Fundamental of Software Engineering Project

University Access Pass

3150300210 Ariel Shih (施萱黛)

3150300211 Micheal Xie (谢文轩)

3150300213 Karmen Yung (翁海凌)

3150300216 Soobin Rim (林水彬)

Index

1. Introduction

In this project, we were tasked with coming up with a software application for helping University students to do some activity. We decide to create a virtual student id card application, which we called University Access Pass (UAP). UAP has six features which are verification, account balance, key-card, transportation, payment, and booking service. In this report, we will explain the user stories, UML use case models, UML sequence diagram, product backlog, component diagram, User Interface and System Testing that are involved in creating UAP.

2. User Stories

A user story is a description of one or more features of a software system. Depending on the project, user stories may be written by various stakeholders including clients, users, managers or development team members. Also, user stories are a type of boundary object. They facilitate sense-making and communication, that is, they help software teams organize their understanding of the system and its context. Our user stories for a software system are recorded on story cards as follows:

STORY CARD NO: 1	Project Name: University Access Pass	Estimation: 4 hours
Story Name: Verification		Date: 03/06/2017 10:00pm
STORY: The verification feature will allow the user to use the app as your student id, in times of security check and to have student discount even outside the campus.		Acceptance Test: 1. Try when the user information that showed up on the application is inaccurate
Note: User needs to log in to their own account first in order to do this operation.		Risk: High

STORY CARD NO: 2	Project Name: University Access Pass	Estimation: 4 hours
Story Name: Account Balance		Date: 03/06/2017 10:00pm

STORY: The account balance feature will allow the user to check the balance of the account, top up money from their bank account, and transfer money between users.	Acceptance Test: 1. Try when there is an insufficient amount of money in the user bank account 2. Try when there is an insufficient amount of money in the user account 3. Try if the user input the wrong information when linking to a new bank account
Note: User needs to log in to their own account first in order to do this operation.	Risk: Medium

STORY CARD NO: 3	Project Name: University Access Pass	Estimation: 4 hours
Story Name: Payment		Date: 03/06/2017 10:00pm
STORY: The payment feature will allow the user to pay for school Wi-Fi, dormitory utility fees, library fees, on campus food and beverage services (cafeteria, convenience stores, coffee shops, and restaurants), and university athletics services.		Acceptance Test: 1. Try when the amount of money that has been deducted from the account does not match with the calculated fee 2. Try when the payment has fail (i.e., may due to connection error)
Note: User needs to log in to their own account first in order to do this operation.		Risk: Medium

STORY CARD NO: 4	Project Name: University Access Pass	Estimation: 4 hours
Story Name: Transportation		Date: 03/06/2017 10:00pm

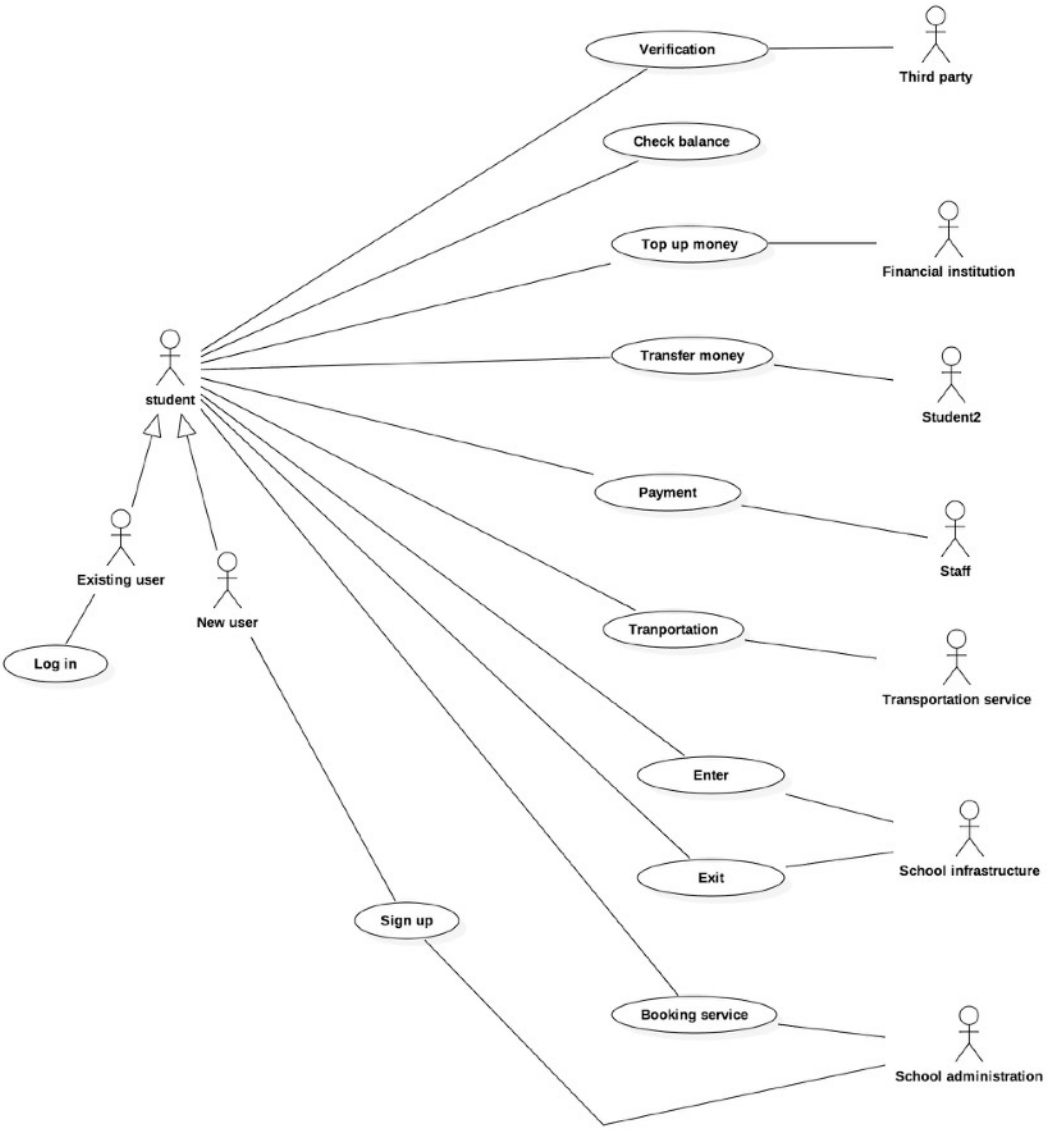
STORY: In order to facilitate the ease of transport on and between campuses for students by doubling as a transportation card. The transportation feature will allow students to use school shuttle, mini, and public buses; along with bike rental.	Acceptance Test: 1. Try when the distance is not calculated correctly 2. Try when the amount of money that has been deducted from the account does not match with the calculated fee 3. Try when there is an insufficient amount of money in the user account
Note: User needs to log in to their own account first in order to do this operation.	Risk: Low

STORY CARD NO: 5	Project Name: University Access Pass	Estimation: 4 hours
Story Name: Key-Card		Date: 03/06/2017 10:00pm
STORY: The key-card feature will allow the user to access various buildings on campus, such as laboratory, computer labs, library, gym, and dormitory.		Acceptance Test: 1. Try when the user does not have access to the infrastructure (i.e., staff only building) 2. Try when the amount of money that has been deducted from the account does not match with the calculated fee (i.e., fee of overstay)
Note: User needs to log in to their own account first in order to do this operation.		Risk: Low

STORY CARD NO: 6	Project Name: University Access Pass	Estimation: 4 hours
Story Name: Booking Service		Date: 03/06/2017 10:00pm

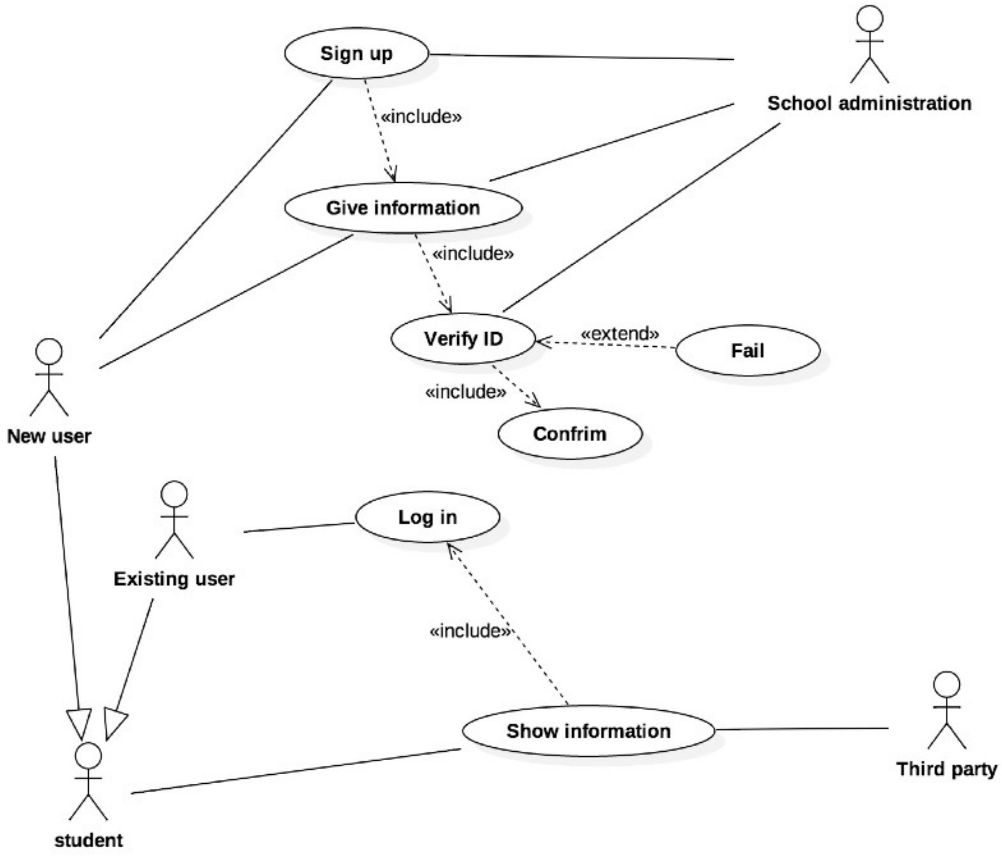
STORY: The booking service feature will allow the user to book lab, study room, and conference room for a specific time slot.	Acceptance Test: 1. Try when the user input the wrong information (i.e., incorrect length of student id/username) 2. Try when the infrastructure has already been booked
Note: User needs to log in to their own account first in order to do this operation.	Risk: Low

3. Unified Modeling Language use cases models



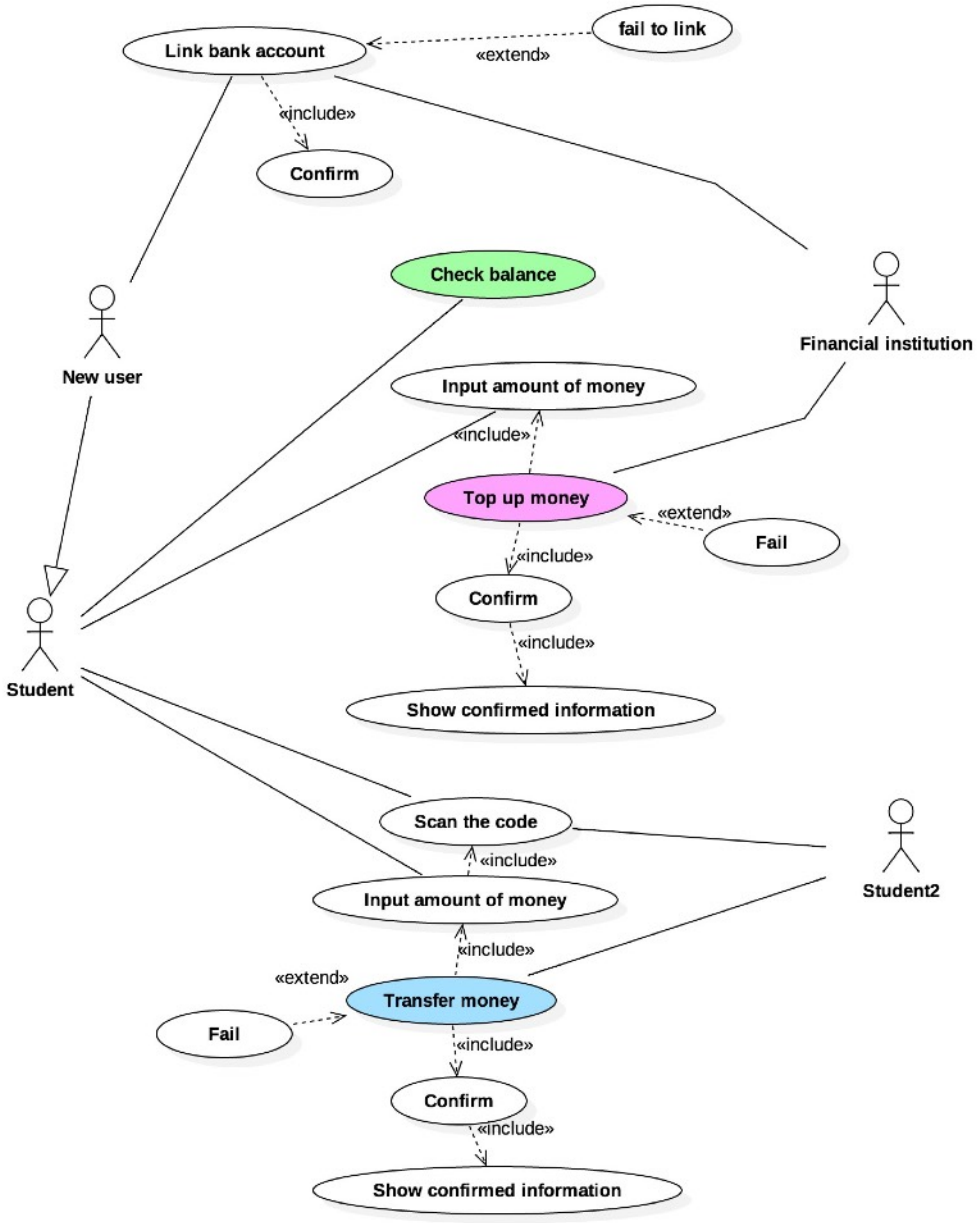
This is our main use case model that illustrates how our software that has six different features works in between different actors. UML use case model illustrates the primary interactions between the relevant actors and the system. Each model consists of the system's intended functions(use cases), its surroundings(actors), and the relationship between actors and use cases(associations). And the detailed use case models for each feature are as follows:

- Verification



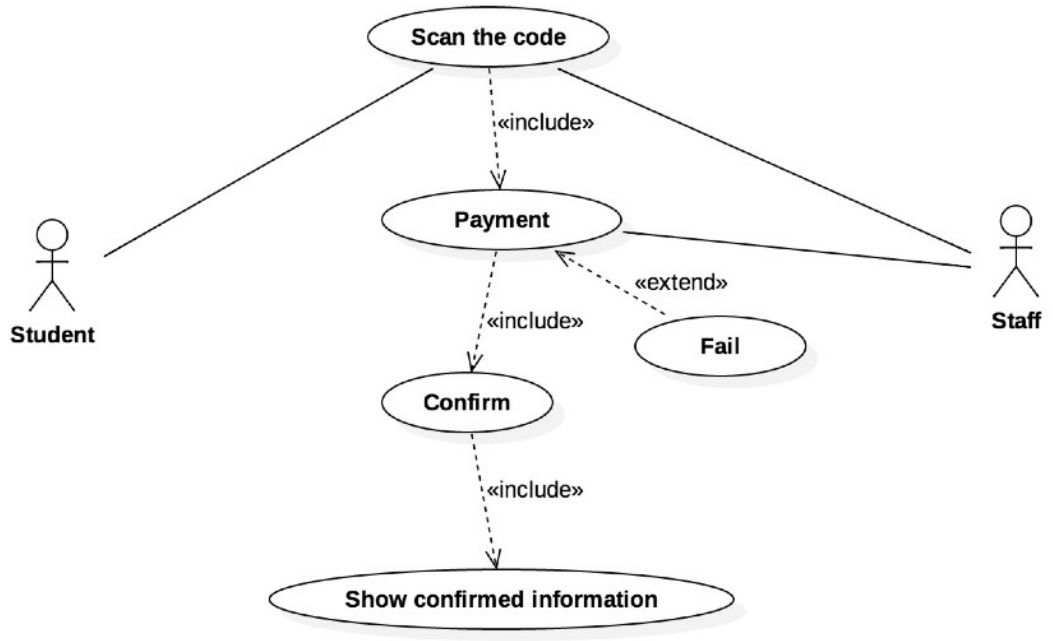
The first thing we need to do is verifying the new user's identification before they actually start using it. The use cases for new users look like this: they need to sign up for the very first time, providing their information(name, student number, passport number and photo) to school administration. After that, the school administration will verify the person's ID and give the system an approval. The existing user who has already signed up for the app, just simply log in and show their ID to the id checker.

- Account balance



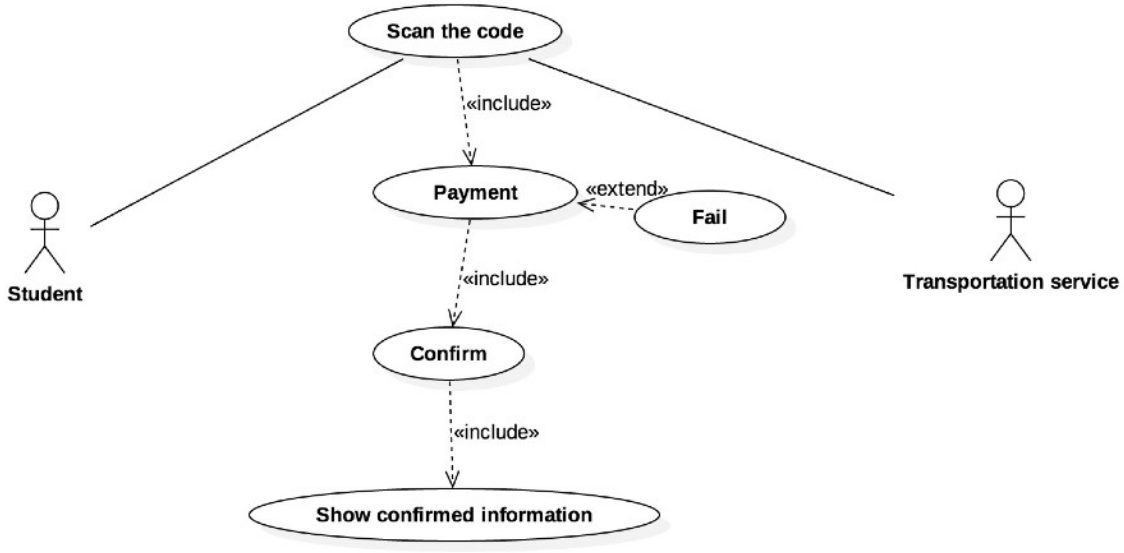
We also have a special case; new user, he needs to link his bank account for the very first time. But in general, a student who has already linked to his bank account, can easily top up money, check balance and transfer money to another students if they have the same app. First, to top up money from the bank account they need to input the amount of money. This is the pre-condition of 'top up money', then our system will show the user either success or failure message depends on whether the user has sufficient amount of money in his bank account or not. Second, to transfer money, they first need to scan each others' code. And the student needs to type the amount of money that he's going to transfer to the other student. At the end, the system will show the user a confirmed message based on his balance status.

- Payment



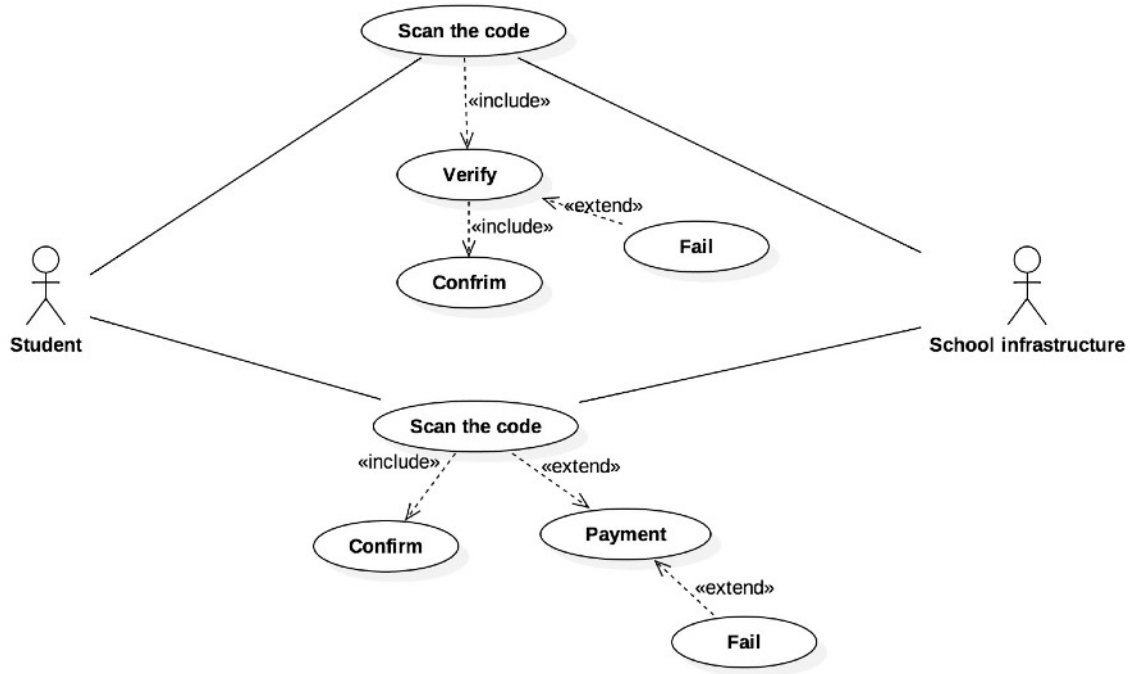
To use the payment feature, student must show their QR code to service staff, who will then scan the code to receive payment and finish the transaction. If the payment is successively done, then the user will receive a payment confirmation message. If the account has insufficient balance or if there is a connection error, then the transaction will fail.

- Transportation



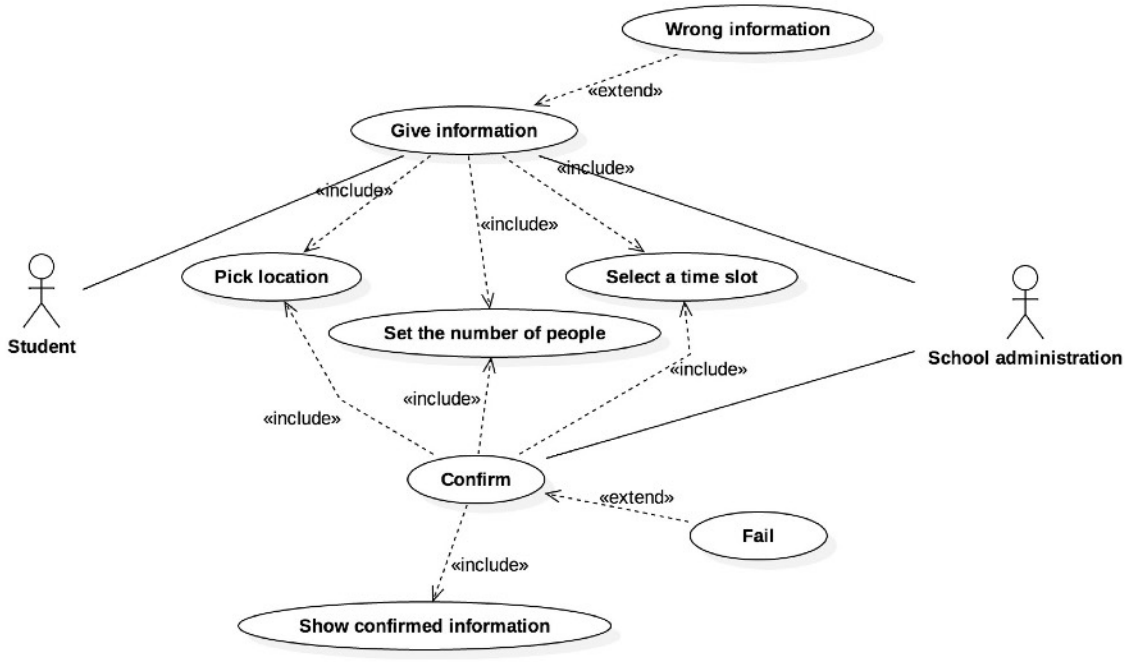
To use the transportation feature, student must allow their QR code to be scanned by the QR code scanner on the bus, which will then collect the fixed fee for the route. If the payment is successively done, then the user will receive a payment confirmation message. If the account has insufficient balance or if there is a connection error, then the transaction will fail.

- Key-card



To use the key-card feature, student must allow their QR code to be scanned by the QR code scanner by the entrance of the buildings they would like to access. After the building scans the QR code, it will verify their identity; if the student is an authorized user, then the door will unlock and let the user into the school facility. Exiting the building is a similar process, unless the facility is a gym or pay-per-use laboratory, in which case the building will collect the fixed usage fee from the student and let them out. If their account balance is insufficient to pay the fee, then the exit will remain lock and the student will have to go to the front desk.

- Booking Service



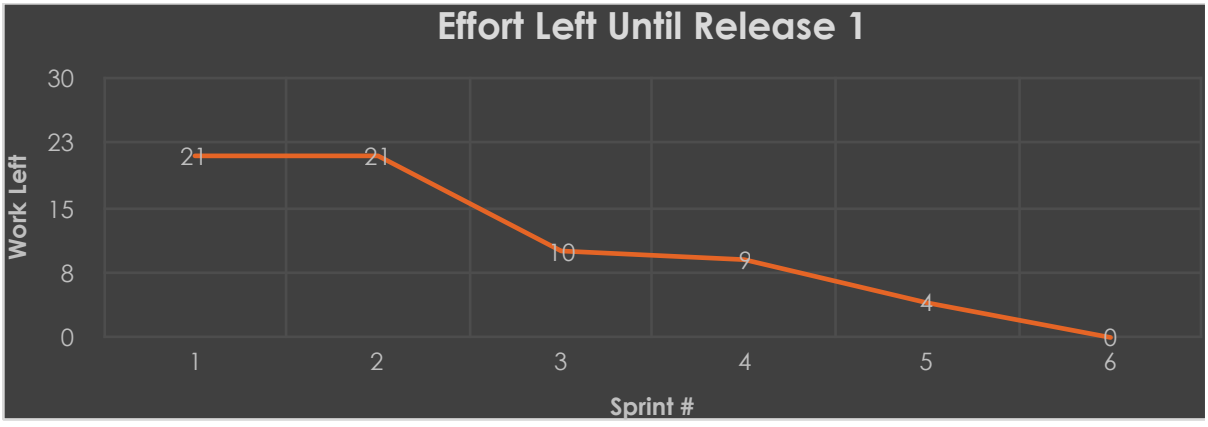
To use the booking service feature, student enters the booking feature and provides information such as location, time, and number of people they want to book a study room for. The information is then send to the school administration, who will then confirm whether the room/time slot is available; if it is available then the confirmation message will be send to the student, otherwise the student will be told to select a different room/time slot.

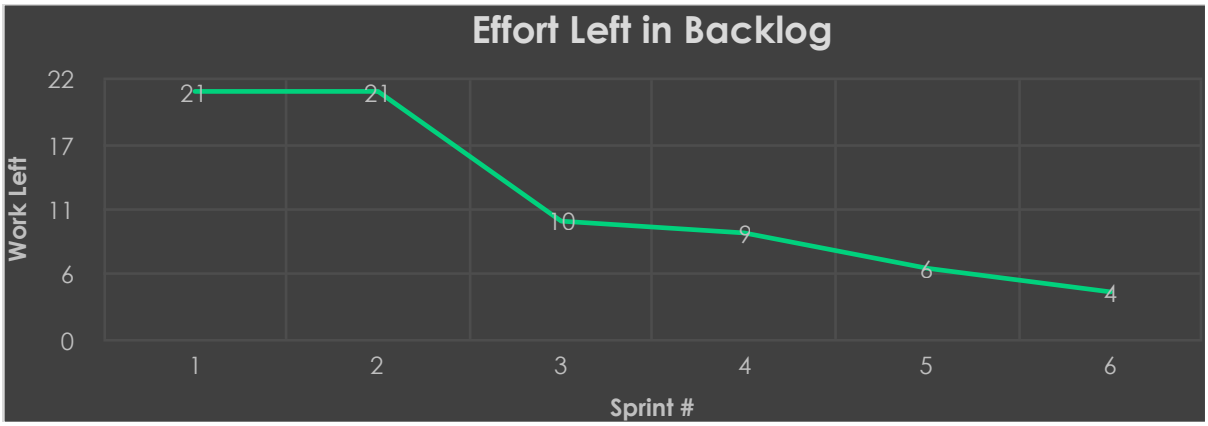
4. UML sequence diagrams

5. Product Backlog

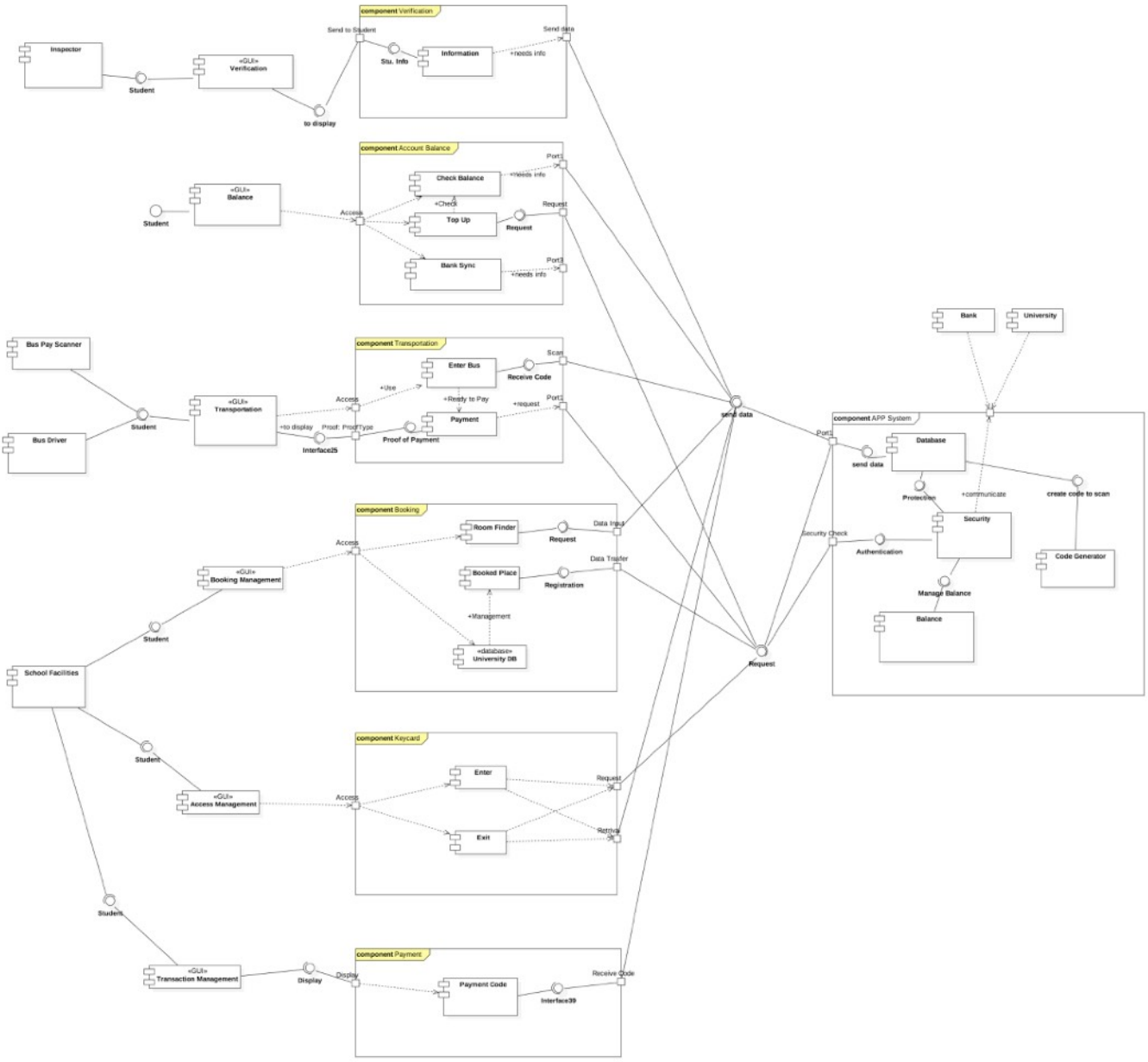
ID	Description	Sprint #	1	2	3	4	5	6
		Effort needed for Release 1 as in the beginning of the sprint	21	21	10	9	4	0
	/*As a user I want to use the application as my student id*/							
1	create application skeleton		5	5				
2	set up integration system (for school server and app to communicate)		2					
3	set up server to display student information (picture, student #, etc.)		2					
4	set up account registration system (only allows enrolled student to view their student infromation on the app)		5					
5	As administration, I want to set up QR code that allows for money transaction and use it as another unique identification		3	2				
Sprint 1	Makes data from school server appear on device							
	/*As a user I want to be able to top up and check account balance*/							
6	set up integration system (for app and selected bank servers to communicate)		2	2				
7	As administration, I want to verify if the user has an account with the selected bank		1	1				
8	As administration, I want to set up an account where the user can transfer money into		1	1				
9	As administration, I want to link the bank account to the user account (allows for top up from their bank)			4				
10	As user I want to be able to know how much money I have left in the account			1				
Sprint 2	Makes monetary transaction possible between the bank server and the device							
	/*As a user I want to be able to transfer money and make a payment*/							
11	As a user I want to be able to transfer money to another student's UAP account			1	1			
12	set up integration system (for school facilities and app to communicate)			2	2			
13	As a user I want to make a payment inside of the campus			2	5			
Sprint 3	Makes monetary transaction possible between the school facilities and the device, as well as user-to-user transaction							
	/*As a user I want to be able to book study spaces, labs and conferences on campus*/							
14	set up a graphical interface for booking information				1	1		
15	As administration, I want to allow user to choose time slot, campus, room type, and number of occupants				1	2		
16	As administration, I want to verify if the room is available or not					1		
17	As administration, I want to send the confirmation message					1		
18	As administration, I want to add auto-refresh feature					2		
Sprint 4	Makes booking service possible							
	/*As a user I want to be able to access various buildings on campus (i.e., labortary, gym, and library, etc.)*/							
19	set up integration system (for school infrastructure server and app to communicate)					2	2	
20	As a user I want to be able to pay for entrance fee to gym or computer labs						2	
Sprint 5	Makes monetary transaction possible between the school infrastructure server and the device							
Release 1	Usable Version							
	/*As a user I want to be able to use school transportation service (i.e., shuttle, mini buses and bikes)*/							
21	set up integration system (for school transportation server and app to communicate)						2	2
22	As a user I want to be able to pay for the transportation service fee							2
Sprint 6	Makes monetary transacation possible between the transportation server and the user account							
Release 2	Add Transportation Service							
		Effort in the Whole Backlog	21	21	10	9	6	4

Backlog state taken after the end of Sprint #5 = after Release 1





6. Component diagrams



- Verification: the student enters the verification feature, which will then retrieve their information from the student data base and display it on their phone.
- Account balance:

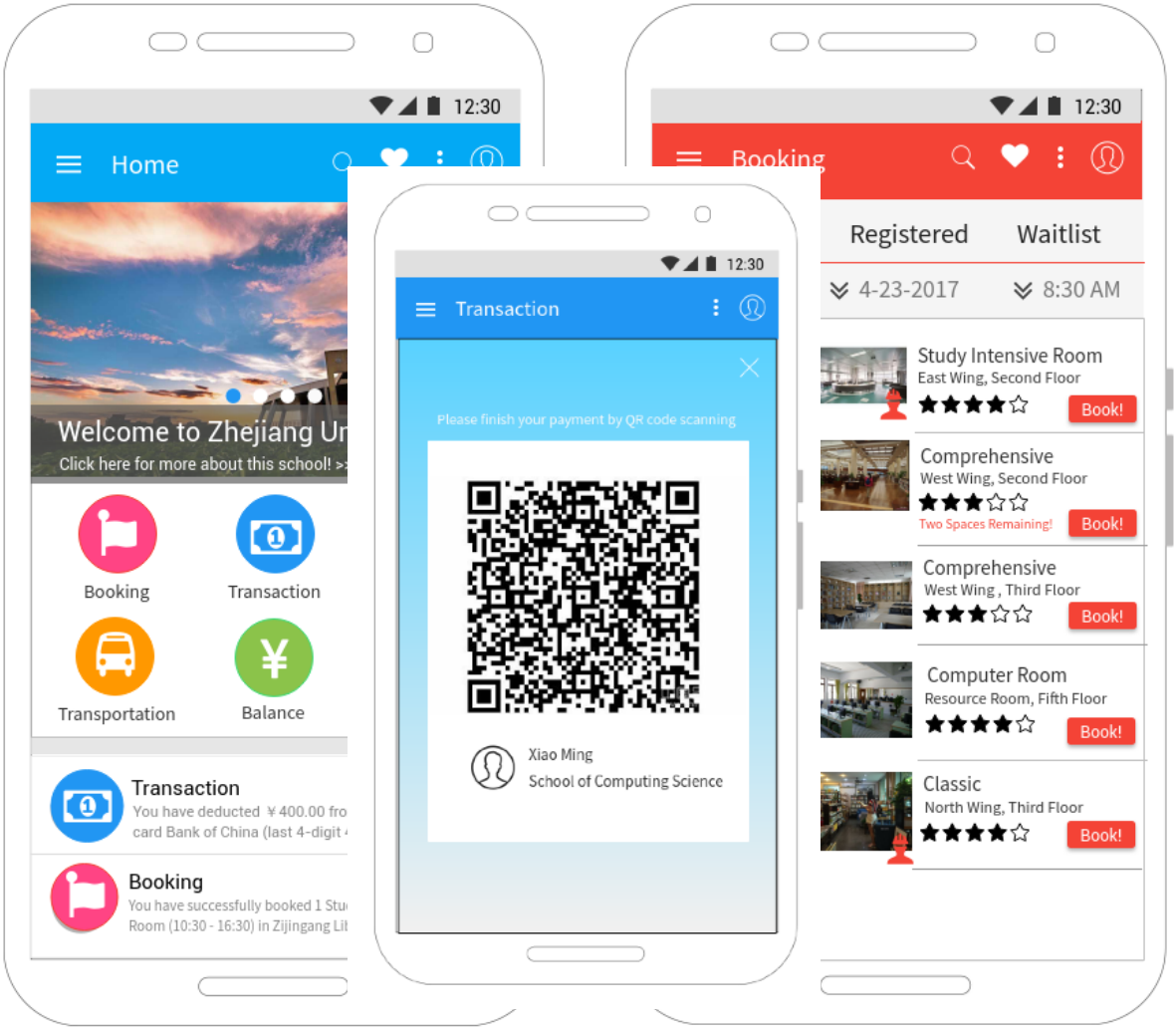
- Transportation:

- Booking: simply our booking management system will allow the user to book school facilities such as study rooms or meeting rooms. First, our room finder component will get the data from the user, and based on our university database, our system will be able

to either confirm or deny the user request.

- Key-card: whenever the student enters and exits the building, our system will get the student's information from database and also send this data back to school server so that they can authorize the request in both cases.
- Payment: the student enters the payment feature, which will then retrieve their QR code from the application data base and display it for third party to scan and collect payment from the user.

7. User Interface



8. System Testing

1. **Usability Testing** is checking how user-friendliness the application is by checking the flow of system navigation (a.k.a. ease of use). We are going to roll out a Closed Beta Testing version of the application, it will be given out to 800 students randomly selected from student data base. For the Beta test any system errors will prompt the user to submit a feedback form by filling out what type of operating system the user uses, the make and model of their phone, what kind of error they encounter, and what were they doing when the error occurred. We will also hold a live testing section by setting up a booth on campus so that students walking by can tryout our application, they will be handed a questionnaire when they are done; questionnaire will ask questions such as:

    - Was the application easy to use?
    - Was the sign-up process straightforward?
    - Was your personal information that was retrieved from the school server correct?
    - How do you find the user interface, is it clutter or confusing to use in any way?
    - Any suggestions or difficulties?

2. **Load / Stress Testing** is testing beyond the limits of normal operation.

    Example: Booking Service

    - Start by having 50 people using the booking service at the same time, if no errors occur, continue to increment by 50 people until a system error or crash occurs.

    Example: Payment

    - Start by testing user-to-user transfer with ¥100, increment by ¥100 until a system error occurs

    Example: Registration / Verification

    - Start by having 50 people registering an account using our application, increment by 50 people until exceeding the maximum number of users that can hold an account
    - Start by having 50 people registering an account at the same time, increment by 50 people until a system error or crash occurs

3. **Functional Testing**

    - Security: our application deals a lot of money transaction, so we will need some sort of security that we would be able to use to protect our user information as well as their monetary accounts.
    - Transportation: so far our transportation feature allows us to use transportation on campus, such as school mini bus and school shuttle bus, we can try to extend it to allow users to use public transportation as well.