Reportal

Akshay - Celeste - Maha - Prateek
- Pranay - Rafif - Sonali

Problem Statement

Public Incident Report is a web form that is open to all Stevens students, faculty, and staff for addressing incidents, concerns across Stevens campus. This platform does not have the feature to post the picture, get exact location of the incident and is also limited with web access.

Reportal allows users to report issues along with posting image with location details via android mobile application and get timely updates on issues from the maintenance team. The maintenance team will use webportal to manage the issues.

Project Description

Reportal is an Android application that aims to assist the Stevens community by providing an easy and interactive way for people to report issues on or around campus. The application will have two main interfaces, one for the public and the other is for the maintenance team.

Who Cares aka Target Audience

- Stevens' Students
- Stevens' Staff
- Hoboken Community

Platform

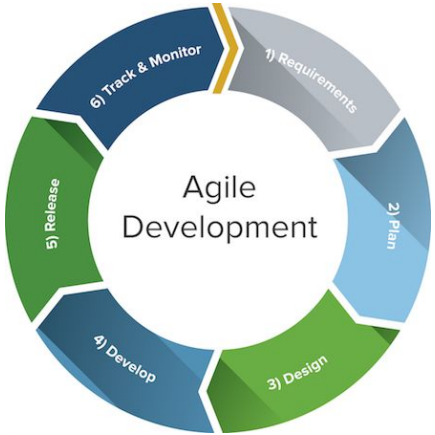
- Reportal app will be available on Android platform for users.
- The maintenance team can access the admin portal via any browser.

Development Plan and Process

- Project method used: Agile
- Project management framework: Scrum
- Communication tools used: Whatsapp & Gmail
- Code collaboration tool used: Github
- Weekly scrum/stand up meetings
- Code reading and Pair programming methods used to review code and test cases

Development Process

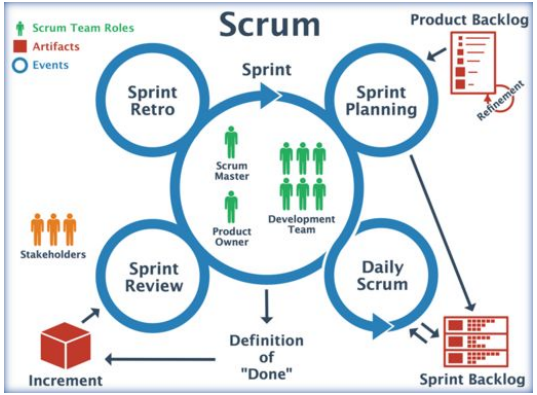
- Team used Agile methods while developing this project.
- Continuous integration practice was followed and Github was used for code collaboration.
- Team used the Feature driven development process to build the application.
- Weekly scrum/stand up meetings were held to track the progress.



Communication Plan

Weekly scrum/stand up meetings

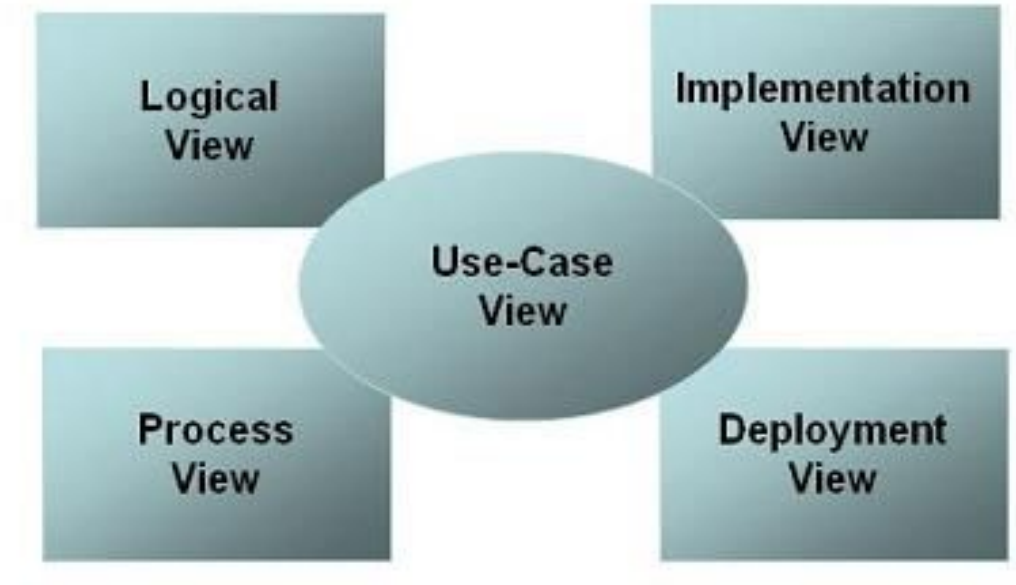
- Scrum meetings took place every week at Altorfer or any other place depending on availability.
- Each member shared the progress and the tasks at hand with the team.
- The target and goals for the next scrum were decided and agreed upon in the meeting.
- Team members worked after the meetings and collaborated together to review tasks and resolve issues.



Documents

- Project plan document
- Product backlog
- API document

Architecture - 4+1 view



Use-Case View

Scenario

1- User Registers

2- User logs in

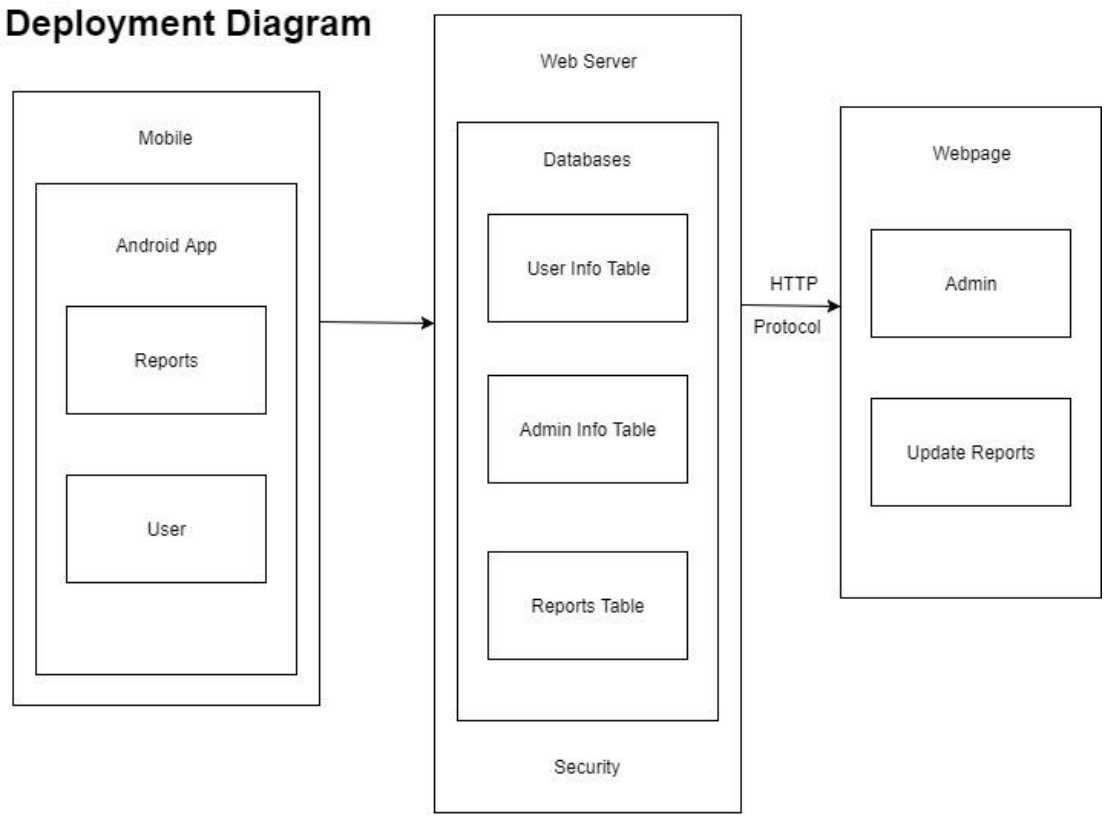
3- User posts an Issue

4- User views all posts

5- User views their post history

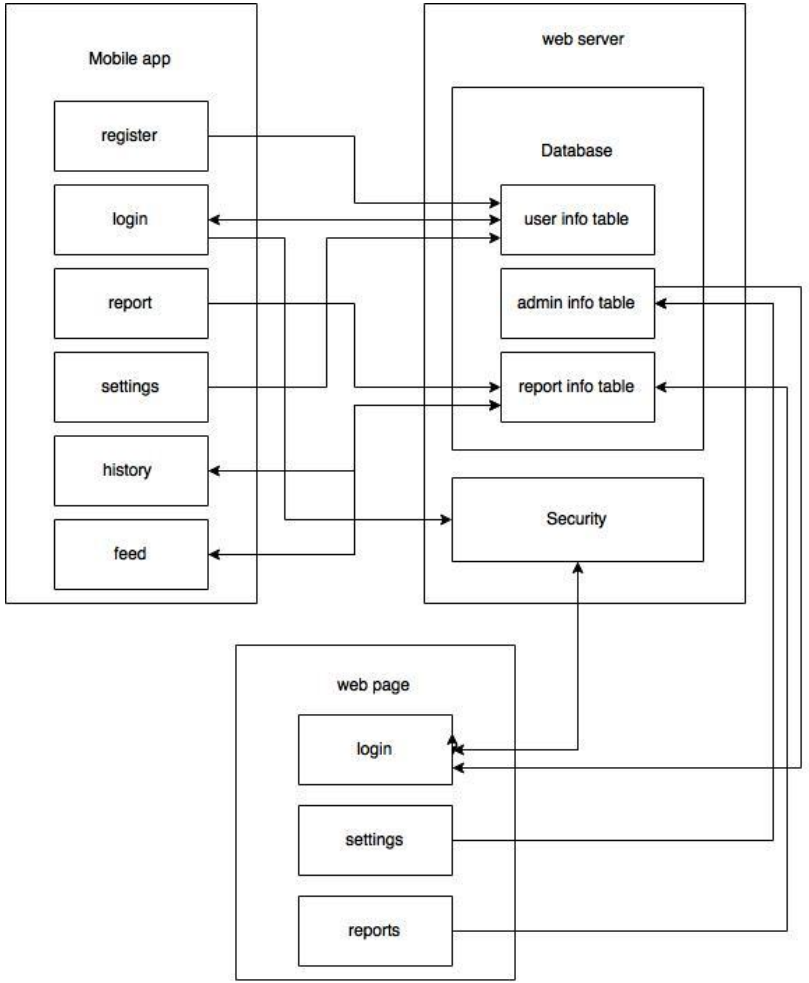
Physical View

Deployment Diagram



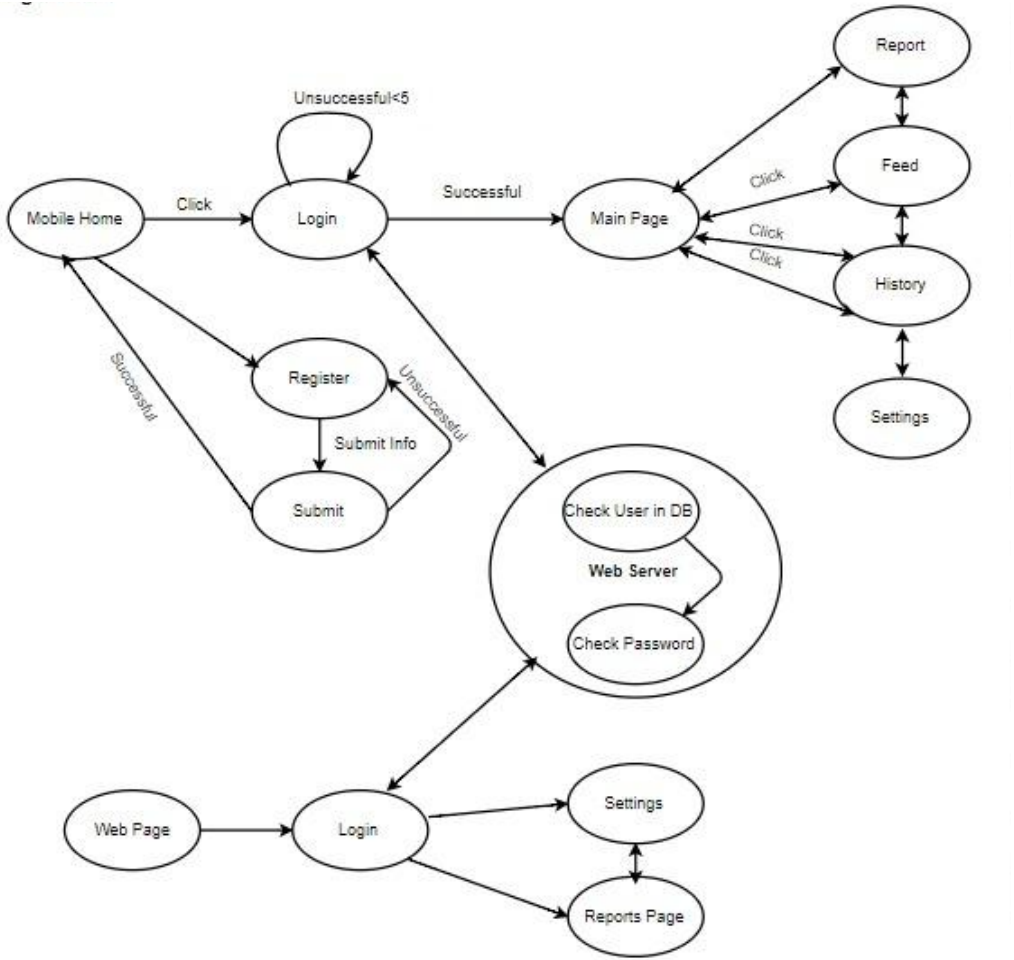
Development View

Component Diagram



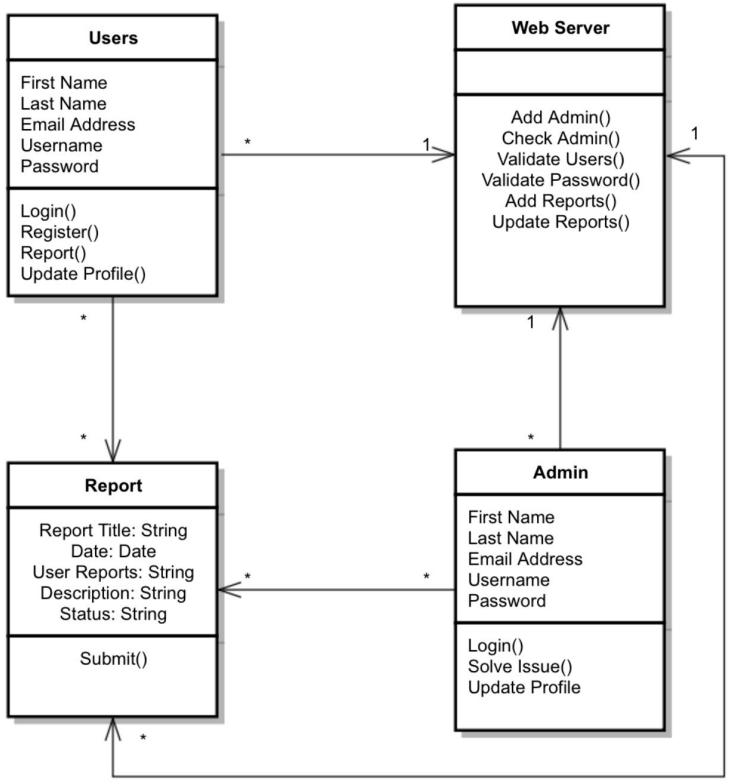
Logical View

State Diagram



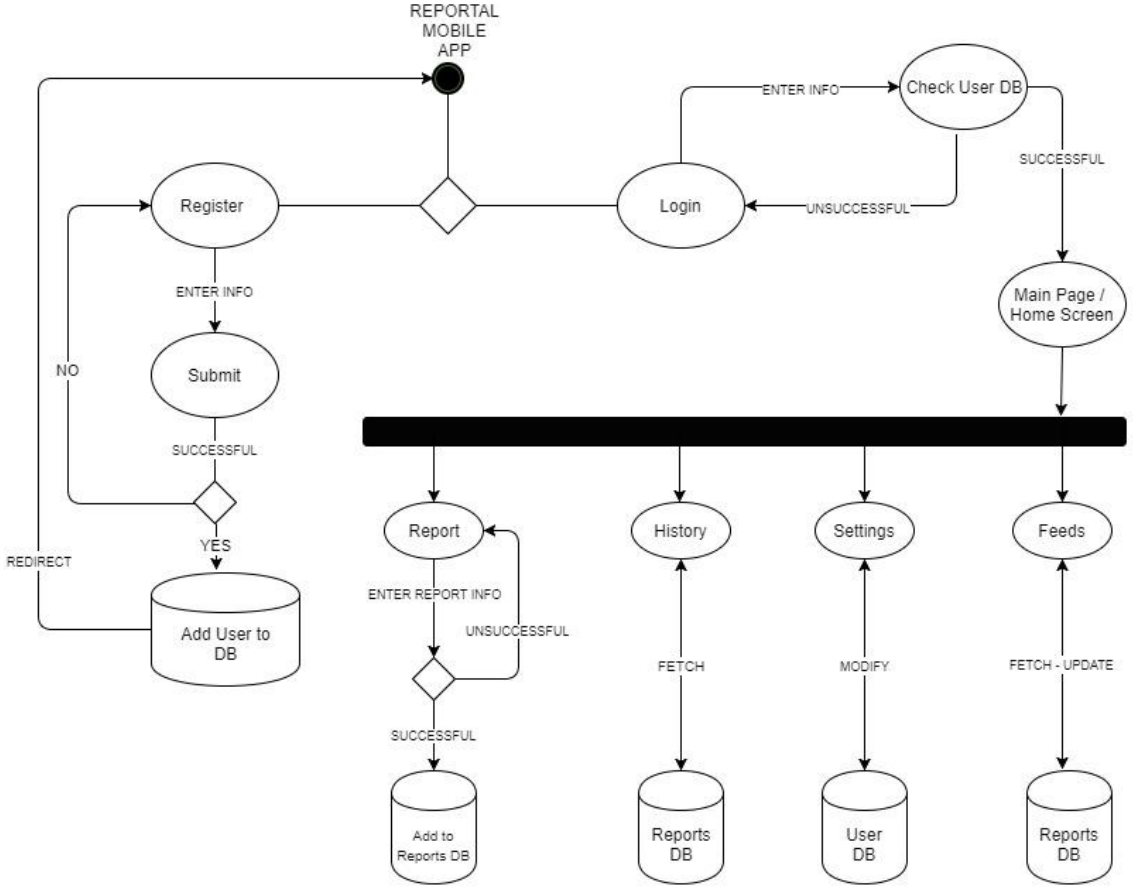
Logical View

Class Diagram



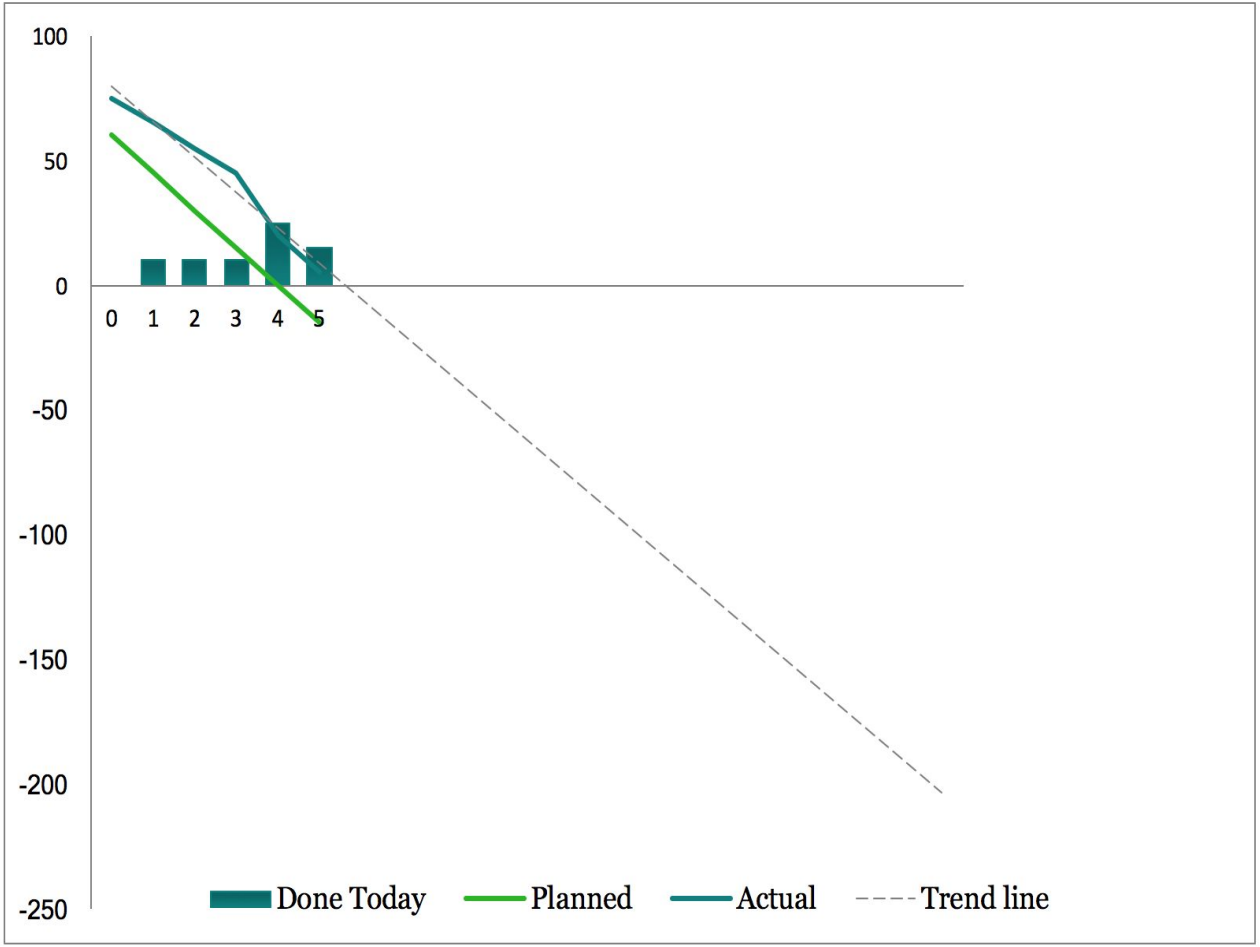
Process View

Activity Diagram



Burn Down Chart

Sprint	Burned down (User stories) Planned	Actual	Balance (User Stories) Planned	Actual	Done Today
0			60	75	N/A
1	15	10	45	65	10
2	15	10	30	55	10
3	15	10	15	45	10
4	15	25	0	20	25
5	15	15	-15	5	15



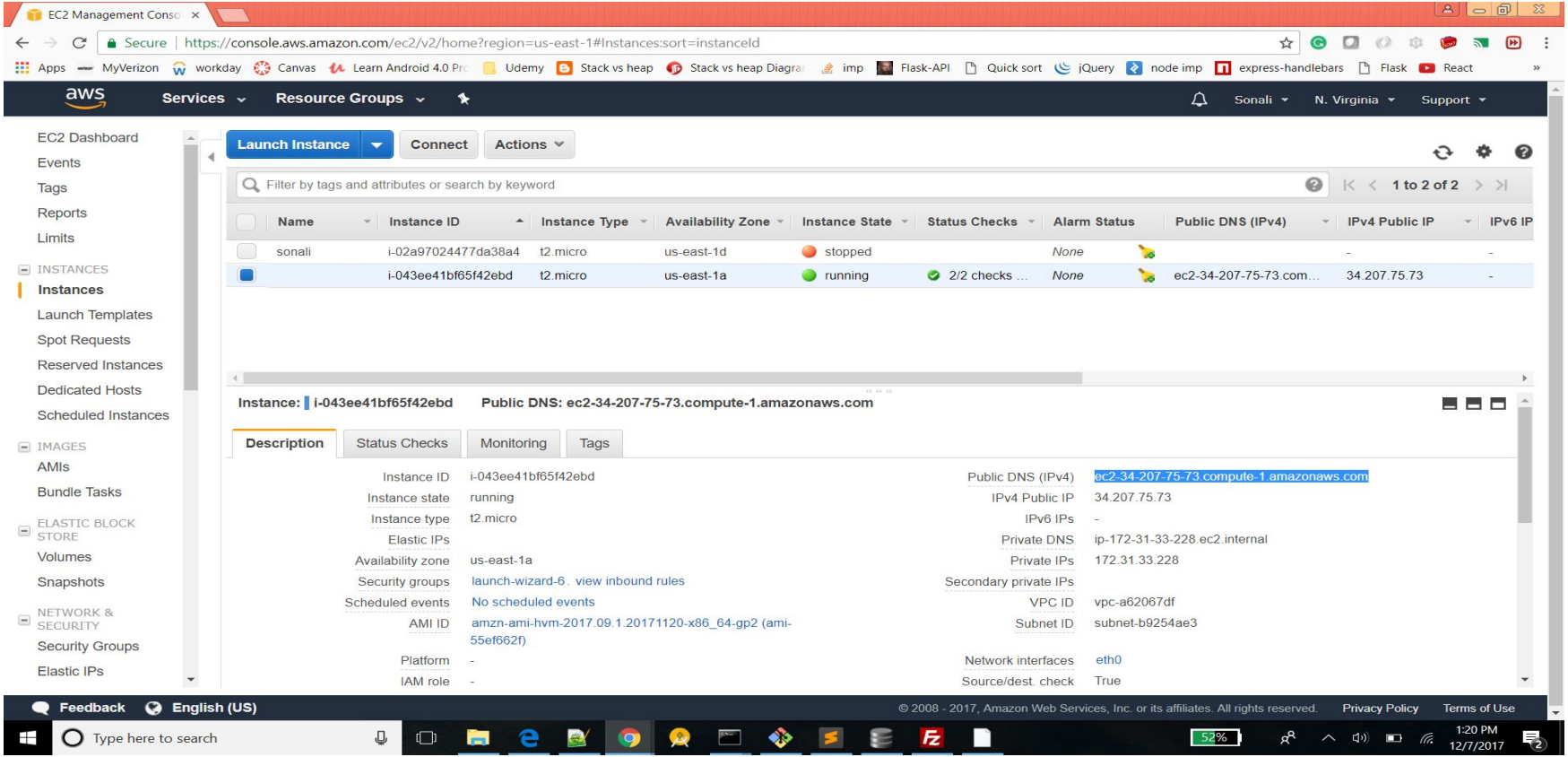
Web Services

Pranay - Sonali

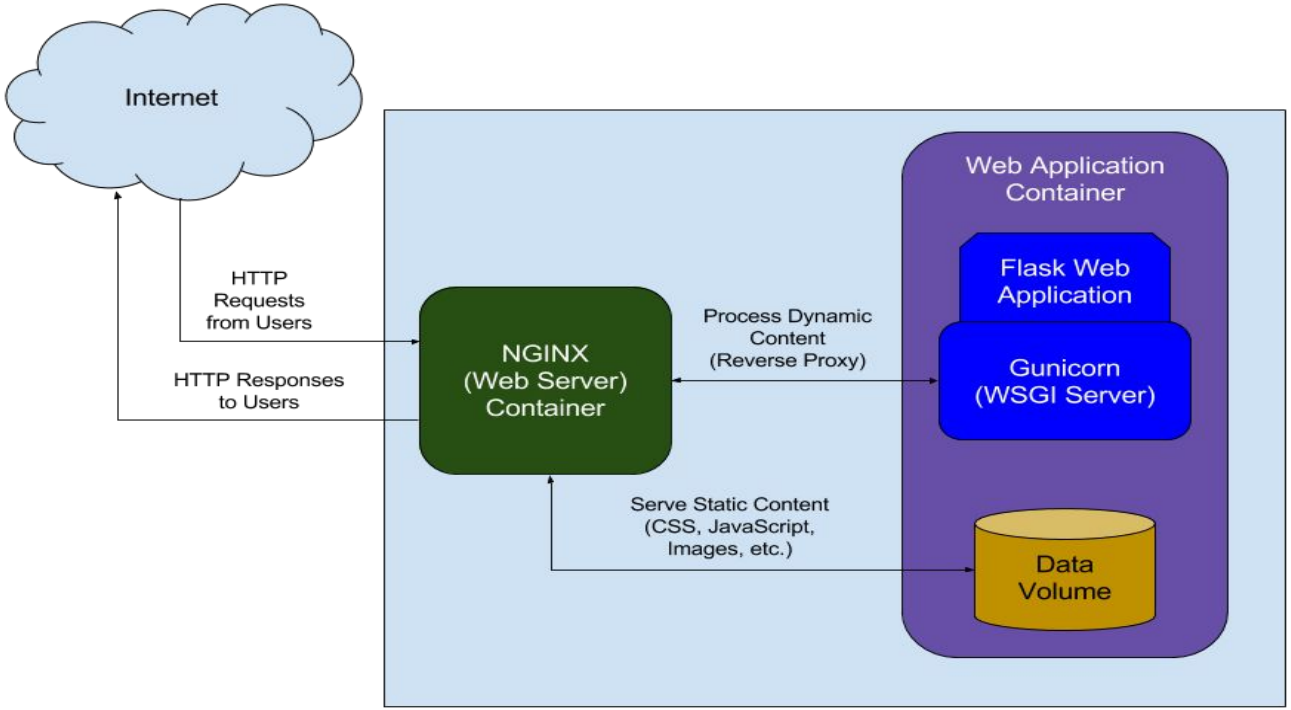
Web Services - Technology Used

- **Programming Language:** Python
- **Web Framework:** Flask, Flask_sqlalchemy
- **API Testing:** Postman
- **File Transfer:** Filezilla
- **Cloud Provider:** AWS
- **Web Server:** Nginx
- **Application Server:** Gunicorn

AWS EC2 Instance Console



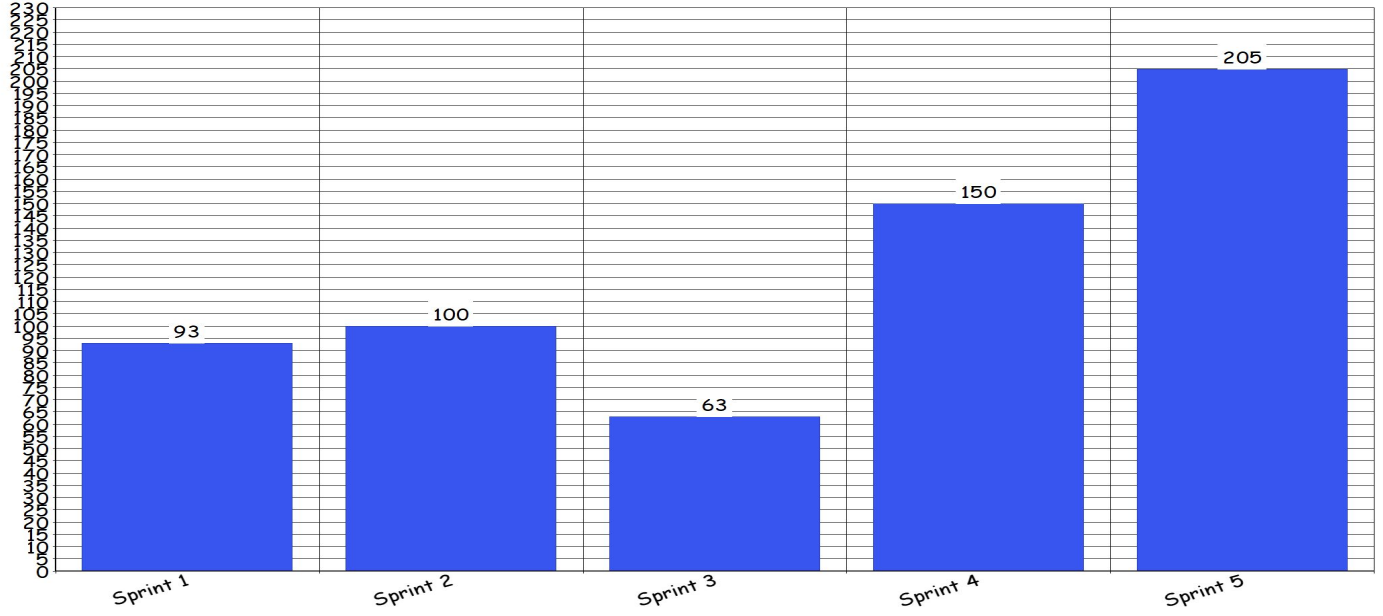
Server Architecture



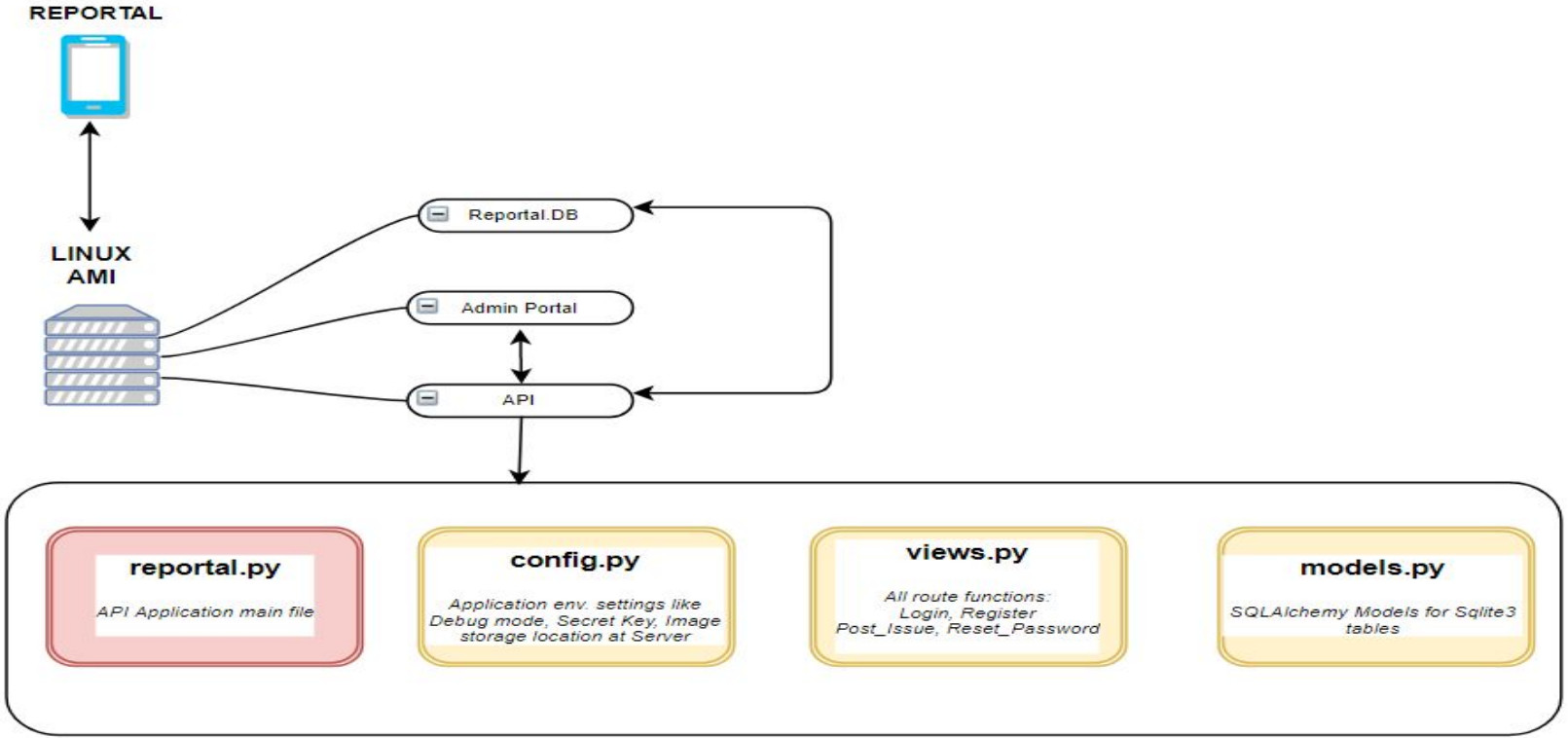
Web Services

SPRINTS	API Endpoints
Sprint 1	(1) /api/login (2) /api/register
Sprint 2	(3)/api/update_user_settings (4) /api/get_user_list/<int:page> (5) /api/get_user/<int:user_id> (6) /api/priority (7) /api/category
Sprint 3	(8) /api/logs (9) /api/get_notification/<int:user_id> (10) /api/delete_notification
Sprint 4	(11) /api/get_priorities (12) /api/get_categories (13) /api/post_issue (14) /api/forgot_password
Sprint 5	(15) /api/get_issue (16) /api/static/img/<string:folderName>/<string:filename> (17) /api/reset_password/<string:token> (18) /api/get_issue_list/<int:assignedToID> (19) /api/update_issue

Web Services LOC/Sprint



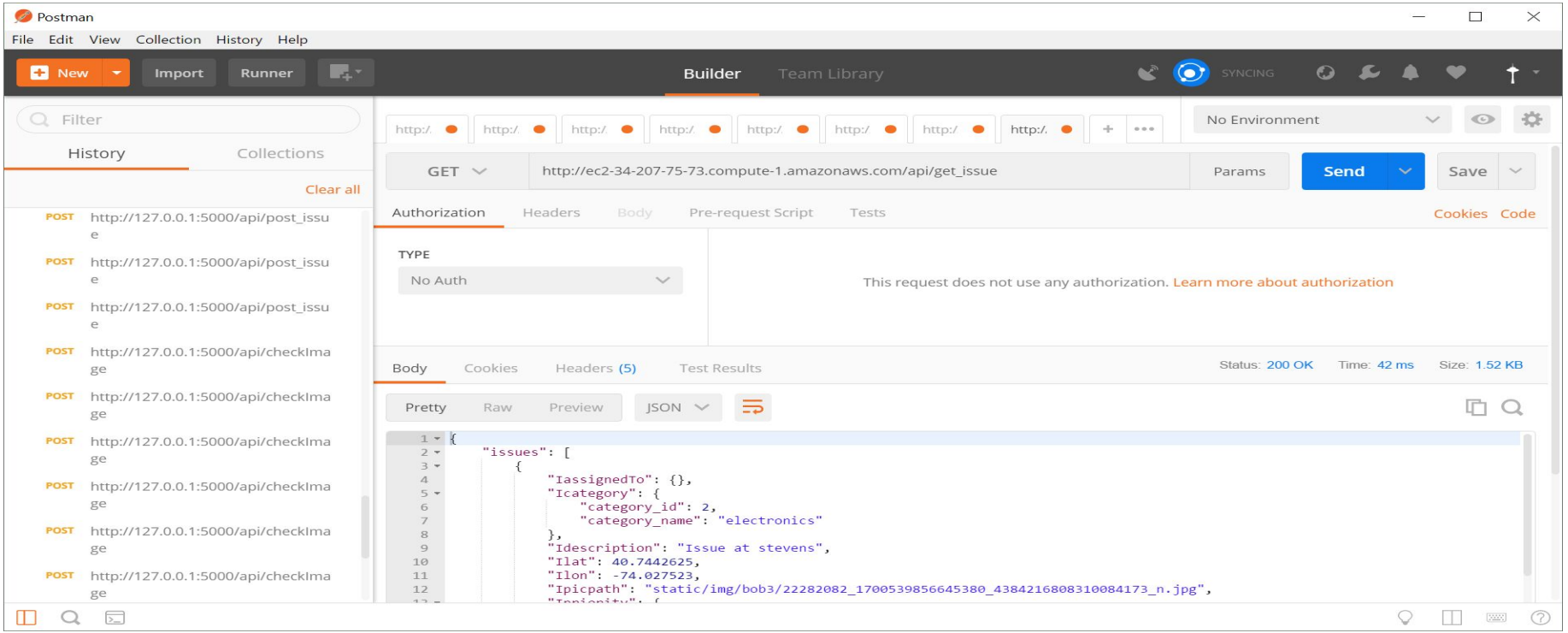
Web Services Structure



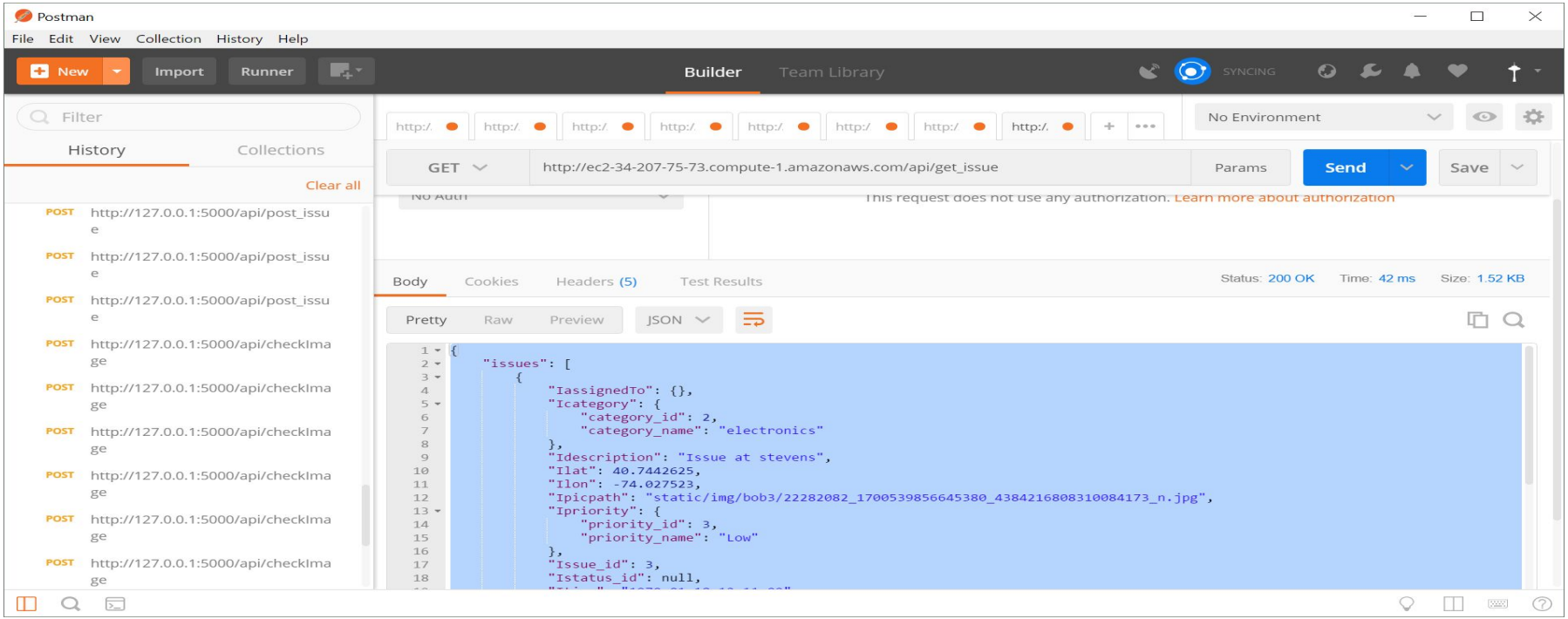
Function - Wise LLOC

Functions	Request Type	Logical LOC
login, register	POST	93
category, priority, user_update_settings	POST, PUT	100
logs, get_notification, delete_notification	GET, DELETE	63
post_issue, get_issue_list, get_categories, get_priorities	GET, POST	150
update_issue, get_issue, forgot_password, getPic	GET, POST	205

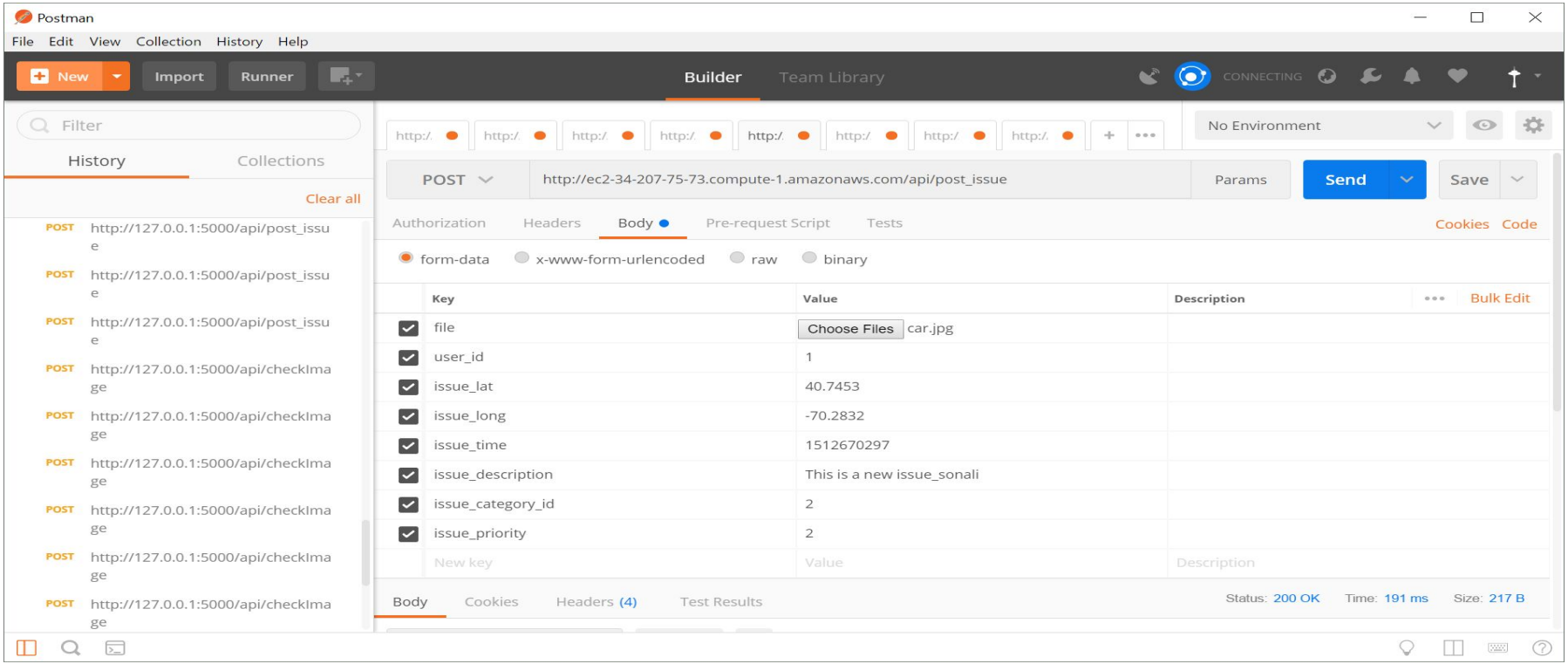
POSTMAN Web Service Testing - Get Issue



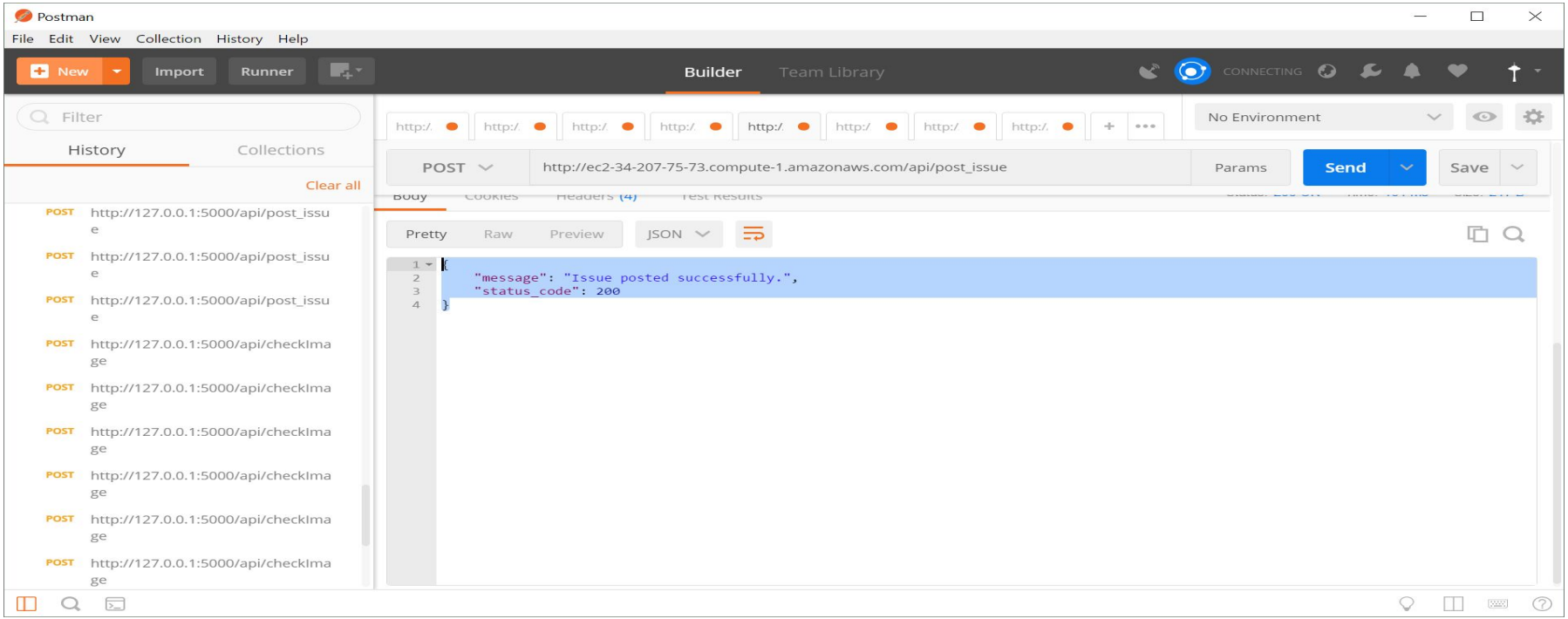
POSTMAN Web Service Testing - Get Issue contd..



POSTMAN Web Service Testing - Post Issue



POSTMAN Web Service Testing - Post Issue



Android

Celeste - Prateek

Android - Technology Used

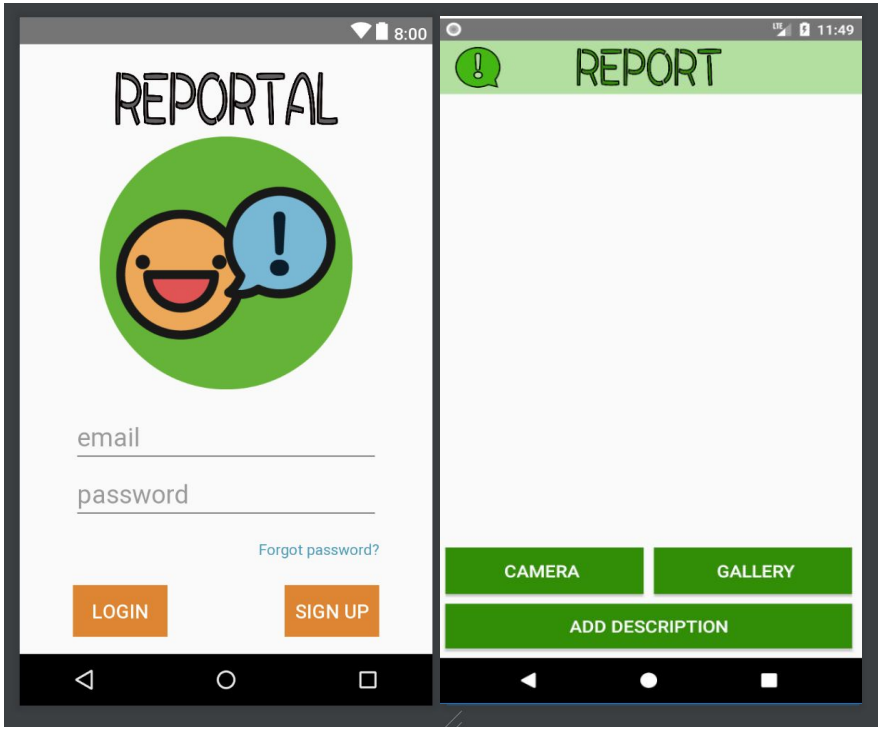
Reportal android application

Android Studio

- **Programming Languages:** xml, kotlin
- Overview of design
- Checking credentials
- Connecting to server
- Saving data

Android - Feature List

- Login
- Sign up
- Forgot password
- Post issue
- View history
- View Feed
- Update settings



Reportal application design and development

Languages used:

- Kotlin
- XML

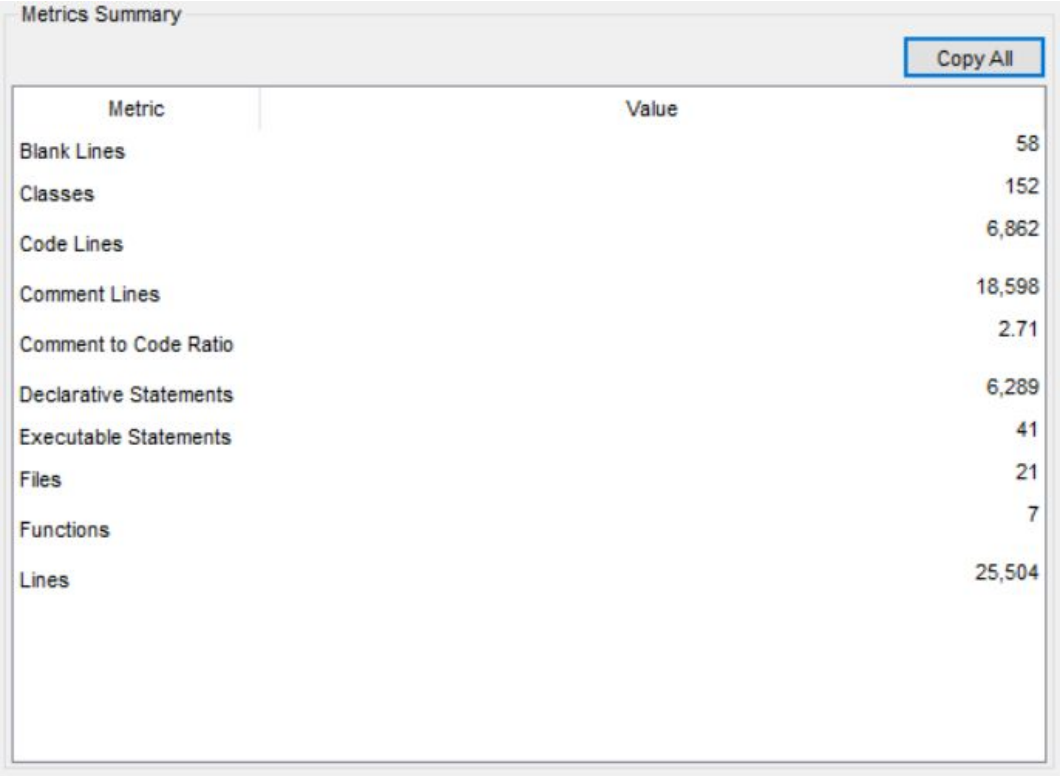
Development weeks: 10

Development hours per week: 8

Total LOC: 6,862

Build Rate: 85 lines of code per hour

Android application code metrics

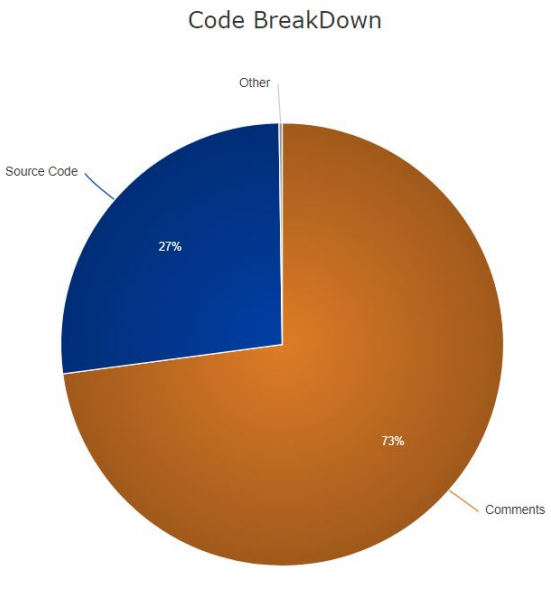


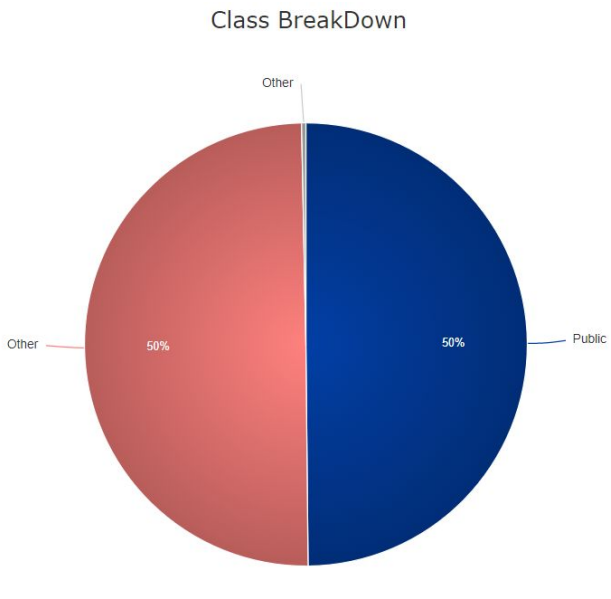
Metrics Summary

Copy All

Metric	Value
Blank Lines	58
Classes	152
Code Lines	6,862
Comment Lines	18,598
Comment to Code Ratio	2.71
Declarative Statements	6,289
Executable Statements	41
Files	21
Functions	7
Lines	25,504

Android application code metric graphs

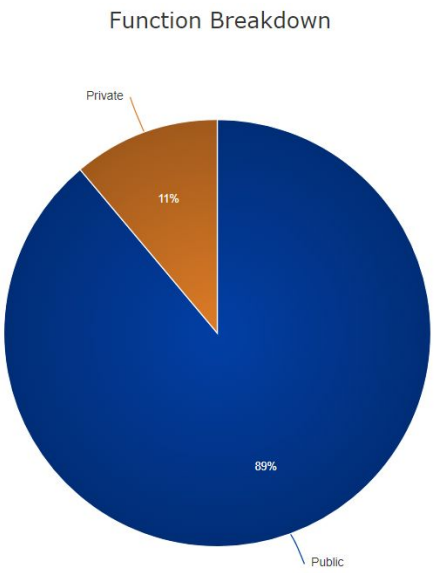


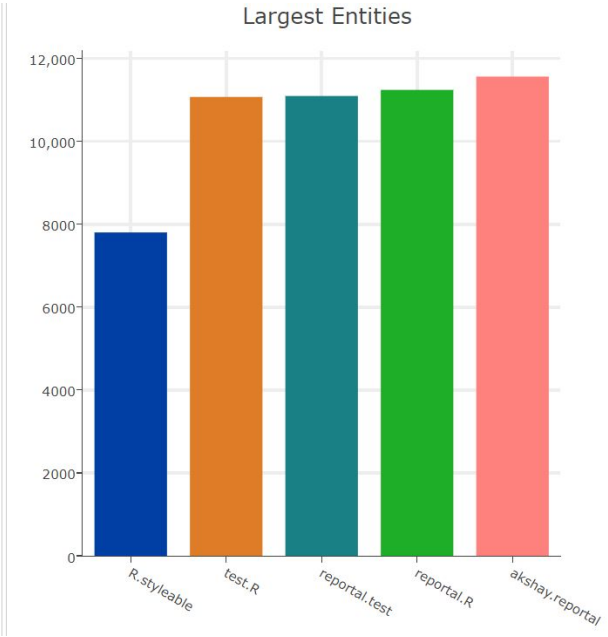


Project Metrics

Files:	21
Program Units:	7
Lines:	25504
Blank Lines:	58
Code Lines:	6862
Comment Lines:	18598
Statements:	41

Android application code metric graphs





Reportal application testing

Test Type	Description
Functional Test	Covers application functionality in a basic fashion
Stress Test	Stresses functionality through extreme repetition, depth, or variation
Unit Test	Validates an application component feature; white-box test
Performance Test	Tests performance issues

Reportal intent sample test cases

Test case	Description
1	Check if the Camera button responds after the user clicks on it.
2	Test the camera functioning of the device by checking if the camera is able to display the picture.
3	Test preview functionality by checking the preview image.
4	Test if the user is able to capture the desired image once the image capture button is pressed.
5	Check if the user is able to recapture the image by testing the revert button functionality in the camera.
6	Check if the user is able to cancel the image captured by testing the cancel button functionality in the camera.

Reportal application camera intent testing

- Unit Testing performed using jUnit.

```
1 class MyTest {
2
3     @Test
4     fun testsWork() {
5         assertTrue(true)
6     }
7 }
```

CMS

Maha - Rafif

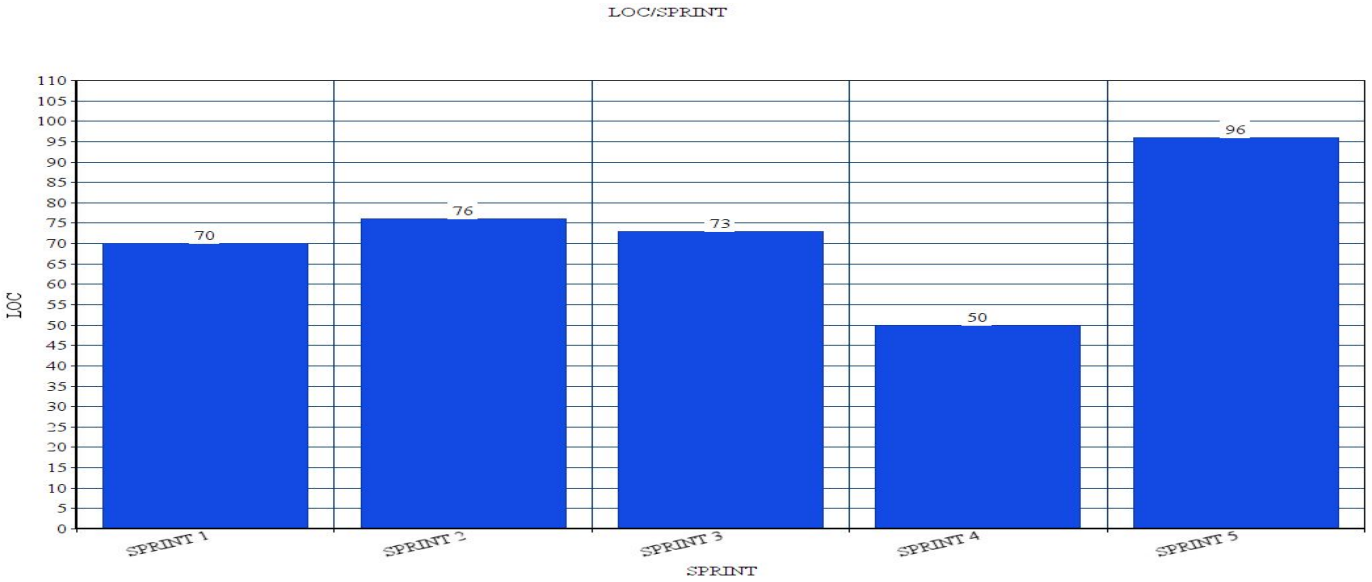
Admin portal - Technology used

- Python
- JavaScript
- HTML - CSS
- Flask
- Jinja
- Bootstrap

Admin portal - Feature List

- Login
- View (users, issues, categories, priorities)
- update(issues, categories, priorities)
- View profile
- Update profile

ADMIN PORTAL - LOC/ sprint



ADMIN PORTAL - USABILITY TESTING

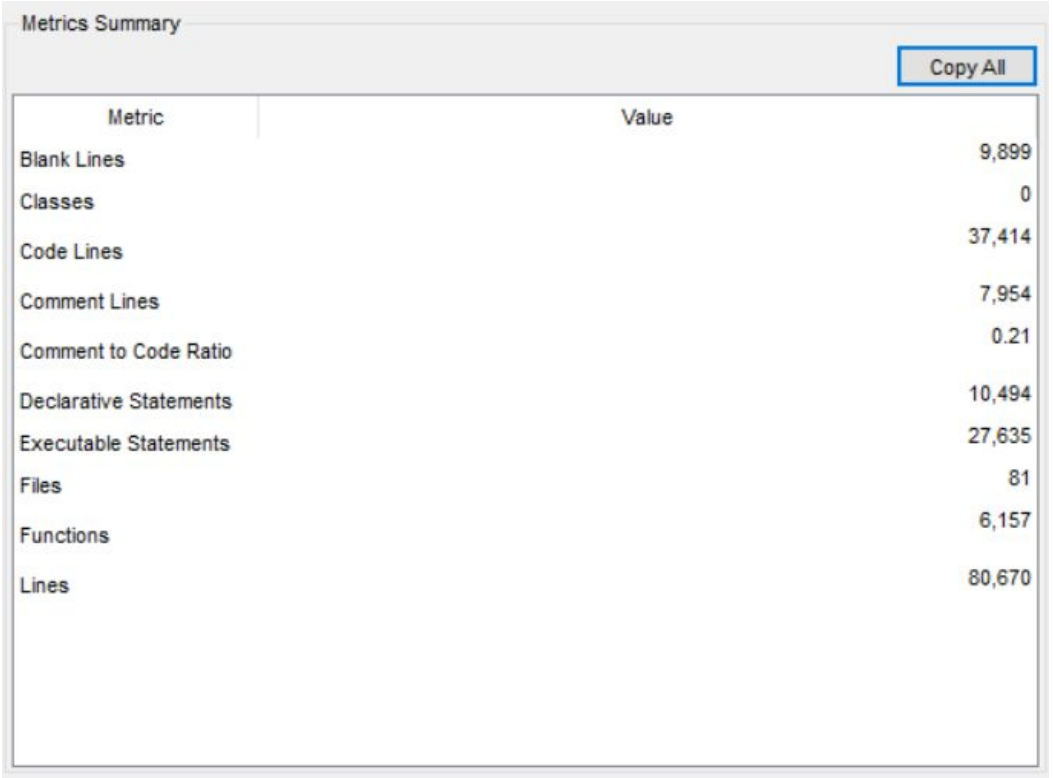
We have chosen 3 different subjects with an average on computer savvy 4 out of 5 to perform one scenario task:

- Log in to Reportal
- View and update Profile

We have gathered some system usability scale (SUS) using JOHN BROOK scale. As a result, interpreted the Individual SUS score in order to analyze the measurement of our system. The result was successful as our score is above 80%.

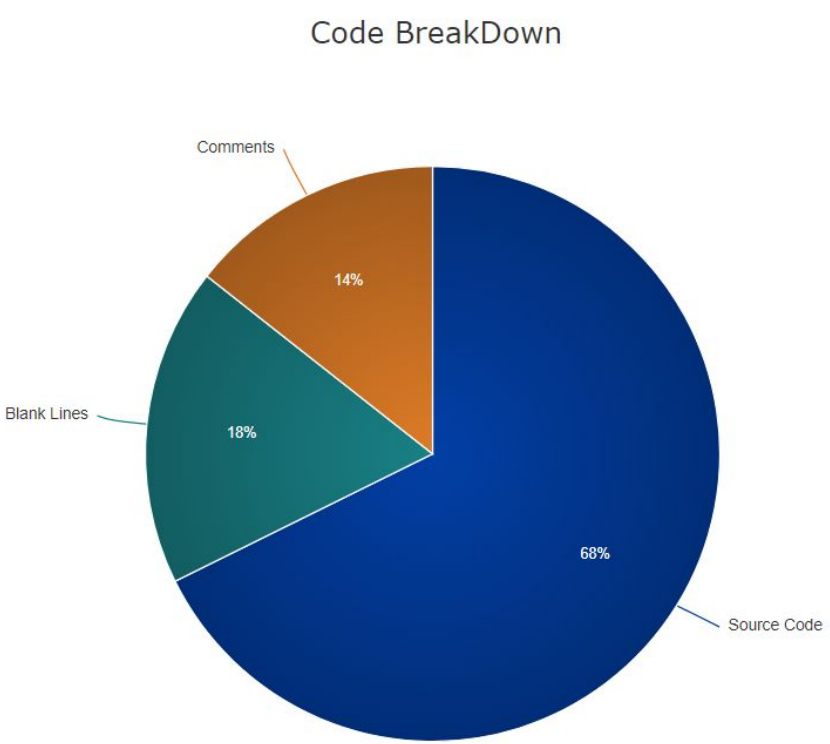
SYSTEM USABILITY SCALE (JOHN BROOKE)	A	B	C
I think I would like to use this system frequently	5	4	4
I found the system unnecessarily complex	1	2	1
I thought the system was easy to use	5	5	4
I think that I would need the support of a technical person to be able to use this system	1	1	1
I found the various functions in the system were well integrated	5	5	5
I thought there was too much inconsistency in this system	1	1	2
I would imagine that most people would learn to use this system very quickly	4	5	4
I found the system very cumbersome to use	1	2	1
I felt very confident using the system	5	5	5
I needed to learn a lot of things before I could get going with this system	1	1	1
Individual SUS Score	**29**	**31**	**28**
Average SUS Score	29+31+28/3 = **29.33**		

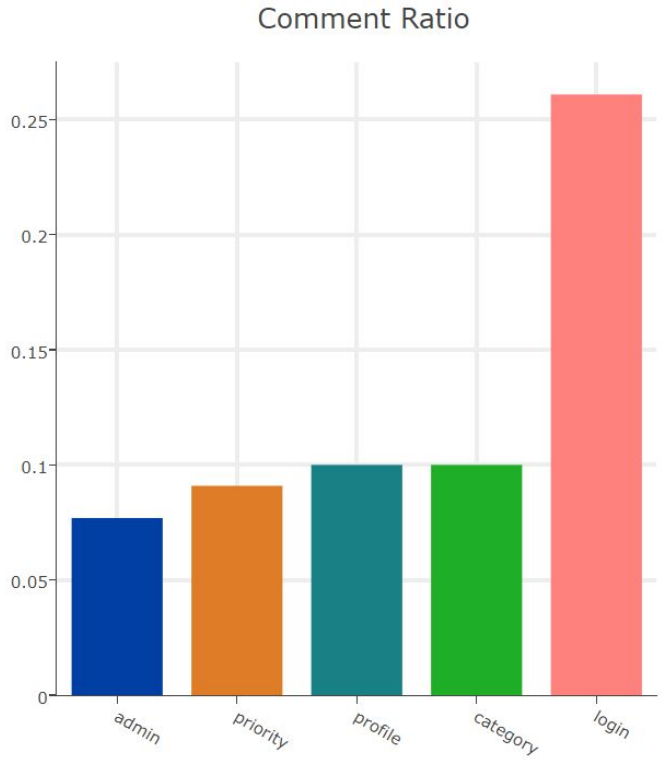
ADMIN PORTAL METRICS



Metric	Value
Blank Lines	9,899
Classes	0
Code Lines	37,414
Comment Lines	7,954
Comment to Code Ratio	0.21
Declarative Statements	10,494
Executable Statements	27,635
Files	81
Functions	6,157
Lines	80,670

ADMIN PORTAL GRAPHS





ADMIN PORTAL - TOTAL LOC

```
host-b430-19:Webportal Mahalidrisi$ radon raw __init__.py
__init__.py
    LOC: 436
    LLOC: 350
    SLOC: 365
    Comments: 28
    Single comments: 23
    Multi: 0
    Blank: 48
    - Comment Stats
        (C % L): 6%
        (C % S): 8%
        (C + M % L): 6%
```

ADMIN PORTAL - CYCLOMATIC COMPLEXITY

```
host-b430-19:Webportal Mahalidrisi$ radon cc -s -a __init__.py
__init__.py
    F 363:0 updatePriority - A (5)
    F 102:0 addUser - A (4)
    F 134:0 updateUser - A (4)
    F 176:0 addAdmin - A (4)
    F 259:0 addCategory - A (4)
    F 287:0 updateCategory - A (4)
    F 333:0 addPriority - A (4)
    F 53:0 login - A (3)
    F 23:0 connection - A (2)
    F 85:0 users - A (2)
    F 159:0 admin - A (2)
    F 211:0 issues - A (2)
    F 226:0 updateIssue - A (2)
    F 243:0 category - A (2)
    F 317:0 priority - A (2)
    F 397:0 maintenance - A (2)
    F 33:0 page_not_found - A (1)
    F 37:0 method_not_found - A (1)
    F 42:0 login_required - A (1)
    F 77:0 logout - A (1)
    F 415:0 dashboard - A (1)
    F 422:0 blankpage - A (1)
    F 429:0 timectime - A (1)

23 blocks (classes, functions, methods) analyzed.
Average complexity: A (2.39130434783)
host-b430-19:Webportal Mahalidrisi$
```

What went well?

- Learnt and explored new technologies, softwares and programming languages.
- Regular meetings and communication helped in resolving issues.
- Improved team-player and collaborative skills.

What went bad?

- Integration
- Estimate user stories
- Estimate story points
- Slow start
- Irregular meetings
- Amazon instance issue

Things to improve

- Continuous integration.
- Code collaboration.
- Decision making and tasks organization.

Code Repository

- The full code repository can be found on Github.

  Github link: https://github.com/akshya672222/SSW695_Team7

Next Version

- The application has a certain potential to go live and used by the Stevens' and Hoboken community. In order to make the application operational, a certain amount of investment and time would be required to make it a success.
- Current version of the app has not been scaled with a lot of users. Also, to make this application operational, the team would require to make it available for download on Play stores.

Next Version

- The next version of this application will also include an iOS version to cater apple products and users along with android.
- Reportal 2.0 will also include some additional features like auto detecting of location, contact services directly, have an emergency alert button, sharing the issue on social media etc.
- The application will also decrease bugs/issues faced in real time by users in the next version.

Future Work

- The future scope for the application is immense.
- Number of functionalities and features can be added to the app.
- If given an opportunity, the team will surely like to work and add more features like auto-detecting the location, sharing the issue on social media etc.
- The team highly recommends to take up and continue this amazing project.

Special Thanks to professor Vesonder

Thank you

Questions?