

GENESIS - Learning Outcome & Python Mini-project Summary Report



LTTTS
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	SHANDHIYA.V.S [99002477]			

Contents

CONTENTS	3
LIST OF FIGURES:	4
TABLE OF TABLES:.....	4
MINIPROJECT -1 [ADVANCED PYTHON AND SDLC FOR REQUIREMENT ANALYSIS].....	5
MODULE/s	5
<i>Topic and Subtopics</i>	<i>5</i>
OBJECTIVES & REQUIREMENTS	5
DESIGN	6
TEST PLAN	7
IMPLEMENTATION SUMMARY	8
<i>Video Summary.....</i>	<i>9</i>
<i>Video Link.....</i>	<i>9</i>
<i>Git Link</i>	<i>10</i>
<i>Link for Github repository</i>	<i>10</i>
<i>Git Dashboard.....</i>	<i>10</i>
<i>Summary.....</i>	<i>10</i>
INDIVIDUAL CONTRIBUTION & HIGHLIGHTS	10
<i>Challenges faced and how were they overcome.....</i>	<i>11</i>
<i>Future Scope</i>	<i>11</i>
MINIPROJECT -2 [SDLC, JAVASCRIPT AND JASMINE FRAMEWORK]	12
MODULES.....	12
<i>Topic and Subtopics</i>	<i>12</i>
OBJECTIVES & REQUIREMENTS.....	12
DESIGN	13
TEST PLAN	14
IMPLEMENTATION SUMMARY	15
<i>Video Summary.....</i>	<i>16</i>
<i>Git Link</i>	<i>16</i>
<i>Git Dashboard.....</i>	<i>16</i>
<i>Summary.....</i>	<i>17</i>
INDIVIDUAL CONTRIBUTION & HIGHLIGHTS	17
<i>Challenges faced and how were they overcome.....</i>	<i>17</i>
MINIPROJECT -3[WEB AUTOMATION AND TESTING USING JAVA BASED SELENIUM AND CUCUMBER]	18
MODULES.....	18
<i>Topic and Subtopics</i>	<i>18</i>
OBJECTIVES & REQUIREMENTS	18
DESIGN	19
TEST PLAN	21
IMPLEMENTATION SUMMARY	22
<i>Video Summary.....</i>	<i>23</i>
<i>Git Link</i>	<i>23</i>
<i>Git Dashboard.....</i>	<i>23</i>
INDIVIDUAL CONTRIBUTION & HIGHLIGHTS	23
<i>Challenges faced and how were they overcome.....</i>	<i>24</i>

Future Scope	24
--------------------	----

List of Figures:

Figure 1 Server Client Configuration	6
Figure 2 Client side functionality	6
Figure 3 Sever side functionality	7
Figure 4 UseCase Diagram	13
Figure 5 Component Diagram	14
Figure 6 Website Overview	15
Figure 7. Git Repo Screenshot	16
Figure 8- Behavioral Diagram	19
Figure 9- Structural Diagram	20
Figure 10- Functional Checking	22
Figure 11- Git Dashboard.....	23

TABLE OF TABLES:

Table 1- Low Level and High Level Requirements	5
Table 2- Git Repo Snapshot	10
Table 3 Low and High level requirements	12
Table 4. Unit Level Test Cases	15
Table 5.Integration Level Test Cases	15
Table 6.Individual Contributions	17
Table 7 - Low Level and High Level Requirements	19
Table 8- Unit Test Cases	21
Table 9- Integration Test Cases	22

Miniproject -1 [Advanced Python and SDLC for requirement analysis]

Module/s

“Modules linked to mini project → Advanced Python and SDLC for requirement analysis”

Topic and Subtopics

“Briefly list the core topics and subtopics being implemented and how”

Core topics:

1. Network programming.
2. Speech recognition.

Sub-topics:

1. Server – Client (Socket Programming).
2. Hosting server and client.
3. Run configuration – Windows Command prompt.

Objectives & Requirements

Objective

1. Converting the Audio file into text file.
2. To establish communication between server and client and share the content present in audio file via communication network.

ID	Description
HL_01_L_01	Low level 01 – Require an IDE to edit the code – Notepad ++ with python language support. High level 01 – Need to launch the server and client, require the Python Ver. 3.8.0
HL_02_L_02	Low level 02 – Setting up the server, using command prompt. High level 02 –Launching the server and running it in the background.
HL_03_L_03	Low level 03 – Setting up the client, using command prompt. High level 03 – Launching the client and running it in the background.
HL_04_L_04	Low level 04 – Uploading the audio file in the client side. High level 04 – Recognizing the content from the audio file.
HL_05_L_05	Low level 05 – Display the recognized content on the server. High level 05 – Load the text on the server.
HL_06_L_06	Low Level 06 – Write the text displayed on the server to a file and save it as a text file. High Level 06 – Save the text that is displayed on the server side.

Table 1- Low Level and High Level Requirements

Design

“System Level and subsystem level UMLs – Structural and Behavioral”

Behavioral diagram:

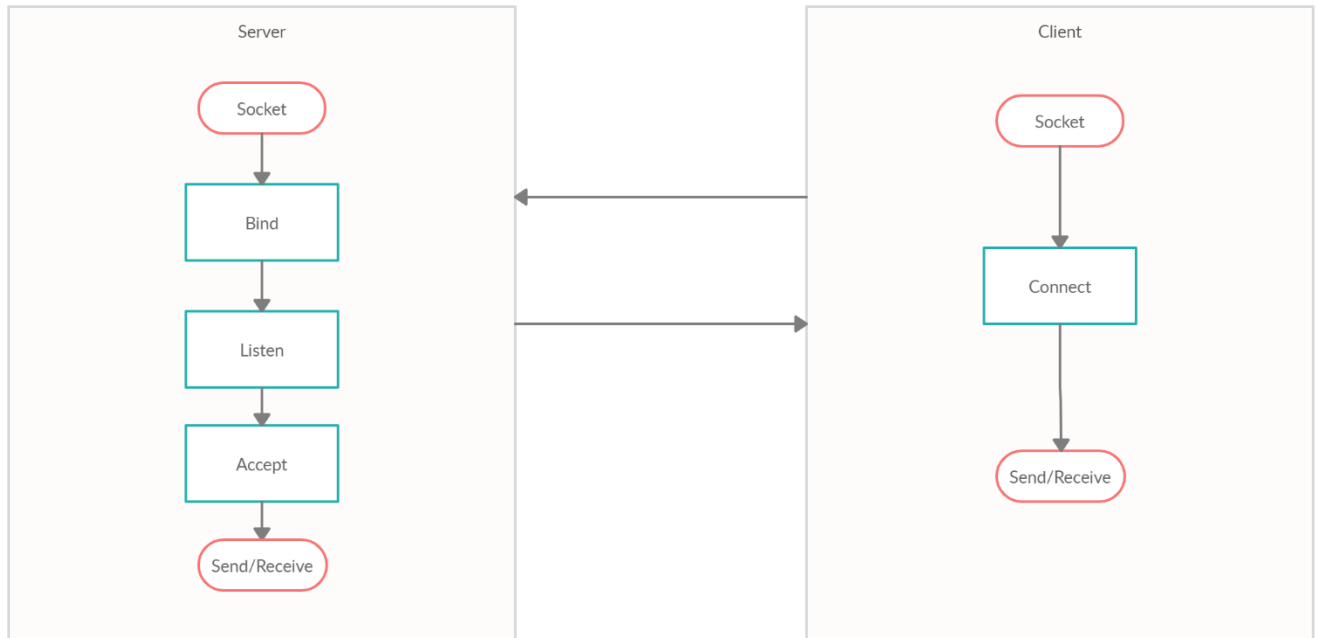


Figure 1 Server Client Configuration

Structural Diagram:

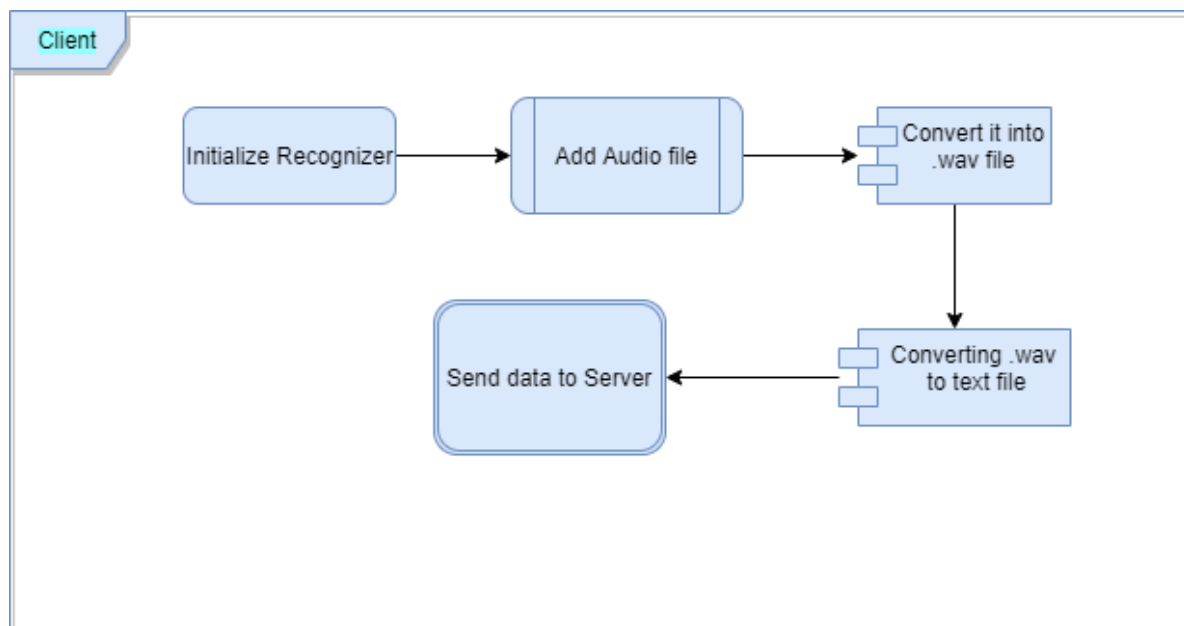


Figure 2 Client side functionality

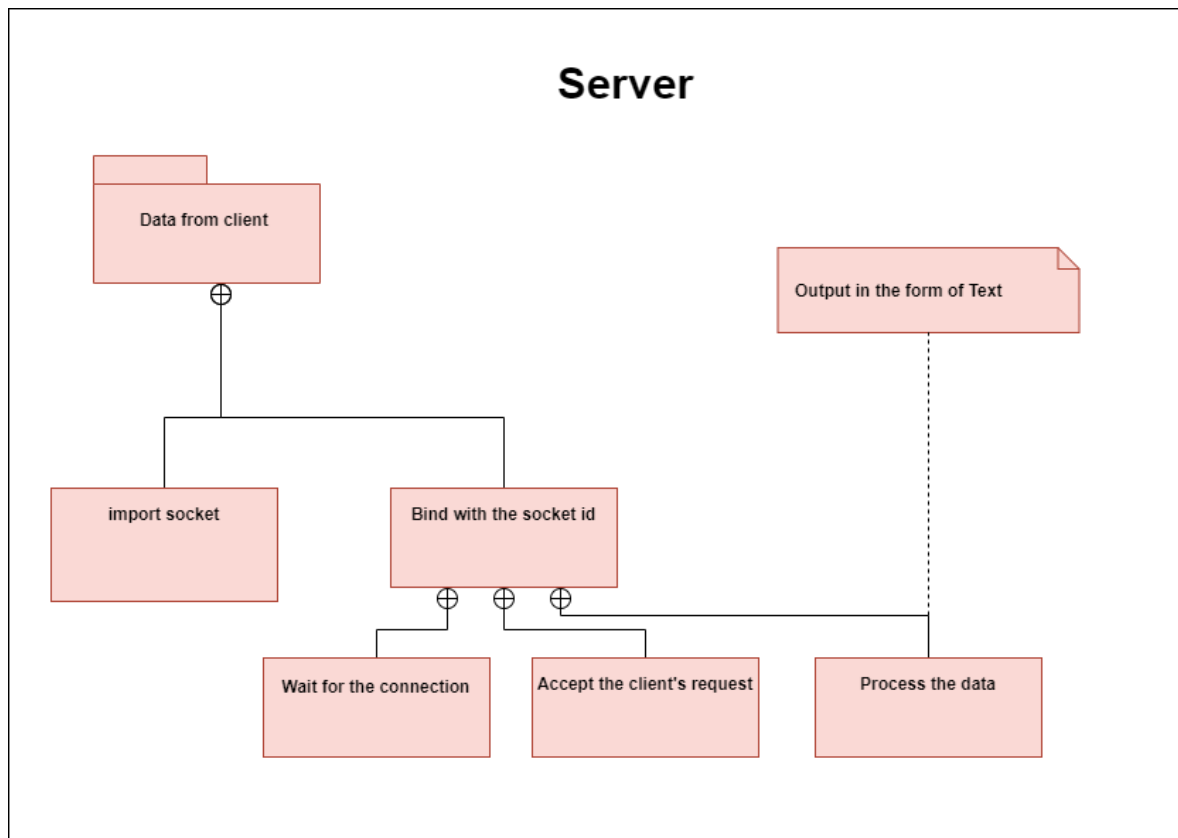


Figure 3 Server side functionality

Test Plan

ID	Description	Precondition	Excepted input	Expected output	Actual output
TC_1	Check the Python installation in the local machine	Downloaded the python ver. 3.8.0	Pyhton.exe	Launch the python	Launching the python
TC_2	Check the python path is set in the environment variables	Set required environment variables and path.	Check the python path in the command prompt	Display the installed path for python	Display the installed path for python
TC_3	Check the notepad++ IDE is installed and has the Python language support.	Python language support	Program execution on the command prompt	Launching the programs with '.py' extensions	Launching the programs with '.py' extensions
TC_4	Check whether the server is launched	Open the command prompt	server.py	Launching the server	Launching the server

TC_5	Check the audio file is ready to load at the client	Audio file with .wav format	File path to the audio file	Launching the audio file at the client	Launching the audio file at the client
TC_6	Check whether the server is running in the background	Opened server in the command prompt	Opened server and running	Running server in the background	Running server in the background
TC_6	Check whether the client is launched	Open the command prompt	Client.py	Launching the client	Launching the client
TC7	Check the audio file is launched and processed text file is sent to the server	Server running in the background	Text file sent from the client	Displaying the text on the server	Displaying the text on the server
TC_8	Check if the text is displayed on the server	Successfully launched the text file from the client	Text file from the client	Text output on Server side	Text output on Server side
TC_9	Check the text displayed is written to a text file	Opened the text file to write	Text displayed on the server	Text written to the text file	Text written to the text file
TC_10	No audio file launched at the client	Server running	Client program launched without the audio file	No output to display the text	Shall not write the text to the file (text file empty)

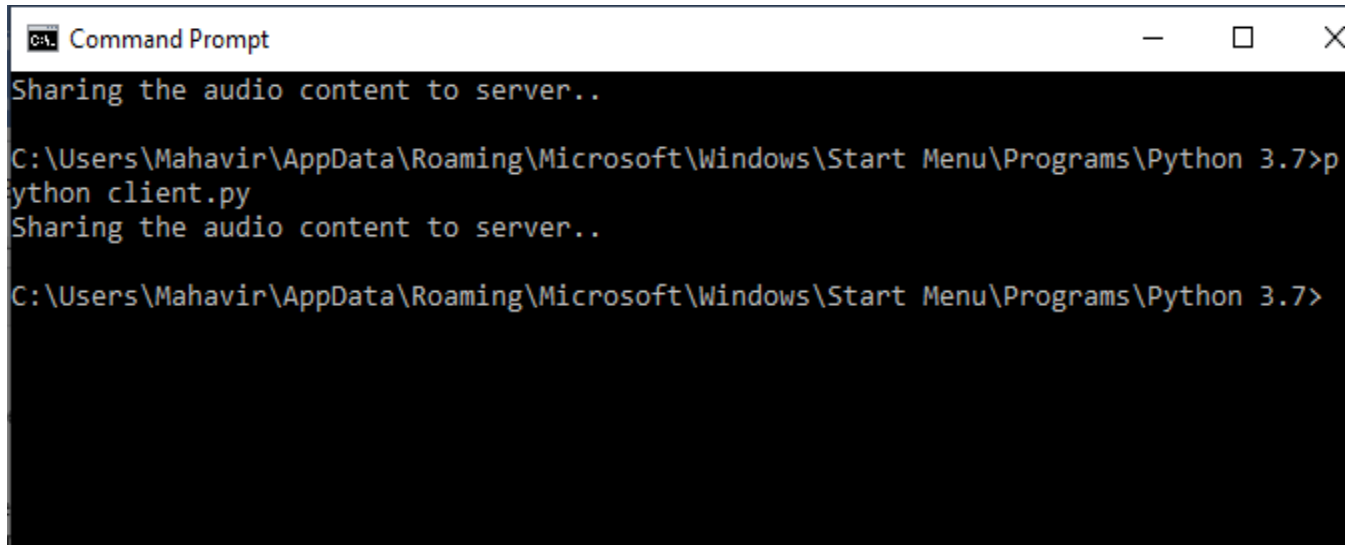
Table 2- Test Plan

Implementation Summary

“Section focused toward’ s implementation aspects. Here it is only core summary while all the details are in the Git Repo

To establish the communication between client and server using Network programming. The audio file is uploaded at the client side and with the help of python libraries, the audio file is converted to text and the text is displayed in server. And the audio file that is converted to text is stored in a file.

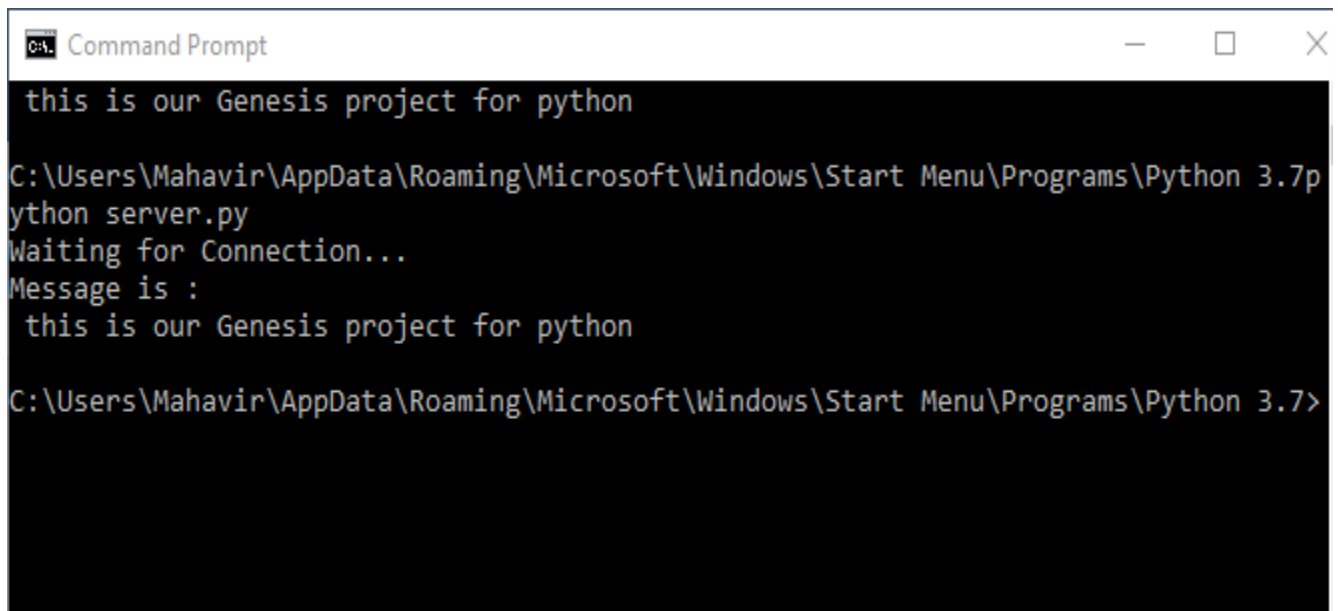
Snapshots of the output from the server and the snippets. (Images)



```
Command Prompt
Sharing the audio content to server..

C:\Users\Mahavir\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Python 3.7>python client.py
Sharing the audio content to server..

C:\Users\Mahavir\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Python 3.7>
```



```
Command Prompt
this is our Genesis project for python

C:\Users\Mahavir\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Python 3.7>python server.py
Waiting for Connection...
Message is :
this is our Genesis project for python

C:\Users\Mahavir\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Python 3.7>
```

Figure 4- Implementation

Video Summary

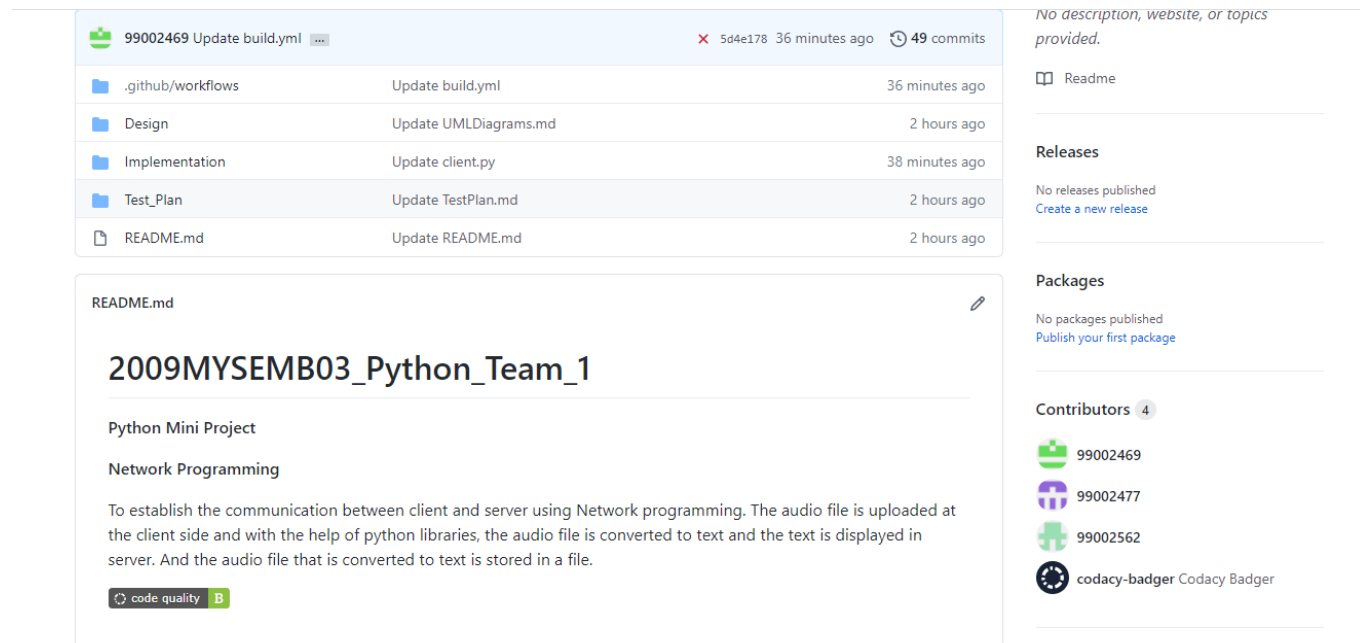
The video which we attached includes the team members who did the project and shows the operation of data transfer from client to server.

[Video Link](#)

Git Link

[Link for Github repository](#)

Git Dashboard



The screenshot shows a GitHub repository page. At the top, there's a commit summary: '99002469 Update build.yml' with a commit hash '5d4e178' and '36 minutes ago', and '49 commits'. Below this is a table of recent changes:

File	Update	Time
.github/workflows	Update build.yml	36 minutes ago
Design	Update UMLDiagrams.md	2 hours ago
Implementation	Update client.py	38 minutes ago
Test_Plan	Update TestPlan.md	2 hours ago
README.md	Update README.md	2 hours ago

Below the table is the README content for '2009MYSEMB03_Python_Team_1', which describes a 'Python Mini Project' for 'Network Programming'. It explains the goal: to establish communication between client and server using network programming, converting audio files to text using Python libraries. A 'code quality' badge shows a 'B' grade.

On the right side of the dashboard, there are sections for 'Releases' (No releases published), 'Packages' (No packages published), and 'Contributors' (4 contributors listed: 99002469, 99002477, 99002562, and codacy-badger).

Table 3- Git Repo Snapshot

Summary

“Section focused toward’ s implementation aspects. Here it is only core summary while all the details are in the Git Repo

To establish the communication between client and server using Network programming. The audio file is uploaded at the client side and with the help of python libraries, the audio file is converted to text and the text is displayed in server. And the audio file that is converted to text is stored in a file.

Individual Contribution & Highlights

S.No	Name	Contributions
1	Shandhiya V S (99002477)	Documentation of report and code quality checking.
2	Yuvateja C (99002469)	Done with PPT and Github activities.
3	Vinay Kumar V (99002562)	Done with coding.

Challenges faced and how were they overcome

Problem:

Unable to read the audio format (.mp3) in the client side and display the text on the server.

Solution:

Converted the audio format from .mp3 file to .wav, and found that the file is able to process on the client end
And successfully displays the text on the server .

Future Scope

1. Shall be able to communicate from the server and able to read the data on the client end.
2. Shall allow the user to upload the text file on the client end and output the speech on the server end using analog devices.
3. One server shall be able to communicate with multiple client devices.
4. Establish the continuous communication between the server and the client.

Miniproject -2 [SDLC, JavaScript and Jasmine Framework]

Modules

Modules linked to the miniproject – SDLC, JavaScript and Jasmine Framework.

Topic and Subtopics

Core Topic:

Web page designing

Web page designing using HTML, CSS and JavaScript.

Sub Topics:

Testing using Jasmine Framework

Creation of spec files, source file and specrunner.

Objectives & Requirements

Objective: To design a webpage for shopping the baby products using HTML, CSS, JavaScript in Visual studio code and testing the script using Jasmine framework.

Understanding JavaScript and Jasmine Framework.

Requirements (High level and low level):

ID	Description
HL_01_L_01	Low level 01 – Invoke Visual Studio Code High level 01 –Creation of HTML file with JavaScript and CSS.
HL_02_L_02	Low level 02 – Any browser of choice High level 02 –launching the created HTML page in the default browser.
HL_03_L_03	Low level 03 – Invoke Jasmine framework in VS code. High level 03 – passing of all Suites during test.
HL_04_L_04	Low level 04–HTML page with valid script. High level 04 –Able to make purchases
HL_05_L_05	Low level 05 – HTML page with valid script. High level 05 –Able to register in site with valid details.
HL_06_L_06	Low Level 06 – HTML page with valid script and corresponding regular expression. High Level 06 –Throws error for invalid details.

Table 4 Low and High level requirements

Design

Behavioral Diagram (UseCase Diagram):

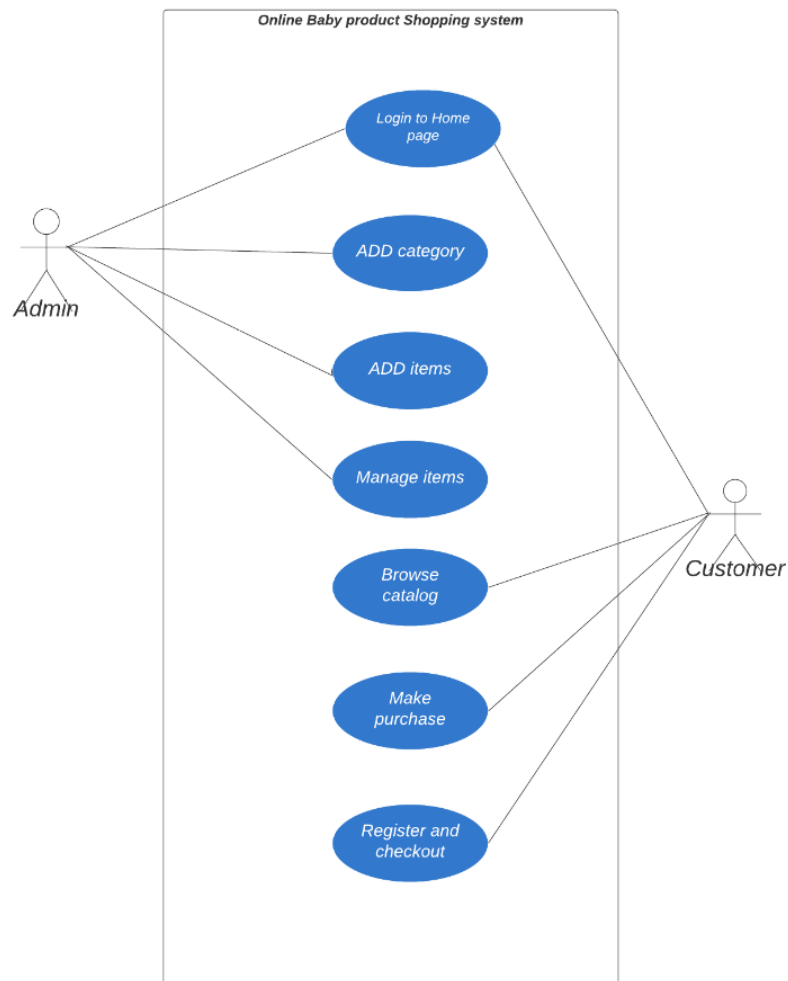


Figure 5 UseCase Diagram

Structural Diagram (Component Diagram):

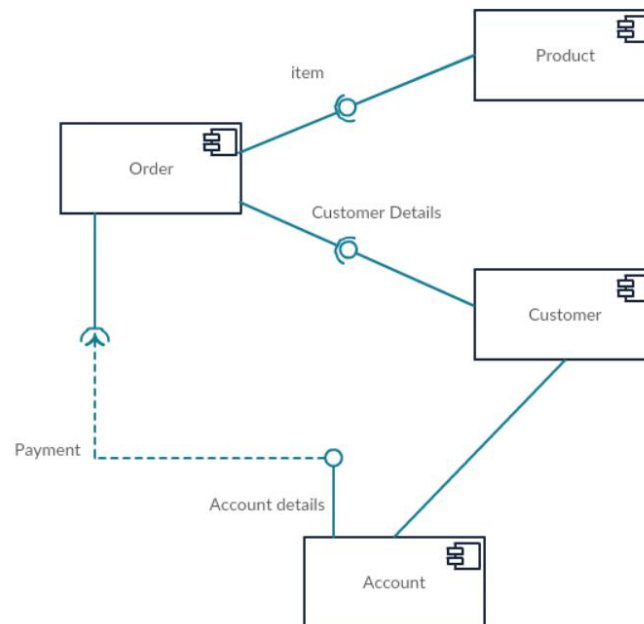


Figure 6 Component Diagram

Test Plan

Unit Level Test Cases:

ID	Description	Precondition	Excepted input	Expected output	Actual output
Tc1	Check whether it's able to launch HTML page.	Any Browser of choice.	Respective code in VS code.	URL launch.	URL is launched.
Tc2	Check whether all hyperlinks are functioning correctly.	Any Browser of choice.	Click on the functionality to check.	All functionalities are correctly executing.	All functionalities are correctly executing.
Tc3	Should check if email is of valid format	Any Browser of choice.	Valid email format	Able to register to the page.	Able to register to the page.
Tc4	should check if password and confirm password fields are same	Any Browser of choice.	Same password in both field.	Successful registration.	Successful registration.
Tc5	Check with different password and confirm password fields.	Any Browser of choice.	Different password in both field	Throws Error.	Throws Error.

Tc6	Should check if password has min 8 characters.	Any Browser of choice.	Password with more than 8 characters.	No error and Successful registration.	No error and Successful registration.
Tc7	Check for password less than 8 characters.	Any Browser of choice	Password with less than 8 characters.	Throws Error.	Throws Error.

Table 5. Unit Level Test Cases

Integration level Test cases:

Tc1	Check whether all page functionalities are passing the jasmine test suits.	Launch chrome browser.	Corresponding code and valid entering valid and invalid details.	Only able to submit valid details.	Submit button works only for valid details and throws error for invalid details.
-----	--	------------------------	--	------------------------------------	--

Table 6. Integration Level Test Cases

Implementation Summary

We have worked with HTML, CSS and JavaScript for designing a website “FIRST CRY”, where the customer can browse through the catalog and make purchases after registration. Jasmine Framework is implemented for testing each functionality of the webpage.

Output :

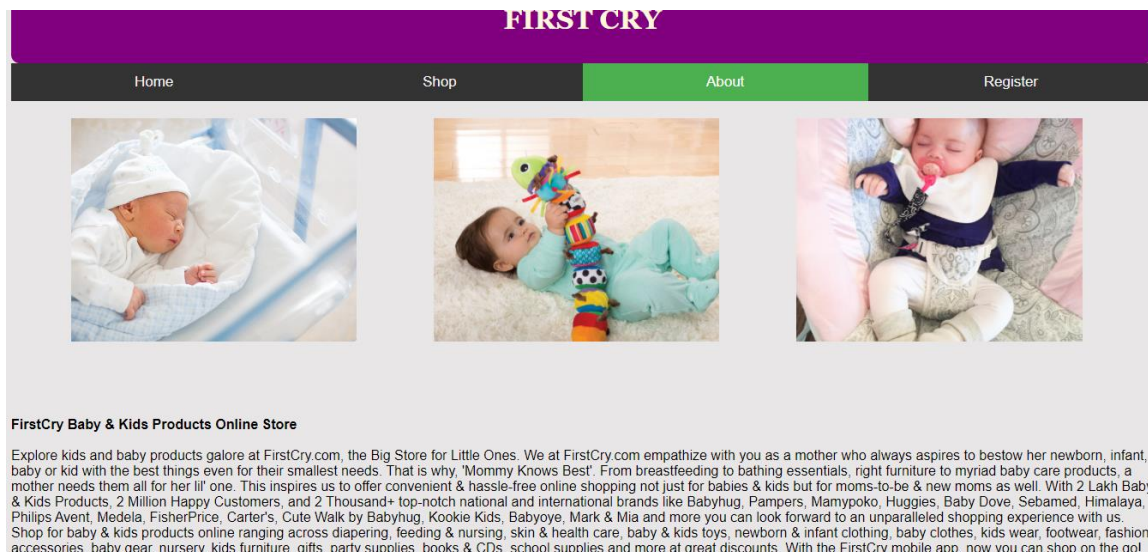
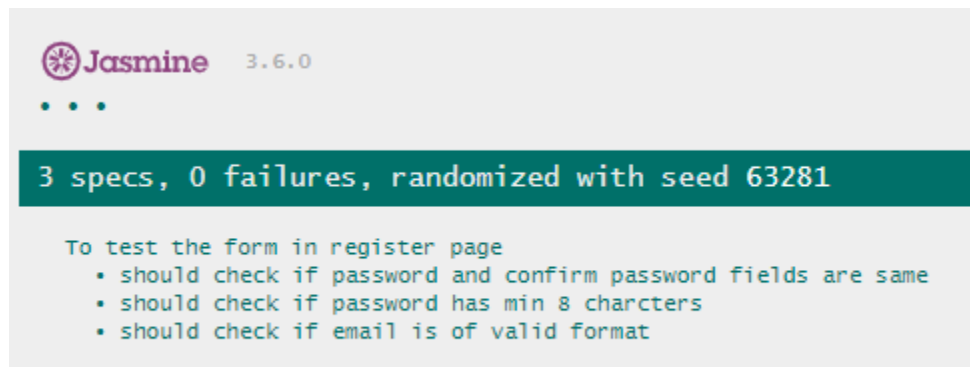


Figure 7 Website Overview

**Figure 8 Jasmine Test results**

Video Summary

The video includes a small look into our miniproject workflow and shows what are all the implementations we have done. The later part of our video will show about the testing of our website using Jasmine framework and the output of each test case.

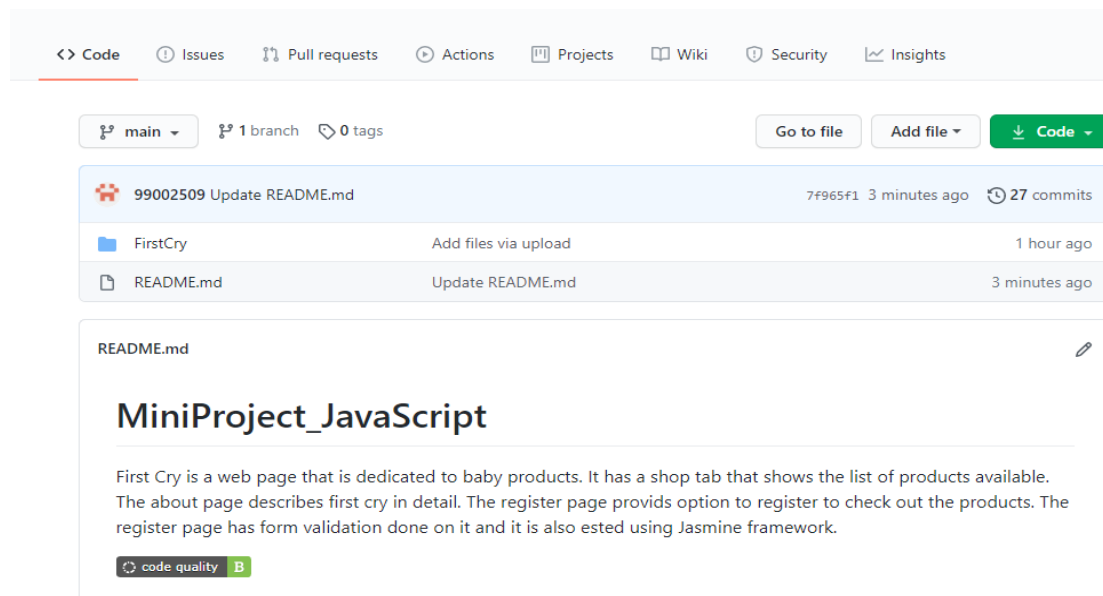
Video Link:

[Video Link](#)

Git Link

[Git Repository Link](#)

Git Dashboard

**Figure 8. Git Repo Screenshot**

Summary

We have worked with HTML, CSS and JavaScript for designing a website “FIRST CRY”, where the customer can browse through the catalog and make purchases after registration. Jasmine Framework is implemented for testing each functionality of the webpage.

Individual Contribution & Highlights

S.No	Name	Contributions
1	Shahna S.S (99002550)	Done with Html, CSS and JavaScript coding. Documentation of report and ppt.
2	Shandhiya V S (99002477)	Done with Html, CSS and JavaScript coding. Documentation of report and ppt.
3	Hridya M (99002509)	Done with Jasmine testing, git repository creation, readme file creation and code quality checking. Documentation of report and ppt.

Table 7.Individual Contributions

Challenges faced and how were they overcome.

Challenge 1: Aligning images in HTML.

Solution: By using CSS style, alignment issues are sorted out.

Challenge 2: Column Layout.

Solution: Set a fixed width for your wrap, and then float the navigation `div`.

Challenge 3: Test suites running in random order for Jasmine framework.

Solution: Unchecking “Test in random order” during the testing of suites.

Miniproject -3[Web Automation and testing using Java based Selenium and Cucumber]

Modules

“Modules linked to the miniproject–SDLC, Java, Selenium and Cucumber”

Topic and Subtopics

Core Topic:

Web page automation

Automation of a web page using Java based selenium scripts run using Eclipse IDE

Sub Topics:

Testing the automated page using cucumber framework

Creation of feature files

Run configuration – Cucumber feature

Objectives & Requirements

Objective: To automate a sample webpage and to test if all the functionalities on the webpage is working correctly. A travel webpage, <https://www.phptravels.net/home> was chosen. Two selenium script were written in JAVA. First script is for entering the user's booking information like destination, check-in and checkout, number of members to accommodate. From this page, it has to navigate to the payment and hotel selection page which is written in the next script. Following which cucumber based testing of these web automation was done.

- Understanding Java and selenium tool
- Testing the automated web page using Cucumber framework

Requirements (High level and low level):

ID	Description
HL_01_L_01 HL_01_L_02	High level 01 – Automate a webpage using Selenium JAVA Low level 01 – Invoke Eclipse IDE Low level 02 – Adding the required Selenium and JAVA dependencies
HL_02_L_01 HL_02_L_01	High level 02 – Launching a chrome browser Low level 01 – Create a MAVEN repository Low level 02 – Create a driver folder and setup an executable chrome driver
HL_03_L_01	High level 03 – Automate a registration page Low level 01 – Find a suitable web page Low level 02 – Invoke the selenium automation framework through eclipse

HL_04_L_01	High level 04 – Feature file creation and BDD test case execution using cucumber framework Low level 01 – Invoke cucumber framework
HL_05_L_01	High level 05 – Execute and pass a few mentioned scenarios Low level 01 – Implement the test cases for following scenarios: <ul style="list-style-type: none">• Given launching chrome• When launching URL• Then check functionality

Table 8 - Low Level and High Level Requirements

Design

Behavioral Diagram

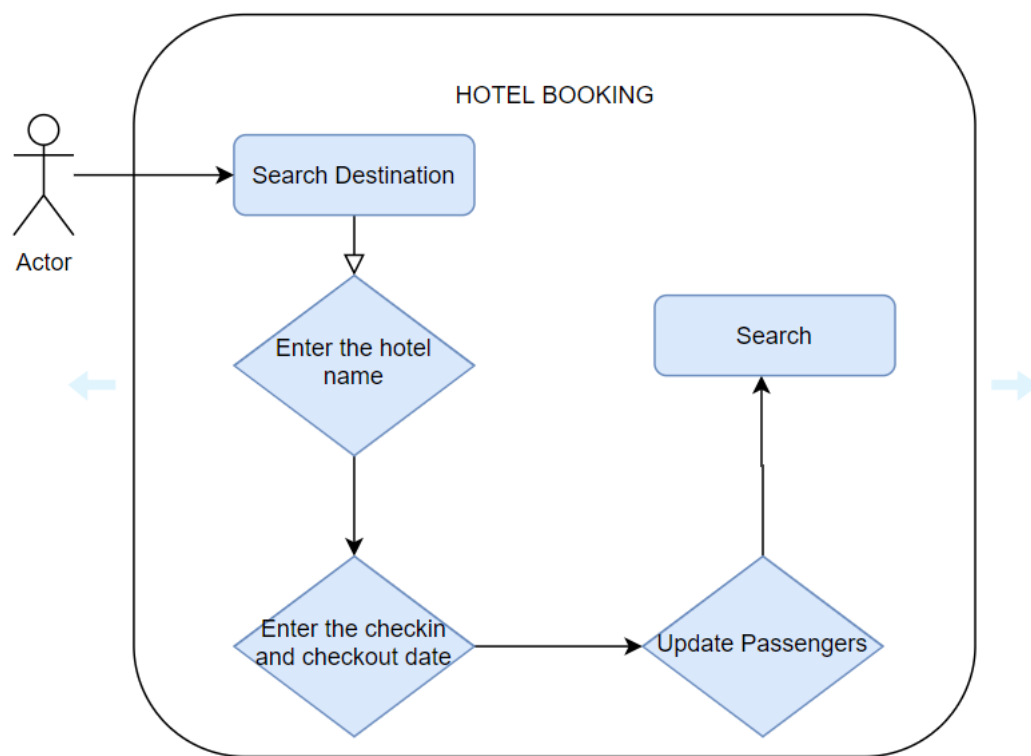


Figure 9- Behavioral Diagram

Structural Diagram

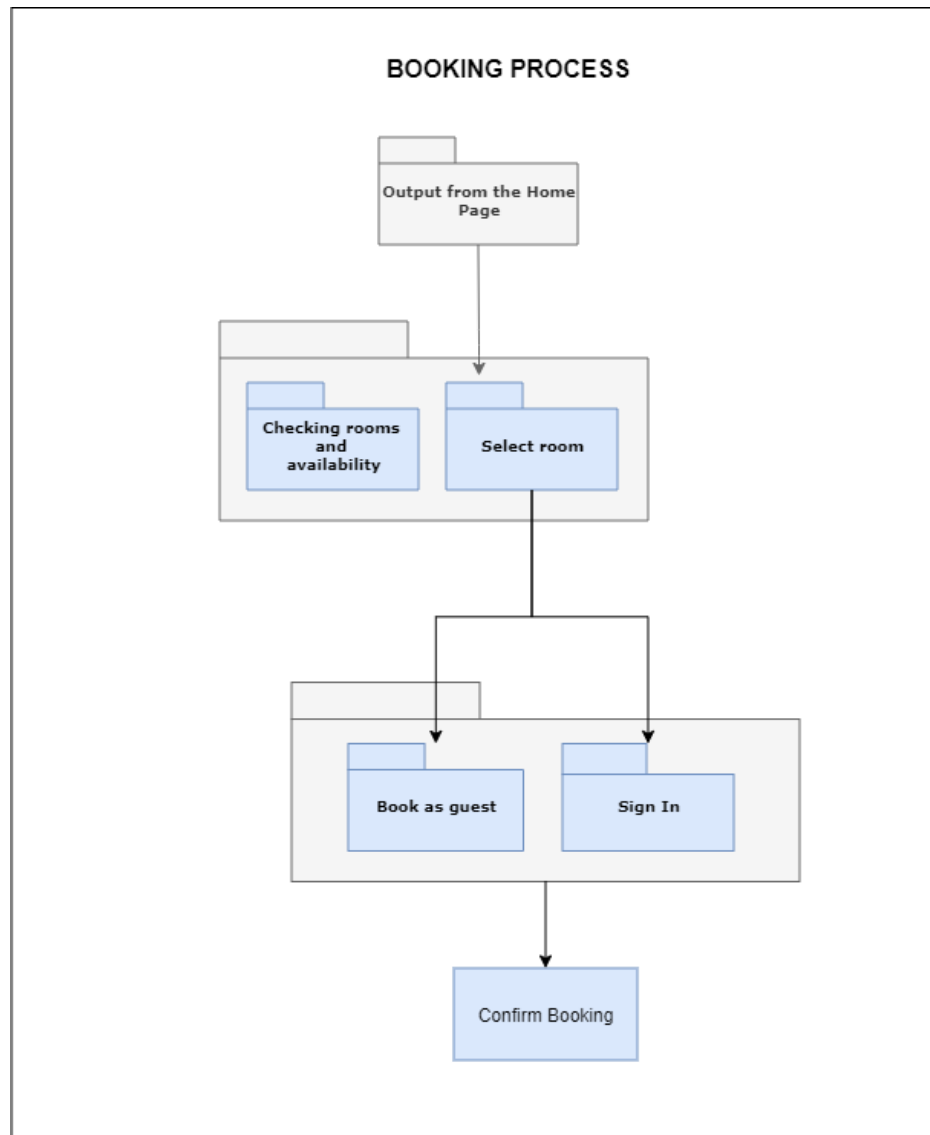


Figure 10- Structural Diagram

Test Plan

Unit Level Test Cases:

ID	Description	Precondition	Expected input	Expected output	Actual output
Tc1	Check if the chrome driver has launched the URL	An executable chrome driver is present in the driver folder	Hit the webpage using chrome driver	Browser launch	Browser launched
Tc2	Check whether destination can be chosen from the dropdown	Dropdown for destination has unique id or tags	Java code to navigate to the destination element and to select the desired destination	Destination should be selected	Destination selected
Tc3	Check whether check-in and checkout can be chosen from the dropdown	Dropdown for check-in and checkout can be accessed	Java code to navigate to the check-in and checkout element and to select the desired dates	Dates should be selected	Dates are selected
Tc4	Updating the number of adults and children should be possible	Button to increment the number of members should be accessible	Java code to access the update adult and children tag	Number of adults and children should be selected	Number of children and adults selected
Tc5	Check whether checkout details entered can be submitted	Button to submit present	Java code to navigate to checkout tag and click checkout	Redirects to the next page	Redirected to the next page after submission
Tc6	Select one of the rooms from the listed hotels	Option to select the room available	Java code to navigate to the select tag and select	Room should get selected	Room selected
Tc7	Navigate to the next section with the details entered	Book now option available	Code to click the book now button	Move to the payment page	Redirected to the payment page

Table 9- Unit Test Cases

Integration level Test cases:

Tc1	Check whether the functionalities of all the pages are passing the cucumber tests	Launch chrome browser	Corresponding Java code and cucumber tags	Launch the corresponding page with all functionalities	Launched the corresponding page with all functionalities
-----	---	-----------------------	---	--	--

Table 10- Integration Test Cases

Implementation Summary

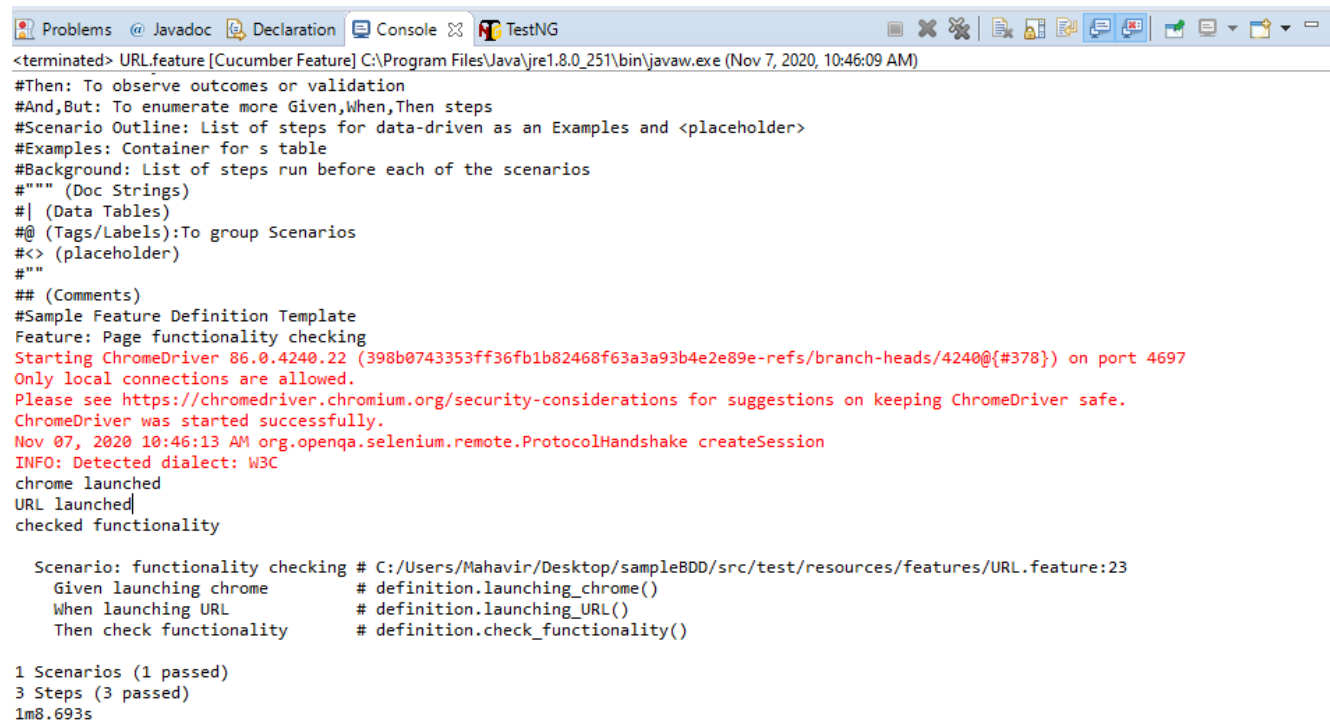
A travel webpage was automated using Selenium tool with and all the functionalities in the page was automated and Behavioral Driven Development (BDD) testing was done using cucumber framework. The following scenarios were tested using cucumber tags:

Feature: Page functionality testing

Scenario:

- Given launching chrome
- When launching URL
- Then check functionality

Output:



```

<terminated> URL.feature [Cucumber Feature] C:\Program Files\Java\jre1.8.0_251\bin\javaw.exe (Nov 7, 2020, 10:46:09 AM)
#Then: To observe outcomes or validation
#And,But: To enumerate more Given,When,Then steps
#Scenario Outline: List of steps for data-driven as an Examples and <placeholder>
#Examples: Container for s table
#Background: List of steps run before each of the scenarios
#"" (Doc Strings)
#| (Data Tables)
#@ (Tags/Labels):To group Scenarios
#<> (placeholder)
#""
## (Comments)
#Sample Feature Definition Template
Feature: Page functionality checking
Starting ChromeDriver 86.0.4240.22 (398b0743353ff36fb1b82468f63a3a93b4e2e89e-refs/branch-heads/4240@{#378}) on port 4697
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Nov 07, 2020 10:46:13 AM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: W3C
chrome launched
URL launched
checked functionality

Scenario: functionality checking # C:/Users/Mahavir/Desktop/sampleBDD/src/test/resources/features/URL.feature:23
  Given launching chrome        # definition.launching_chrome()
  When launching URL            # definition.launching_URL()
  Then check functionality       # definition.check_functionality()

1 Scenarios (1 passed)
3 Steps (3 passed)
1m8.693s

```

Figure 11- Functional Checking

Video Summary

The video includes the navigation of one page to another through automation and the output of the each functions.

Video Link: [Video Link](#)

Git Link

[Github Repo Link](#)

Git Dashboard

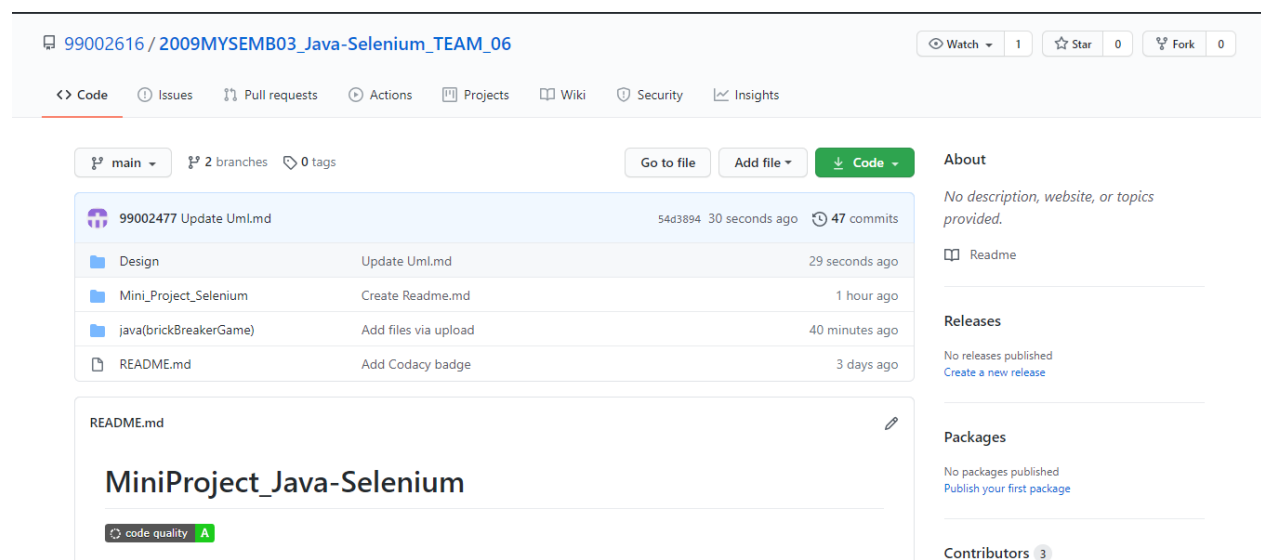


Figure 12- Git Dashboard

Individual Contribution & Highlights

Sl.No	Name	Contributions
1	Shandhiya V.S (99002477)	Done with the selenium automation of web page. Documentation of report and ppt.
2	Raj Shekhar Mishra (99002616)	Done with the BDD testing of automated page with cucumber framework. Documentation of report and ppt.
3	Soniya Chandran (99002587)	Done with git repository creation and code quality checking. Documentation of report and ppt.

Challenges faced and how were they overcome

Challenge 1: Timeout issue and delay in loading the webpage

Solution: This was handled by including Thread.sleep() in the code

Challenge 2: Fetching a strong Xpath was difficult with certain tags

Solution: hence, gave relative Xpath with stronger tags at certain places.

Future Scope

1. The online amount payment function can be implemented.
2. More functions for sorting the hotels can be done in future.