GENESIS - Learning Outcome & Mini-project Summary Report

LTTS
GLOBAL
ENGINEERING
ACADEMY

L&T Technology Services

Details

Ver. Rel. No.	Release Date	Prepared. By	Reviewed By	To be Approved	Remarks/Revision Details
1	11-11-2020	Amit Das			

Contents

TABLE OF FIGURES

Miniproject -1 [Individual] : HOUSE RENT PORTAL

Modules

JAVA, Maven and javaDoc

Topic and Subtopics

The project is a multiclass program that makes use of objects to interact within themselves to present a house rent portal system that runs in the terminal or console. The sensitive data is hidden inside private classes and made inaccessible to the outside environment. The program is modular and almost every component can be individually tweaked without affecting the other as long as the arguments and return types conform to the program needs. The major topics covered in the project are as follows

- Object Oriented Programming
- Abstract Classes
- Implementation of abstract classes
- Default methods
- Inheritance
- File operations
- Concrete and Instance methods
- Callback functions

Objectives & Requirements

Objective

The objective is to develop a house rent portal system where the Owners can register and list their properties and the prospective tenants can register and search for available houses for rent based upon their search criteria.

Requirements

HIGH LEVEL REQUIREMENTS

ID	Requirement
HH_01	The portal should serve both admin and users.
HH_02	Admin should have access to entire database but privilege should be restricted.
HH_03	User portal should offer both owners and tenants to use the facility.
HH_04	Owners should be able to register and list their properties.
HH_05	Owners should be able to login to review their entered details.
HH_06	Tenants should be able to register and subsequently login.
HH_07	Tenants should be able to search for available properties based on rent and city.

LOW LEVEL REQUIREMENTS

ID	Requirement
HH_01_LL_01	Have a common class to choose between user and admin entry.
HH_01_LL_02	Call the adminObject or userObject as user the user input choice.
HH_02_LL_01	Admin privileges are password protected. Private class takes password as input from admin and return Boolean
HH_03_LL_01	Common interface for both owners and tenants to register or login
HH_04_LL_01	File Writer class used to save owner registration and property listed data.
HH_05_LL_01	File Reader class used to read owner data in case of successful login.
HH_06_LL_01	File Writer class used to save tenant details into file.
HH_07_LL_01	Recursively search after fetching data from file using Buffered Reader.

Design

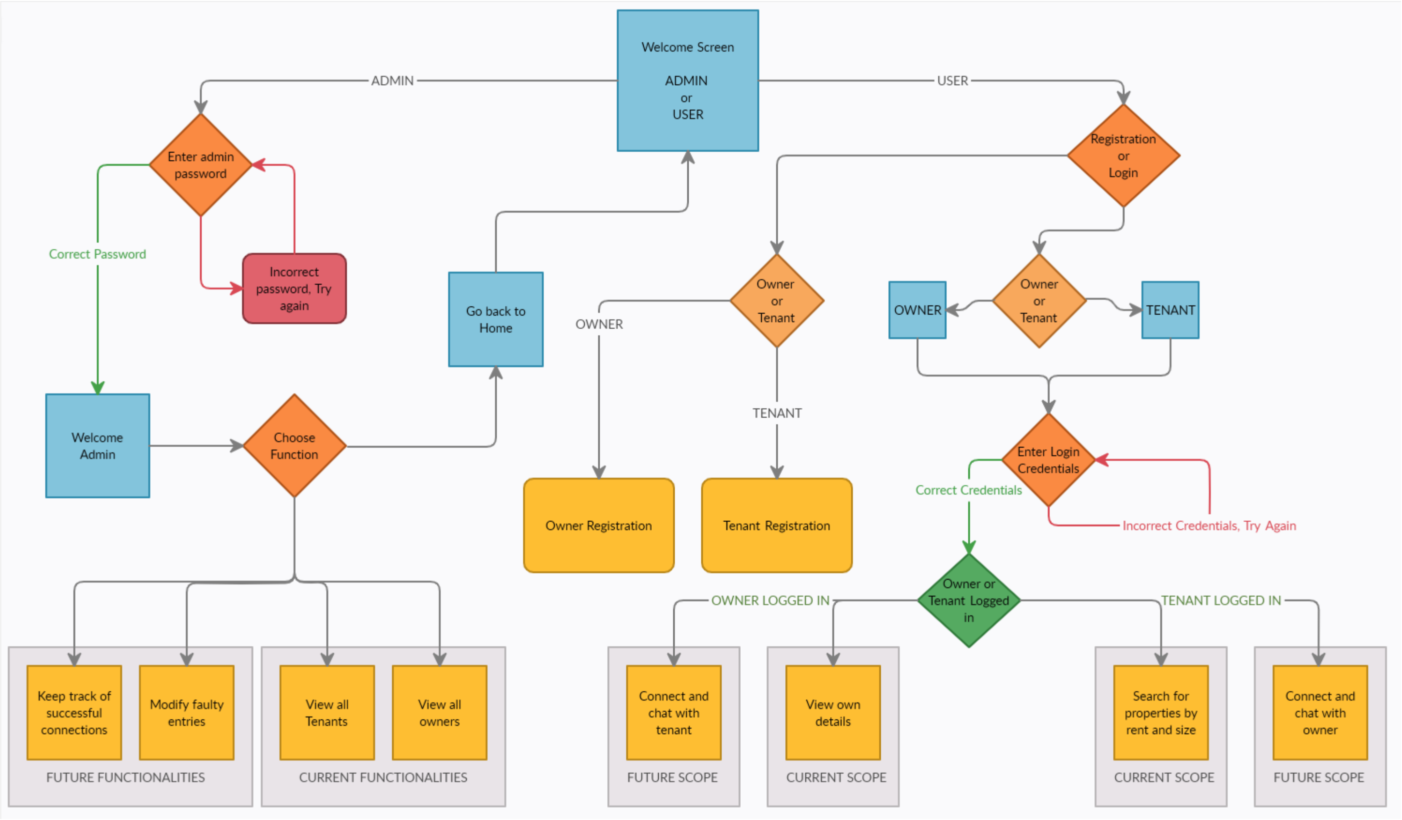


Figure 1 : Activity Diagram for JAVA

Test Plan

ID	Description	Expected	Result
TT_1_HH_01_LL_01	Check for valid inputs	Proceed to admin/users' welcome page.	Passed
TT_2_HH_01_LL_02	Check for invalid inputs	Display please try again.	Passed
TT_1_HH_02_LL_01	Check for invalid admin password	Access denied, try again	Passed
TT_2_HH_03_LL_01	Check with actual admin password	Admin privilege granted	Passed
TT_1_HH_04_LL_01	Check for valid Registration input	Redirect to registration	Passed
TT_2_HH_04_LL_01	Check for valid Login input	Redirect to Login	Passed
TT_3_HH_04_LL_01	Check for invalid input	Invalid selection, please try again	Passed
TT_1_HH_05_LL_01	Check for correct owner credentials	Login successful	Passed
TT_2_HH_05_LL_01	Check for incorrect owner credential	Login failed, try again	Passed
TT_1_HH_06_LL_01	Check for correct user credentials	Login successful	Passed
TT_2_HH_06_LL_01	Check for incorrect user credentials	Login failed, try again	Passed
TT_1_HH_07_LL_01	Check if search parameters work	Results matching search parameters	Passed

Implementation Summary

The program starts with the "Execute" class calling the Welcome Page class. From then on based upon the user's choice he/she can get admin access or just continue as a consumer of the service. To get admin access one must enter through a password protected layer. Upon choosing to continue as a consumer of the service, the user gets the choice to either register or login. Owners and tenants have a common login interface but the system can detect what category the user belongs to by identifying them through a unique userID. Upon logging in the owner gets the option of viewing either all the Owner entries at once or Tenant entries at once. If the logged in user is a tenant, then he/she gets the option to search for properties available by the virtue of monthly rent and city of choice.

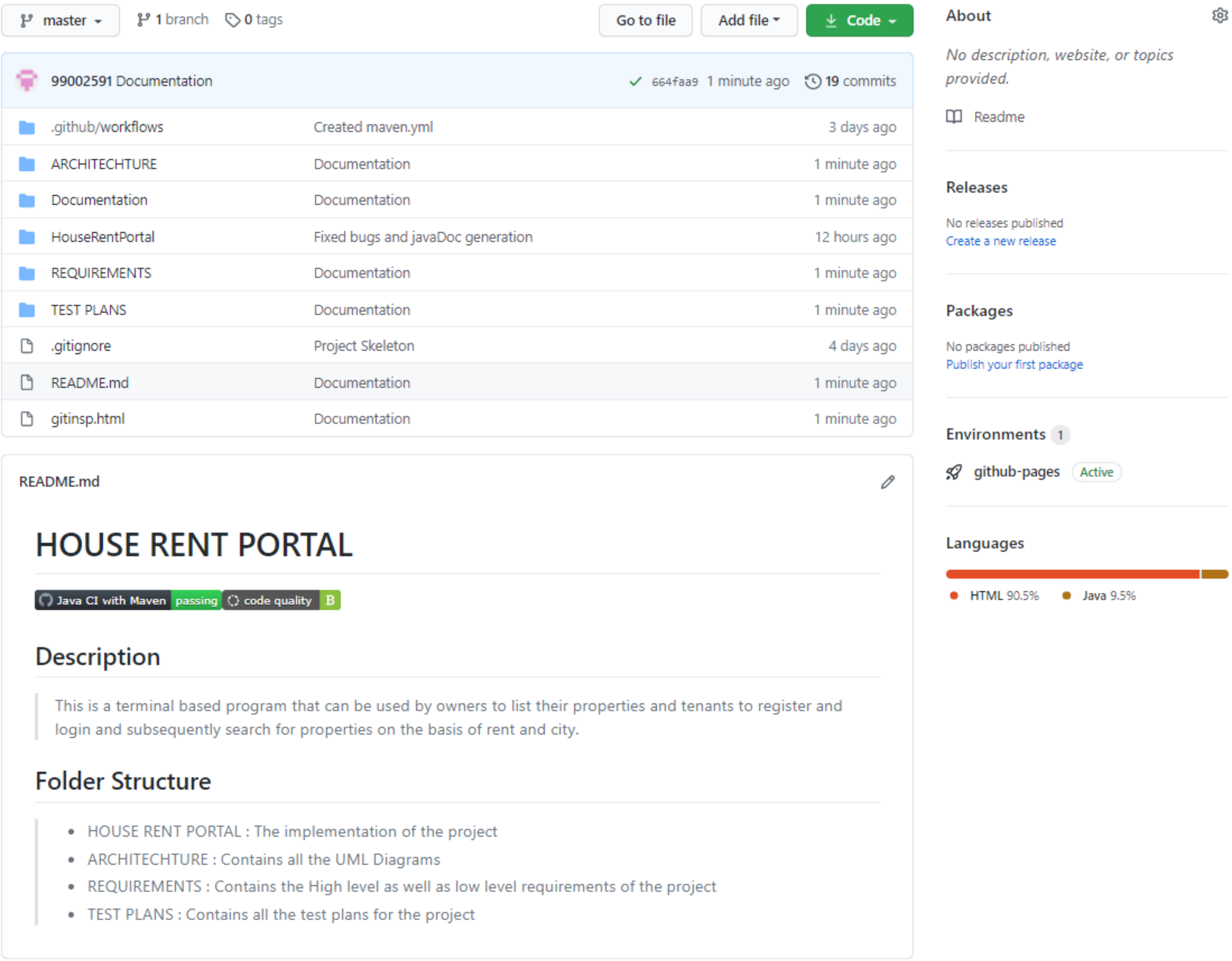
Video Summary

Link to Video showing program execution

Git Link

https://github.com/99002591/2009MYSEMB03-JAVA-10

Git Dashboard



99002591 Documentation ✓ 664faa9 1 minute ago 19 commits

.github/workflows	Created maven.yml	3 days ago
ARCHITECHTURE	Documentation	1 minute ago
Documentation	Documentation	1 minute ago
HouseRentPortal	Fixed bugs and javaDoc generation	12 hours ago
REQUIREMENTS	Documentation	1 minute ago
TEST PLANS	Documentation	1 minute ago
.gitignore	Project Skeleton	4 days ago
README.md	Documentation	1 minute ago
gitinsp.html	Documentation	1 minute ago

About

No description, website, or topics provided.

Readme

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Environments 1

github-pages Active

Languages

● HTML 90.5% ● Java 9.5%

README.md

HOUSE RENT PORTAL

Description

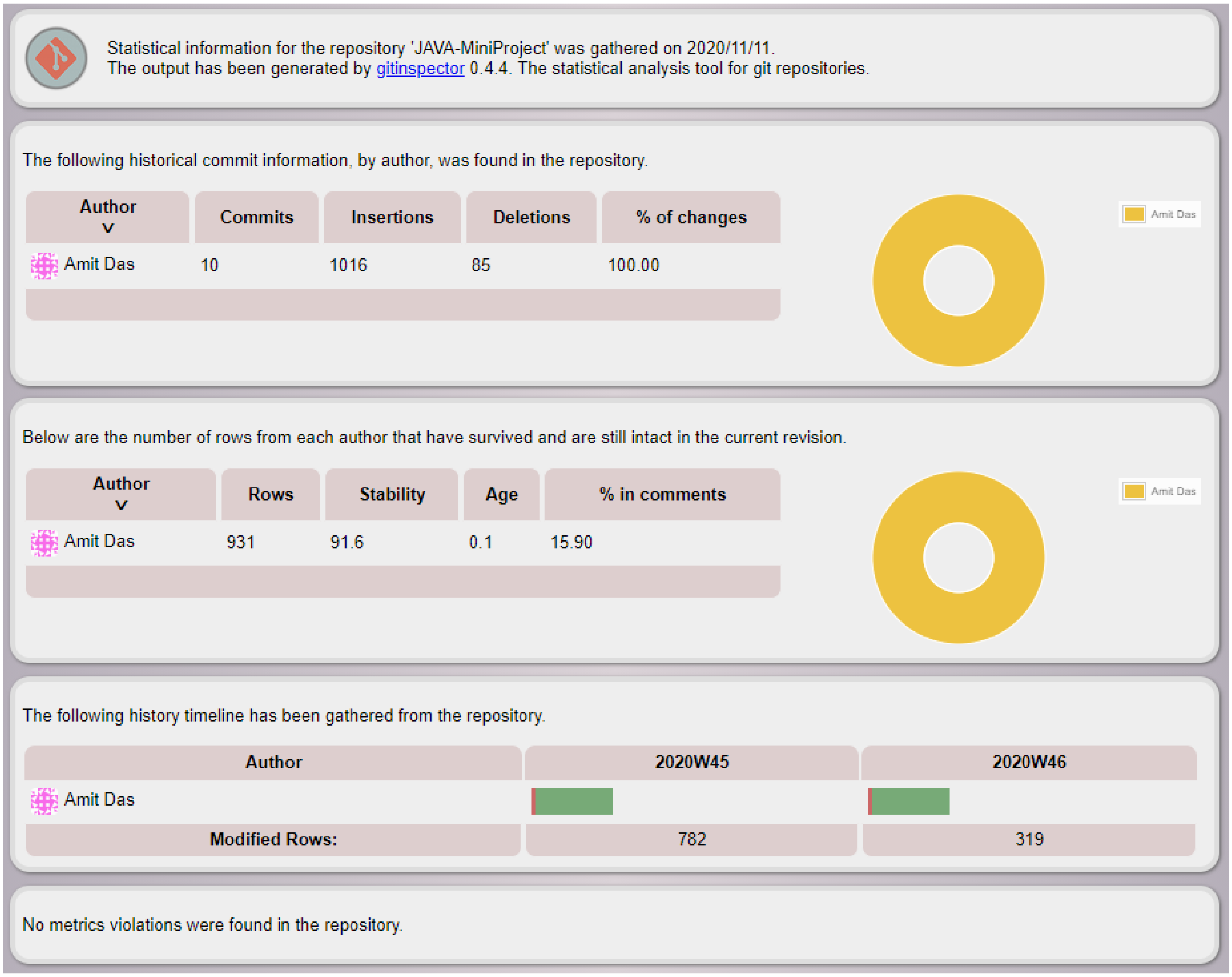
> This is a terminal based program that can be used by owners to list their properties and tenants to register and login and subsequently search for properties on the basis of rent and city.

Folder Structure

> - HOUSE RENT PORTAL : The implementation of the project
> - ARCHITECHTURE : Contains all the UML Diagrams
> - REQUIREMENTS : Contains the High level as well as low level requirements of the project
> - TEST PLANS : Contains all the test plans for the project

Figure 2 : Dashboard JAVA

Git inspector summary



Statistical information for the repository 'JAVA-MiniProject' was gathered on 2020/11/11.
The output has been generated by gitinspector 0.4.4. The statistical analysis tool for git repositories.

The following historical commit information, by author, was found in the repository.

Author v	Commits	Insertions	Deletions	% of changes
Amit Das	10	1016	85	100.00

Below are the number of rows from each author that have survived and are still intact in the current revision.

Author v	Rows	Stability	Age	% in comments
Amit Das	931	91.6	0.1	15.90

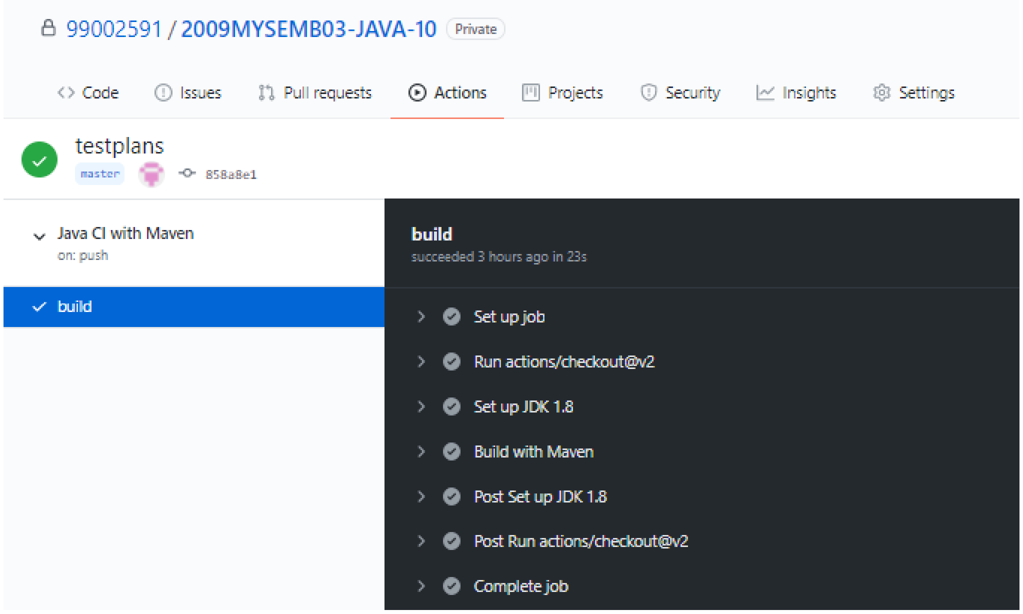
The following history timeline has been gathered from the repository.

Author	2020W45	2020W46
Amit Das	🟥🟩	🟥🟩
Modified Rows:	782	319

No metrics violations were found in the repository.

Figure 3 : Gitinspector JAVA

Build



99002591 / 2009MYSEMB03-JAVA-10 Private

testplans
master 858a8e1

Java CI with Maven
on: push

build

build
succeeded 3 hours ago in 23s

- Set up job
- Run actions/checkout@v2
- Set up JDK 1.8
- Build with Maven
- Post Set up JDK 1.8
- Post Run actions/checkout@v2
- Complete job

Figure 4 : Java Maven build

Run: “maven -B package --file pom.xml” to build the entire project.

Code quality and Issues or Bug Tracking

- Code quality was improved after taking into account the warnings thrown by Codeacy.
- No unused variables
- Use of easily understandable variable names.

Individual Contribution & Highlights

I worked individually on this project and completed the program myself end to end.

Summary

The project runs on the terminal and is very easy to comprehend and use. The program is almost entirely modular and each component of the code can be easily tweaked. Exceptions are handled in the program that ensures the program doesn’t terminate in case of unexpected events.

Challenges faced and how were they overcome

- Gitinspector is not able to fetch all commits. While there are 24 commits up until the time of uploading, git inspector can only fetch 10 commits.
- Had to work Individually because of teammate overlap and non-availability of vacancy in other teams.
- Had to skip unit testing because of time constraints.

✓ 858a8e1 4 hours ago 24 commits

Future Scope (If applicable)

- Integrating a system for the user to send a connection request to the Owner.
- GUI implementation.