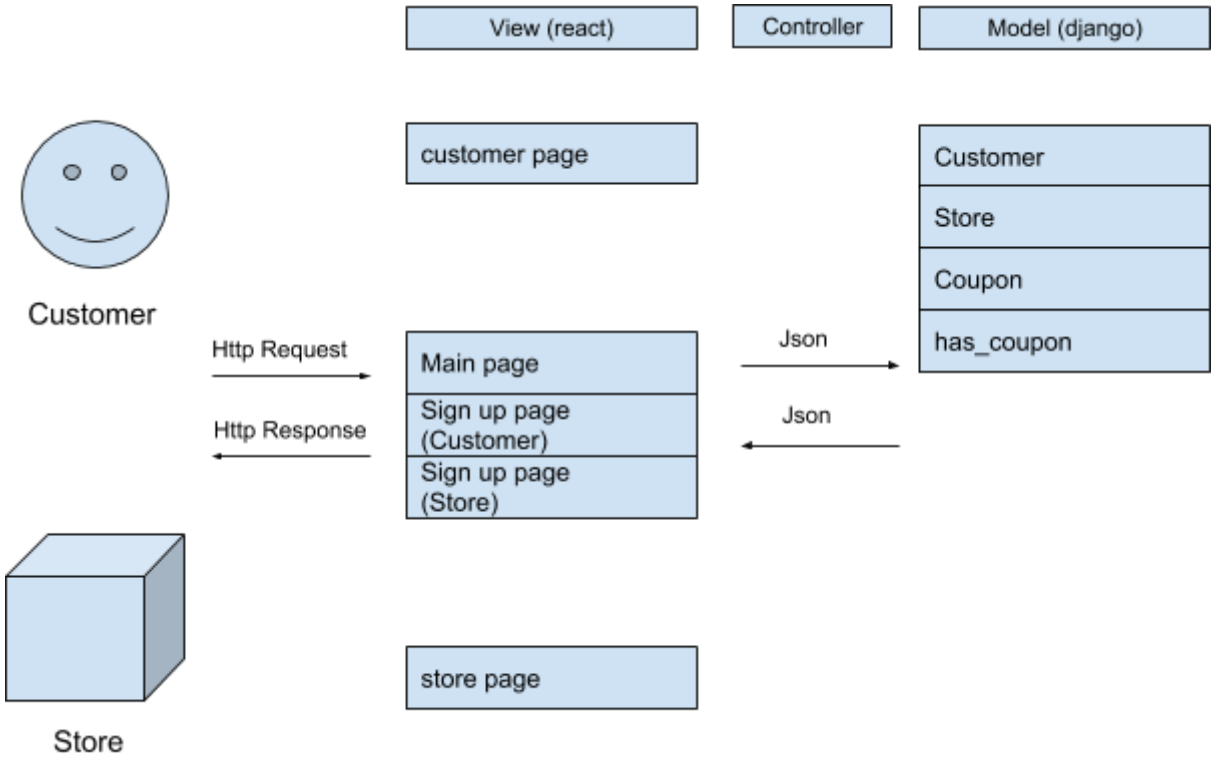
티끌모아

Design and Planning
2018. 4. 30 (version 1)

Kim Shin Hwi
Kim Ji Hoon
Seol Jae Wan
Yu Geun Gook

System Architecture

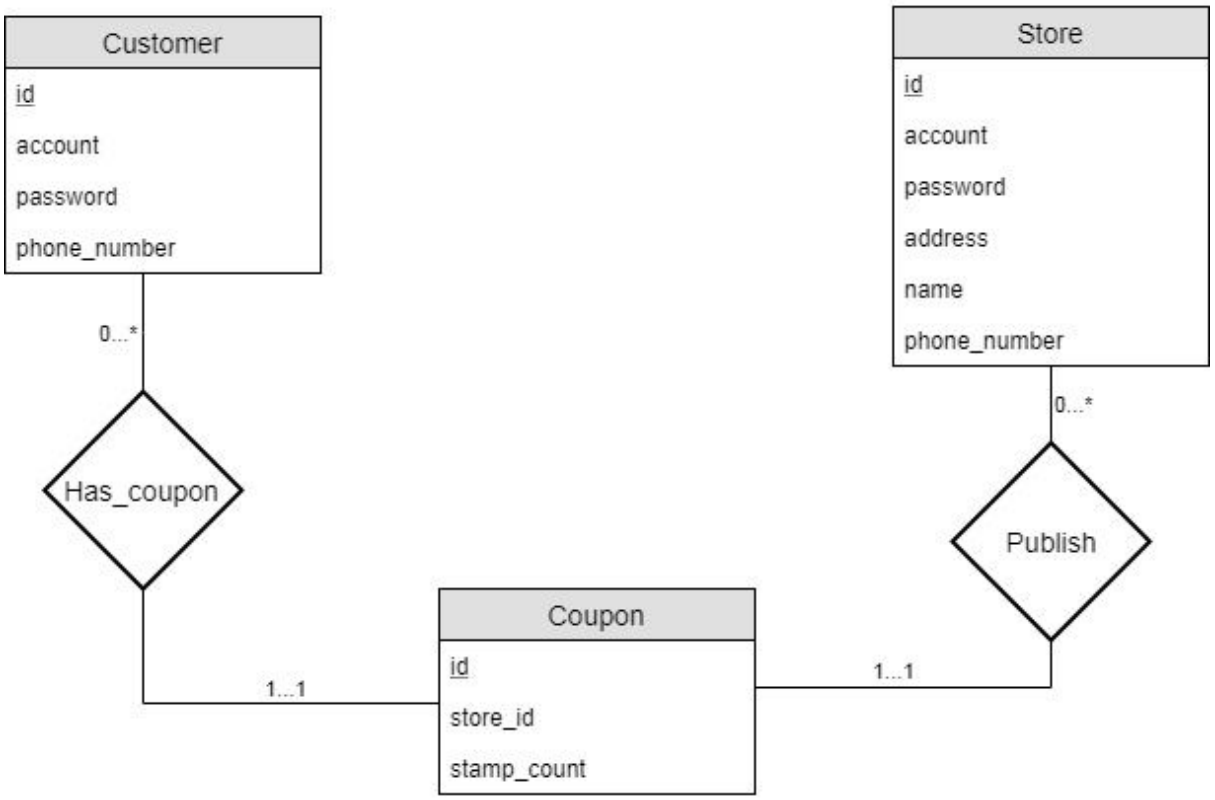
1. MVC



There are 4 views and 4 models. detail will be described further below

2. Model

- E-R Diagram

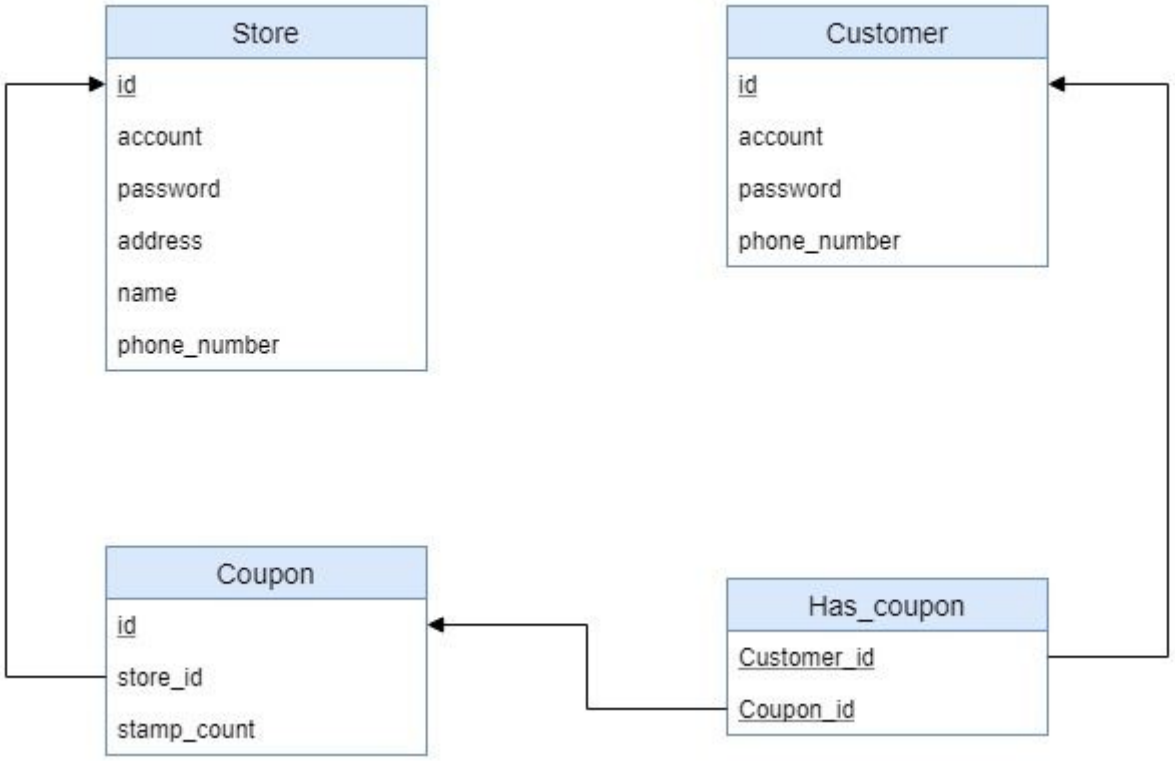


In E-R diagram, rectangle means entity set, diamond means relationship set and numbers above line represent cardinality constraints. All attributes are listed inside each rectangle. Underlined attributes are primary key of each entity set.

There are three entity sets and two relationship sets. Each ‘customer’ and ‘store’ entity stands for customer account and store account. Users sign in with ‘account’ and ‘password’ attributes. ‘Coupon’ entity has relationship with ‘customer’ entity and ‘store’ entity. ‘Has_coupon’ relationship has information about which customers have which coupons. 'Publish' relationship indicates which coupons were issued in which stores.

A customer can have no coupon or can have many coupons. Likewise, a store can publish no coupon or can publish many coupons. Therefore, cardinality constraints should be 0…*. On the other hand, a coupon has to have only one owner(customer) and is published by certain store. Therefore, cardinality constraints should be 1...1.

- Relation schema diagram

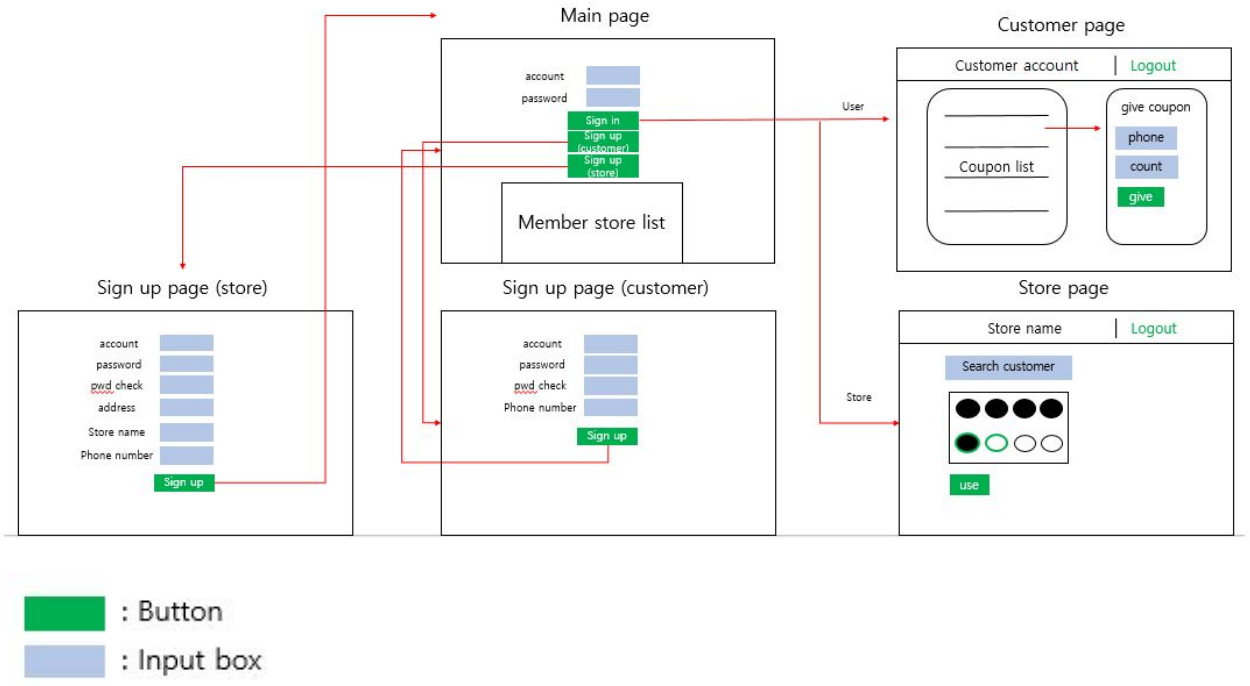


Based on E-R diagram above, relation schema diagram can be made like this. Rectangle means relation schema, and all schema attributes are listed inside each rectangle. Underlined attributes are primary key of each relation, and arrow stands for foreign key constraints.

Converting E-R diagram to relation schema diagram, ‘publish’ relationship set is simply represented by foreign key constraint. ‘Coupon’ relation has ‘store_id’ that is foreign key attribute pointing ‘id’ of ‘store’ relation.

On the other hand, the ‘Has_coupon’ relationship set is converted to a relation. It represent relationship between ‘customer’ and ‘coupon’ by having ‘customer_id’ and ‘coupon_id’ attributes.

3. View



Main page (“/”)

- Sign in
- Move to sign up page
- Show member store list

Sign up page - customer (“/sign_up_customer”)

- Get account, password, password check, phone number as input
- Sign up

Sign up page - store(“/sign_up_store”)

- Get account, password, password check, address, name, phone number as input
- Sign up

customer page (“/customer/id”)

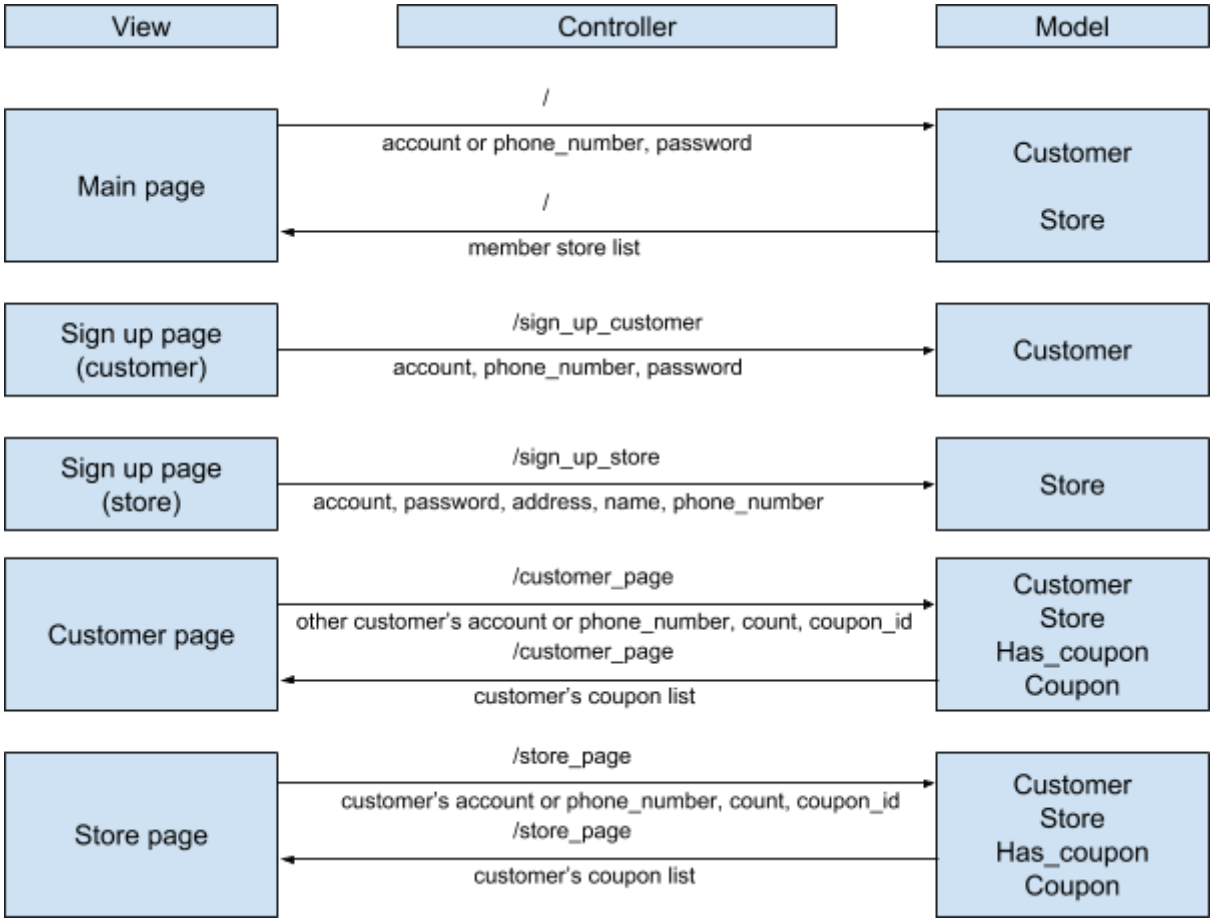
- List my coupons
- When click a coupon, show ‘give’ window
- In ‘give’ window, get target’s phone number and count and transfer my coupon

store page (“/store/id”)

- Search customer by phone number or account
- Show the customer’s coupon status
- Give or delete coupons
- Use coupons

4. Controller

Here is controller design.



Left side is view part (frontend) and right side is model part (backend). Left-to-right arrow represents http-request with user inputs from view, and right-to-left arrow represents http-response with data from model. Above the arrow, there is an API that controller uses to transfer JSON data below the arrow

Design Detail

5. Frontend

We have 5 pages and each page has components and methods.

main
account: input password: input signIn: button signUpCustomer: button signUpStore: button memberStoreList: listComponent
onClickSignIn onClickSignUpCustomer onClickSignUpStore

sign up store	sign up customer
account: input password: input passwordCheck: input address: input name: input phoneNumber: input signUp: button	account: input password: input passwordCheck: input phoneNumber: input signUp: button
onClickSignUp	onClickSignUp

customer page	store page
customerAccount: text logout: button couponList: couponListComponent .phoneNumber: input .count: input .give: button .components are shown only when onClickToggleUp	storeAccount: text logout: button customerAccount: input .couponPanel: couponPanelComponent .components are shown only when onClickSearch
onClickToggleUp onClickToggleDown onClickLogout onClickGive	onClickSearch onClickLogout onClickUse

6. Backend

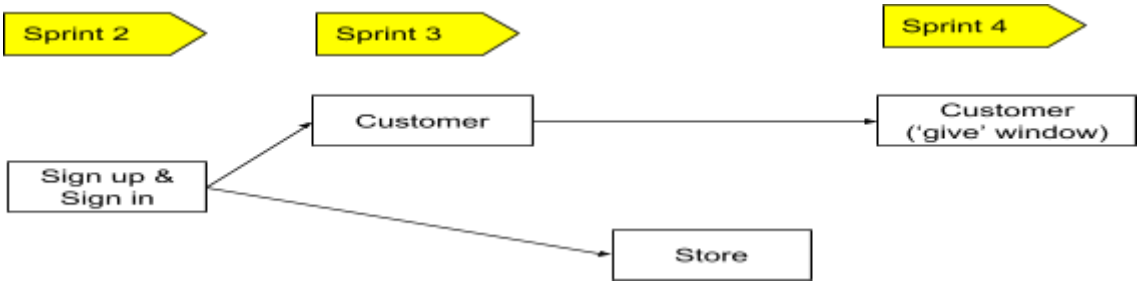
Detailed specifications of RESTful APIs are as following:

Model	API	GET	POST	PUT	DELETE
Customer	/api/sign_in	X	validate log-in	X	X
	/api/sign_up_customer	X	create new customer	X	X
	/api/customer/:id	get coupon list	X	X	X
Store	/api/sign_in	X	validate log-in	X	X
	/api/sign_up_store	X	Create new store	X	X
	/api/store/:id	get coupon list	X	transfer coupon	transfer coupon
Coupon	/api/customer/:id	get coupon list	X	X	X
	/api/store/:id	get coupon list	X	X	X
Has_Coupon	/api/customer/:id	get coupon list	X	X	X
	/api/store/:id	get coupon list	X	transfer coupon	transfer coupon

Plan

Our user stories are broken down into programming tasks below. Rectangle stands for page, including all programming tasks related with the page.

Arrows show hierarchy of implementation. ‘Sign up’ and ‘sign in’ pages and all features related with these pages should be implemented before other implementations. Therefore, ‘sign up’ and ‘sign in’ implementations are in sprint2. ‘Customer’ and ‘Store’ implementations are in sprint3. In sprint4, implementation of ‘giving stamp to other customer’ feature(which is part of customer page) is included.



Page	Feature	Difficulties (1-5)	Time (mins)	Sprints	Person
Main	Go to sign up page	1	60	2	1
Main	Show member store list	2	120	2	1, 2
Main	Sign in and go to customer or store page	3	180	2	2, 3, 4
Customer sign up	Sign up and go to main page	2	120	2	4, 3
Store sign up	Sign up and go to main page	2	120	2	1, 3
Customer	Show coupon status	2	120	3	2, 4
Customer	Show ‘give’ window	3	180	4	1, 2, 3, 4
Customer	Give stamps to other customer	2	120	4	1, 2, 3, 4
Store	Search customer and show the customer’s coupon status	3	180	3	2, 4
Store	Give stamps to customer	2	120	3	1, 2, 3
Store	Take(delete) stamps of customer	2	120	3	1, 3
Store	Use stamps of customer	2	120	3	2, 4

(person - 1: shinhwi, 2: jihoon, 3: jaewan, 4: geongook)