**Project Init**

**Group 5**

Donovan Colton

Faisal Almansour

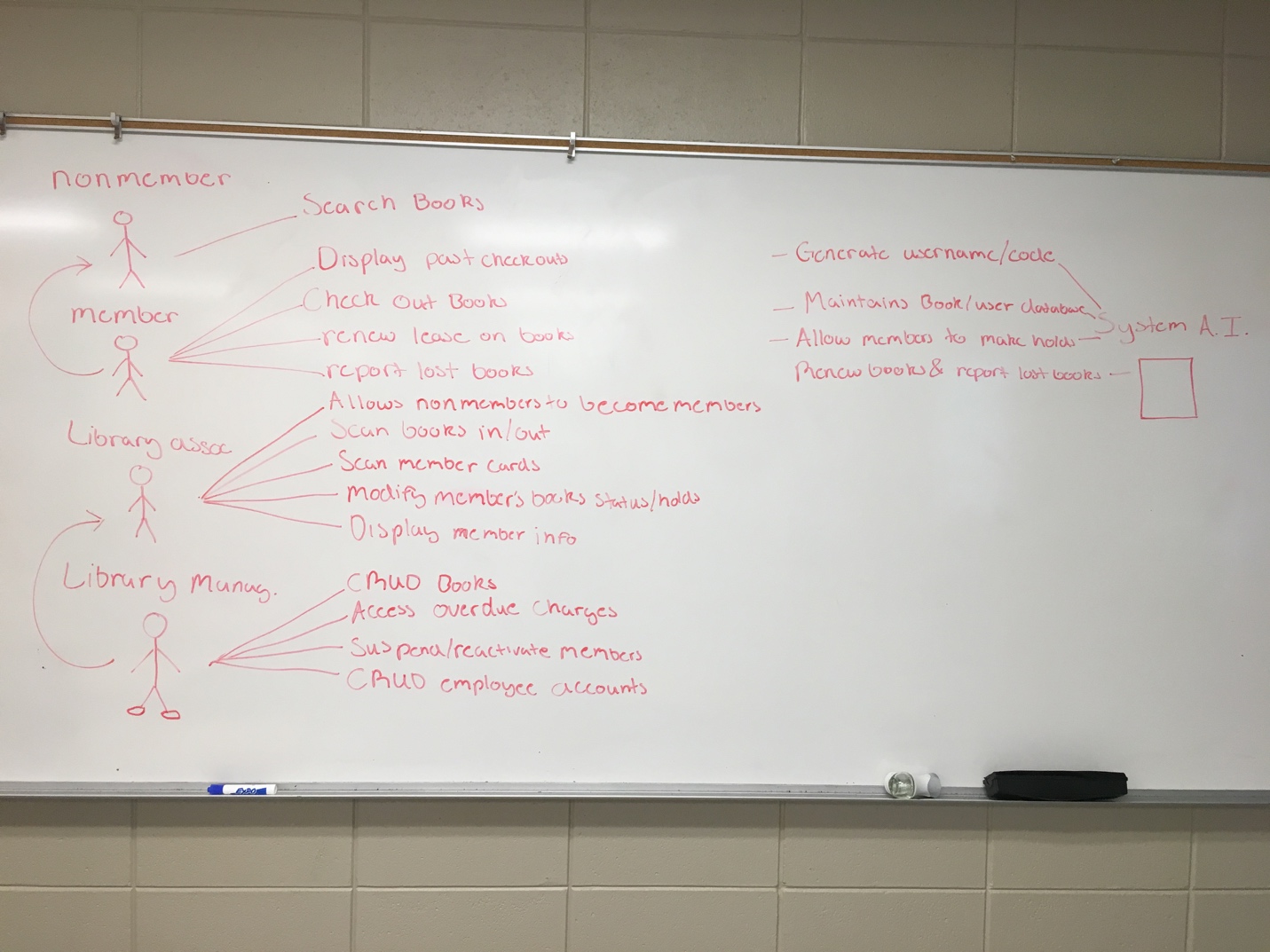
Chris Rushing

Cameron Kirby

**1.Actors**

1. Nonmembers
   1. Nonmembers can only use the system to search for books. To check out books, they must create an account (became a member) by supplying name, address phone and username.
2. Members
   1. Members must approach library associates to check out books. A member can have a **maximum of 10 books checked out**.
   2. Can make holds on books or renew books, either available or checked out, through the system or by contacting a library associate.
      1. If hold is place on a book, the member has 4 days to check it out once it becomes available
3. Library Associate
   1. Check out books to members (scan IBSN and 4-digit code)
   2. Makes returned books available again by retrieving them from the drop-in box, unless there is a hold.
   3. Display member info
4. Library Manager
   1. Can do all an associate can do and more
   2. Control book inventory
   3. Assess overdue charges
   4. Suspend/reactivate users
   5. Create/edit library associate accounts
5. System AI
   1. Generates username and 4-digit code
   2. Record data
      1. Book checkout date and time
      2. Overdue fees
      3. Book information
         1. Status –available or checked out
         2. Title, author, IBSN, year, keywords
      4. User info
         1. Book history
         2. Due dates
         3. Fine history
         4. Holds
   3. Allow members to make holds/renew books/ report books lost.
      1. Once book is reported lost, member is charged an replacement fee, and not charged late fees (must be paid in full)

**2. Use Case Diagram**



**3. System Requirements**

**US1 –  Nonmembers (search books)**

**Description**

allow anyone to search the library’s db for books

**Functional/System Requirements**

implement a search of db by different attributes of books

**Non-Functional Requirements**

**Notes**

**US2 – Members (display information)**

**Description**

Members need to be able to display all of there account information (fines, past/current checkouts, etc). This can be done through the library’s system or by contacting a associate.

**Functional/System Requirements**

A mechanism to query the db and display required information

**Non-Functional Requirements**

User dashboard to request to display these thing

**Notes**

**US3 –Members(checkout/renew/hold)**

**Description**

Allow users, with the library card (4 digit code), to checkout renew and hold

**Functional/System Requirements**

Recognize unique library code

Check whether the member currently has 10 books checked out or a suspended account (lost book)

**Non-Functional Requirements**

**Notes**

**US4 –Library Associate(scan books out to users/ check books back in)**

**Description**

Given 4 digit code and IBSN an associate can check out a book to a valid member

Given a ibsn an associate can scan a book back into the library

**Functional/System Requirements**

Recognize unique codes from users

Recognize isbn in db

**Non-Functional Requirements**

Once a book is checked back in the system must verify there is no holds, if there is the member with the hold has for days to pick up. There can be many holds.

**Notes**

**US5 –Assoc (display member info, modify members status/holds, sign up nonmembers)**

**Description**

Associate can pull up any member's information by request and also create new members

**Functional/System Requirements**

Associate verification process

A db of all users and the name, address, phone, four digit code. also includes their library information such as holds, status,

**Non-Functional Requirements**

**Notes**

**4. User Story Prioritization**

|  |  |  |  |
| --- | --- | --- | --- |
| US ID | User Story Title | Prioritization | US points |
| 1 | Users - Search Book DB | 8 | 1 |
| 2 | Members - Display info (checkouts, fines, etc) | 6 | 3 |
| 3 | Members - Checkout/Renew | 7 | 3 |
| 4 | Assoc. - Scan books in/out | 5 | 3 |
| 5 | Assoc. - Sign Up members/ modify status | 3 | 2 |
| 6 | Manag. - CRUD inventory | 4 | 5 |
| 7 | Manag. - access charges, suspend/reactivate users | 9 | 5 |
| 8 | Manag. - CRUD assoc. accounts | 1 | 2 |
| 9 | System AI. - generate usrnm/code,maintain DB | 2 | 8 |
|  |  |  |  |
|  | **CRUD - Create,Retrieve, Update,Destroy** | |  |

Interestingly enough Dr. Gibson, we decided to approach our prioritization technique based on the number of database transactions a particular case would use. Simply, the more database access a case requires determines it’s priority in ascending order. A lot of database retrieval/updating is costly so this is why higher priorities are at the bottom. The lower priorities are simple features that will be implemented first.

Our 5th user story was given 2 points, why? because it will take less time and has a lower complexity in the actual API we’re constructing. The point acquisition algorithm, similar to the above, aggregates complexity and frequency. For instance, our System that handles a lot of request from different users will be more complex in time comparable with a single use story branch so to speak.