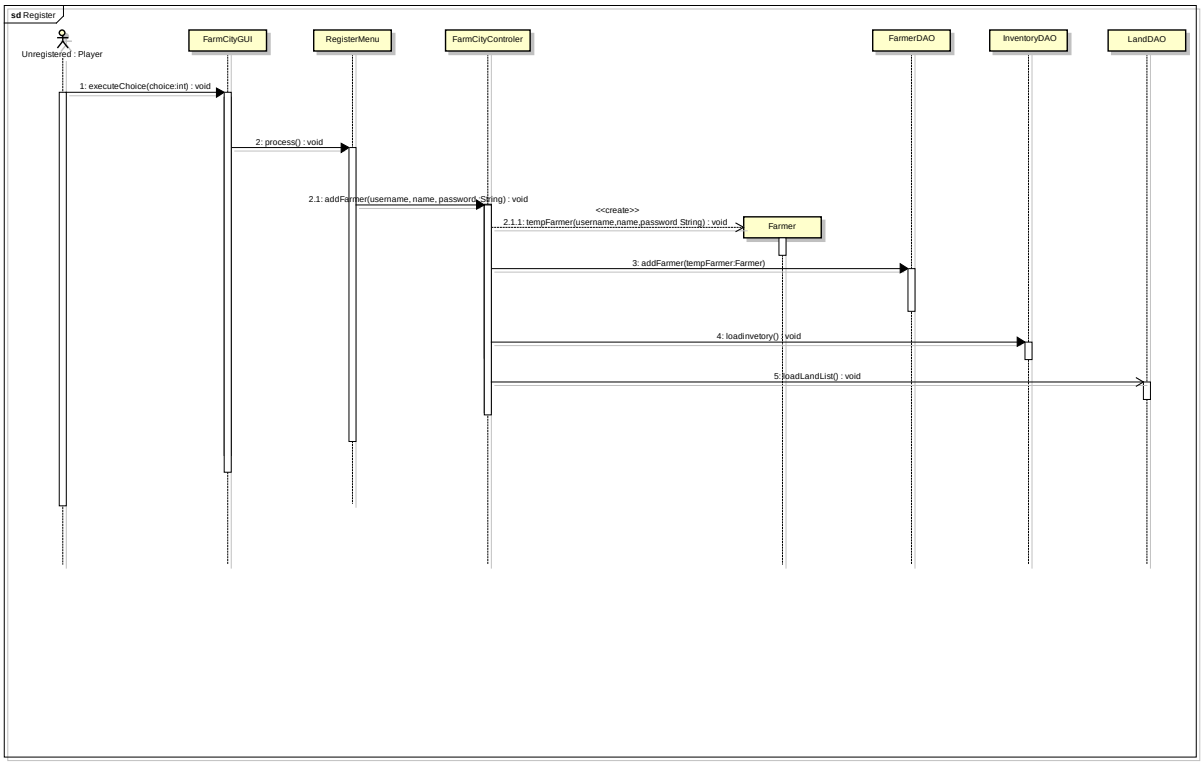
5.12.1 SSD



6. SEQUENCE DIAGRAM

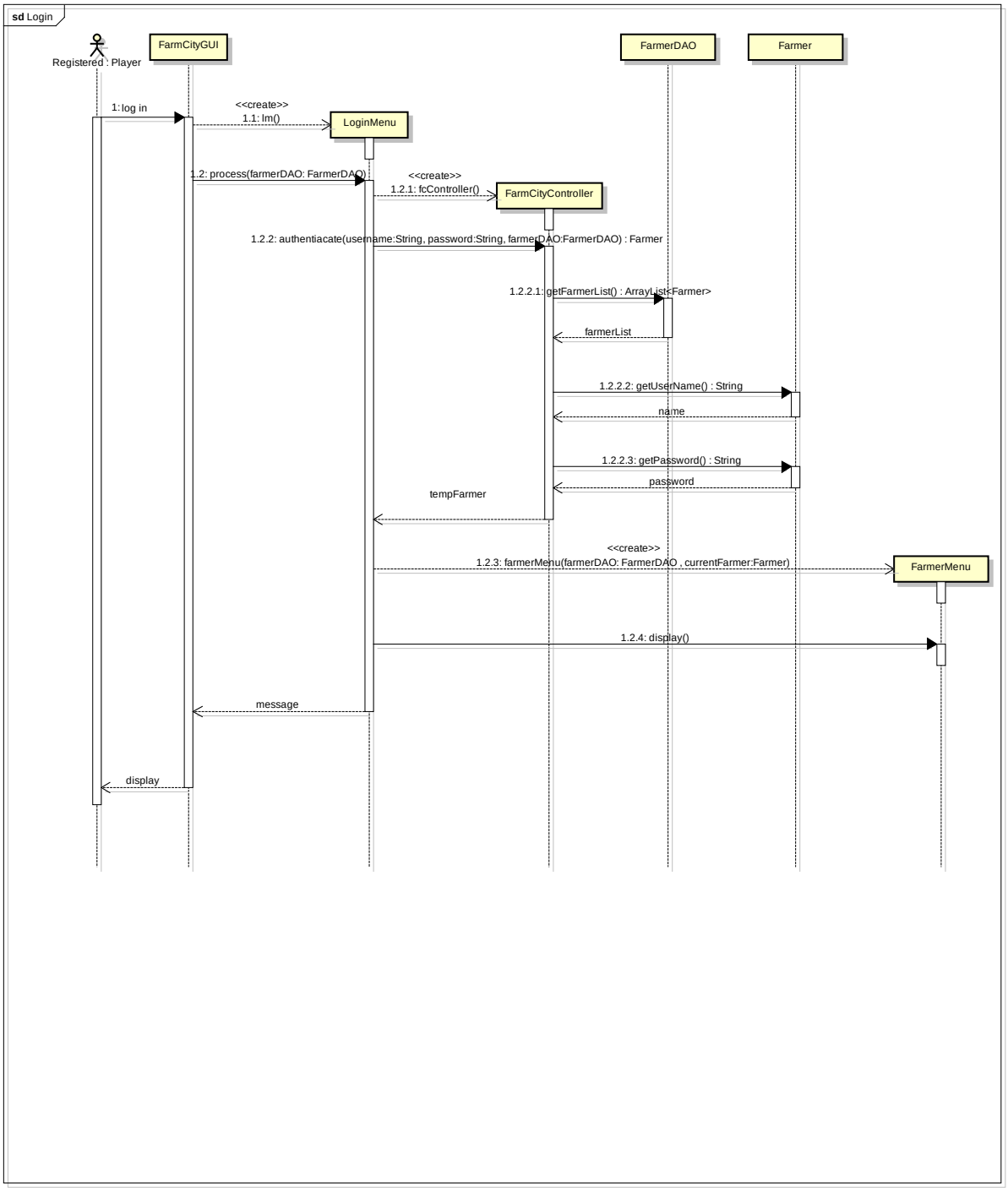
6.1 Register

6.1.1 SD



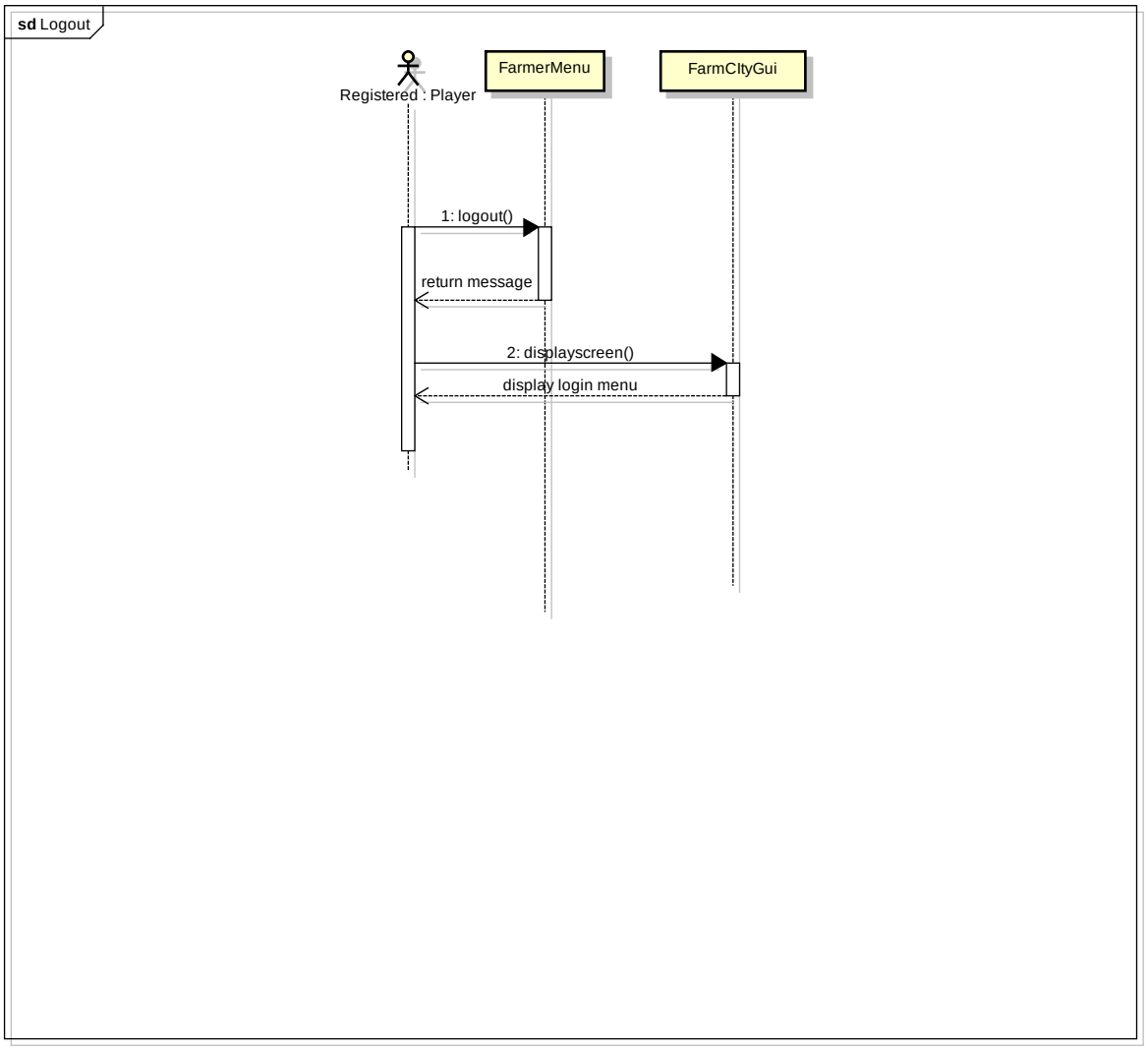
6.2 Login

6.2.1 SD



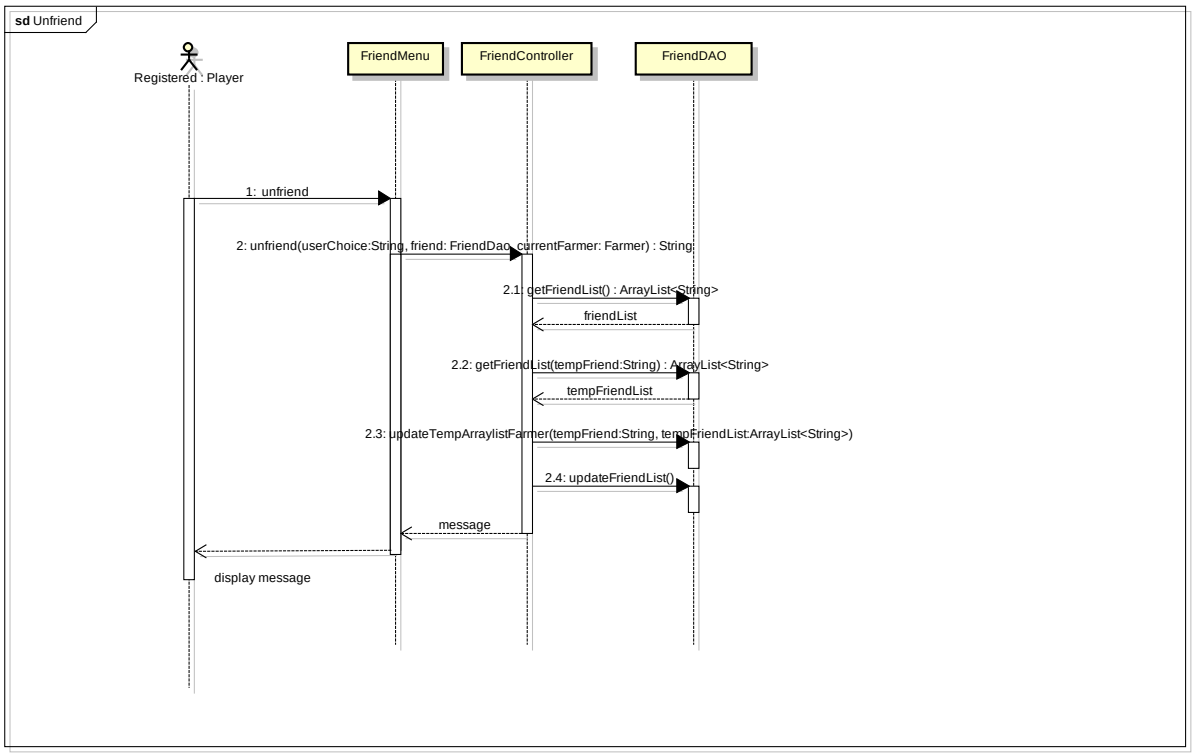
6.3 Logout

6.3.1 SD



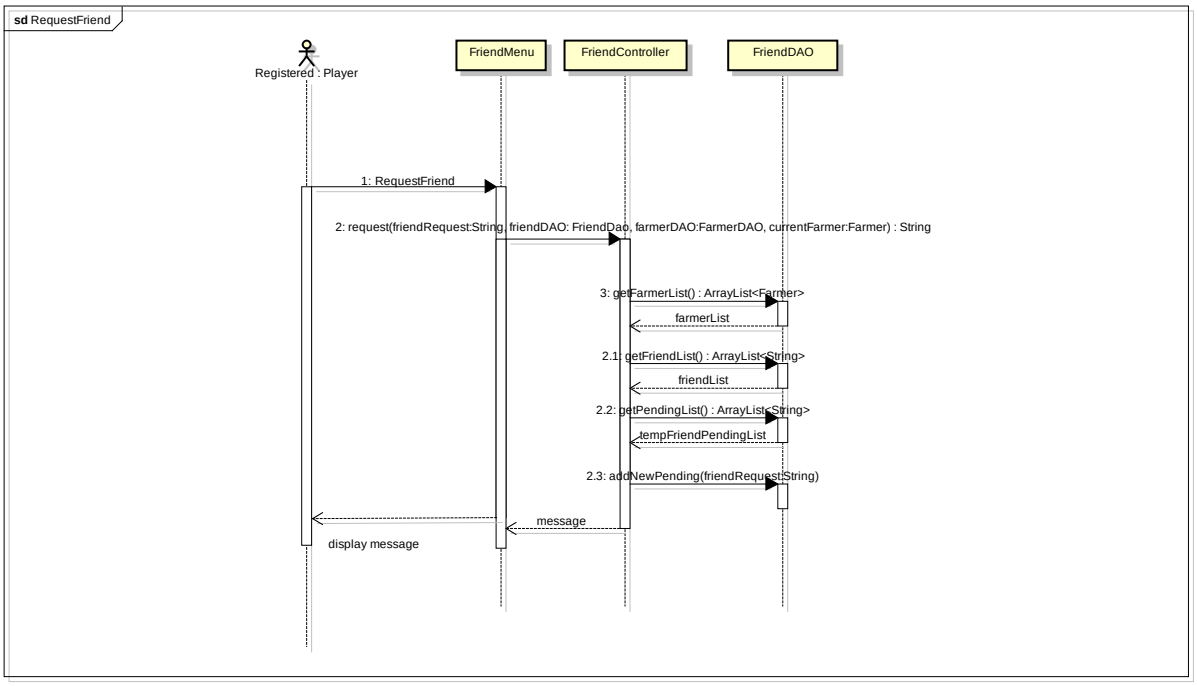
6.4 Unfriend

6.4.1 SD



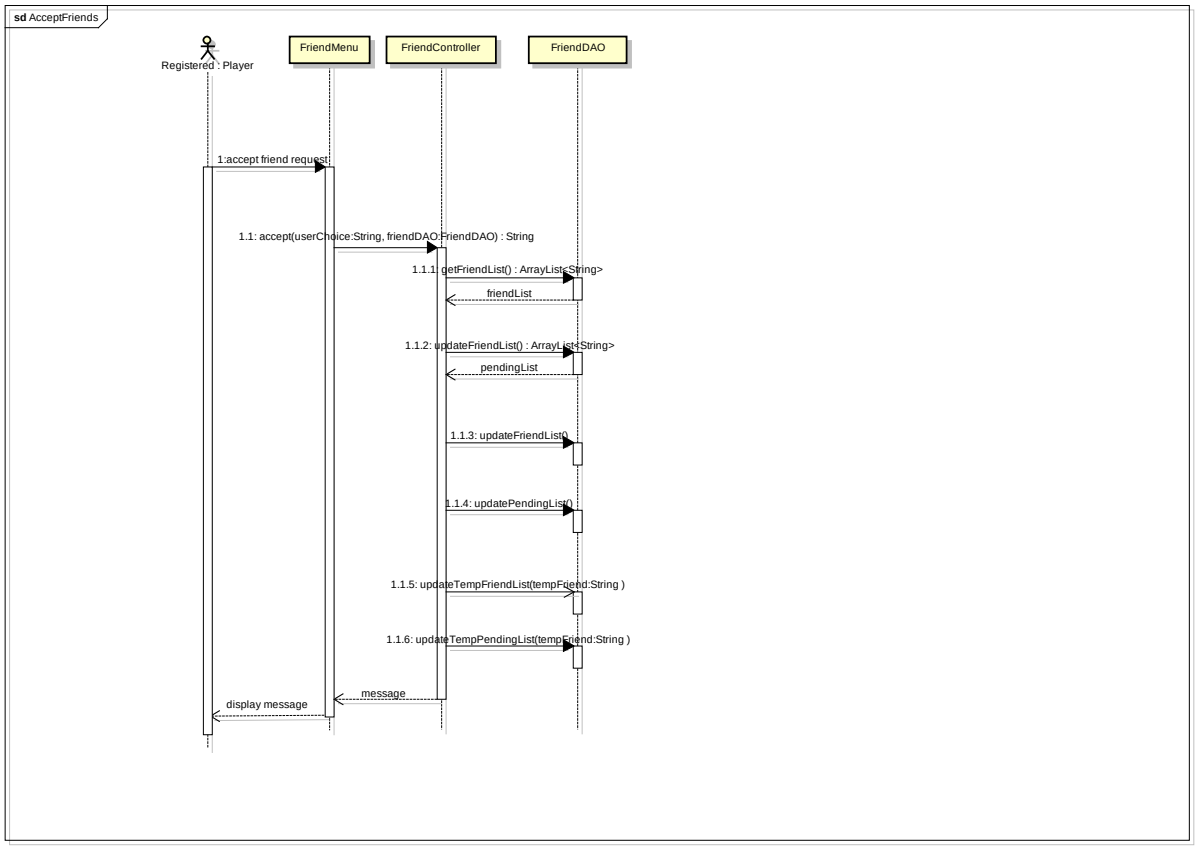
6.5 Request friend

6.5.1 SD



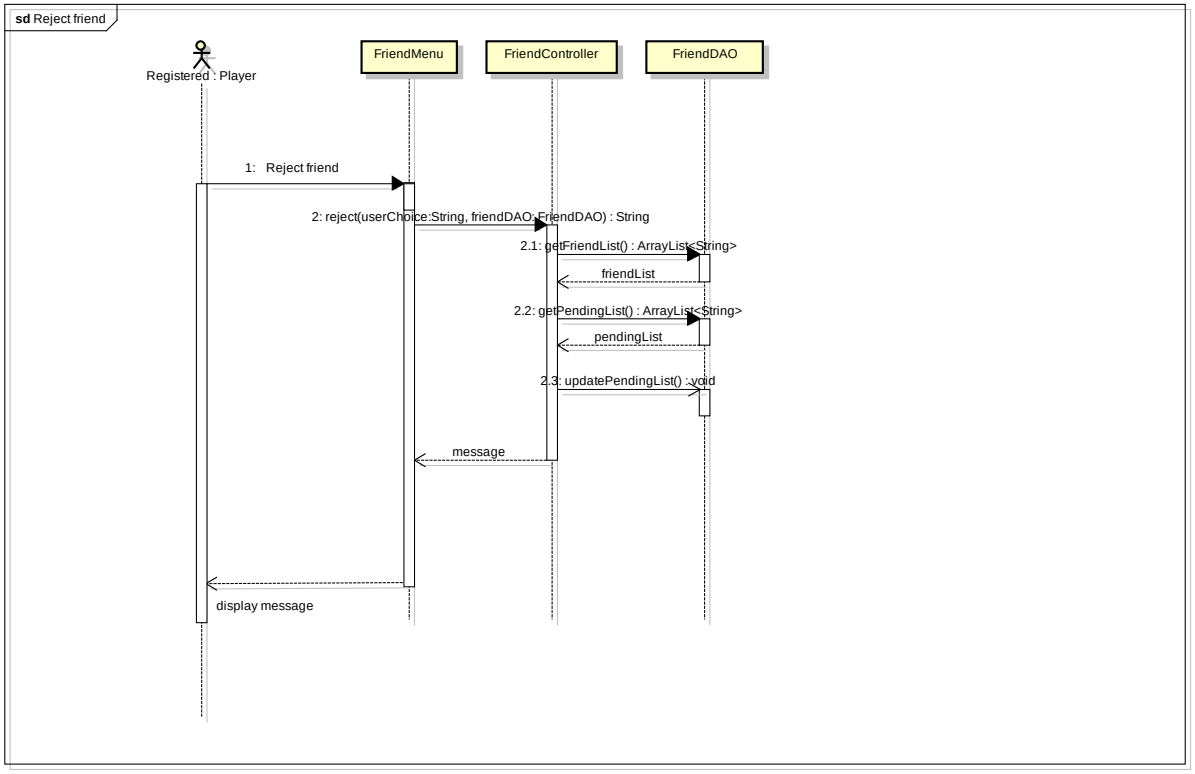
6.6 Accept request

6.6.1 SD



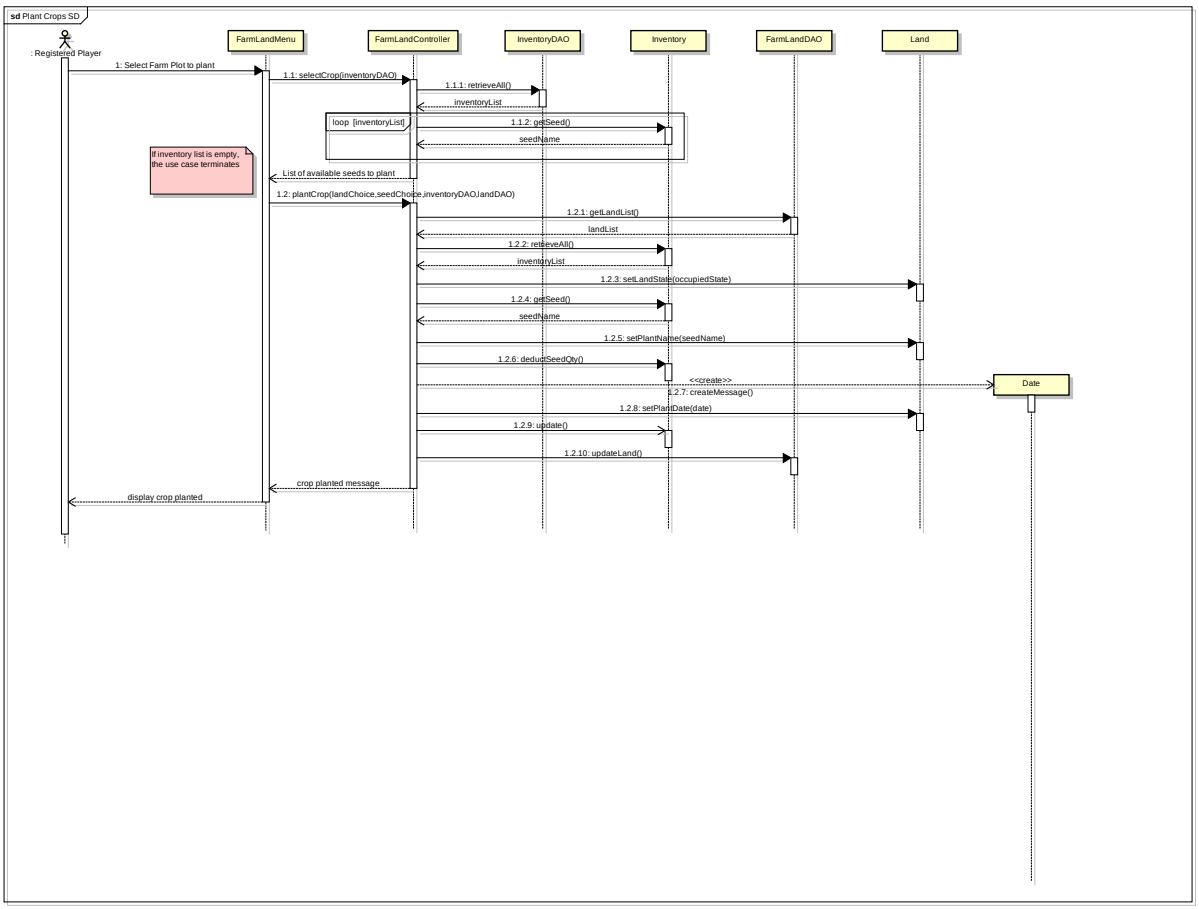
6.7 Reject request

6.7.1 SD



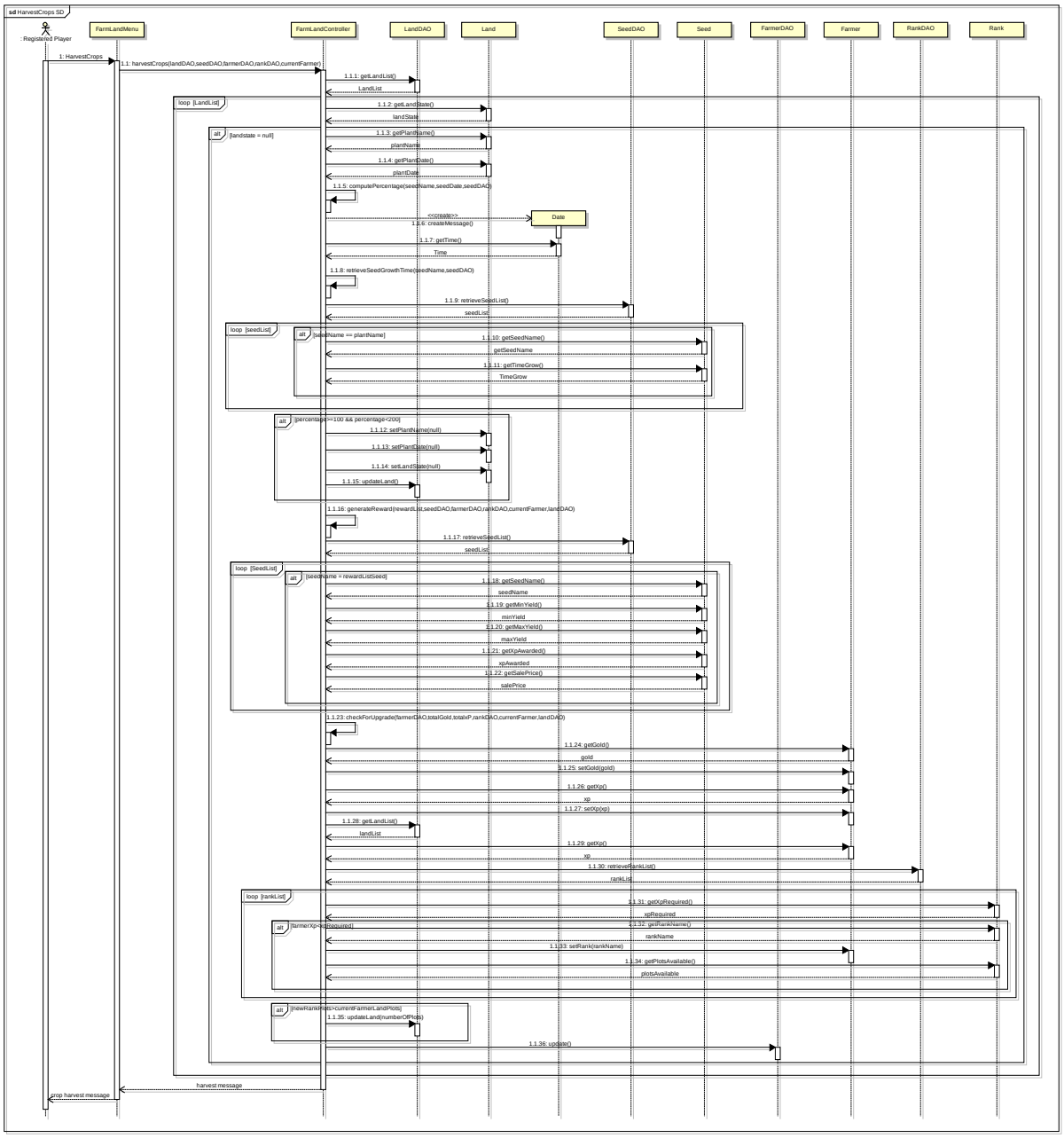
6.8 Plant crop

6.8.1 SD



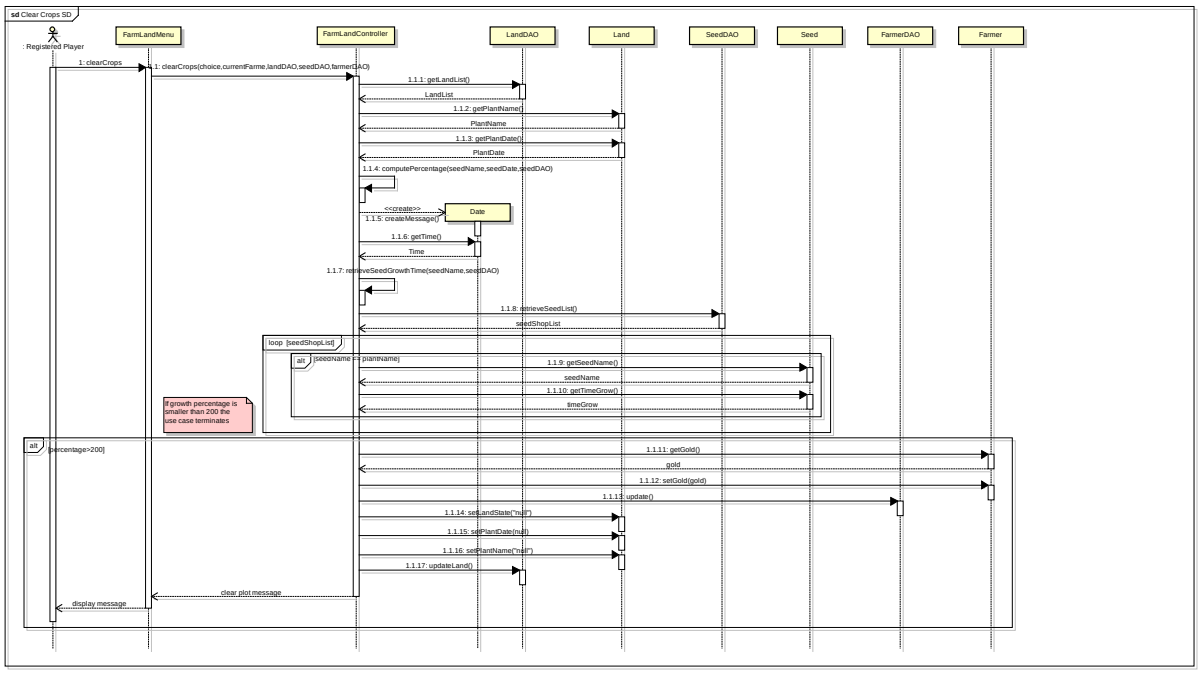
6.9 Harvest crop

6.9.1 SD



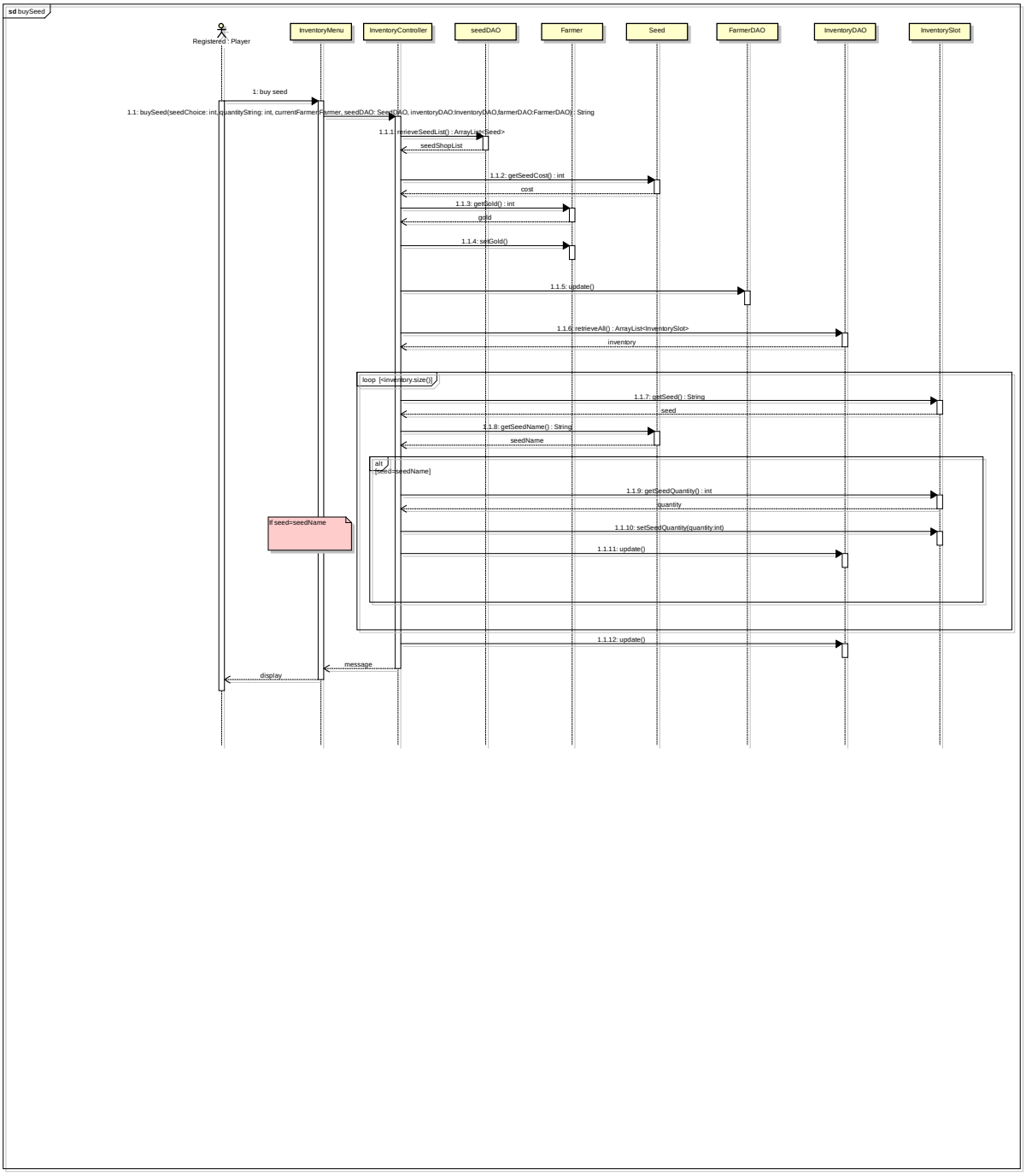
6.10 Clear crops

6.10.1 SD



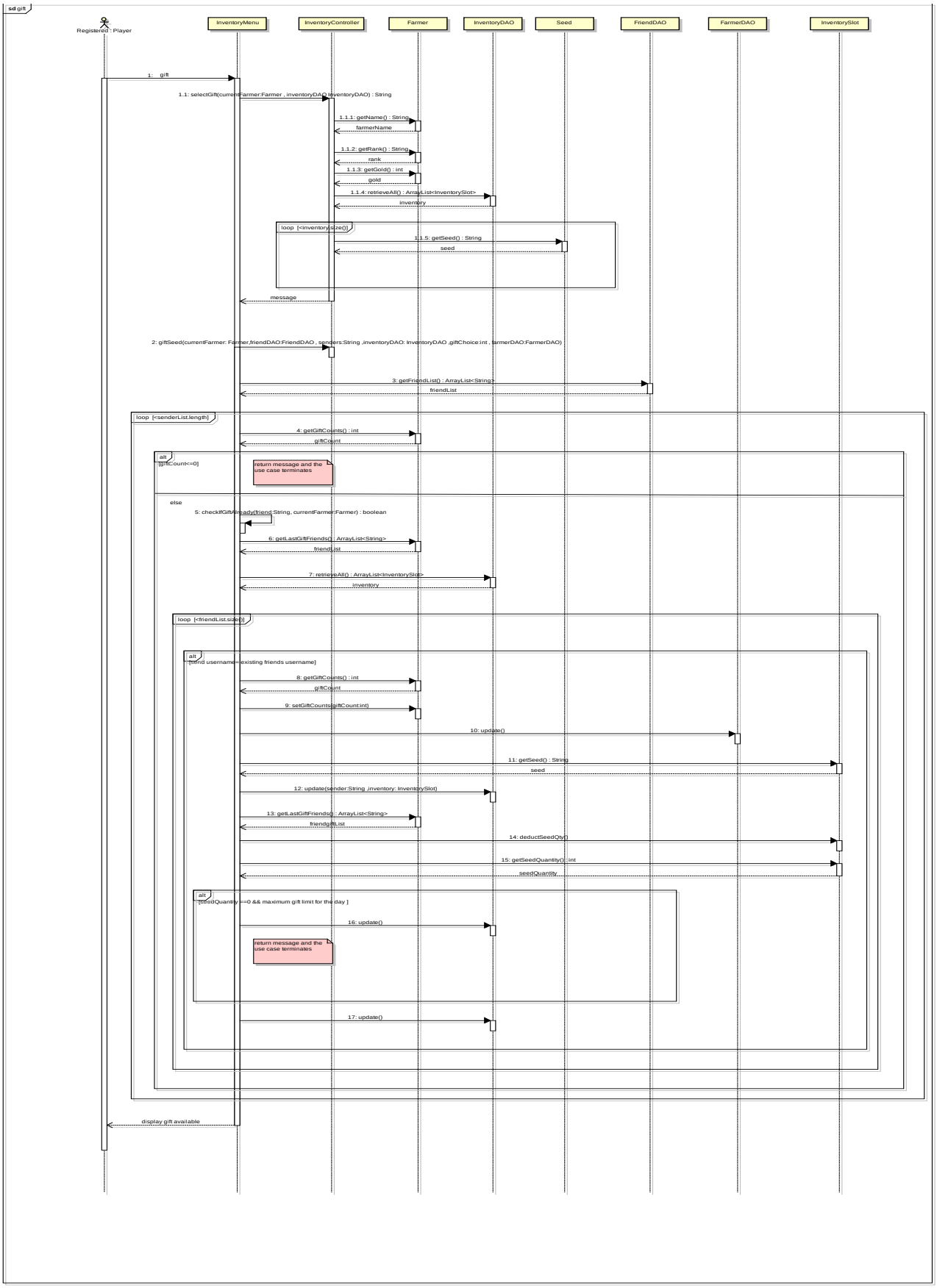
6.11 Buy seed

6.11.1 SD

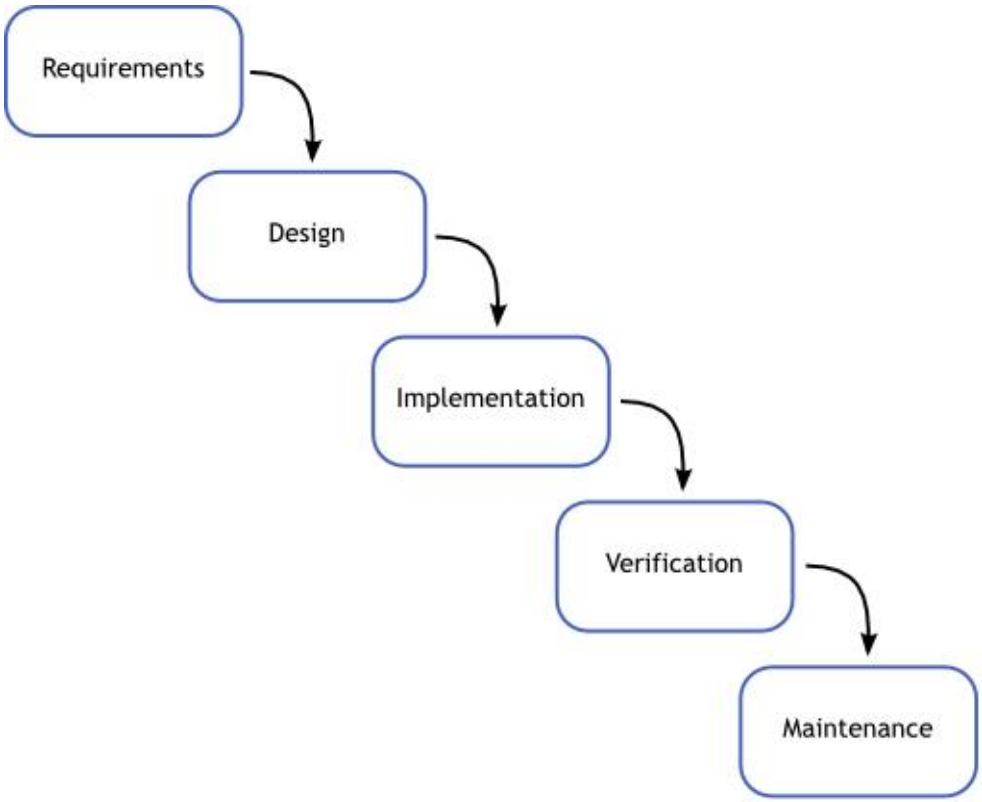


6.12 Gift seed

6.12.1 SD



7. OBJECT-ORIENTED DESIGN CONSIDERATIONS



Waterfall model

For our team's FarmVile Project, we took the Waterfall Model Approach because we wanted to make sure that we fully understood the prior conditions such as requirements and design before implementing code.

For the requirement phase, our group took 1 week to understand the requirements of the project before going on to the designing phase. During the design phase, our group proposed multiple designs for our application. For example, Hong Kun proposed storing all the farmer's land plot information into 1 file while Thao suggested using multiple files. In the end, we used Thao's design approach as we felt this approach was easier as it required less processing and it had lesser overheads when the project becomes big.

During implementation, different members had different ways of writing and storing the arraylist data from the DAO (Data Access Objects). When that happened, we moved back to the design phase and we all came together to standardize that DAO will only be initialized once during program start up and when farmer login.

8. MISCELLANEOUS

1.) LESSONS LEARNT

Project Management:

This is considerably one of the biggest application developed from scratch as a year1 SIS student. Therefore the workload may seem intimidating and overwhelming. Initially it is difficult to visualise and understand where to start and how exactly are we going to complete the project. However as time passes, we understood how to work together as a team and manage the project better. We followed a step by step process and adhered to datelines.

Work allocation

In terms of work allocation, we learned to dedicate and split the project equally among each team member. Every team member is involved in the holistic design of the entire application. We sat down to discuss how we should build the entire application. Thereafter we split the work by functionalities and came back together to integrate the entire application.

Time management

With various other projects and modules at hand, we had to learn to manage our time effectively. More importantly we had to ensure that we are on time and ready for project submission. Therefore we had to learn to divide the project and set completion datelines along the way for ourselves to ensure that we stay on track and be able to complete the application on time.

Team dynamics & morale

Initially before we started working on the project we barely knew each other and we all had different approaches to doing things. However after much time spend together, we learned to work together and complement each other's strength and weakness and actively participated in peer learning and helping. Along the way when the morale of the team was low, we also learned to encourage and support each other to ensure that the team progressed holistically and uniformly.

2.) CHALLENGES

Our team consist of people with varying expertise and background, therefore coming together and learning to work together can be a challenging task. Everyone has a different take on the matter at hand and have a different preference in the way we choose to execute and carry out the project. However after some time we put aside our differences and worked together to complement each other's strength and weaknesses.

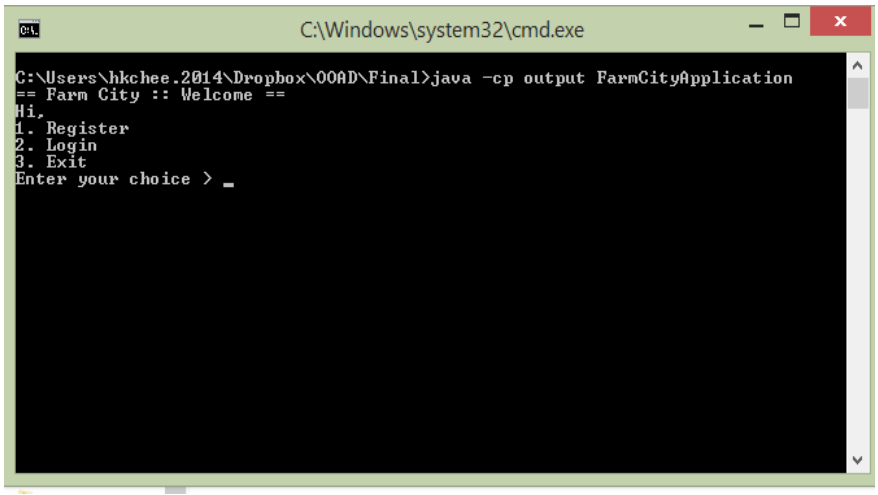
Overcoming challenges

To overcome this challenge faced, we complemented our strengths through peer learning and mutual support. Hong Kun who is more experience in coding would provide tips of advice on how we can debug our individually written codes. On the other hand, Jeremy who is familiar with Astah would help out with difficulties regarding Sequence Diagram (SD) and System Sequence Diagram (SSD) designs. Last but not least, Thao who is more comfortable in writing and design would provide additional support to team when writing API documents and system design. Thus complementing our individual strengths and overcoming our differences to learn and help each other as a team.

9. USABILITY AND SCREENSHOTS

9.1 Welcome Menu

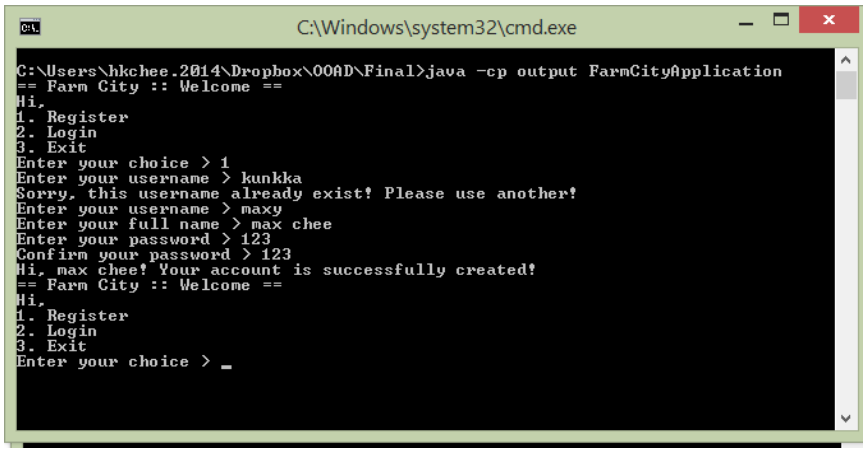
9.1.1 Screenshot



1. The application starts by showing the registered user the welcome page.

9.2 Register

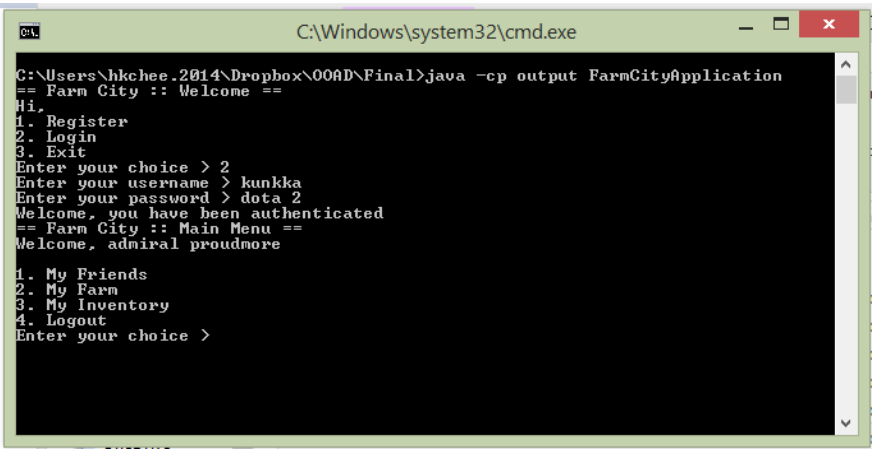
9.2.1 Screenshot



1. To register user has to select “1”.
2. Input a **unique** Username (Username should contain only alphanumeric characters).
3. On successful registration, the registered user gets 1000 gold coins. A new farmer is said to be a novice by rank with 0 XP.
4. Registration success message is displayed.
5. Welcome page will be re-displayed.

9.3 Login

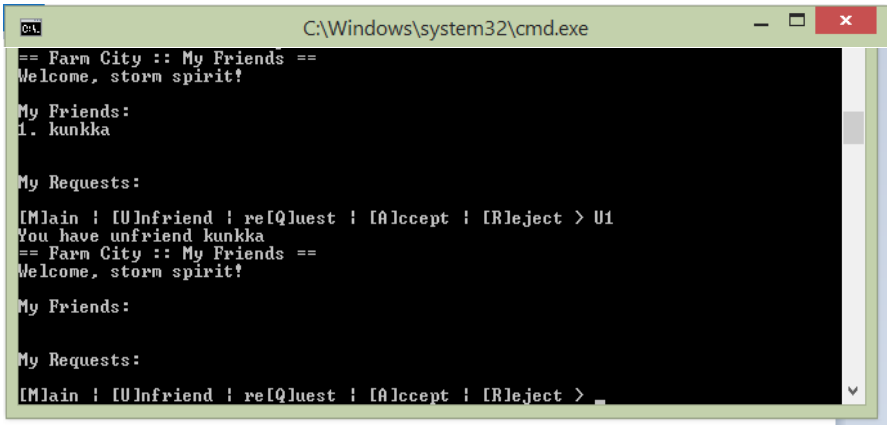
9.3.1 Screenshot



1. To Login user has to select “2”
2. Registered user to enter Username and password.
3. Upon validation success, welcome message is displayed.
4. Main menu will be displayed.

9.4 Unfriend

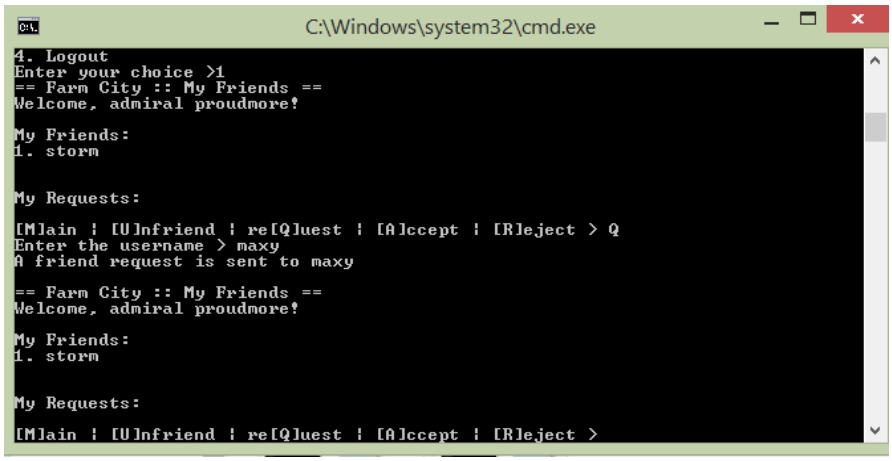
9.4.1 Screenshot



1. The registered user sees his/her friends in the list.
2. The registered user enters U<ID> to unfriend an existing friend.
3. The system displays an unfriend message.
4. The system then displays the "My Friends" menu.

9.5 Request friend

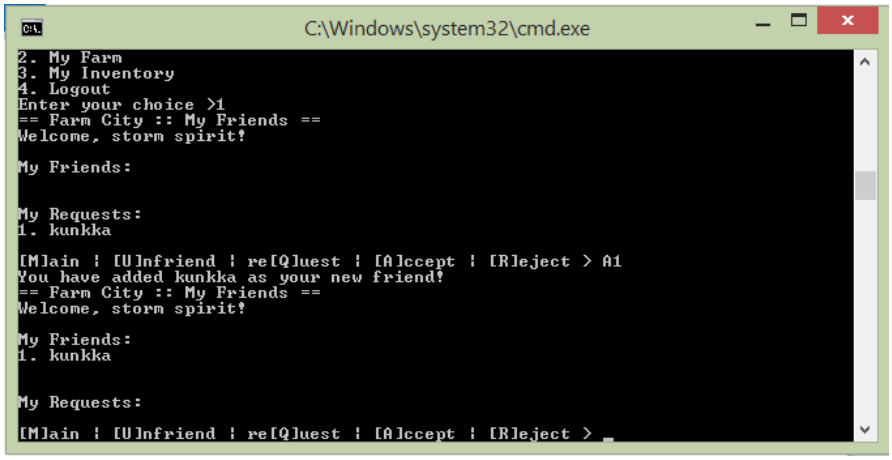
9.5.1 Screenshot



1. The registered user sees his/her friends in the list.
2. The registered user enters “Q” to request friend.
3. The registered user enters the username he wishes to send a friend request to.
4. The system displays a success friend request message.
5. The system then displays the "My Friends" menu.

9.6 Accept request

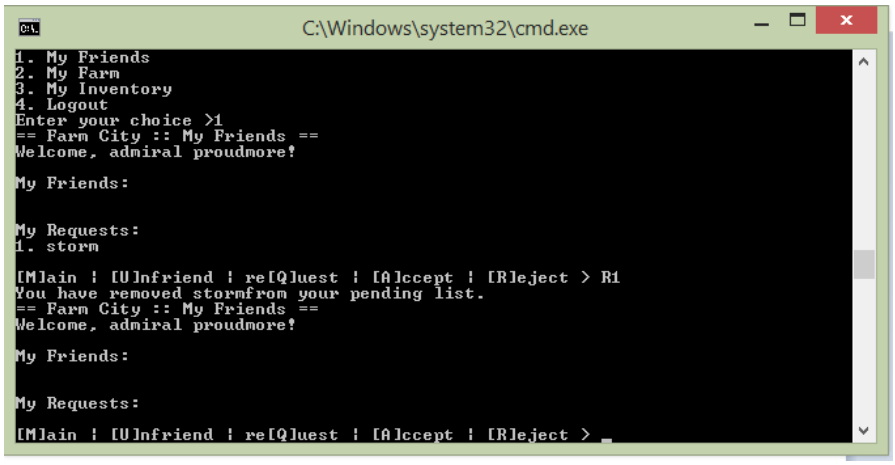
9.6.1 Screenshot



1. The registered user sees his/her friends in the list.
2. The registered user enters A<ID> to accept a Friend Request.
3. The system displays an added friend message.
4. The system then displays the "My Friends" menu.

9.7 Reject request

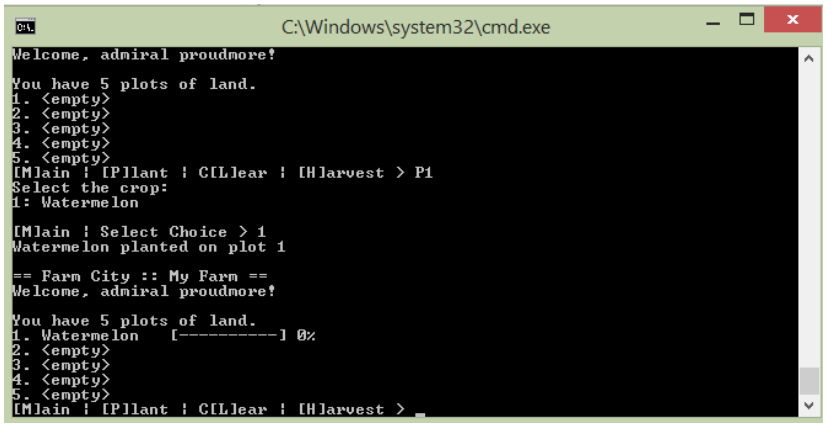
9.7.1 Screenshot



1. The registered user sees his/her friends in the list.
2. The registered user enters R<ID> to reject a Friend Request.
3. The system displays a reject request message.
4. The system then displays the "My Friends" menu.

9.8 Plant crop

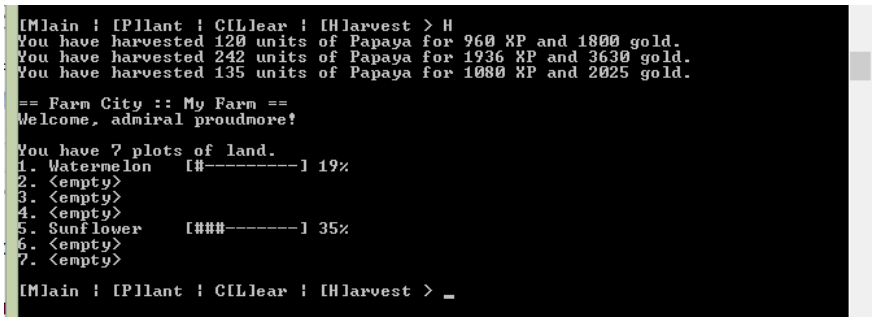
9.8.1 Screenshot



1. The player selects P<ID> to plant a crop.
2. The crops shown above for planting are the ones that the farmer has in his inventory.
3. The seed that the player choose will then be planted on the specific plot.

9.9 Harvest crop

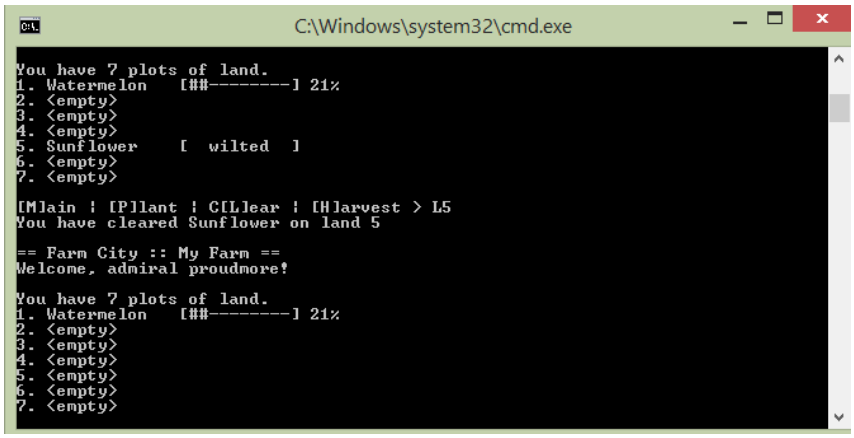
9.9.1 Screenshot



1. The player enters H to harvest. Crops ready for harvest (i.e. crops grown 100%) on any plot of land will be harvested.
2. When the player / farmer harvests, it is considered sold. And the gold corresponding to number of units produced Harvest crop

9.10 Clear crops

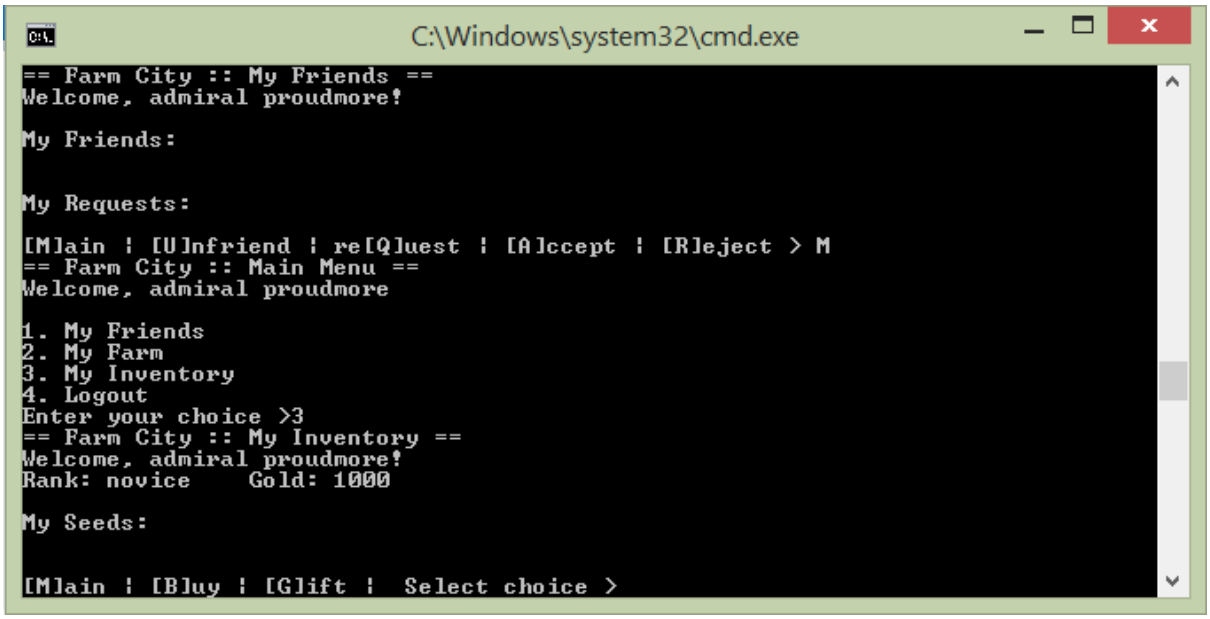
9.10.1 Screenshot



1. The player selects L<ID> to clear away wilted crop. Clearing the plot of wilted crop costs him 5 gold coins.
2. After performing any operation like harvest / plant / clear, the system displays the "My Farm" menu.

9.11 Inventory menu

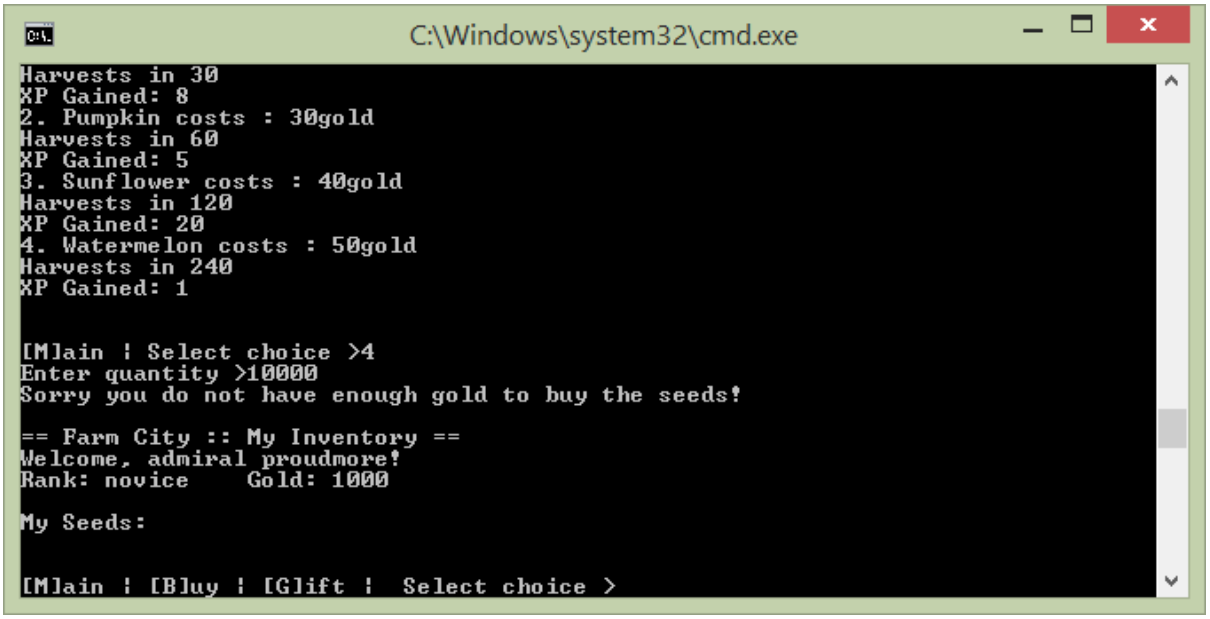
9.11.1 Screenshot

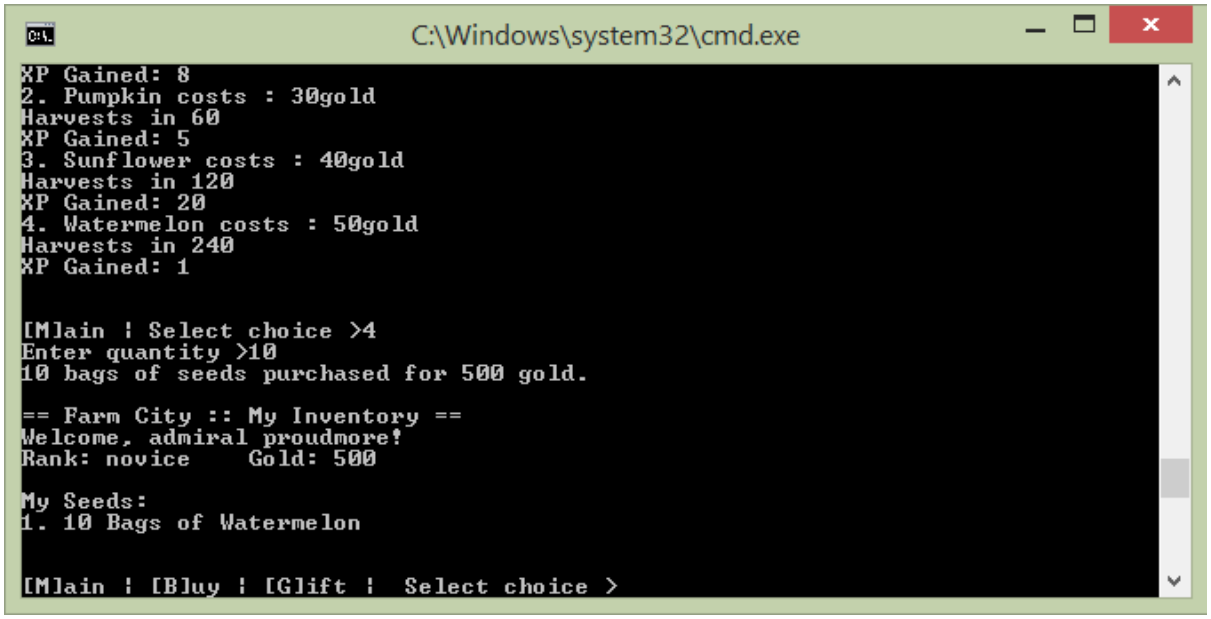


1. From the Main Menu, user enters <3> to see his/her Inventory Menu.
2. Inventory Menu will be displayed.
3. In his Inventory Menu, user will see his/her rank, amount of Gold and seeds in his Inventory.
4. User then has choice to return to Main Menu or Buy Seed or Send Gift.

9.12 Buy seed

9.12.1 Screenshot

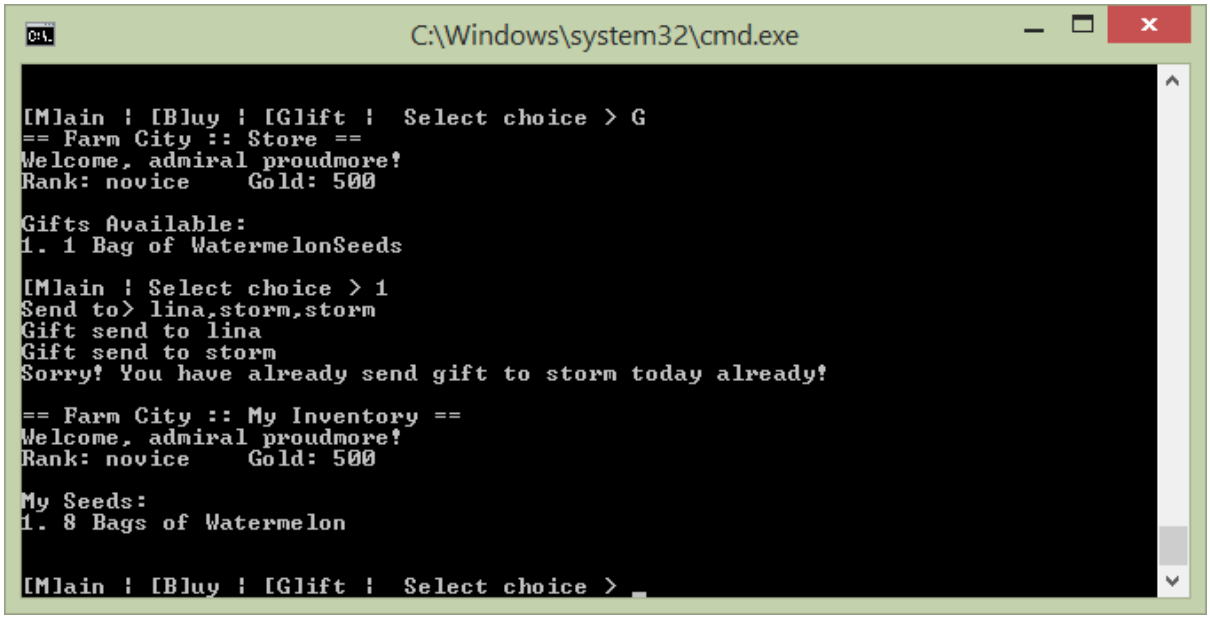




1. Upon seeing his/her Inventory Menu and seeds available inside his/her Inventory. The users enters <B> to buy seed.
2. Shop Menu will be displayed with seeds available to be bought.
3. User keys in his/her choice of seed and the quantity.
4. Invalid seed choice or shortage of gold to purchase for the seed from user will be rejected by the system and reject message will be sent out. Otherwise, user will receive a confirmation of success of buying seed.
5. The system then displays Inventory Menu.

9.13 Gift seed

9.13.1 Screenshot



1. Upon seeing his/her Inventory Menu and seeds available inside his/her Inventory. The user enters <G> to send gift.

2. Store Menu will be displayed with gifts available to be sent.
3. User keys in his/her friends' name whom he/she wants to send gift to.
4. The system will display reject message when his friend has received a gift within the day, invalid gift choice is keyed in and gift sent has reached its limit of 5 per day. Otherwise, the user will receive a confirmation of success of sending gift.
5. The system then displays Inventory Menu.