SAP Documentation

April 29, 2022

- Interviewed by Mr. Lee, discussed documentation and planning with DSDM approach
- Planned further for second sprint
    - Decide on specific functionality to include in later work periods
    - Include more detail about what to add specifically
- Made progress on UML diagram
    - Worked out all connections and split responsibilities for the night
- Figured out basic functions for each class while working on UML diagram
- Divided responsibilities for first commit
- Created repository for Super Auto Pets and added all group members as collaborators

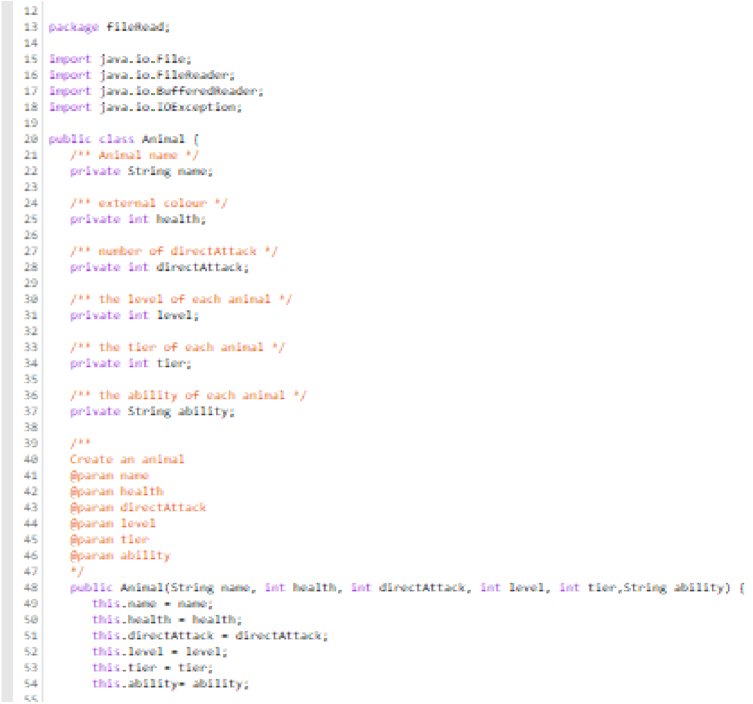
May 1, 2022

- Finished UML diagram
    - Some changes to be made the next work period
    - Background class in UML doesn't match class submitted
    - Most of methods should be in profile class, the background can't "do" anything it's just a blueprint for all backgrounds
    - Double check multiplicity with other group members
- Both background class and team class submitted for sprint 1, missing Kiarash's class

May 2, 2022

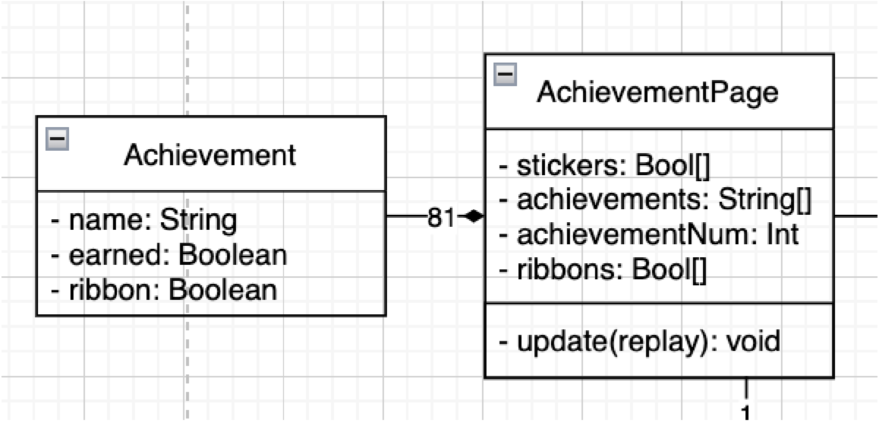
- Edited UML diagram to include achievements portion
- Assigned specific classes and functionality for each individual per sprint
    - Sprint 1:
        - ~~Team class (Christina)~~
        - ~~Hat class (Anthony)~~
        - ~~Background class (Shuhan)~~
        - ~~Animal class (Kiarash)~~
    - Sprint 2:
        - ~~Achievement class (Kiarash)~~
            - Retrieve data from animal class
            - Display list of achievements and ribbons for each
        - ~~History class (Anthony)~~
            - Display 10 most recent games
            - Display 10 most recent wins
        - ~~Shop class (Christina)~~

      - Spawn food and pet
      - Retrieve data from player class about turn number
      - Update coin value in player class after purchases made
    - ~~Battle (Shuhan)~~
      - Calculate damage dealt and ability effects
      - Return outcome of battle to player class
      - Determine if game over and send data to save class if true
    - ~~Game class (Anthony)~~
      - Connect with player class after most recent game has completed to collect key data
      - Store data for as long as game remains in most recent ten games in history
  - Sprint 3:
    - Purchase class (Christina) <- removed
    - ~~Profile class (Anthony)~~
    - AchievementPage (Anthony) <- removed, functionality shifted to profile
      - Update based on most recent game and whether player won with new tier three
      - Update list of achievements if profile earns one
    - ~~Player class (Shuhan)~~
    - Food class (Kiarash)
    - Save class (Shuhan)
- Artifacts
  - Preparing file reader for animal class, unfinished (Kiarash)



```java

package fileRead;

import java.io.File;
import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

public class Animal {
    /** Animal name */
    private String name;

    /** external colour */
    private int health;

    /** number of directAttack */
    private int directAttack;

    /** the level of each animal */
    private int level;

    /** the tier of each animal */
    private int tier;

    /** the ability of each animal */
    private String ability;

    /**
    Create an animal
    @param name
    @param health
    @param directAttack
    @param level
    @param tier
    @param ability
    */
    public Animal(String name, int health, int directAttack, int level, int tier,String ability) {
        this.name = name;
        this.health = health;
        this.directAttack = directAttack;
        this.level = level;
        this.tier = tier;
        this.ability= ability;

```

    - Made minor update to UML to include missing achievement class info (Anthony)



May 3, 2022

- Adjusted goals to take into account limited time
- Decided to focus efforts only on tier one pets to remove repetitive coding for effects
- All team members made progress with classes
    - Shuhan almost finished Battle class, effects not implemented yet
    - Anthony completed game class without certain variables because Player and Team class not finalized yet (missing playerOne, playerTwo, teamOne, and teamTwo)
    - Anthony to finish history class by end of day
    - Christina to finish shop class
- Functionality to prove on main method
    - Home page
        - History
        - Achievements page
        - User purchase a hat skin
        - Hats and backgrounds page
        - User buys new hat with coins and new background
    - Gameplay
        - Sample battle between two players where player wins
        - Saving game mid-game
        - Saving game after completing the History file
        - Update achievements page after level three animal wins
        - In shop class, buy pets and food with coins
- Team Discussion: https://drive.google.com/file/d/13FodNXaAaQu2EhvH9Ca_pV2levxi_BEO/view?usp=sharing

May 4, 2022

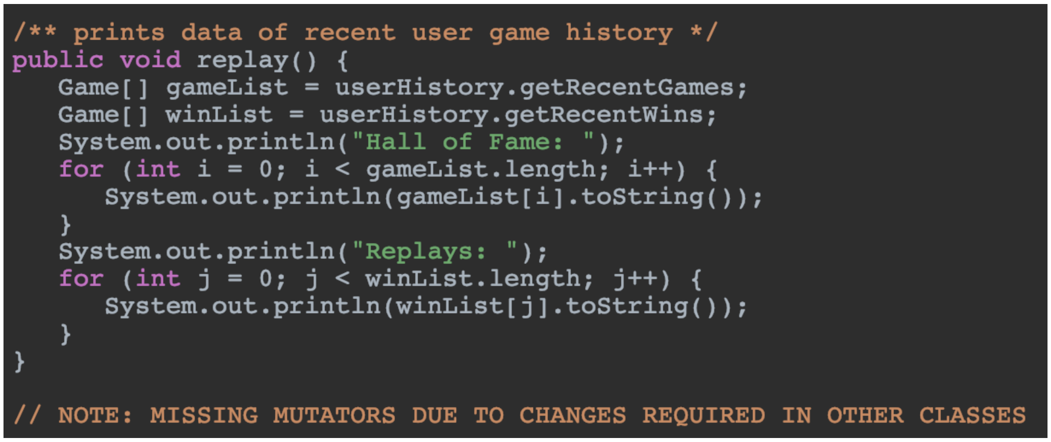
- Set new goals for sprint 3 that are more achievable
    - Purchase class (Christina)
        - Scan inputted user and password to check if information is correct
        - Check if card info entered matches saved card info
        - Update hat or background class once purchase has been made
    - Profile class (Anthony)
        - All essential functionality relating to accessing/changing user info and initiating the game
        - Returning data for recent games, player stats, current settings, achievements, and owned data
    - AchievementPage (Anthony)
        - Update based on most recent game and whether player won with new tier three
        - Update list of achievements if profile earns one
    - Player class (Shuhan)
        - Organize data for team and update changes to team made by the user
        - Connect shop class to team class via buy pet/food method
        - Pair user with random player to battle
        - Connect with battle class to calculate and return outcome of battle
        - Update data in save class when player has left app without completing game
        - Update data in history when game has been completed
    - Food class (Kiarash)
        - Return set of random foods at user's command
        - Return all foods and effects
    - Save class (Shuhan)
        - Store data from most recent game
        - Delete data when player has completed game for efficiency

May 8, 2022

- Meeting notes covers the meeting prior to third sprint deadline: https://docs.google.com/document/d/1TUyTxd8xMjEJdVvGF6L_i3msIoL5Us5hz-PAktgQQ7k/edit?usp=sharing

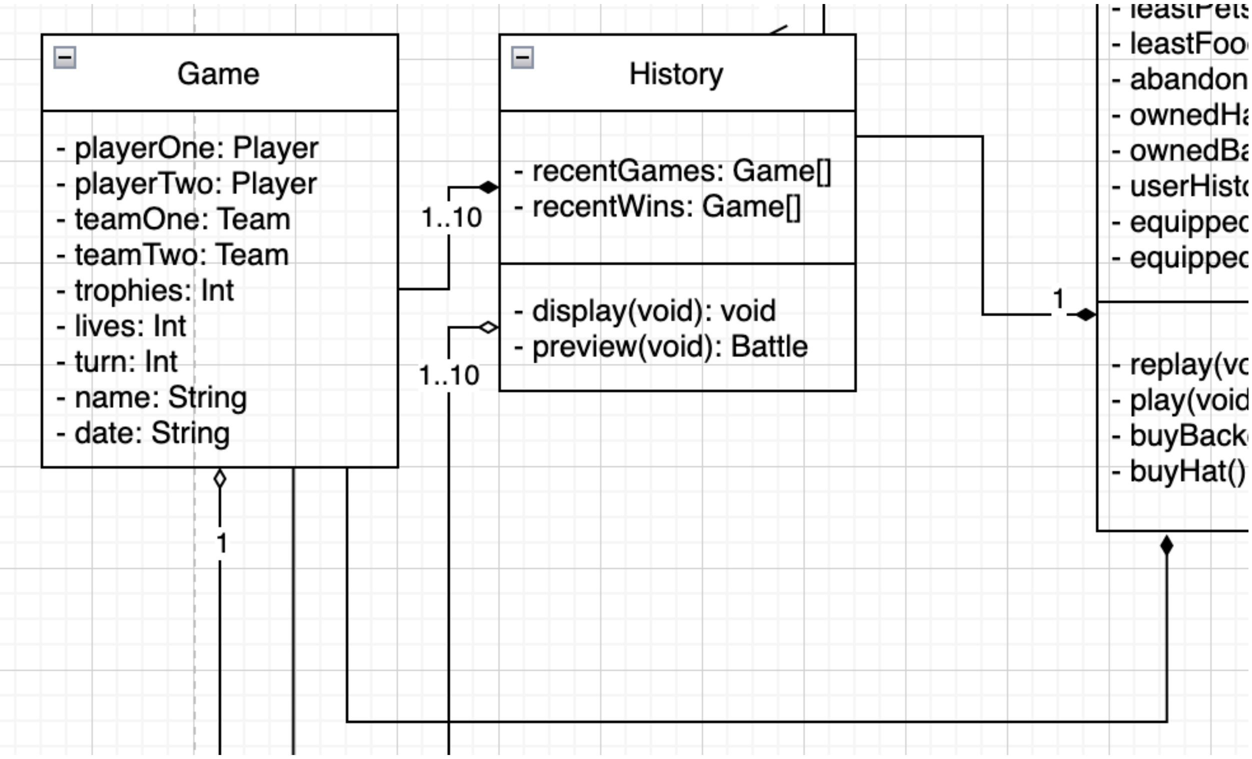
May 9, 2022

- Updated profile class to include majority of functions and accessors
- Screenshot below shows some accessors/mutators that couldn't be included, the update methods for food and animal instances
- Achievement page not yet complete, should be finished by end of Tuesday

- 

```
/** prints data of recent user game history */
public void replay() {
   Game[] gameList = userHistory.getRecentGames;
   Game[] winList = userHistory.getRecentWins;
   System.out.println("Hall of Fame: ");
   for (int i = 0; i < gameList.length; i++) {
      System.out.println(gameList[i].toString());
   }
   System.out.println("Replays: ");
   for (int j = 0; j < winList.length; j++) {
      System.out.println(winList[j].toString());
   }
}

// NOTE: MISSING MUTATORS DUE TO CHANGES REQUIRED IN OTHER CLASSES
```

May 10, 2022

- Christina, Shuhan, and Anthony finished basic functions of all assigned classes
- Achievement page class removed, deemed unnecessary
- Coordinated demonstrating overall functionality on one main java file settled on splitting this into two mains due to complexity of profile class
- Checked other team members' classes such as Hat class to maintain quality
    - Made significant changes to Hat class, added set methods to allow the user to select active hats
- Made some minor changes to profile class on UML diagram, changed history-profile relationship to composition
- 

May 11, 2022

- Completed profile class with the exception of play function and background access
- Finalized SAP_Testing class and updated hat class, history class stayed more or less the same
- Updated UML diagram to include latest additions to profile class
    - Chose to keep methods that haven't been completed yet in the UML to remind future self to finish
- NOTE: Demonstration will be split into two parts:
    - Part one will contain all the user info and show how it can be changed and accessed from various files
    - Part two will demonstrate the actual game and its mechanics

Profile.java Documentation

*Refer to notes above for an additional record of changes made to profile class.

May 6:

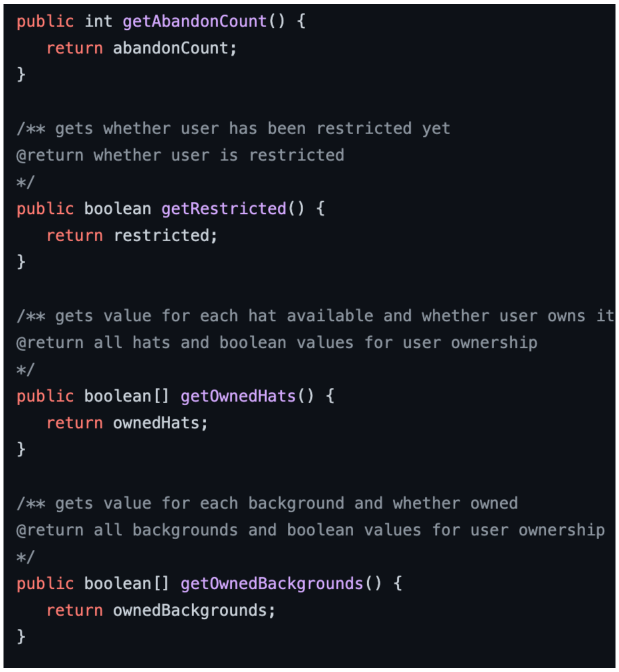
- Created Profile.java file, added basic attributes and get methods listed on the UML diagram
- Added placeholder methods to be replaced at later date when more is known about other classes

May 7:

- Ran into issues with replay() method due to history class missing certain variables

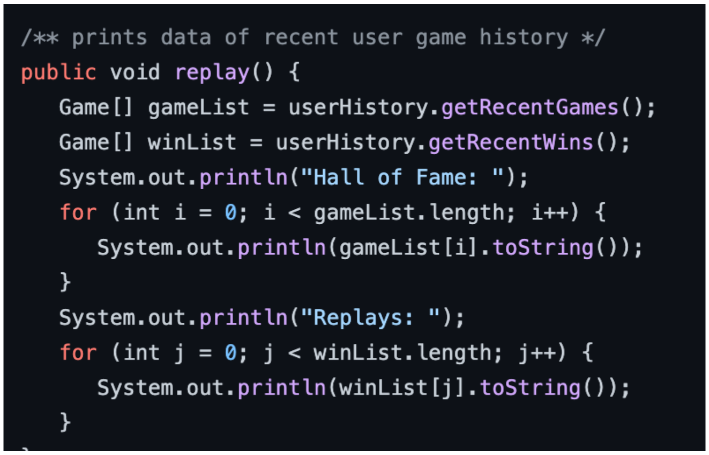
May 9:

- Added get methods and missing set methods, most of remaining methods complete with the exception of four
- Updated ownedHats and ownedBackgrounds variables to booleans rather than Hat[] and Background[]



```java
public int getAbandonCount() {
   return abandonCount;
}

/** gets whether user has been restricted yet
@return whether user is restricted
*/
public boolean getRestricted() {
   return restricted;
}

/** gets value for each hat available and whether user owns it
@return all hats and boolean values for user ownership
*/
public boolean[] getOwnedHats() {
   return ownedHats;
}

/** gets value for each background and whether owned
@return all backgrounds and boolean values for user ownership
*/
public boolean[] getOwnedBackgrounds() {
   return ownedBackgrounds;
}
```

- Completed replay() function

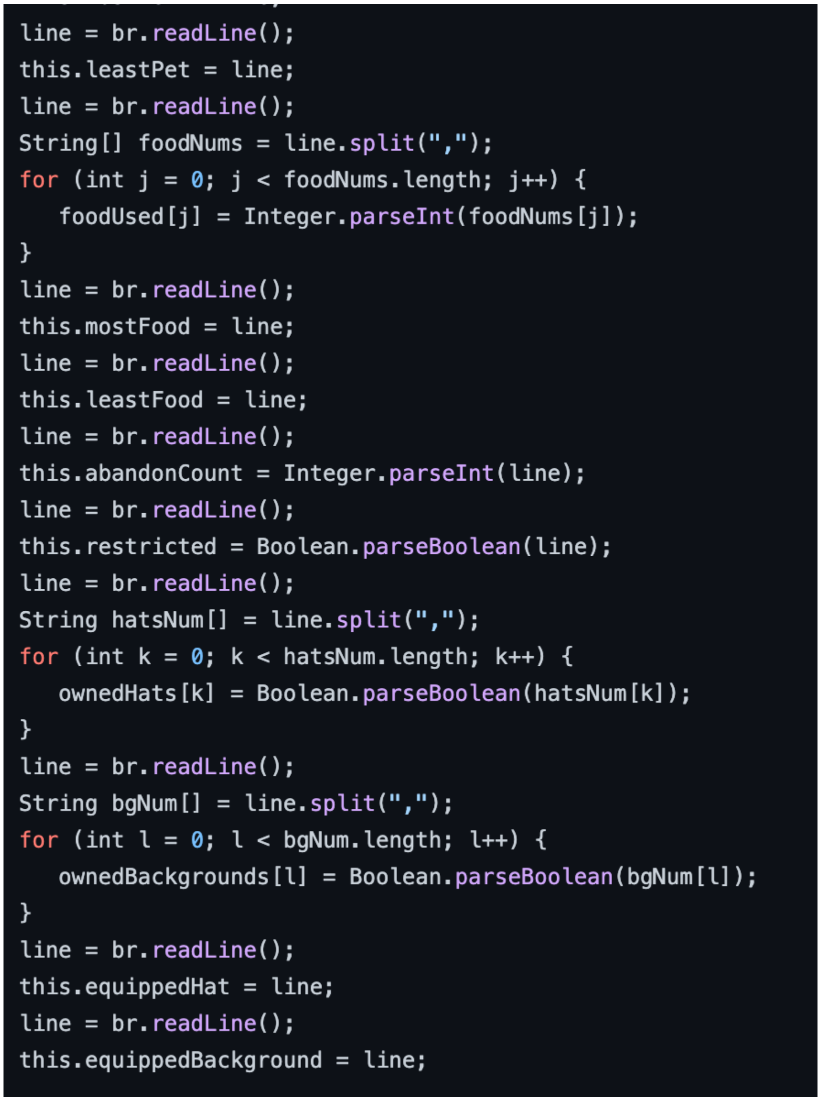


```java
/** prints data of recent user game history */
public void replay() {
   Game[] gameList = userHistory.getRecentGames();
   Game[] winList = userHistory.getRecentWins();
   System.out.println("Hall of Fame: ");
   for (int i = 0; i < gameList.length; i++) {
      System.out.println(gameList[i].toString());
   }
   System.out.println("Replays: ");
   for (int j = 0; j < winList.length; j++) {
      System.out.println(winList[j].toString());
   }
```

- Couldn’t get play function to work due to lack of communication about team/player classes

May 10:

- Added file reading function, profile should be able to access txt file with information on user now

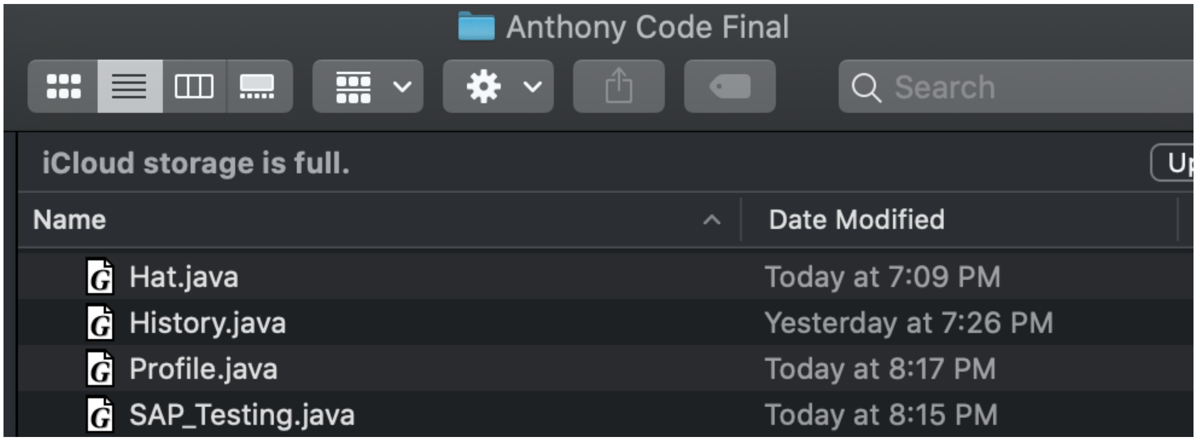
- 

```
line = br.readLine();
this.leastPet = line;
line = br.readLine();
String[] foodNums = line.split(",");
for (int j = 0; j < foodNums.length; j++) {
   foodUsed[j] = Integer.parseInt(foodNums[j]);
}
line = br.readLine();
this.mostFood = line;
line = br.readLine();
this.leastFood = line;
line = br.readLine();
this.abandonCount = Integer.parseInt(line);
line = br.readLine();
this.restricted = Boolean.parseBoolean(line);
line = br.readLine();
String hatsNum[] = line.split(",");
for (int k = 0; k < hatsNum.length; k++) {
   ownedHats[k] = Boolean.parseBoolean(hatsNum[k]);
}
line = br.readLine();
String bgNum[] = line.split(",");
for (int l = 0; l < bgNum.length; l++) {
   ownedBackgrounds[l] = Boolean.parseBoolean(bgNum[l]);
}
line = br.readLine();
this.equippedHat = line;
line = br.readLine();
this.equippedBackground = line;
```

- Instead of calling team class to generate a team name, shifted name generator to profile class
- Started on buy hat method, ran into numerous issues connected to the [username]_Hats.txt file
- Pushed submission by one day in order to resolve issues with accessing/mutating profile information (especially in relation to the hat file)
- Added equippedHat and equippedBackground variables to profile class, since these are attributes that the user should ideally have

May 11:

- Completed buyHat() method and did extensive testing to resolve minor issues
- Delayed play and achievementpage function until more is known about relationships between team/profile/player/achievement classes

- Added additional commenting to certain areas, not in all locations due to complexity of code
- Successfully connected profile class to SAP_Testing.java file and tested various values to maximize quality
- Updated UML to include new changes to profile class
- Removed .info txt file because it was deemed unnecessary for the purposes of this assignment, no need to save user info to a text file if the program would restart anyways once the testing file finishes running
- Submitted Profile.java file with the most up-to-date changes and as much functionality as is possible with current group situation
- 

iCloud storage is full.

Name	Date Modified
Hat.java	Today at 7:09 PM
History.java	Yesterday at 7:26 PM
Profile.java	Today at 8:17 PM
SAP_Testing.java	Today at 8:15 PM