

ECE 18-652

Final Project Presentation





Hello!

We are **Team A1**

Congshan Lv, Guangyu Chen, Jin Gu, Jiyu Shi, Pan Li



Overview

- Software Architecture
- Features of Our App
- How to Test
- Practices and Quality attributes
- Lessons learned

1

Software Architecture



Software Architecture

HTML, CSS, JavaScript	Basic web building blocks
Bootstrap	Mobile compatibility

Express.JS	Server framework
Node.JS	Runtime environment
SQLite3	Database



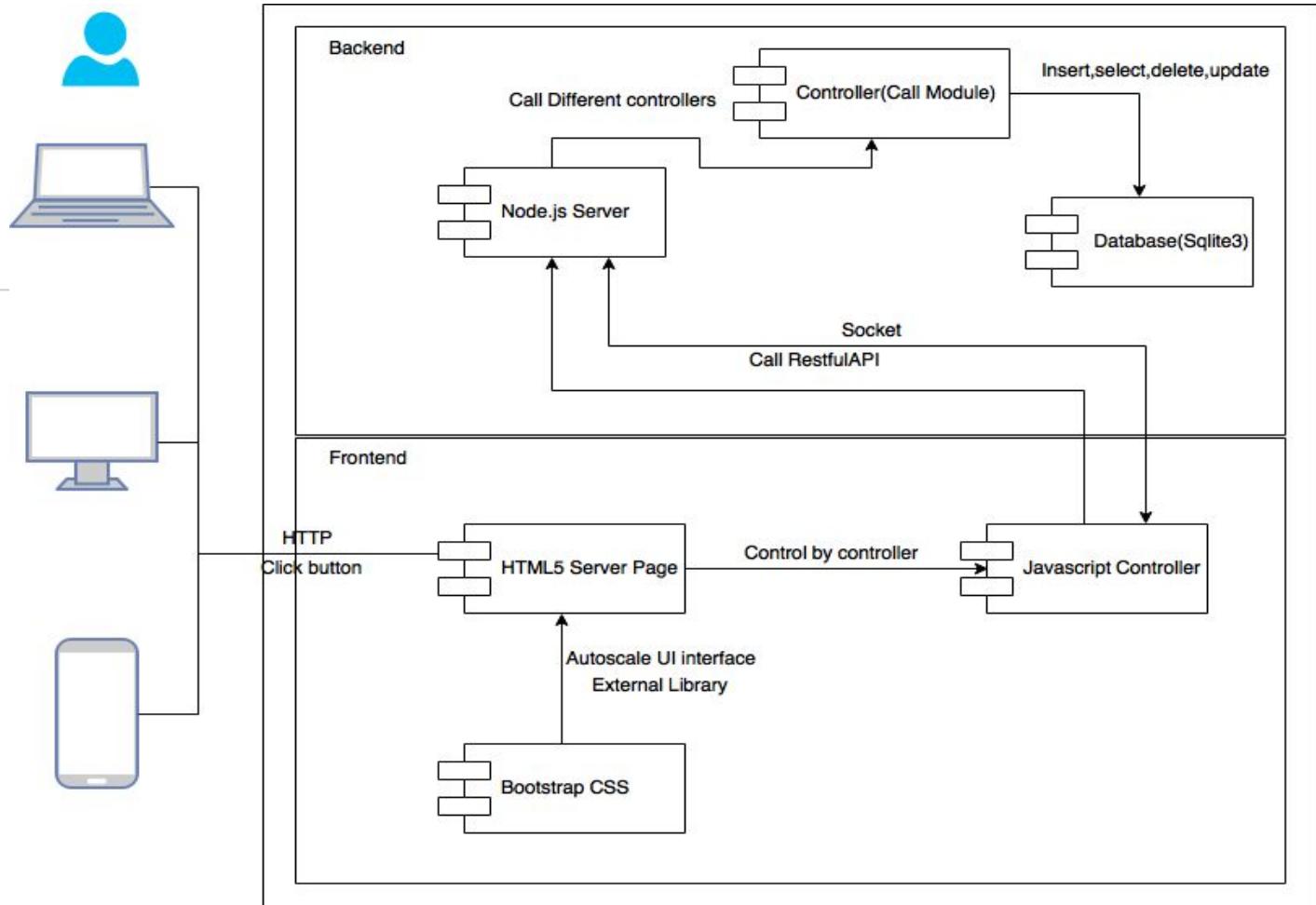
Software Architecture

Github	Version control and issue tracking
Trello	Project and backlog management
CircleCI	Continuous integration



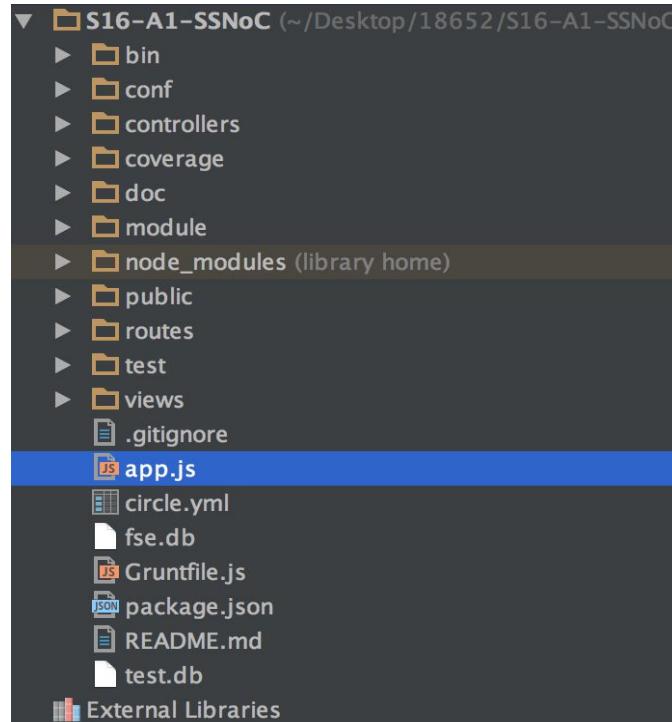
Software Architecture

Selenium IDE	UI, system and acceptance testing
Excel	Documenting for application's RESTful API
StarUml	Diagram editing, creating UML artifacts
Webstorm	Integrated Development Environment
Istanbul	CodeCoverage
Mocha & Grunt	Used for unit test



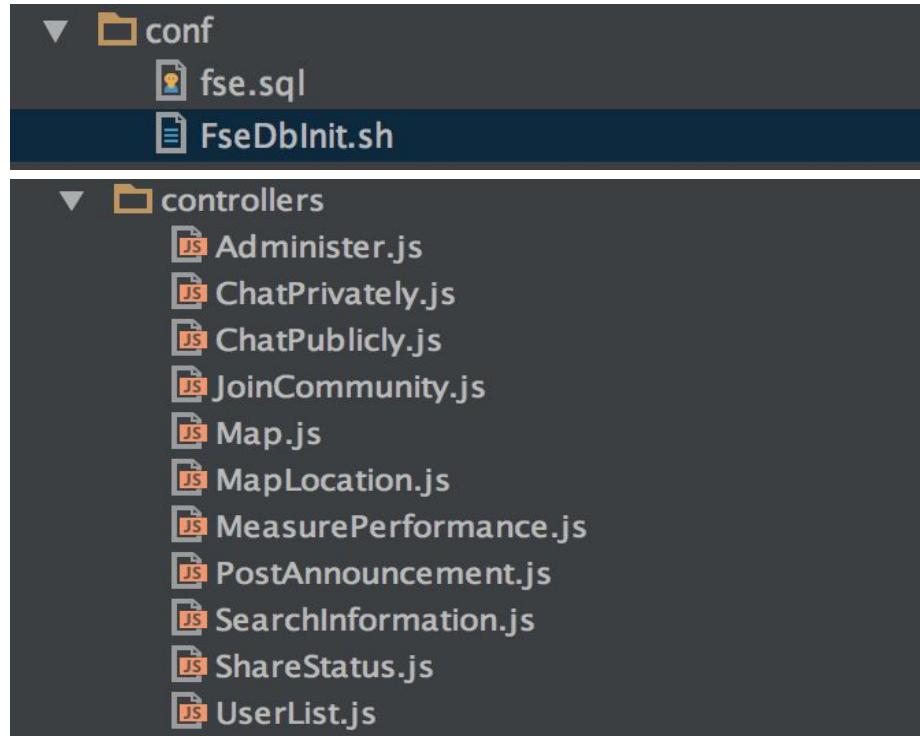


Code Architecture



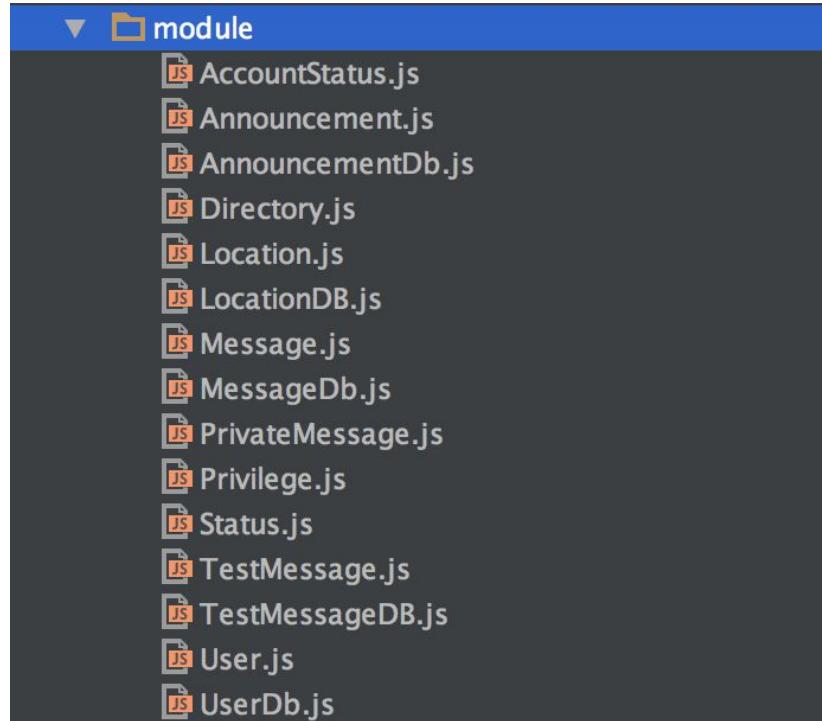


Code Architecture



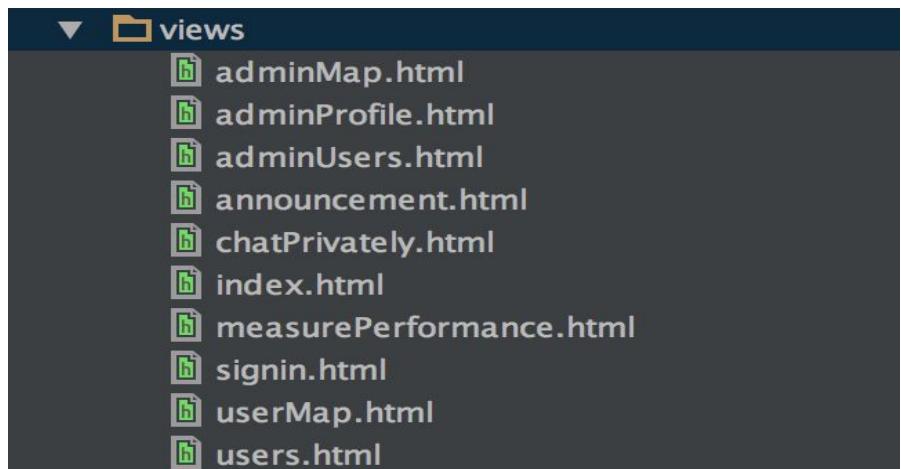
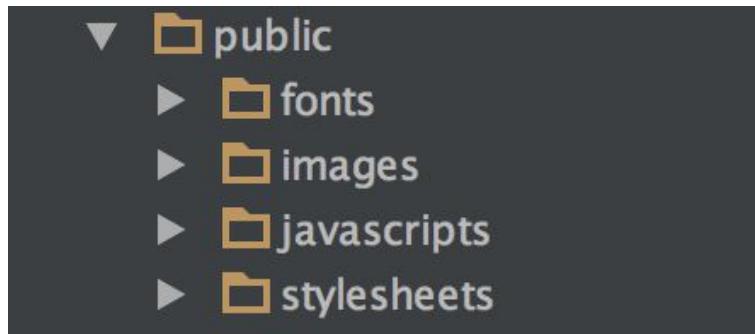


Code Architecture



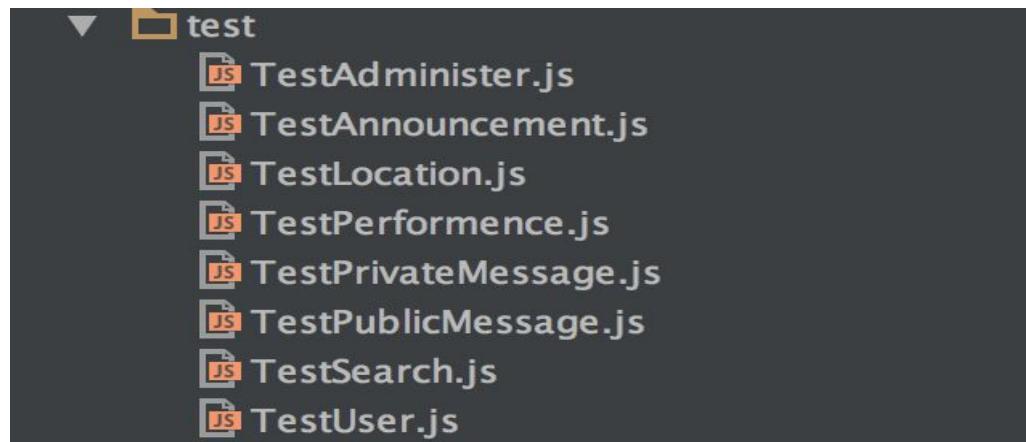


Code Architecture





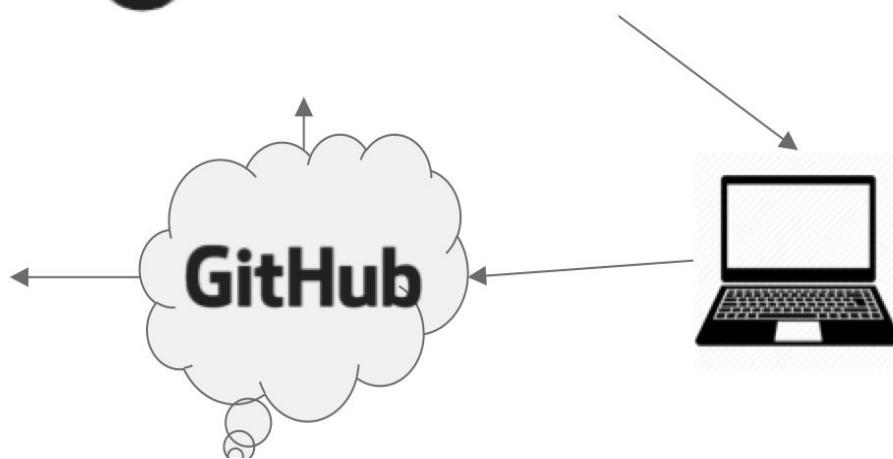
Code Architecture





Code Push

circleci





2

Features of Our App



What can citizens do?



Login

As an existing user, you can login to SSNoC with your username and password.





Login as New User

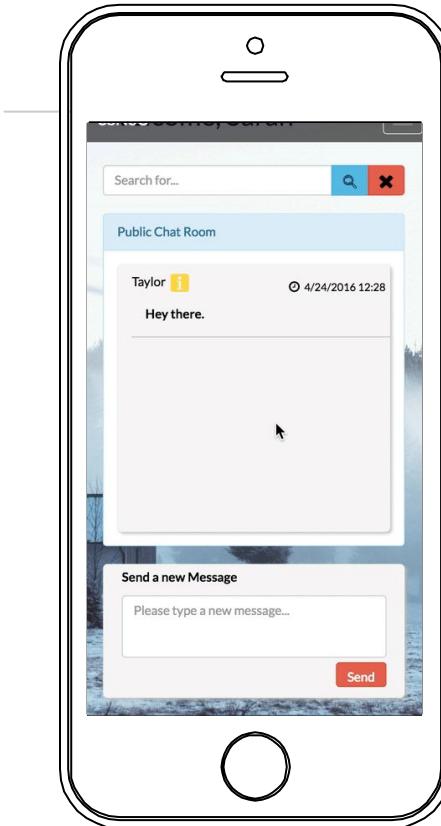
As a new user, you can register and login to SSNoC with your username and password.





Chat Publicly

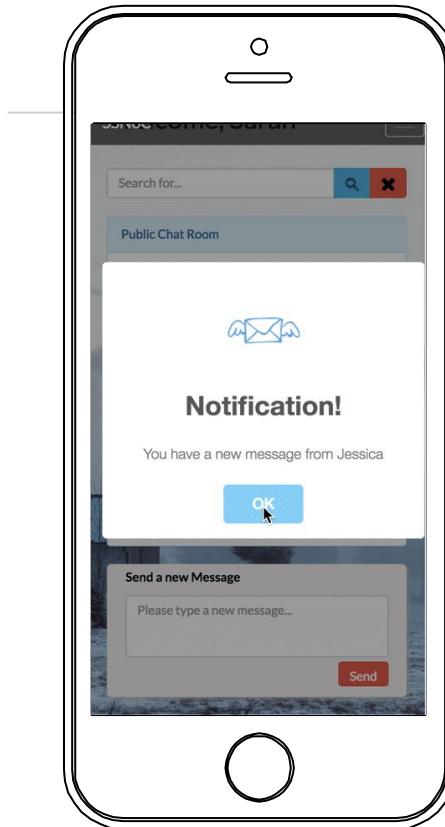
Users would enter the public chat room after login, where they would be able to join the public chat.





Chat Privately

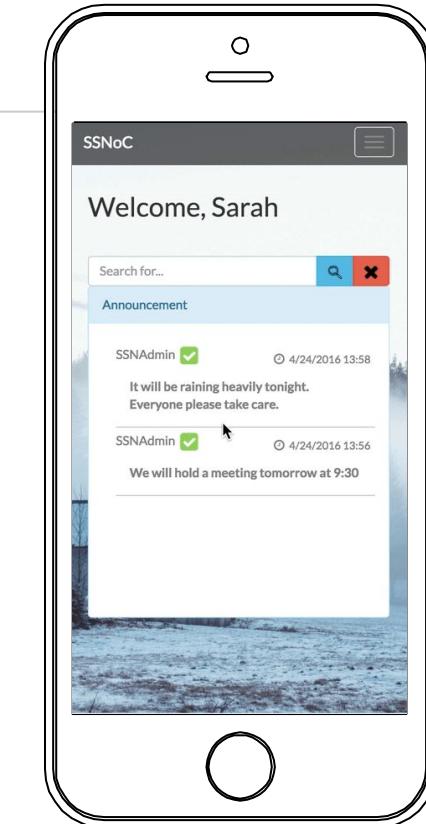
Users can also have one-on-one conversation with another user.





Announcements

Citizens can view/search the announcements posted by the administrator.

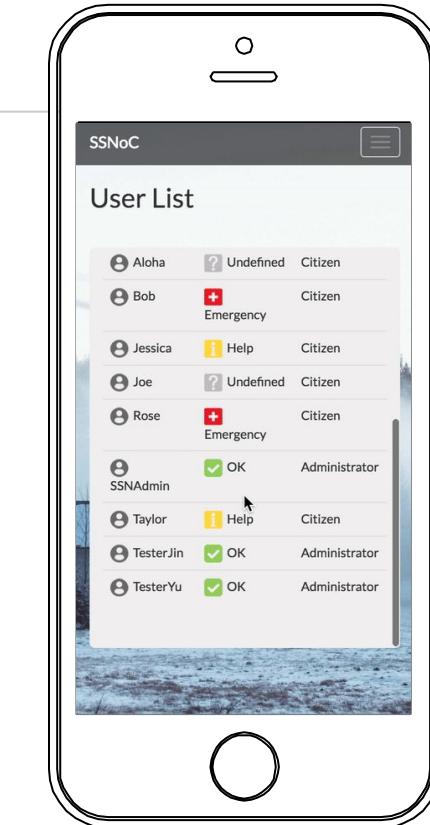




User List

Users can view a list of all users and their status.

Users can also search a user by status or username.



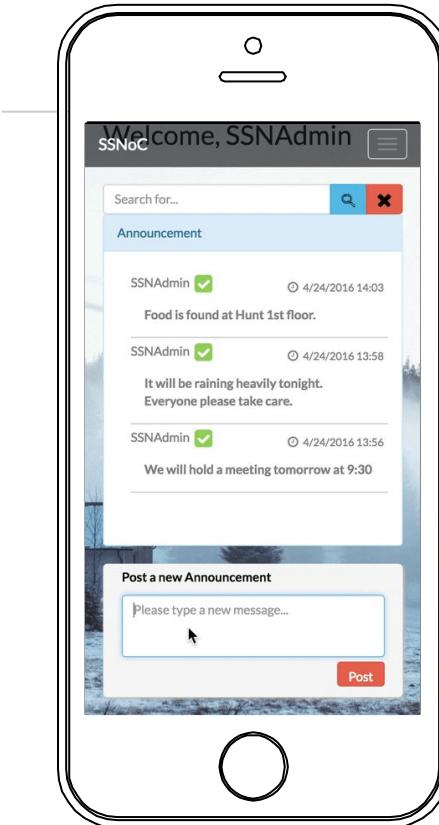


What can administrators do?



Announcements(admin)

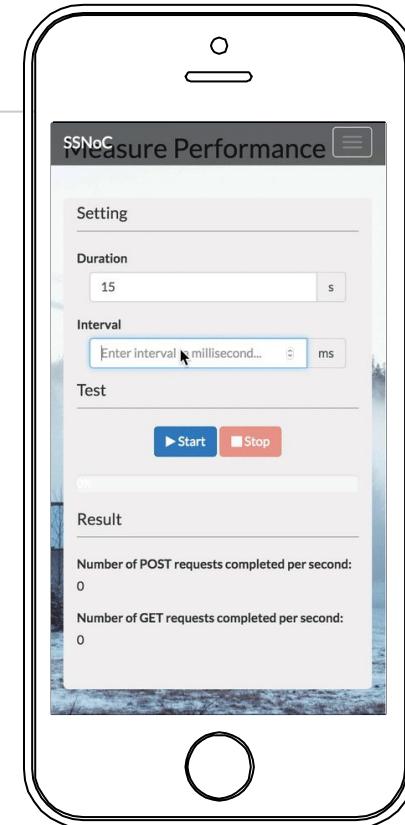
Besides viewing and searching, the administrator also have the privilege to post an announcement.





Measure Performance(admin)

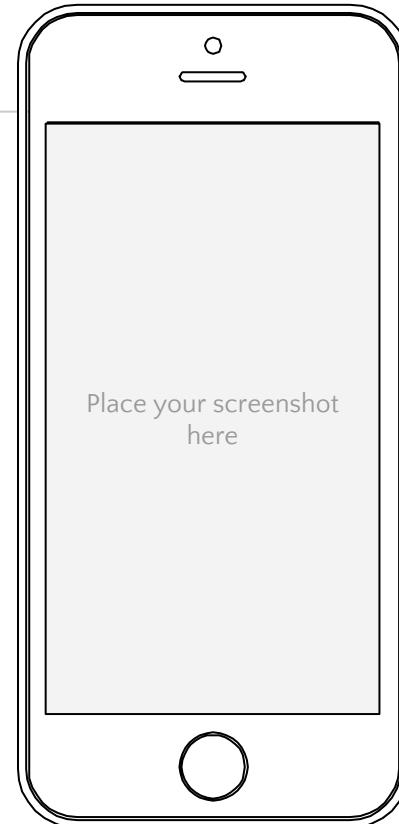
Administrators can measure the performance of the system by evaluating the running time of GET/POST requests.





Administer User Profile(admin)

Administrators can modify user profile. The user would be logged out if there is a modification on his profile.





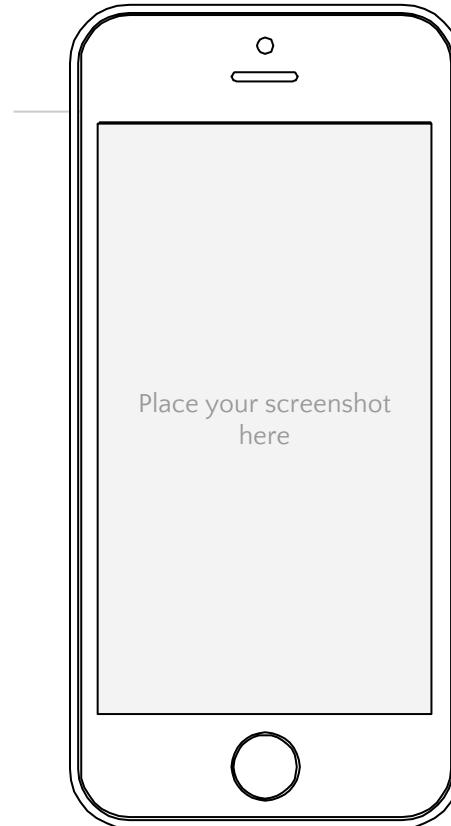
Unique Features



Map(unique feature)

Users can share their locations on the map, and also see where other users are.

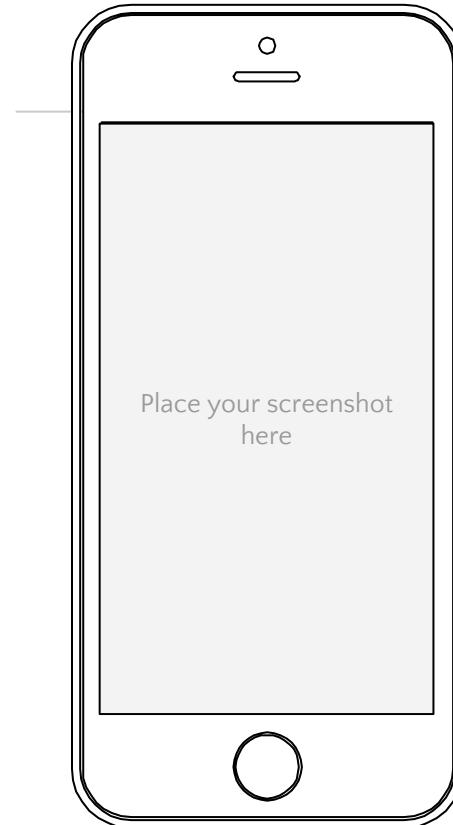
- Status: OK
- Status: Help
- Status: Emergency
- Status: Offline





Map(unique feature)

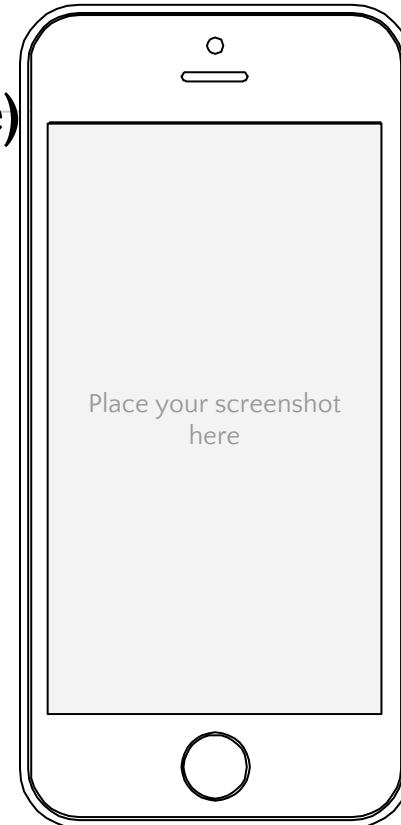
Administrators can mark the map with markers such as food, medicine, and water.





Modify Profile(individual feature)

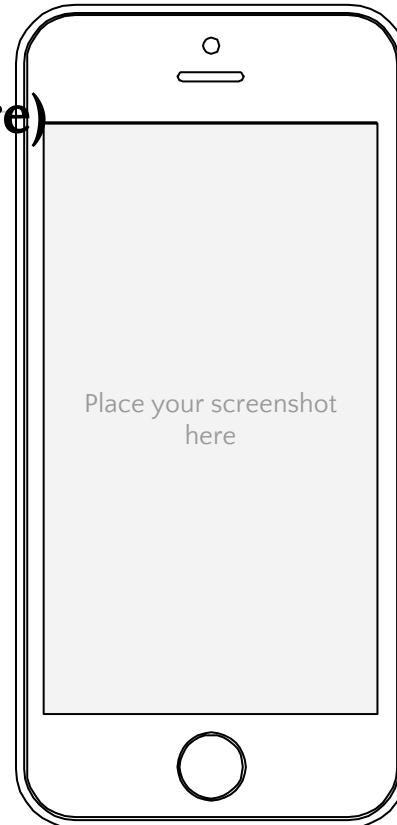
Users can modify their own profiles.





Delete Message(individual feature)

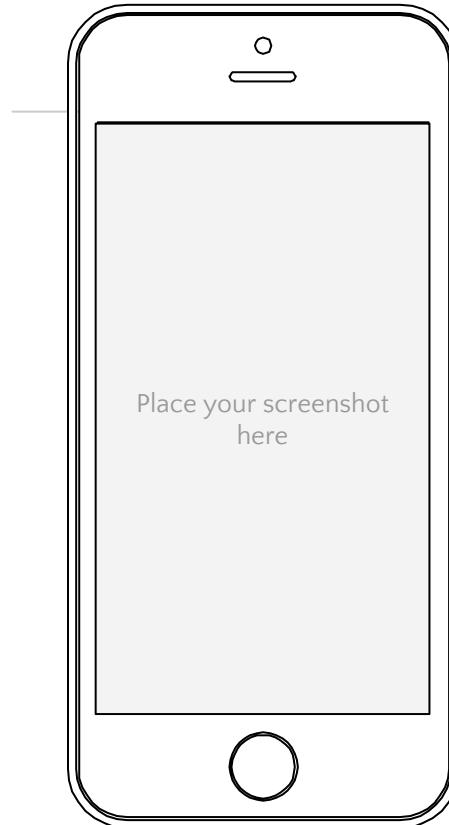
Users can delete messages in multiple ways: single, multiple, time period.





Posts(individual feature)

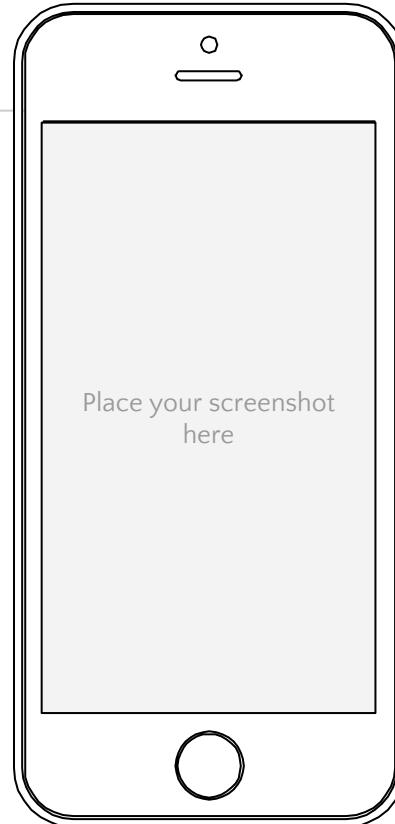
Users can send posts,
and also view posts
from other people.





Group Chat(individual feature)

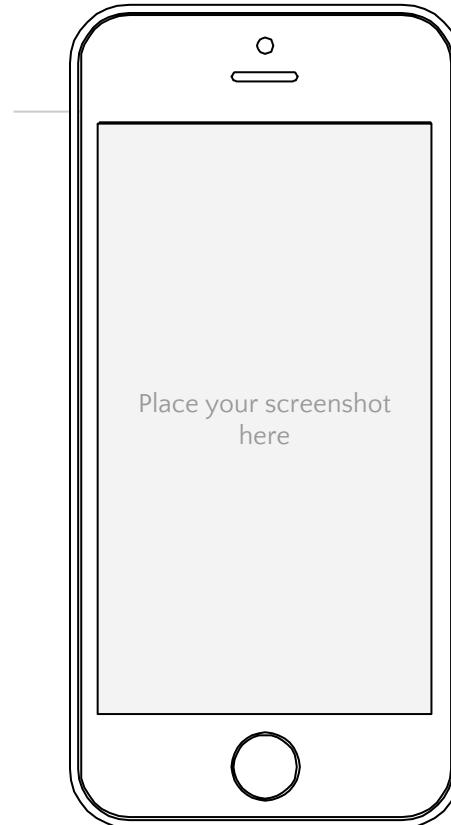
Users can invite other people into a chat group, and hold private conversations within the group.





Alerts(individual feature)

Users can receive alert, in situation of emergency. The sound of the alert is different in different situations.



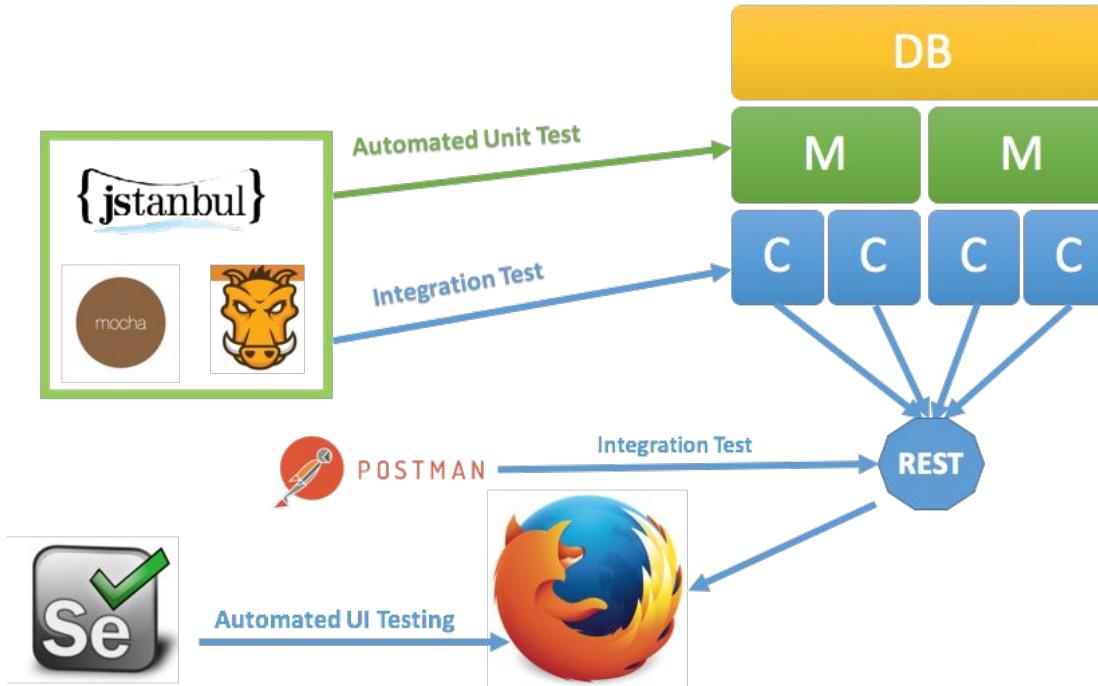


3

How to Test



Test Overview





Automated Unit Test

```
✓ Register_ExistUserIn
✓ Check_Existed_User
✓ Check_Non-Existed_User
✓ Test getUserInfo if the user exists.
✓ Test getUserInfo if the user does not exist.
✓ Test ShareStatus
✓ Test delete logged user
✓ Test get online users
✓ Test get offline users
✓ Test update a non-existed name
✓ Test update a existed user name
✓ Test get user with wrong name
✓ Test get user by wrong password
✓ Test get user by userAuth

42 passing (3s)

Writing coverage object: [/Users/guangyu/Desktop/18652/S16-A1-SSNoC/coverage/coverage.json]
Writing coverage reports at: [/Users/guangyu/Desktop/18652/S16-A1-SSNoC/coverage]

----- Coverage summary -----
Statements : 96.63% ( 459/475 )
Branches   : 81.93% ( 68/83 )
Functions   : 97.78% ( 132/135 )
Lines      : 96.66% ( 455/471 )

>> Done, Check coverage folder.

Done, without errors.
→ S16-A1-SSNoC git:(master) |
```

	SUCCESS	cmusv-fse / S16-A1-SSNoC / master #298
	add doc	
	SUCCESS	cmusv-fse / S16-A1-SSNoC / master #297
	Merge branch 'master' of https://github.com/cmusv-fse/S1...	
	SUCCESS	cmusv-fse / S16-A1-SSNoC / master #296
	unit test	
	SUCCESS	cmusv-fse / S16-A1-SSNoC / master #295
	unit test	
	SUCCESS	cmusv-fse / S16-A1-SSNoC / master #294
	unit test	
	FIXED	cmusv-fse / S16-A1-SSNoC / master #293
	boring test	
	SUCCESS	cmusv-fse / S16-A1-SSNoC / master #291
	boring test	



REST Integration Test

```
PUT localhost:3001/updateProfile Params Send Save
Authorization Headers (1) Body Pre-request Script Tests Generate Code
form-data x-www-form-urlencoded raw binary JSON (application/json)
1+ { "username" : "jiyu1", "oldUsername": "jiyu1", "accountStatus" : "inactive", "oldAccountStatus": "active", "password" : "12345", "oldPassword" : "12345", "privilege" : "Citizen", "oldPrivilege" : "Citizen"
9
10 }
```

```
POST /signin/ HTTP/1.1
Host: localhost:3001
Content-Type: application/json

{
  "username": "SSNAdmin",
  "password": "admin"
}
```

```
Body Cookies Headers (6) Tests Status: 200 OK Time: 29 ms  
Pretty Raw Preview JSON    
1+ {  
2 "statusCode": 200,  
3 "message": "Info Saved."  
4 }
```

```
Body Cookies Headers (6) Tests Status: 200 OK Time: 83 ms  
Pretty Raw Preview JSON 🔍    
1+ {  
2 "statusCode": 200,  
3 "message": "Success"  
4 }
```



Automated UI Test

The image shows a split-screen view. On the left is a screenshot of a web browser displaying a login page titled "SSNoC". The page has fields for "Username" and "Password" and a "Sign In" button. On the right is a screenshot of a test tool interface. The top part shows a list of test cases: NewUser, WrongPassword, ExistingUserJoinCommunity, PreservedUserName, InvalidPasswordLength, InvalidUserNameLength, and SigninUserSessionKeep. The "InvalidPasswordLength" case is selected. The main panel displays a table of test steps:

Command	Target	Value
open	/	
type	id=username	helloworld
type	id=password	123
click	id=signin	
assertText	css=h2	Invalid Pass...
click	css=button...	
type	id=password	1234
click	id=signin	
click	css=button...	
assertText	css=p	Would you ...

Below the table, the "Runs:" and "Failures:" counts are both 0. At the bottom, a log window shows the execution details:

```
[Info] Executing: [type | id=username | helloworld]
[Info] Executing: [type | id=password | 123]
[Info] Executing: [click | id=signin | ]
[Info] Executing: [assertText | css=h2 | Invalid Password!]
[Info] Executing: [click | css=button.confirm | ]
[Info] Executing: [type | id=password | 1234]
[Info] Executing: [click | id=signin | ]
[Info] Executing: [click | css=button.cancel | ]
[Info] Executing: [assertText | css=p | Would you want to create new account?]
[Info] Test case passed
[Info] Playing test case InvalidPasswordLength
[Info] Executing: [open | / | ]
```

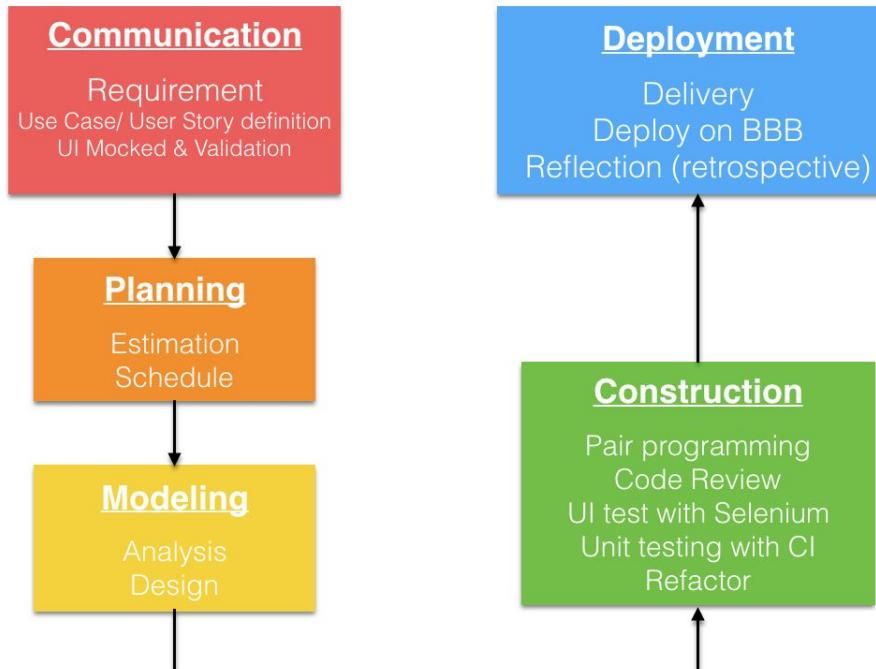


4

Practices and Quality attributes



Software Engineering Practices





Software Engineering Practices



Iteration1:Join Community [V = 3,
P=5]

13/17 uc

This block displays a post from an iteration. The title is "Iteration1:Join Community [V = 3, P=5]". Below the title is a timestamp "13/17" and a "uc" button. At the bottom are five small profile icons representing different users.

Iteration 1:Chat Publicly[V = 2 ,P = 8]

11/17

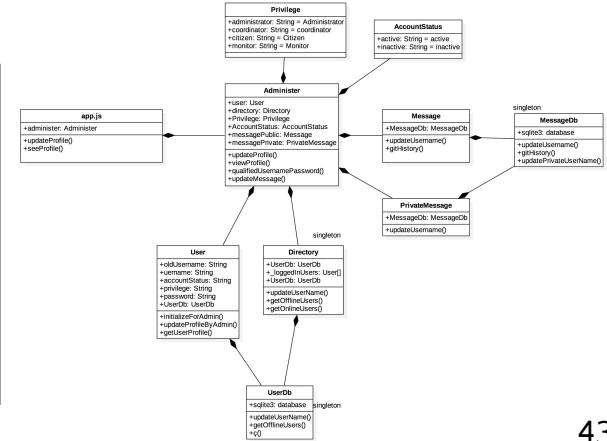
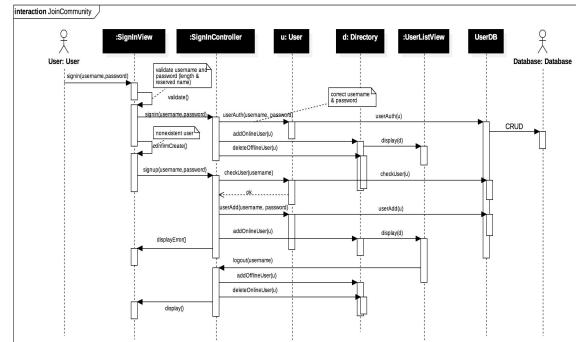
This block displays another post from the same iteration. The title is "Iteration 1:Chat Publicly[V = 2 ,P = 8]". Below the title is a timestamp "11/17". At the bottom are five small profile icons representing different users.



Software Engineering Practices

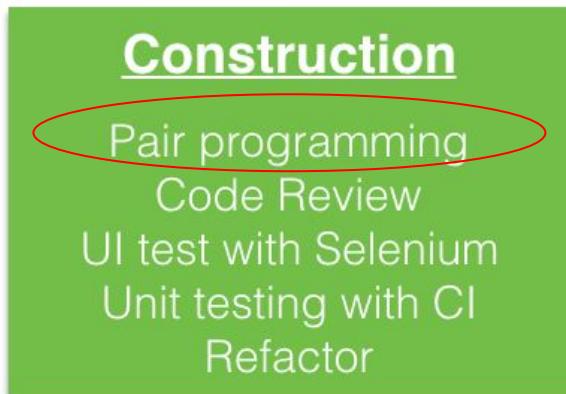
Modeling

Analysis
Design





Software Engineering Practices





Software Engineering Practices

Construction

Pair programming
Code Review
UI test with Selenium
Unit testing with CI
Refactor





Quality Attributes

External/ Runtime/ Visible

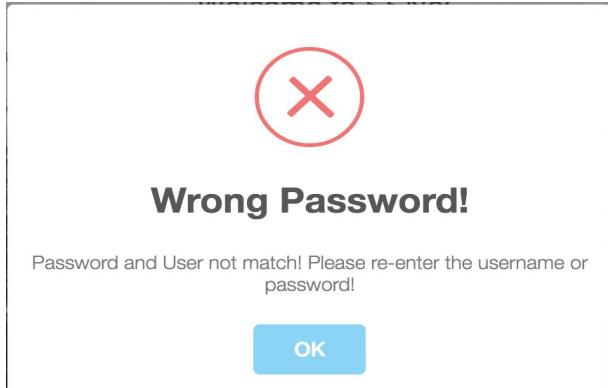
- Performance
- **Robustness**
- Reliability
- ...

Internal/No-Runtime/Invisible

- Testability
- Portability
- Reusability
- **Extensibility**
- ...



Software Engineering Practices



POST ▾ http://localhost:3001/signin

Body Cookies Headers (6) Tests

Pretty Raw Preview JSON ↗

```
1 {  
2   "statusCode": 403,  
3   "message": "Wrong password"  
4 }
```

```
$.post('/signin', {  
  username: username,  
  password: password  
},  
function(response) {  
  console.log(response);  
  if (response.statusCode === 200) {  
    window.location = "/index?username=" + username;  
  } else if (response.statusCode === 403) {  
    swal({  
      title: "Wrong Password!",  
      text: "Password and User not match! Please re-enter the username or password!",  
      type: "error",  
      confirmButtonText: "OK"  
    });  
    $password.val('');  
  } else if (response.statusCode === 405) {  
    swal({  
      title: "Invalid Username and/or Password!",  
      text: "Password and User not valid! Please re-enter the username or password!",  
      type: "error",  
      confirmButtonText: "OK"  
    });  
    $password.val('');  
  }  
});
```

Front-end: Handle “wrong password” login

```
exports.checkSignIn = function(req, res) {  
  var username = req.body.username;  
  var password = req.body.password;  
  new User().initialize(username).exist(function(result){  
    if (result == 401){  
      res.json({"statusCode":401, "message": "User not found"});  
    } else if (result == 403){  
      new User().initialize(username, password).userAuth(function(result,user) {  
        if (result == 403) {  
          res.json({"statusCode": 403, "message": "Wrong password"});  
        } else {  
          // User successfully authenticated  
        }  
      });  
    }  
  });  
};
```

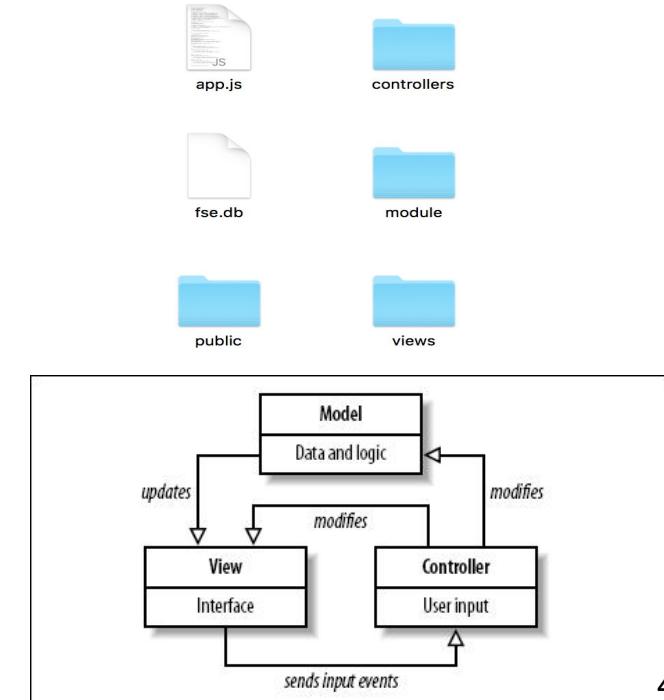
Back-end: Handle “wrong password” login



Software Engineering Practices

Separation of concerns

- Model: data, domain knowledge
- Controller: user interactions, flow
- View: display, layout, presentation, visual elements



5

Lessons Learned



Lessons Learned

- Four Most valuable lessons
 - Give TDD a Try
 - Happy Pair Programming
 - Pay Your Debt Early
 - Build You Project with MVC Architecture



Give TDD a Try

Before

- Write tests later
- No idea about the direction
- Unit tests always broke

After

- Write tests first
- Clearly where to go
- Life is better again

TDD

Pair Programming



Cheapest to fix



Happy Pair Programming

Before

- Stay at home, died alone
- Find bugs and fix them by individual

After

- Efficiency improves
- Find bugs easier
- Team work is happy





Pay Your Debt Early

Before

- Work later before the due day
- Spend much time to pay the debt

After

- Efficiency improves
- A more elegant use case
- Life is better again





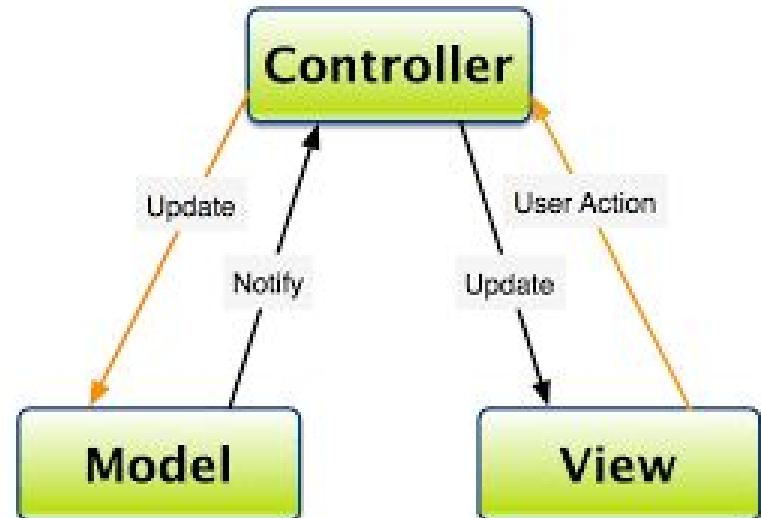
Welcome to MVC Architecture

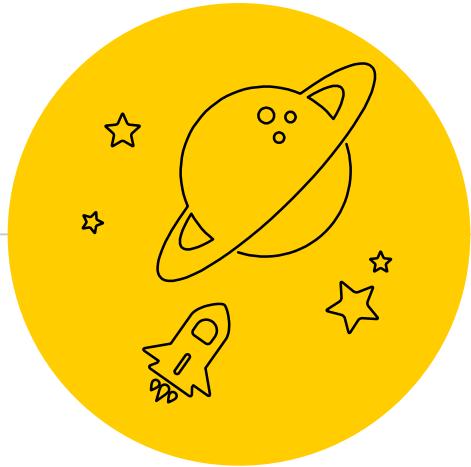
Before

- Everything is in a mess
- Hard to follow the code

After

- Low coupling
- High cohesion
- Easy to add a new user case
- Easy for a beginner to start





Thank you!