50.001 Introduction to Information Systems and Programming
1D Group 1-3

Got-Anot?

Victoria Yong 1004455
Ngui Jia Xuan, Sheriann 1004641
Ivan Tandyajaya 1004572
Cai Xuemeng 1004522
Wang Chenyu 1004515

Background

Problem Statement

As students we find ourselves sourcing for resources for our various projects. SUTD provides a host of resources for us to use.

Current process of sourcing: **Physically** go to the various departments to check for availability of these resources, all while **complying to their rules** like opening timings, staff approval, or dress codes. Furthermore, we sometimes find that they may be **out of stock**.

Proposed process of sourcing: By **digitising** the process, we make it more convenient for both staff and students to **update the inventory** and streamline the sourcing process for **ease of use**.

Application Overview

Got-Anot? Is an inventory management and shopping application for resources in SUTD. It allows students to quickly search for where to get the items they need in school, if provided, and book the resources to ensure they are able to get a hold of the limited stock. The application also allows staff to easily check the remaining stock of all items, and update the inventory when they have replenished stock.

System Design and Implementation

Requirement Checklist

	Student	Staff
Login / Logout	✓	✓
Find Resource - Narrow Search (by Item Name) - Wide Search (by first letter of query) - Search by Location	✓	✓
Check Inventory of Location and Item Stock	✓	✓
Book resource for collection in a certain lab - Some items not bookable by Student/Staff	✓	✓
Confirm item and quantity taken after collection		✓
Verify and update the stock		✓
Add new item		✓

System Architecture

Front-end: Android Studio, Java

Back-end: Firebase, Java API

Back-end API

Classes

ResourceSet

Resource
Nested class in ResourceSet. Object representing an item in the inventory.

To add a resource, simply call *resourceSet.all_can_access_no_need_return_item(args)* and it will be added into the ResourceSet.

Location
Object representing the different locations resources can be found in SUTD.

User
Instance of a user account in Got-Anot?

Staff
Subclass of User. Describes a user with the Staff role and permissions.
Assigns User accounts of this type the "Staff" role.

Student
Subclass of User. Describes a user with the Student role and permissions.
Assigns User accounts of this type the "Student" role.

BookingEvent
Instance of a booking request issued by a User account.

Functions

Function	Parameters	Return Type	Description
accurateSearch	(String query)	int[]	Queries with user-inputted String.
accurateSearch	(String query, String location)	int[]	Same as above but with an extra location filter.
wideSearch	(String query)	int[]	Only implemented if accurateSearch returns NIL. Splits string by space and searches by every char inside the query. Returns resource ID.
wideSearch	(String query, String location)	int[]	Same as above but with an extra location filter
bookingConfirmationHandler	(User student, User staff, Resource resource, int quantity, String timing, Location location)	Boolean	Requires: (resource.amt < required_amt) If a booking request fulfils the requirement, creates a new bookingEvent corresponding to input timing, resource.name and required_amt and returns True. Otherwise, returns False.
login	(int userID, String password)	User	If both userID and password match the database, grab the User object and return it.
confirmCollection	(int id)	void	Updates the bookingEvent status in the database.

updateStock	(Resource resource, int quantity, User user)	void	Updates the stock in the database and returns a notification to the user (staff account only).
getHistoryBySearchingName	(String resource_name)	User[]	Returns a list of users who have booked and collected this resource before.
getHistoryBySearchingId	(int userId)	Resource[]	Returns a list of resources taken by the specific user.
addNewResource	()	void	Adds a new resource to the database

Design

Inheritance

- Student and Staff extend from User class to further specify their roles (Fig. 14 and Fig. 15)
- Easy checking of permissions
- Reduce redundancy: still reuse User attributes and methods

Nested Class

- Outer class ResourceSet with Resource inner class
- More logical grouping of attributes and methods for the respective classes
- Increases encapsulation

Strategy Design Pattern: RecyclerView

- Used in Search (Fig. 5) and Order History (Fig. 8) pages.
- We can set the adapter and layout manager to Recycler view instance

Adapter Design Pattern: RecyclerView Adapter

- Used in Search (Fig. 5) and Order History (Fig. 8) pages.
- We used the adapter to adapt the data originally stored in Firebase to our recycler view.

Global Instance Design Pattern, Singleton and Shared Preferences: Login

- The login system uses the Singleton design pattern and only allows one instance of User to be stored in the LoginInfo at a time. LoginInfo is made global so account information can be accessed by all parts of the application.

- LoginInfo stores 3 types of data:
    1. A User instance fetched from firebase upon login.
    2. An ArrayList of booking events for that user.
    3. A ResourceSet.Resource instance called selectedRecource, indicating which item is currently being selected by the user.

- Login data can be stored as SharedPreferences for future logins.

Fragments

- The main activity that controls the navigation of fragments is the Navigation Drawer activity. The activity swaps out its main view with various fragments listed in the navigation graph (*mobile_navigation.xm*l) upon various button clicks.
- Pages which extends Fragment: Home Page, Profile Page, Search Page, Order History, Add Resource Page (Refer to Appendix A)
- Allows for ease of navigation between pages
- Makes code modular and allows reuse of elements across multiple pages

Task Logs

Victoria: UI/UX design, Poster, Report
Ivan: UX design, Front-end/Back-end Integration
Sheriann: Back-end API, Firebase, Demo Video and Editing, Report
Xuemeng: Back-end API, Firebase, Front-end/Back-end Integration
Chenyu: Back-end API, Firebase, Front-end/Back-end Integration, Demo Video

Future Works

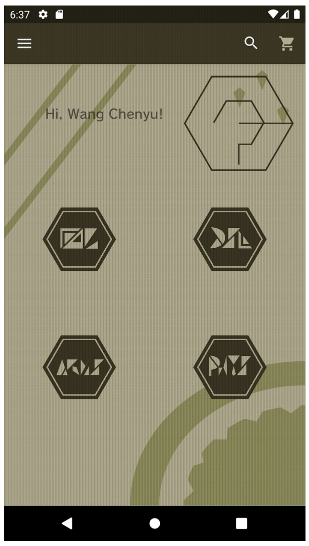
Given more time, we would also like to implement the following ideas to make our application more robust and provide greater ease of use.

1. Allow users to find the item listing from their Order History page and re-order the item again if needed
2. Shopping list function where users can create shopping lists for each of their various projects and save items to the list, which can be accessed through their profile page.
3. Complimentary web application integration
4. Feedback Page/Chat-bot/Chat function for users to inquire about items
5. Chat between user and location staff to inquire and further discuss timing if necessary
6. Implement Search History in the search bar
7. Data analytics of items (Which items are most popular, average rate of stock depletion etc.) which could predict and alert when staffs need to start restocking
8. Implement Search Suggestion in the search bar
9. Add search filters
10. Add search result sorting (By most relevant, by most popular, Alphabetical order etc.)
11. Store/retrieve images from database (for users to set profile photo, staff to set item photo and item listing pages to retrieve item photo when displaying a listing)

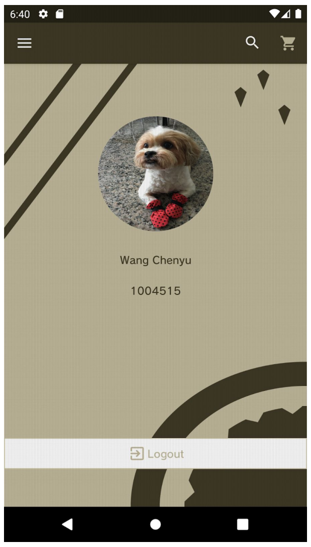
Summary

Got-Anot? is an android inventory management application that provides greater convenience to both staff and students, allowing users to quickly find items they need and book them for collection before stock runs out. This application streamlines the process of sourcing for resources for projects, allowing students to focus more on the building and less on the sourcing. The app also provides a unified platform for the lab technicians and staff to update their inventory, keep track of what they have in stock and monitor where the items are going, allowing them to be more informed of the use of resources by students.

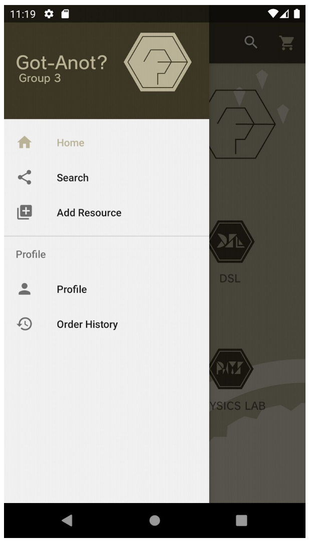
Appendix A: Screenshots



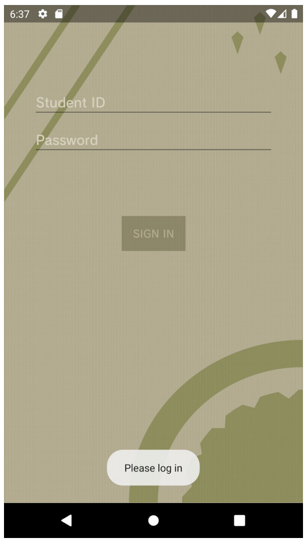
(Fig. 1: Home)



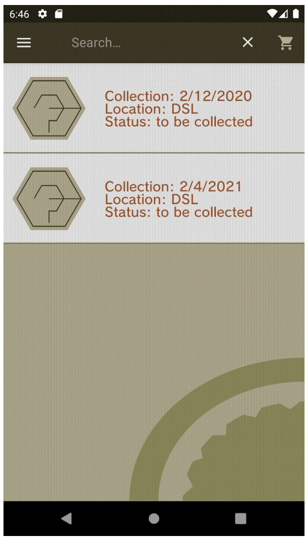
(Fig. 2: Profile Page)



(Fig. 3: Navigation Drawer)



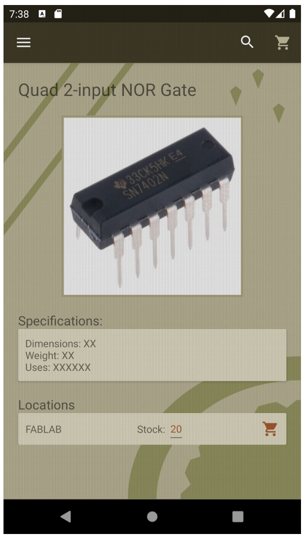
(Fig. 4: Login Page)



Collection: 2/12/2020
Location: DSL
Status: to be collected

Collection: 2/4/2021
Location: DSL
Status: to be collected

(Fig. 5: Search Page)



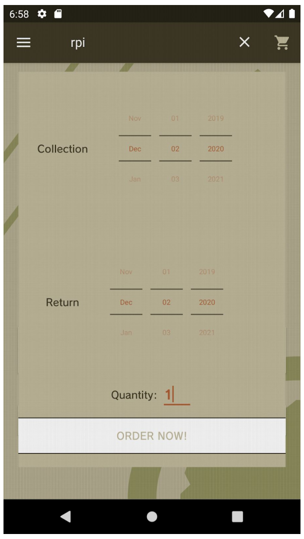
Specifications:

Dimensions: XX
Weight: XX
Uses: XXXXXX

Locations

FABLAB Stock: 20

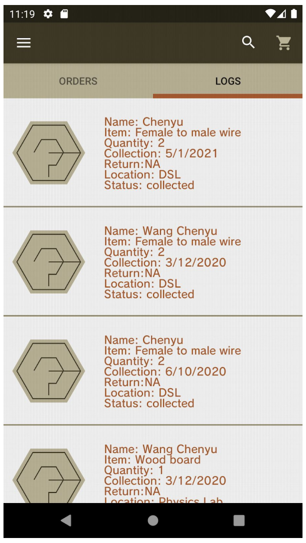
(Fig. 6: Item Listing)



Collection	Dec	02	2020
Return	Dec	02	2020

Quantity: 1

(Fig. 7: Resource Booking)



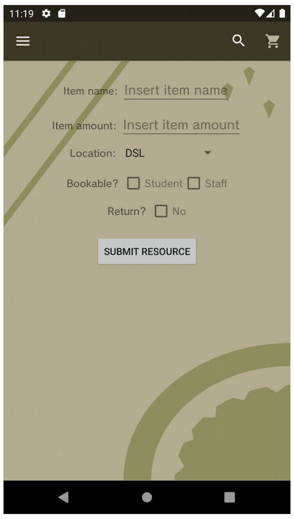
Name: Chenyu
Item: Female to male wire
Quantity: 2
Collection: 5/1/2021
Return:NA
Location: DSL
Status: collected

Name: Wang Chenyu
Item: Female to male wire
Quantity: 2
Collection: 3/12/2020
Return:NA
Location: DSL
Status: collected

Name: Chenyu
Item: Female to male wire
Quantity: 2
Collection: 6/10/2020
Return:NA
Location: DSL
Status: collected

Name: Wang Chenyu
Item: Wood board
Quantity: 1
Collection: 3/12/2020
Return:NA
Location: Physics Lab

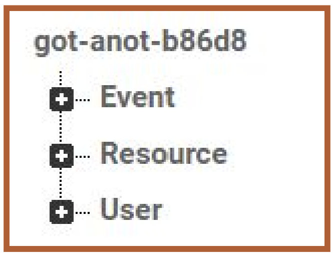
(Fig. 8: Order History)



Item name: Insert item name
Item amount: Insert item amount
Location: DSL
Bookable? ☐ Student ☐ Staff
Return? ☐ No

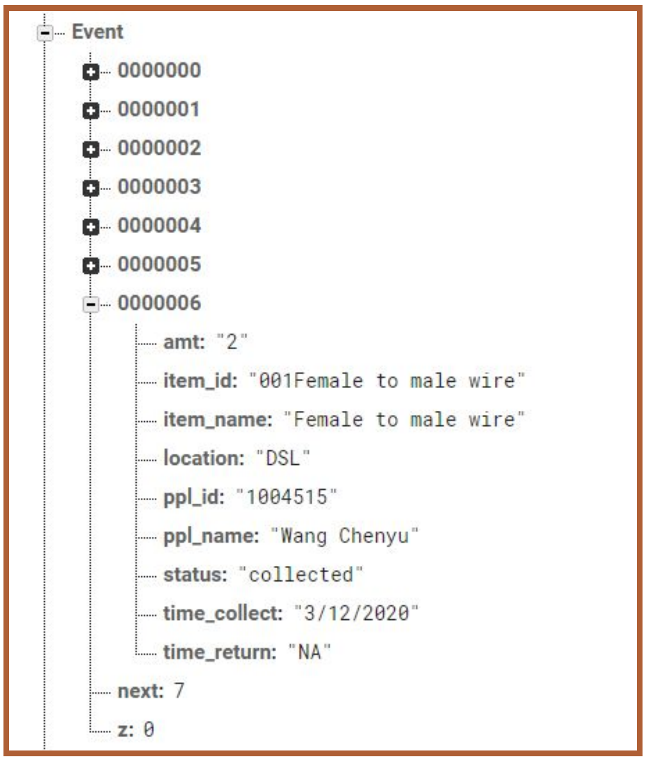
(Fig. 9: Add Resource Page)

Appendix B: Firebase



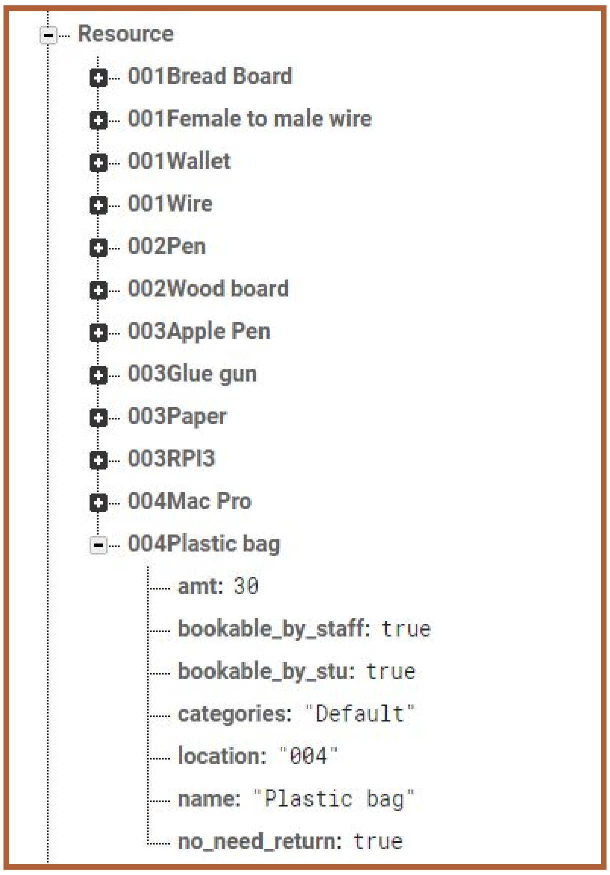
```
got-anot-b86d8
  Event
  Resource
  User
```

(Fig. 10: Firebase General Structure)



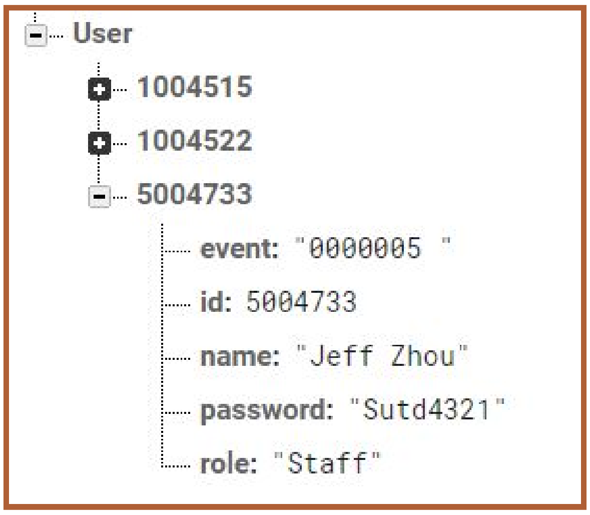
```
Event
  0000000
  0000001
  0000002
  0000003
  0000004
  0000005
  0000006
    amt: "2"
    item_id: "001Female to male wire"
    item_name: "Female to male wire"
    location: "DSL"
    ppl_id: "1004515"
    ppl_name: "Wang Chenyu"
    status: "collected"
    time_collect: "3/12/2020"
    time_return: "NA"
  next: 7
  z: 0
```

(Fig. 11: Event Hierarchy and Attributes)



```
Resource
    001Bread Board
    001Female to male wire
    001Wallet
    001Wire
    002Pen
    002Wood board
    003Apple Pen
    003Glue gun
    003Paper
    003RPI3
    004Mac Pro
    004Plastic bag
        amt: 30
        bookable_by_staff: true
        bookable_by_stu: true
        categories: "Default"
        location: "004"
        name: "Plastic bag"
        no_need_return: true
```

(Fig. 12: Resource Hierarchy and Attributes)



```
User
    1004515
    1004522
    5004733
        event: "0000005 "
        id: 5004733
        name: "Jeff Zhou"
        password: "Sutd4321"
        role: "Staff"
```

(Fig. 13: User Hierarchy and Attributes)