VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Chennai, Tamil Nadu, India.

School of Computer Science and Engineering (SCOPE)

BCSE301P – SOFTWARE ENGINEERING LAB

LAB RECORD

2023-2024

Submitted By:
Amisha Tripathi
21BCE1735
SCOPE

Submitted to:
Dr.D.Jeya Mala
Associate Professor Senior
SCOPE

Certificate

This is to certify that Ms. Amisha Tripathi (21BCE1735) satisfactorily completed the course of experiments of the lab course "Software Engineering Laboratory (BCSE301P) prescribed by VIT-Chennai for the semester V –B.Tech. degree course in the year 2023– 2024.

Date: 16/07/2023

Amisha

Signature of Student Head of the Department Signature of Faculty In-charge

Table of Contents

NAME -AMISHA TRIPATHI

REGISTRATION NUMBER- 21BCE1735

Ex.No. 1

Title – PROCESS MODEL SELECTION FOR "SMART HOME AUTOMATION SYSTEM"

Date: 11-05-2023

Aim :

Analysis and Identification of suitable process model for " SMART HOME AUTOMATION SYSTEM"

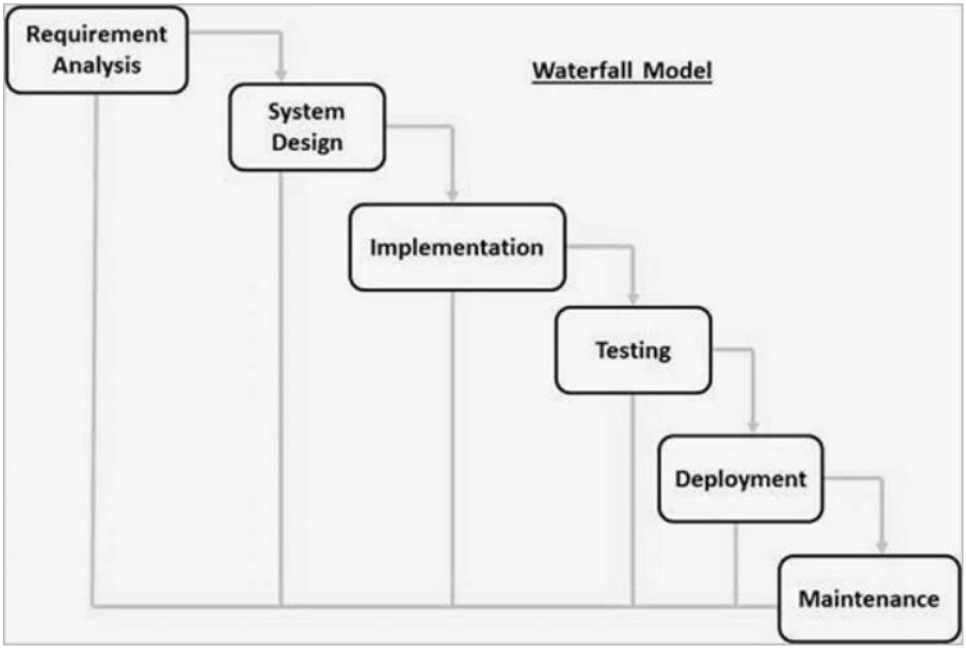
Description:

Process Models:

1. WATERFALL MODELS

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

The following illustration is a representation of the different phases of the Waterfall Model.



The sequential phases in Waterfall model are −

Requirement Gathering and analysis − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

System Design − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.

Implementation − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

Integration and Testing − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

Deployment of system − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

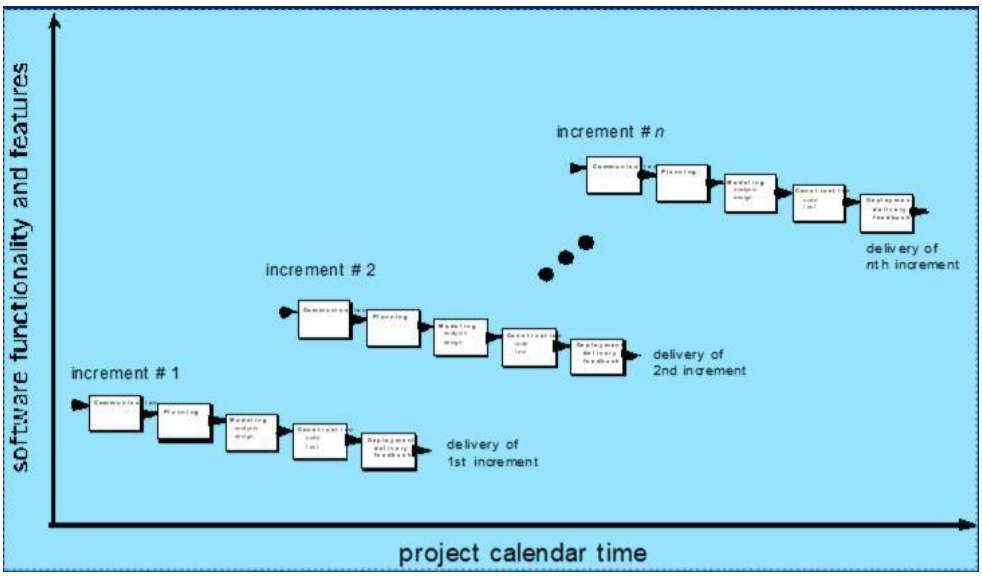
Maintenance − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

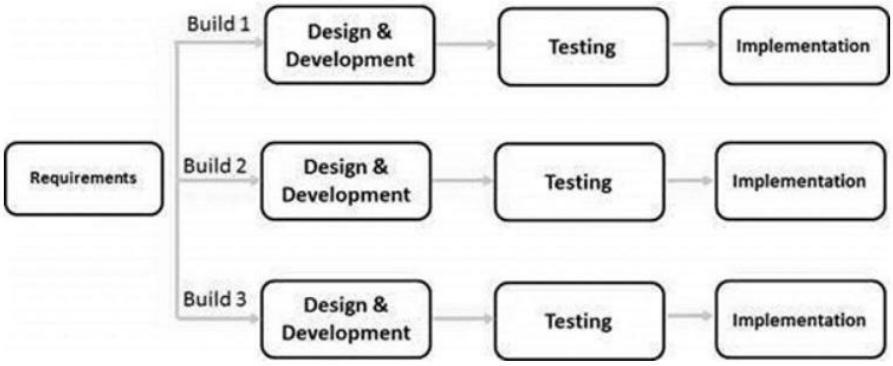
2. Incremental MODEL

The incremental model divides the development process into smaller increments or modules. Each increment delivers a portion of the final software functionality. The increments are developed and delivered in sequence, allowing for early feedback and risk mitigation.

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).

The following illustration is a representation of the Iterative and Incremental model −





Iterative and Incremental development is a combination of both iterative design or iterative method and incremental build model for development. "During software development, more than one iteration of the software development cycle may be in progress at the same time." This process may be described as an "evolutionary acquisition" or "incremental build" approach."

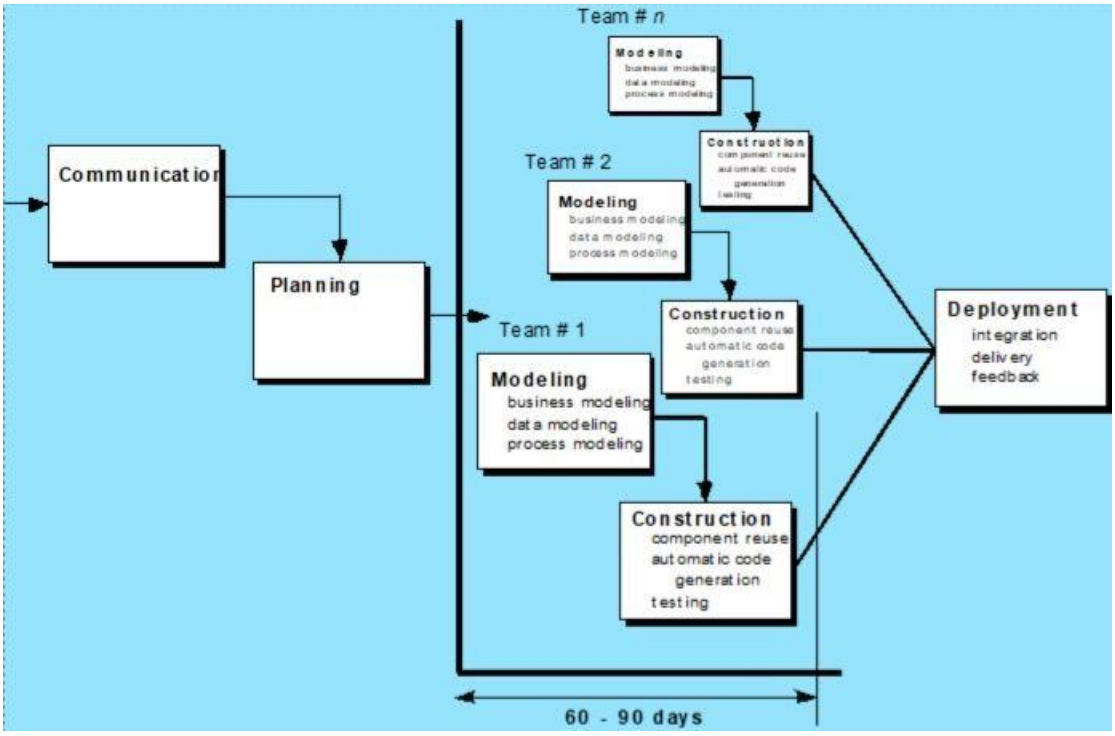
In this incremental model, the whole requirement is divided into various builds. During each iteration, the development module goes through the requirements, design, implementation and

testing phases. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is ready as per the requirement.

3. RAD MODEL

The **RAD (Rapid Application Development)** model is based on prototyping and iterative development with no specific planning involved. The process of writing the software itself involves the planning required for developing the product.

Rapid Application Development focuses on gathering customer requirements through workshops or focus groups, early testing of the prototypes by the customer using iterative concept, reuse of the existing prototypes (components), continuous integration and rapid delivery.



Following are the various phases of the RAD Model −

Business Modelling

The business model for the product under development is designed in terms of flow of information and the distribution of information between various business channels. A complete business analysis is performed to find the vital information for business, how it can be obtained, how and when is the information processed and what are the factors driving successful flow of information.

Data Modelling

The information gathered in the Business Modelling phase is reviewed and analyzed to form sets of data objects vital for the business. The attributes of all data sets is identified and defined. The relation between these data objects are established and defined in detail in relevance to the business model.

Process Modelling

The data object sets defined in the Data Modelling phase are converted to establish the business information flow needed to achieve specific business objectives as per the business model. The process model for any changes or enhancements to the data object sets is defined in this phase. Process descriptions for adding, deleting, retrieving or modifying a data object are given.

Application Generation

The actual system is built and coding is done by using automation tools to convert process and data models into actual prototypes.

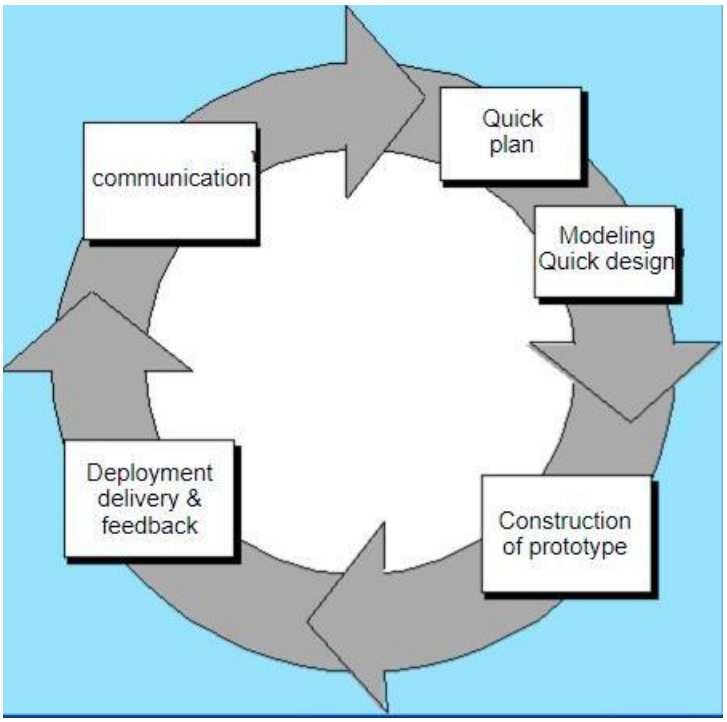
Testing and Turnover

The overall testing time is reduced in the RAD model as the prototypes are independently tested during every iteration. However, the data flow and the interfaces between all the components need to be thoroughly tested with complete test coverage. Since most of the programming components have already been tested, it reduces the risk of any major issues.

4. EVOLUTIONARY - PROTOTYPE MODEL

Prototype is a working model of software with some limited functionality. The prototype does not always hold the exact logic used in the actual software application and is an extra effort to be considered under effort estimation.

Prototyping is used to allow the users evaluate developer proposals and try them out before implementation. It also helps understand the requirements which are user specific and may not have been considered by the developer during product design.



5. SPIRAL MODEL

Prototype is a working model of software with some limited functionality. The prototype does not always hold the exact logic used in the actual software application and is an extra effort to be considered under effort estimation.

Prototyping is used to allow the users evaluate developer proposals and try them out before implementation. It also helps understand the requirements which are user specific and may not have been considered by the developer during product design.

The spiral model has four phases. A software project repeatedly passes through these phases in iterations called Spirals.

Identification

This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.

This phase also includes understanding the system requirements by continuous communication between the customer and the system analyst. At the end of the spiral, the product is deployed in the identified market.

Design

The Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and the final design in the subsequent spirals.

Construct or Build

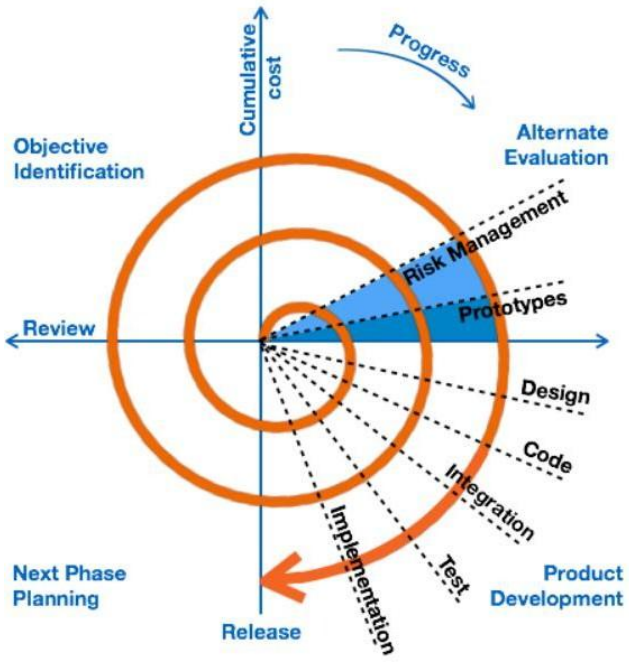
The Construct phase refers to production of the actual software product at every spiral. In the baseline spiral, when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.

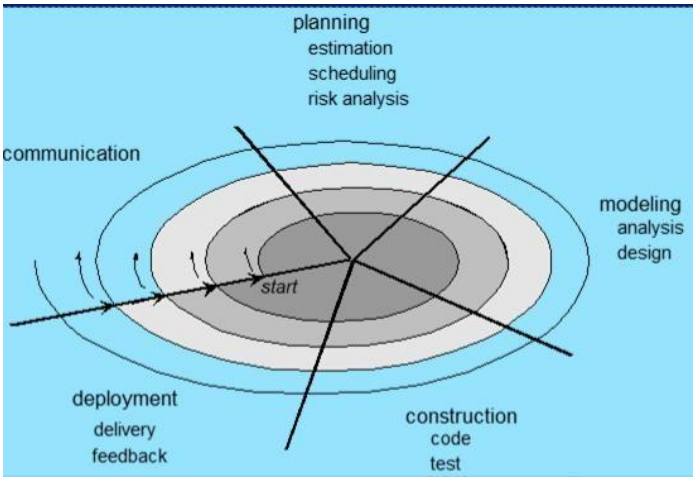
Then in the subsequent spirals with higher clarity on requirements and design details a working model of the software called build is produced with a version number. These builds are sent to the customer for feedback.

Evaluation and Risk Analysis

Risk Analysis includes identifying, estimating and monitoring the technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

The following illustration is a representation of the Spiral Model, listing the activities in each phase.



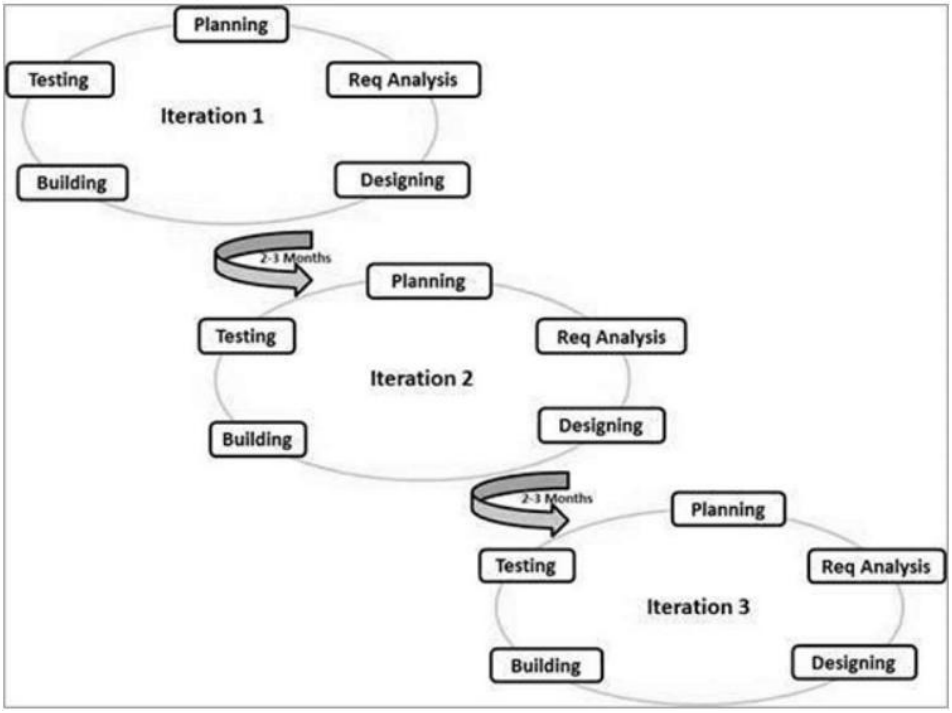


6. AGILE MODEL

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like −

- Planning
- Requirements Analysis
- Design
- Coding
- Unit Testing and
- Acceptance Testing.

At the end of the iteration, a working product is displayed to the customer and important stakeholders.



Following are the Agile Manifesto principles −

Individuals and interactions − In Agile development, self-organization and motivation are important, as are interactions like co-location and pair programming.
Working software − Demo working software is considered the best means of communication with the customers to understand their requirements, instead of just depending on documentation.
Customer collaboration − As the requirements cannot be gathered completely in the beginning of the project due to various factors, continuous customer interaction is very important to get proper product requirements.
Responding to change − Agile Development is focused on quick responses to change and continuous development.

OUTPUT:

MY PROJECT- SMART HOME AUTOMATION SYSTEM

Our Smart Home Automation System is designed to provide convenience, comfort, and energy efficiency to homeowners. It integrates various devices and technologies to create an interconnected ecosystem that can be controlled and monitored remotely. Here are some of the key functionalities of our system:

- Home Security: The system includes features such as smart locks, door/window sensors, motion detectors, and security cameras. Homeowners can remotely monitor their property, receive alerts for any suspicious activity, and control access to their home.

- Lighting Control: The system allows users to automate and control their home lighting. They can set schedules, adjust brightness levels, and even create lighting scenes to create the desired ambiance. Motion sensors can also be integrated to automatically turn lights on or off based on occupancy.

- Climate Control: Smart thermostats enable homeowners to control the temperature of their home from anywhere. They can set heating and cooling schedules, adjust temperature settings remotely, and receive energy usage reports. Integration with weather forecasts allows for optimal temperature management.
- Energy Management: The system provides real-time energy monitoring and helps homeowners identify energy-consuming devices and optimize their usage. Smart plugs and power meters enable remote control and monitoring of appliances, allowing users to conserve energy and reduce costs.

- Entertainment Systems: Integration with audio and video systems allows homeowners to control their home entertainment devices through a centralized interface. They can control speakers, TVs, media players, and streaming services, creating a seamless entertainment experience.

- Voice Control and Integration: Our system supports voice commands through virtual assistants like Amazon Alexa or Google Assistant. This enables homeowners to control various functionalities of the system using voice commands, enhancing the overall user experience.

- Automated Scenes and Routines: Users can create customized scenes and routines that automate multiple devices simultaneously. For example, a "Good Morning" routine can gradually turn on lights, adjust the thermostat, and play soothing music to wake up the residents.

- Remote Access and Monitoring: Homeowners can control and monitor their smart home system remotely through mobile apps or web interfaces. This allows them to check the status of devices, receive alerts, and make adjustments even when they are away from home.

- Integration with Other Smart Devices: Our system is designed to integrate with a wide range of smart devices, such as smart speakers, security systems, doorbells, blinds, and more. This provides homeowners with a unified and streamlined control experience.

Overall, our Smart Home Automation System offers a comprehensive solution for managing and controlling various aspects of the home. It aims to enhance convenience, security, energy efficiency, and entertainment while providing homeowners with the flexibility to customize and adapt the system to their specific needs.

SUITABLE PROCESS MODEL FOR SMART HOME AUTOMATION SYSTEM

For the development of a smart home automation system, an **Agile model, specifically the Scrum framework**, would be a suitable choice. Here's why:

- Flexibility: Smart home automation systems often involve complex and rapidly evolving technologies. An Agile model allows for flexibility and adaptability to accommodate changing requirements, technologies, and customer needs. It enables you to respond quickly to market trends and incorporate new features or technologies as they emerge.
- Incremental Development: Agile models, including Scrum, promote incremental development. You can start with a basic set of features and gradually add more functionality in subsequent iterations. This approach allows you to deliver a working system quickly and get valuable user feedback early on.
- Collaboration and Communication: Smart home automation systems require collaboration among various stakeholders, including developers, designers, product owners, and end-users. Agile models foster regular communication and collaboration through techniques such as daily stand-up meetings, sprint reviews, and retrospectives. This helps ensure that everyone is on the same page and can provide input and feedback throughout the development process.

- Continuous Testing and Integration: Quality assurance is crucial for a reliable smart home automation system. Agile models emphasize continuous testing and integration, enabling frequent validation of the system's functionality and performance. This iterative testing approach helps identify and address issues early, leading to higher-quality software.
- Customer-Centric Approach: Smart home automation systems are designed to enhance the daily lives of end-users. Agile models prioritize customer satisfaction and involve customers throughout the development process. By gathering customer feedback and incorporating it into subsequent iterations, you can ensure that the system meets their needs and expectations.
- Time-to-Market: Agile models promote shorter development cycles and faster time-to-market. This is important in the rapidly evolving smart home market, where competition is fierce. By delivering usable increments of the system in regular intervals, you can get your product to market faster and gain a competitive advantage.

It's worth noting that the choice of process model ultimately depends on the specific requirements, team composition, and project constraints. However, considering the dynamic nature of smart home automation systems and the benefits provided by Agile models, Scrum would be a well-suited approach.

Result : Thus the analysis of various process models and identification of the suitable process model for the project “ SMART HOME AUTOMATION SYSTEM” is successfully done. The identified process model is “an Agile model, specifically the Scrum framework “

Name- Amisha Tripathi
Registration Number- 21BCE1735

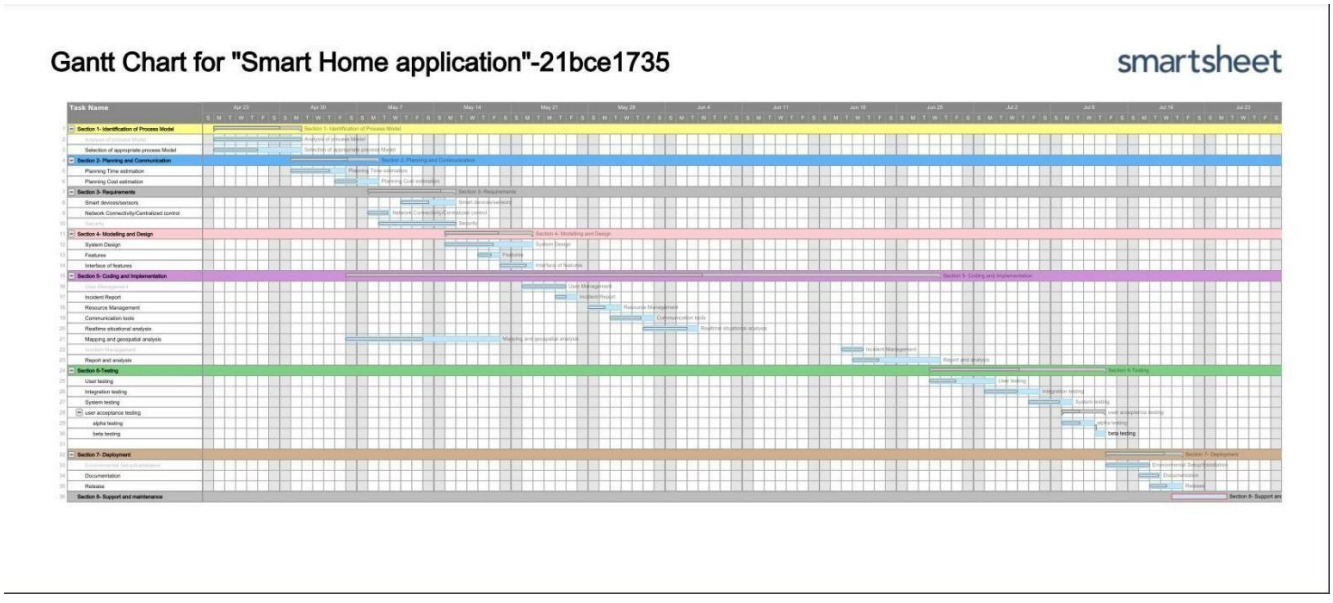
EX-2a

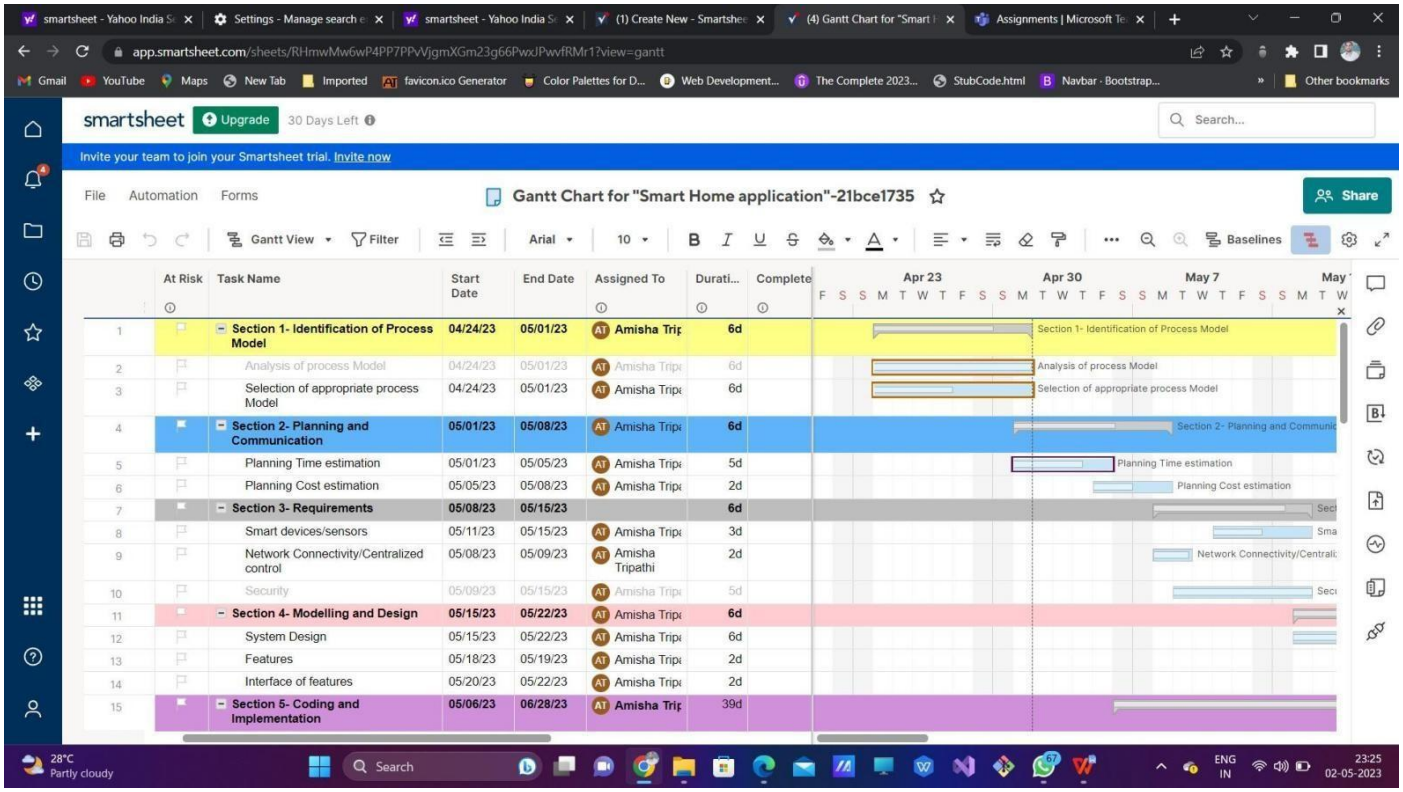
Title – Gantt Chart for our project.
Date: 14 May, 2023

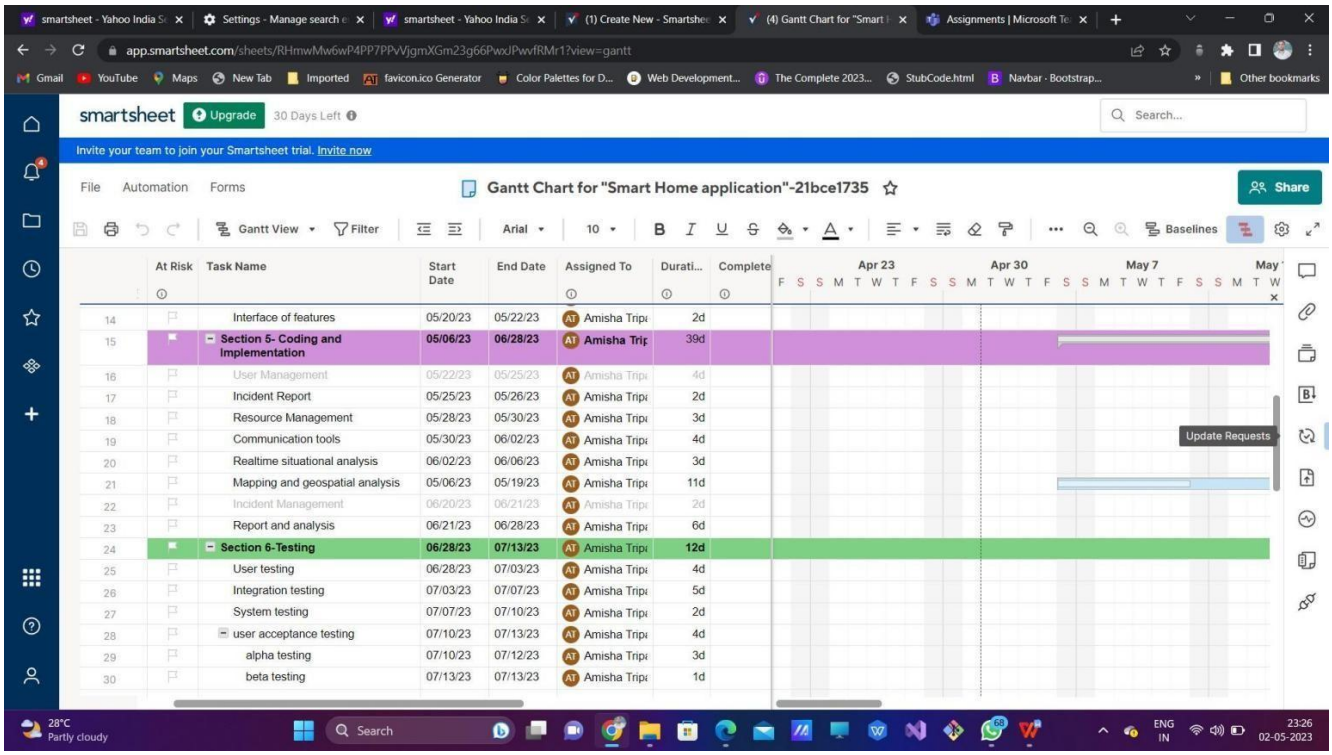
Aim : **To draw Gantt Chart for " Smart Home Automation System App".Description**:

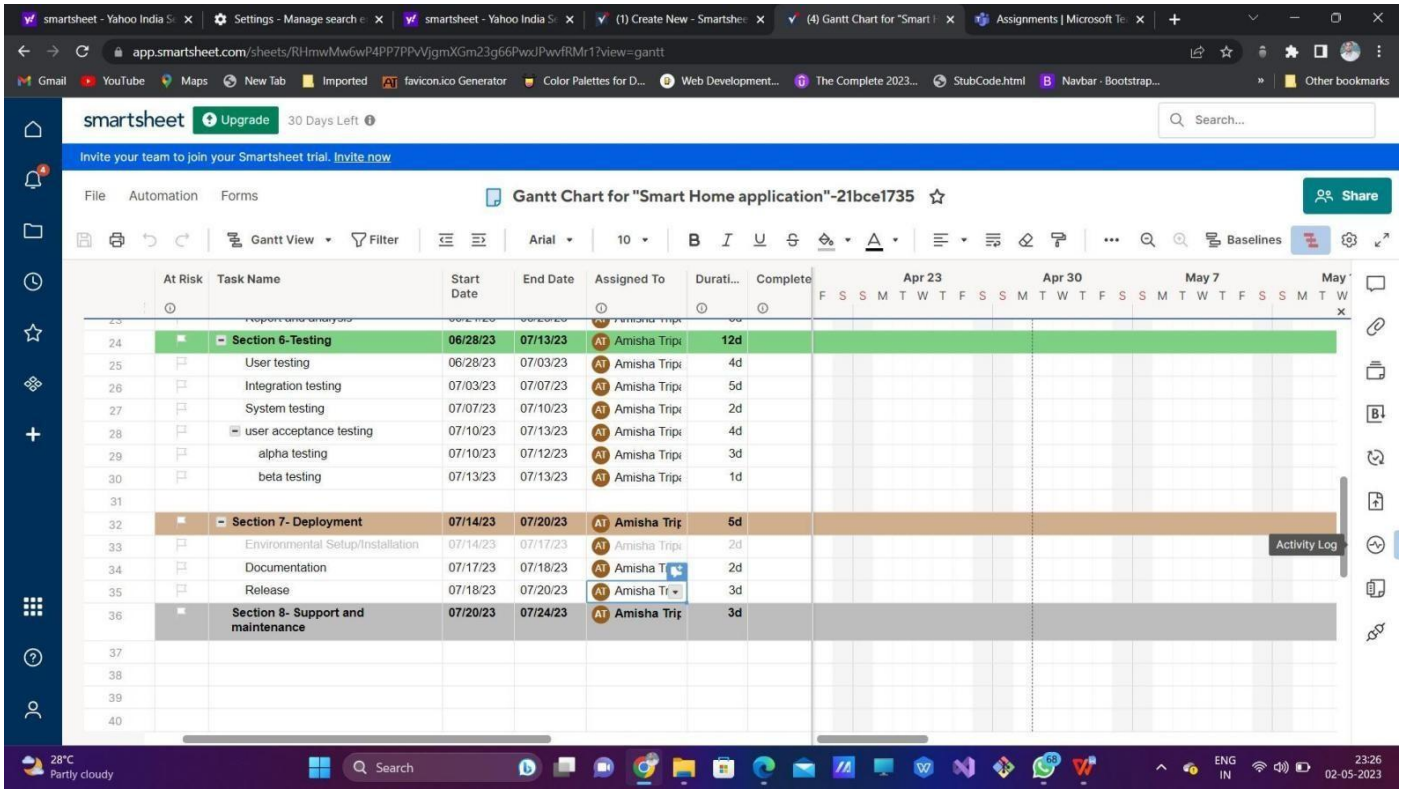
This Gantt Chart shows the timelines of various tasks and subtasks of our project regarding 'Smart HomeAutomation App'. The dates and time assigned to individual tasks and the people in the team who are being assigned are being pictorially represented. The basic structure of the Gantt Chart contains the duration of each task and under that the duration of each subtasks in a histogram format.

Output:




smartsheet
Upgrade
30 Days Left
Search...
Invite your team to join your Smartsheet trial. Invite now
File
Automation
Forms
Gantt Chart for "Smart Home application"-21bce1735
Share
Gantt View
Filter
Arial
10
Baselines
At Risk
Task Name
Start Date
End Date
Assigned To
Durati...
Complete
Apr 23
Apr 30
May 7
1 Section 1- Identification of Process Model 04/24/23 05/01/23 Amisha Trip 6d
2 Analysis of process Model 04/24/23 05/01/23 Amisha Tripa 6d
3 Selection of appropriate process Model 04/24/23 05/01/23 Amisha Tripa 6d
4 Section 2- Planning and Communication 05/01/23 05/08/23 Amisha Tripa 6d
5 Planning Time estimation 05/01/23 05/05/23 Amisha Tripa 5d
6 Planning Cost estimation 05/05/23 05/08/23 Amisha Tripa 2d
7 Section 3- Requirements 05/08/23 05/15/23 6d
8 Smart devices/sensors 05/11/23 05/15/23 Amisha Tripa 3d
9 Network Connectivity/Centralized control 05/08/23 05/09/23 Amisha Tripathi 2d
10 Security 05/09/23 05/15/23 Amisha Tripa 5d
11 Section 4- Modelling and Design 05/15/23 05/22/23 Amisha Tripa 6d
12 System Design 05/15/23 05/22/23 Amisha Tripa 6d
13 Features 05/18/23 05/19/23 Amisha Tripa 2d
14 Interface of features 05/20/23 05/22/23 Amisha Tripa 2d
15 Section 5- Coding and Implementation 05/06/23 06/28/23 Amisha Trip 39d
28°C
Partly cloudy
Search
ENG
IN
23:25
02-05-2023


smartsheet
Upgrade
30 Days Left
Search...
Invite your team to join your Smartsheet trial. Invite now
File
Automation
Forms
Gantt Chart for "Smart Home application"-21bce1735
Share
Gantt View
Filter
Arial
10
Baselines
At Risk
Task Name
Start Date
End Date
Assigned To
Durati...
Complete
Apr 23
Apr 30
May 7
14 Interface of features 05/20/23 05/22/23 Amisha Tripa 2d
15 Section 5- Coding and Implementation 05/06/23 06/28/23 Amisha Trip 39d
16 User Management 05/22/23 05/25/23 Amisha Tripa 4d
17 Incident Report 05/25/23 05/26/23 Amisha Tripa 2d
18 Resource Management 05/28/23 05/30/23 Amisha Tripa 3d
19 Communication tools 05/30/23 06/02/23 Amisha Tripa 4d
20 Realtime situational analysis 06/02/23 06/06/23 Amisha Tripa 3d
21 Mapping and geospatial analysis 05/06/23 05/19/23 Amisha Tripa 11d
22 Incident Management 06/20/23 06/21/23 Amisha Tripa 2d
23 Report and analysis 06/21/23 06/28/23 Amisha Tripa 6d
24 Section 6-Testing 06/28/23 07/13/23 Amisha Tripa 12d
25 User testing 06/28/23 07/03/23 Amisha Tripa 4d
26 Integration testing 07/03/23 07/07/23 Amisha Tripa 5d
27 System testing 07/07/23 07/10/23 Amisha Tripa 2d
28 user acceptance testing 07/10/23 07/13/23 Amisha Tripa 4d
29 alpha testing 07/10/23 07/12/23 Amisha Tripa 3d
30 beta testing 07/13/23 07/13/23 Amisha Tripa 1d
Update Requests
28°C
Partly cloudy
Search
ENG
IN
23:26
02-05-2023



Result : Thus the Gantt chart is designed for planning the timeline of various activities for the project“ Smart Home Automation System App”.

EX-2b

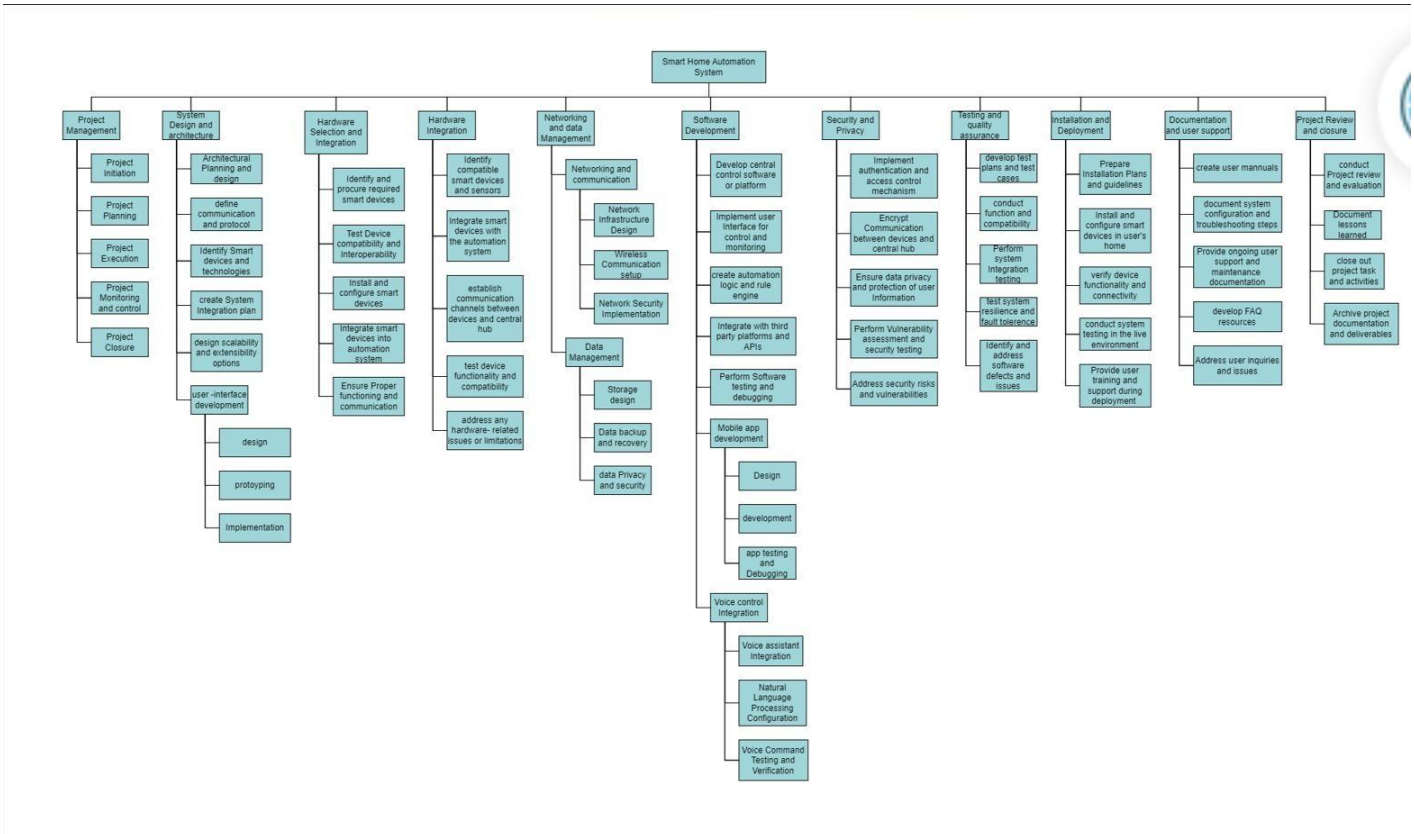
Title - Work Breakdown Structure for our project.
Date: 14 May, 2023

Aim : **To draw Work breakdown structure - role based for " Smart Home Automation System App".**

Description:

A Work Breakdown Structure (WBS) is a hierarchical decomposition of a project or deliverable into smaller, more manageable components. It represents the work that needs to be accomplished to complete the project and serves as a framework for organizing, planning, and controlling the project activities.The WBS is typically represented as a hierarchical chart or tree structure, with the highest level representing the project or deliverable, and subsequent levels breaking down the work into smaller components. Each element in the WBS should be well-defined, mutually exclusive, and collectively exhaustive, ensuring that all the work required for the project is accounted for.
By breaking the project into manageable pieces, the WBS allows for better project planning, resource allocation, and control, enabling project managers to effectively manage and execute complex projects. It serves as a foundation for other project management processes, such as scheduling, cost estimation, risk management, and quality control, facilitating a systematic and structured approach to project execution.

OUTPUT:



Result:
Thus the work breakdown structure is designed for planning the timeline of various activities for the project " Smart Home Automation System App".

EX-2c

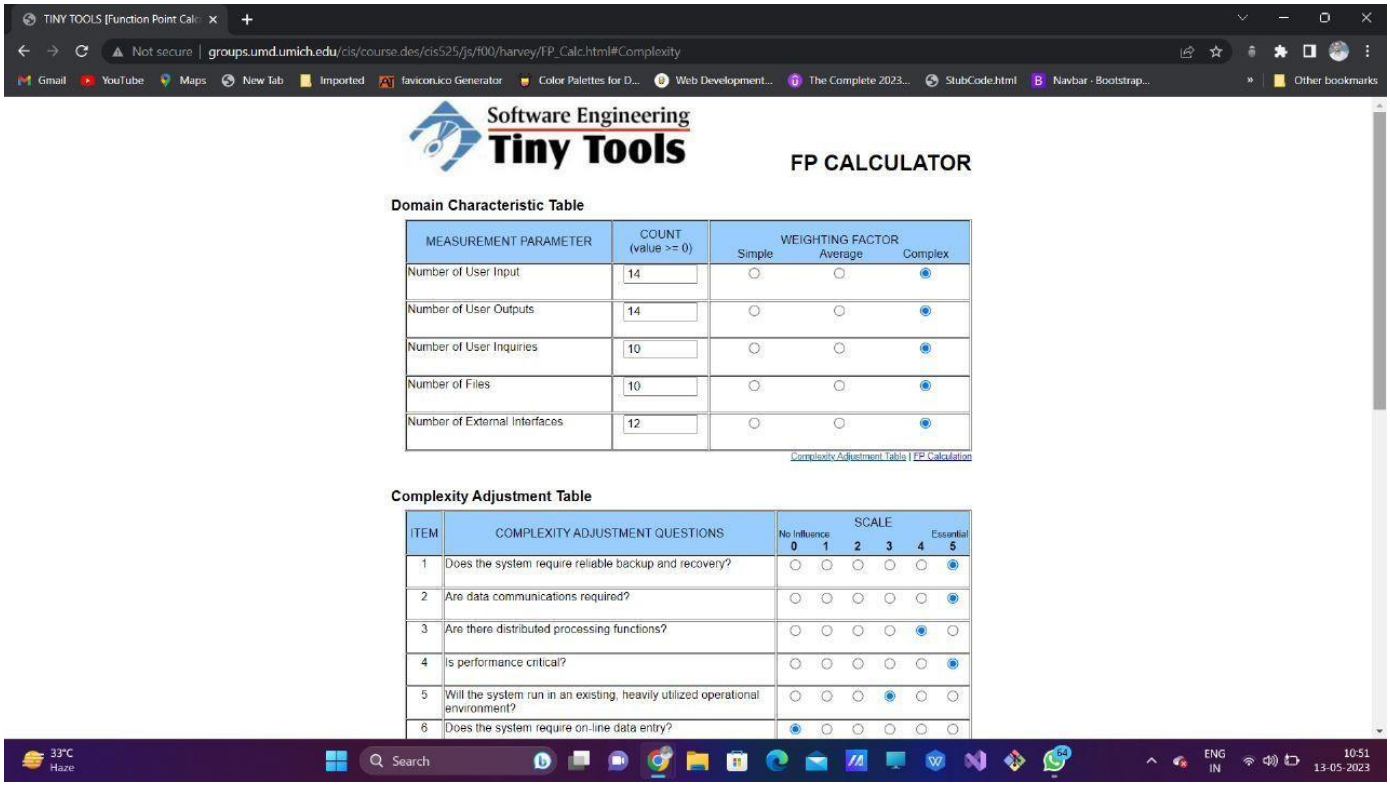
Title - FP based Estimation for our project.
Date: 14 May, 2023

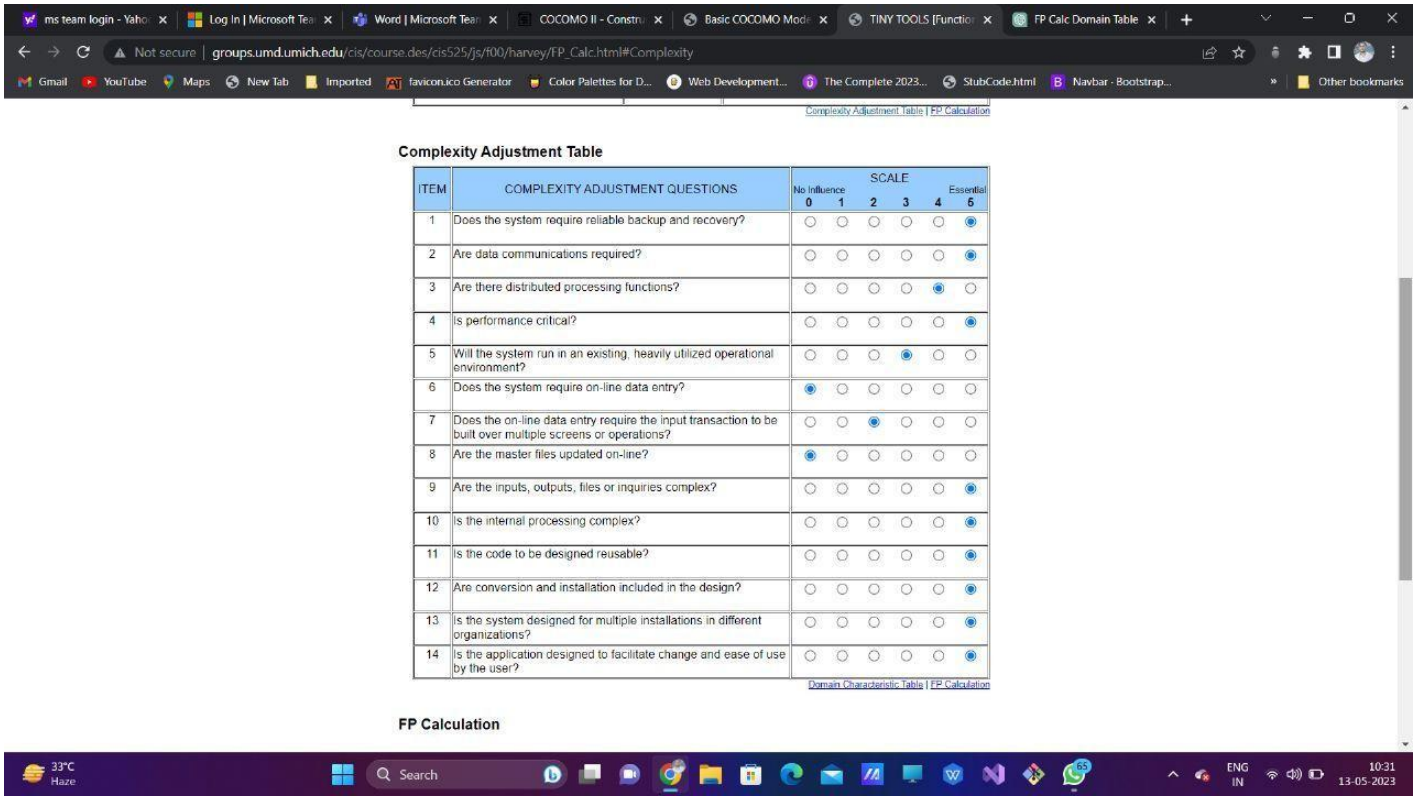
Aim :**To do FP based Estimation for " Smart Home Automation System App".**

Description:

FP-based cost estimation is a method used to estimate the effort, cost, and resources required for software development projects based on the number of function points in the software system. Function points represent the measure of the software's functionality and are calculated based on the inputs, outputs, inquiries, and data files it processes.FP-based cost estimation provides a standardized and objective approach to estimating software development effort and cost. It allows for comparability across different projects and helps in resource planning, budgeting, and project scheduling. However, it is important to note that FP-based cost estimation is based on assumptions and historical data, and the accuracy of the estimation relies on the quality of the function point analysis and the relevance of the productivity factor used. It is recommended to consider other factors, such as project complexity, technology factors, team experience, and environmental constraints, to refine and adjust the estimation to better reflect the specific characteristics of the project.

OUTPUT:





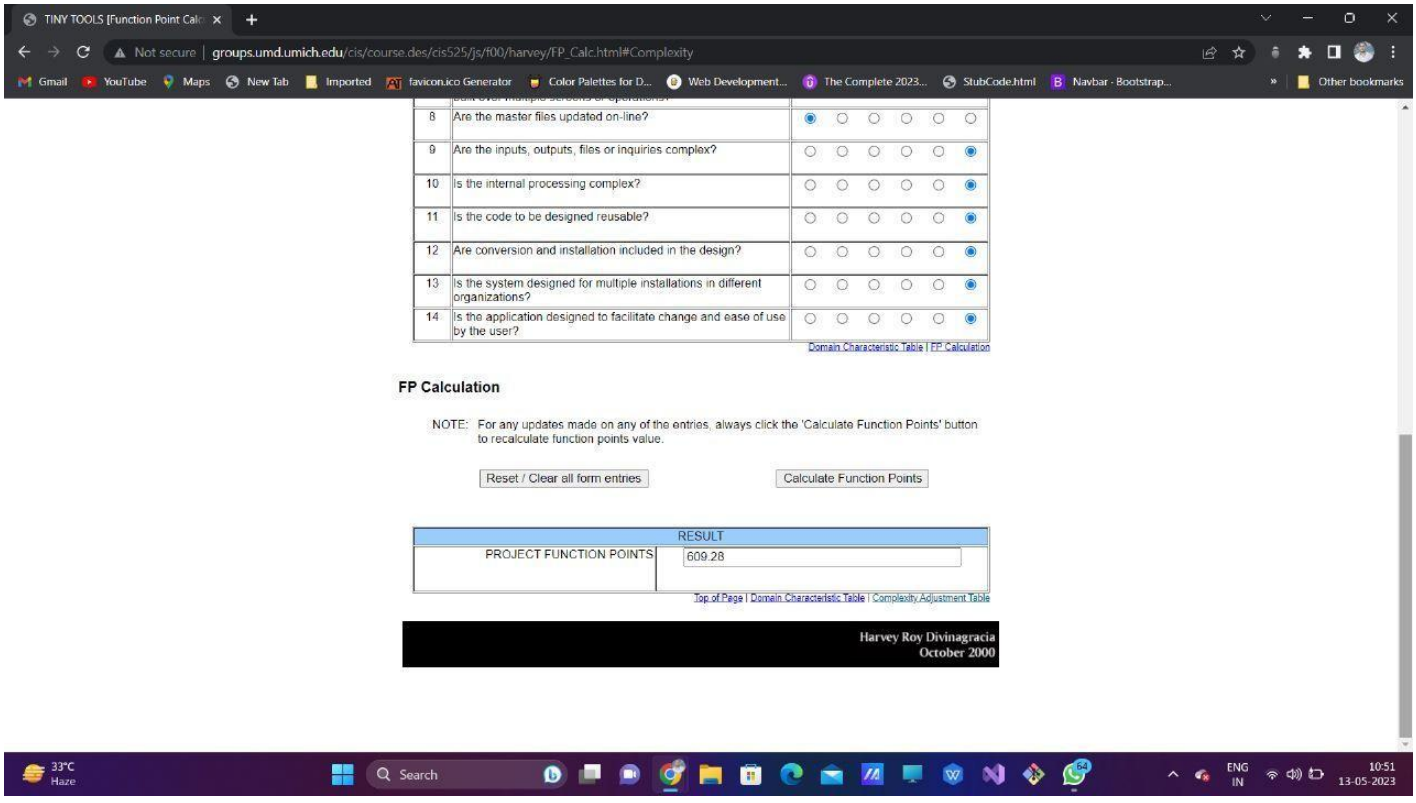
Complexity Adjustment Table | FP Calculation

Complexity Adjustment Table

ITEM	COMPLEXITY ADJUSTMENT QUESTIONS	SCALE (No Influence 0 1 2 3 4 Essential 5)
1	Does the system require reliable backup and recovery?	5
2	Are data communications required?	5
3	Are there distributed processing functions?	4
4	Is performance critical?	5
5	Will the system run in an existing, heavily utilized operational environment?	3
6	Does the system require on-line data entry?	0
7	Does the on-line data entry require the input transaction to be built over multiple screens or operations?	2
8	Are the master files updated on-line?	0
9	Are the inputs, outputs, files or inquiries complex?	5
10	Is the internal processing complex?	5
11	Is the code to be designed reusable?	5
12	Are conversion and installation included in the design?	5
13	Is the system designed for multiple installations in different organizations?	5
14	Is the application designed to facilitate change and ease of use by the user?	5

Domain Characteristic Table | FP Calculation

FP Calculation



8 Are the master files updated on-line? 0
9 Are the inputs, outputs, files or inquiries complex? 5
10 Is the internal processing complex? 5
11 Is the code to be designed reusable? 5
12 Are conversion and installation included in the design? 5
13 Is the system designed for multiple installations in different organizations? 5
14 Is the application designed to facilitate change and ease of use by the user? 5

Domain Characteristic Table | FP Calculation

FP Calculation

NOTE: For any updates made on any of the entries, always click the 'Calculate Function Points' button to recalculate function points value.

Reset / Clear all form entries | Calculate Function Points

RESULT	
PROJECT FUNCTION POINTS	609.28

Top of Page | Domain Characteristic Table | Complexity Adjustment Table

Harvey Roy Divinagracia
October 2000

Result:

Thus the FPbased Estimation is done for planning the timeline of various activities for the project“ Smart Home Automation System App” and Project Function Point is coming as 609.28.

EX-2d

Title - COCOMO based COST Estimation for our project.

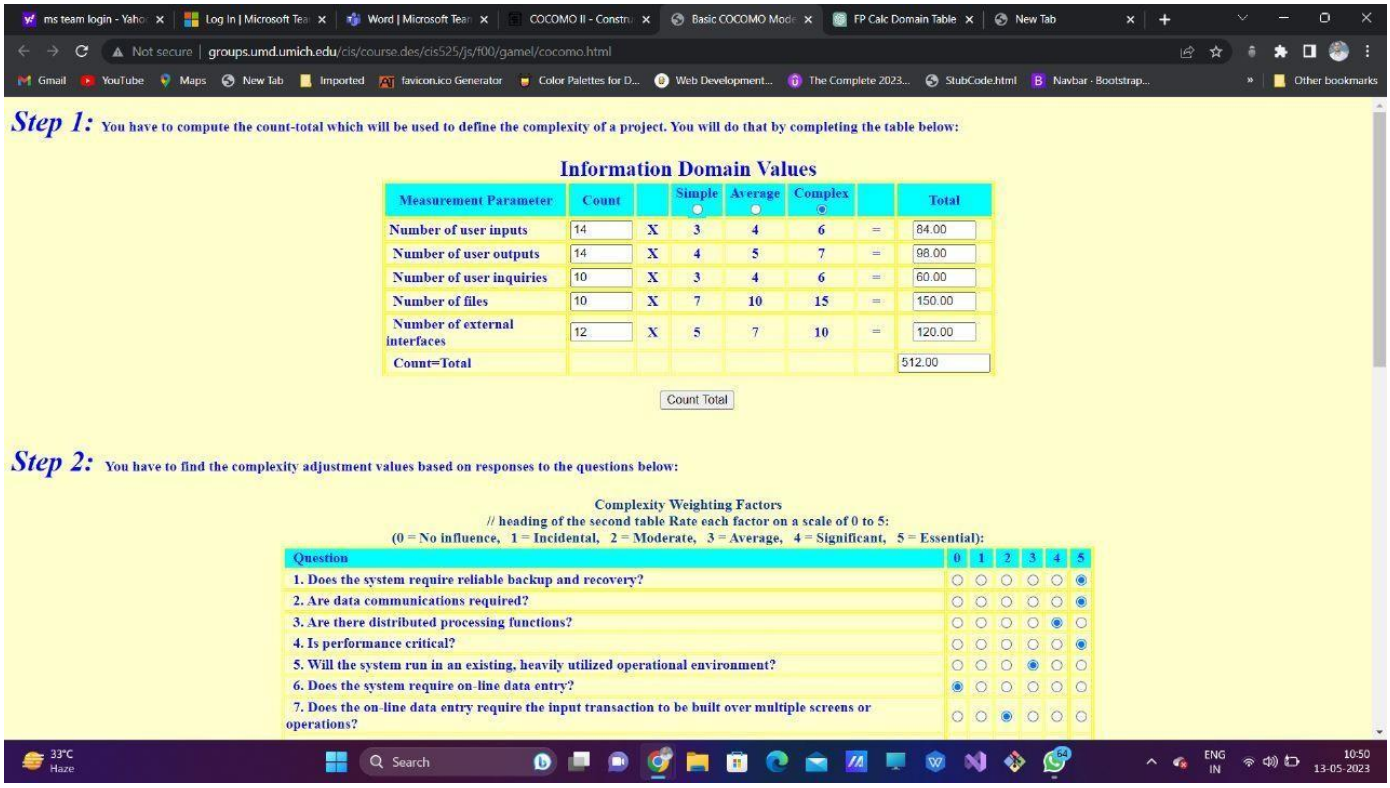
Date: 14 May, 2023

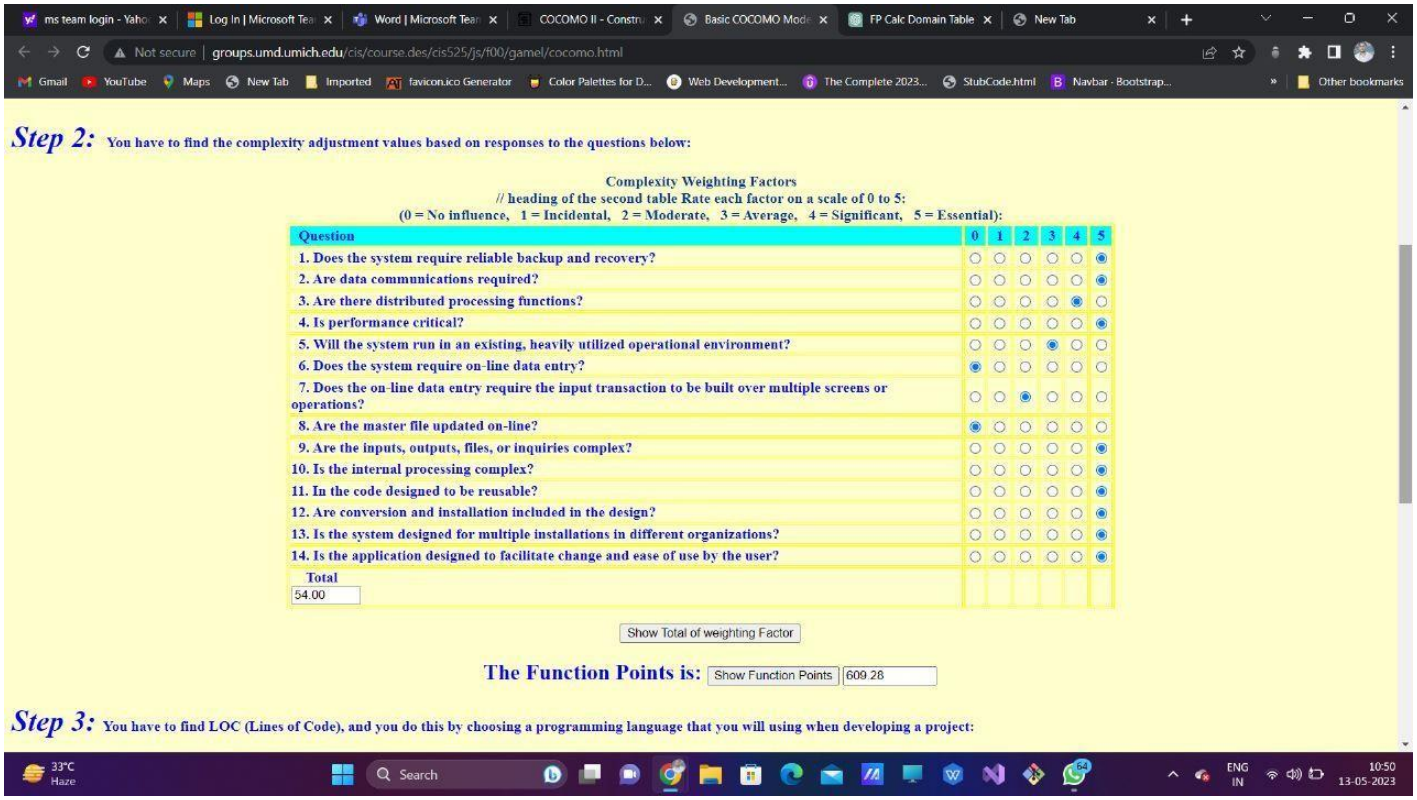
Aim : **To do COCOMO based cost Estimation for " Smart Home Automation System App".**

Description:

COCOMO (COnstructive COst MOdel) is a well-known software cost estimation model that was originally developed by Barry Boehm in the 1980s. It is a parametric model that uses a set of equations and factors to estimate the effort, cost, and duration of software development projects based on various project characteristics.
COCOMO estimation provides a structured and systematic approach to estimating software development effort and cost. It helps in project planning, resource allocation, budgeting, and risk management. However, it is important to note that COCOMO estimation is based on assumptions and historical data, and the accuracy of the estimation depends on the accuracy of the size estimation, the relevance of the cost drivers used, and the quality of the historical data available. It is recommended to calibrate and adjust the estimation based on project-specific factors and expert judgment to improve accuracy.

OUTPUT:





Step 2: **You have to find the complexity adjustment values based on responses to the questions below:**

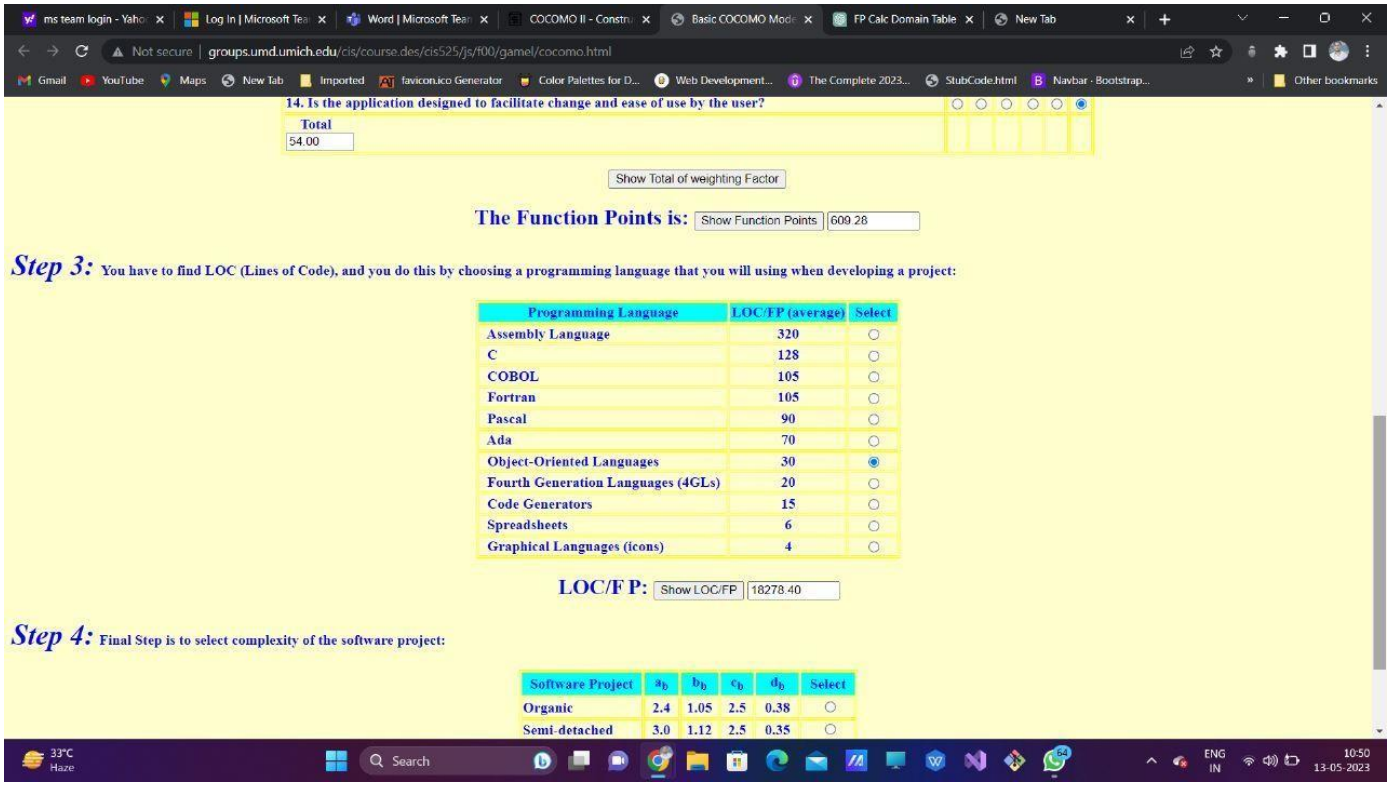
Complexity Weighting Factors
// heading of the second table Rate each factor on a scale of 0 to 5:
(0 = No influence, 1 = Incidental, 2 = Moderate, 3 = Average, 4 = Significant, 5 = Essential):

Question	0	1	2	3	4	5
1. Does the system require reliable backup and recovery?	○	○	○	○	○	◉
2. Are data communications required?	○	○	○	○	○	◉
3. Are there distributed processing functions?	○	○	○	○	◉	○
4. Is performance critical?	○	○	○	○	○	◉
5. Will the system run in an existing, heavily utilized operational environment?	○	○	○	◉	○	○
6. Does the system require on-line data entry?	◉	○	○	○	○	○
7. Does the on-line data entry require the input transaction to be built over multiple screens or operations?	○	○	◉	○	○	○
8. Are the master file updated on-line?	◉	○	○	○	○	○
9. Are the inputs, outputs, files, or inquiries complex?	○	○	○	○	○	◉
10. Is the internal processing complex?	○	○	○	○	○	◉
11. In the code designed to be reusable?	○	○	○	○	○	◉
12. Are conversion and installation included in the design?	○	○	○	○	○	◉
13. Is the system designed for multiple installations in different organizations?	○	○	○	○	○	◉
14. Is the application designed to facilitate change and ease of use by the user?	○	○	○	○	○	◉
Total 54.00						

Show Total of weighting Factor

The Function Points is: Show Function Points 609.28

Step 3: **You have to find LOC (Lines of Code), and you do this by choosing a programming language that you will using when developing a project:**

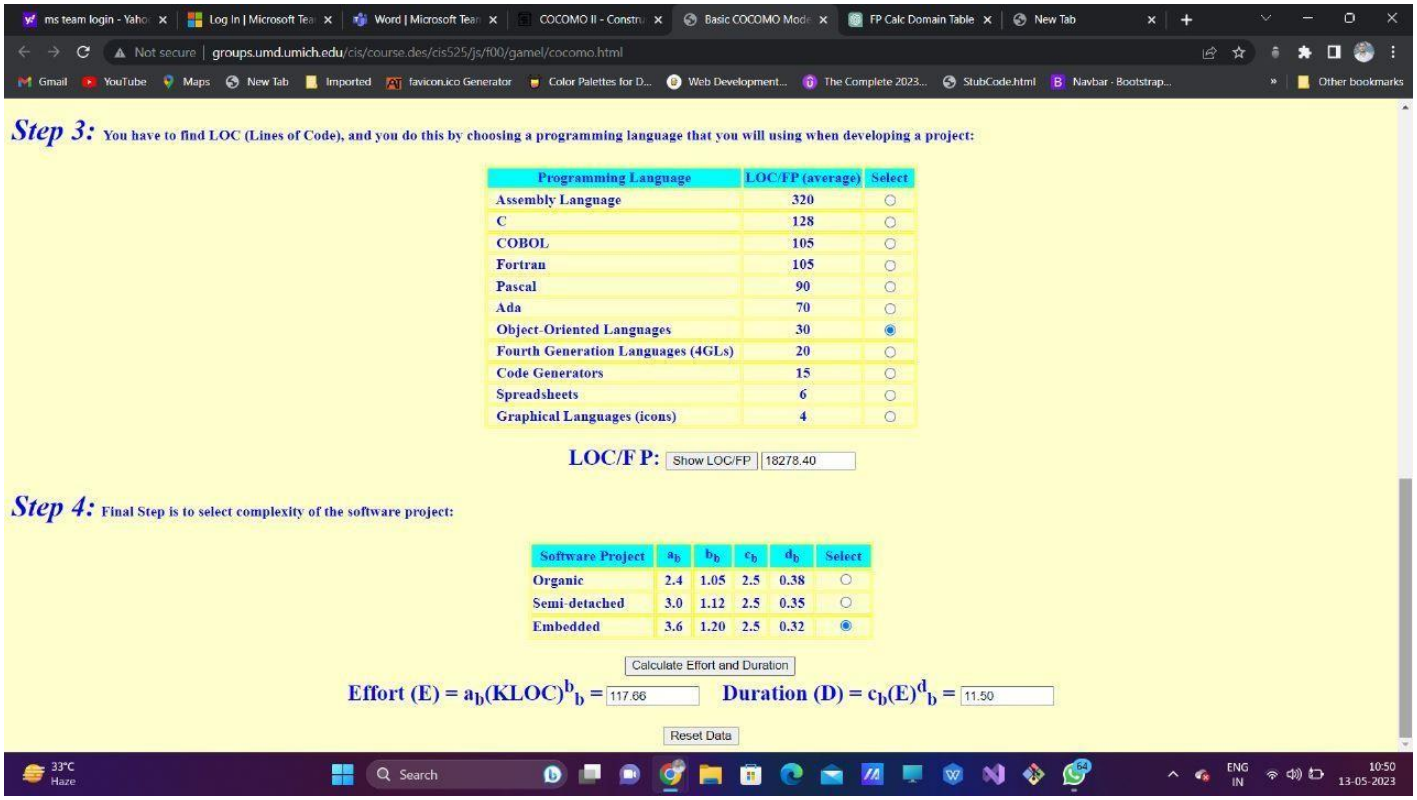


Programming Language	LOC/FP (average)	Select
Assembly Language	320	○
C	128	○
COBOL	105	○
Fortran	105	○
Pascal	90	○
Ada	70	○
Object-Oriented Languages	30	◉
Fourth Generation Languages (4GLs)	20	○
Code Generators	15	○
Spreadsheets	6	○
Graphical Languages (icons)	4	○

LOC/F P: Show LOC/FP 18278.40

Step 4: **Final Step is to select complexity of the software project:**

Software Project	a_b	b_b	c_b	d_b	Select
Organic	2.4	1.05	2.5	0.38	○
Semi-detached	3.0	1.12	2.5	0.35	○



Step 3: **You have to find LOC (Lines of Code), and you do this by choosing a programming language that you will using when developing a project:**

Programming Language	LOC/FP (average)	Select
Assembly Language	320	○
C	128	○
COBOL	105	○
Fortran	105	○
Pascal	90	○
Ada	70	○
Object-Oriented Languages	30	◉
Fourth Generation Languages (4GLs)	20	○
Code Generators	15	○
Spreadsheets	6	○
Graphical Languages (icons)	4	○

LOC/F P: Show LOC/FP 18278.40

Step 4: **Final Step is to select complexity of the software project:**

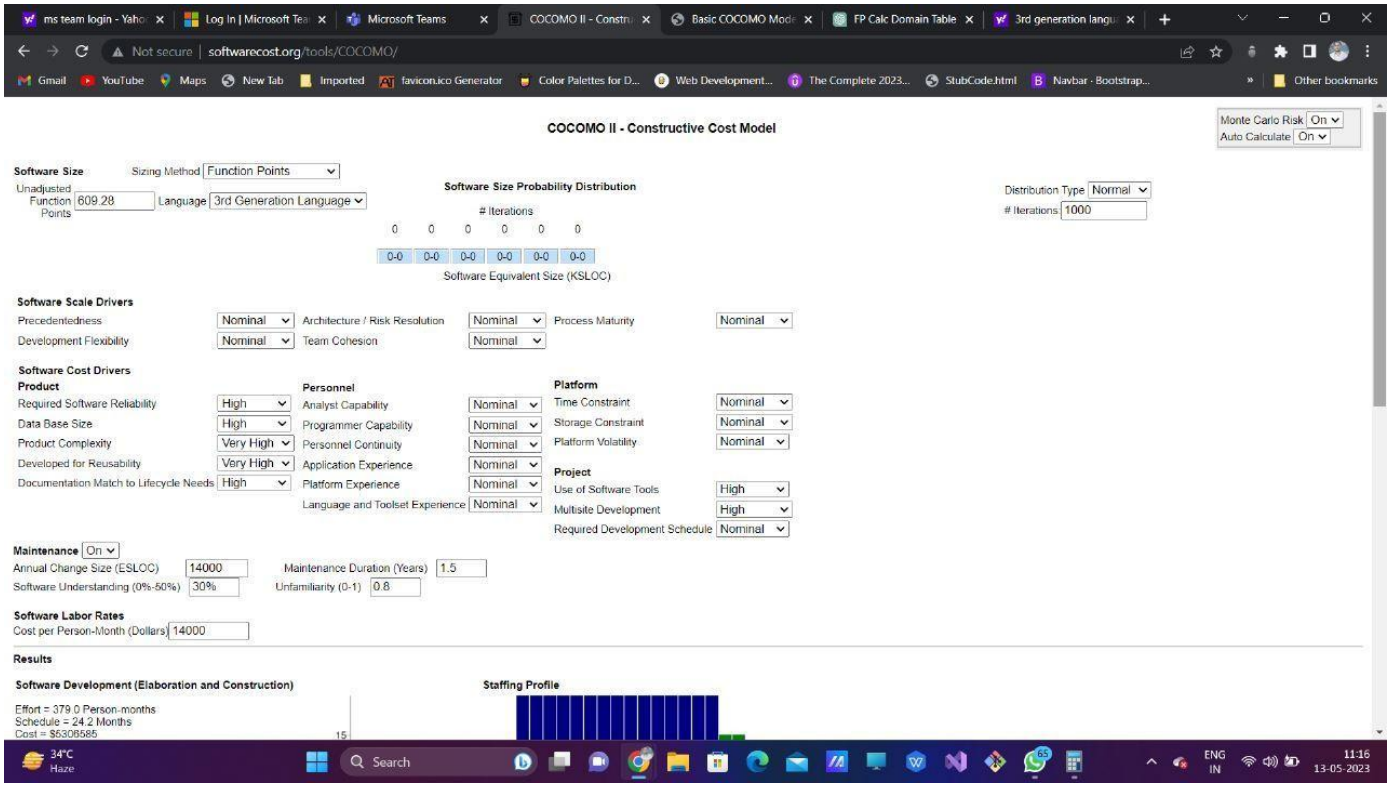
Software Project	a_b	b_b	c_b	d_b	Select
Organic	2.4	1.05	2.5	0.38	○
Semi-detached	3.0	1.12	2.5	0.35	○
Embedded	3.6	1.20	2.5	0.32	◉

Calculate Effort and Duration

Effort (E) = $a_b(KLOC)^b_b$ = 117.66 **Duration (D) = $c_b(E)^d_b$ =** 11.50

Reset Data

COCOMO II CONSTRUCTIVE COST MODEL



COCOMO II - Constructive Cost Model

Monte Carlo Risk On
Auto Calculate On

Software Size Sizing Method Function Points
Unadjusted Function Points 609.28 Language 3rd Generation Language

Software Size Probability Distribution
Iterations
0 0 0 0 0 0
0-0 0-0 0-0 0-0 0-0 0-0
Software Equivalent Size (KSLOC)

Distribution Type Normal
Iterations 1000

Software Scale Drivers
Precedentedness Nominal | Architecture / Risk Resolution Nominal | Process Maturity Nominal
Development Flexibility Nominal | Team Cohesion Nominal

Software Cost Drivers

Product		Personnel		Platform	
Required Software Reliability	High	Analyst Capability	Nominal	Time Constraint	Nominal
Data Base Size	High	Programmer Capability	Nominal	Storage Constraint	Nominal
Product Complexity	Very High	Personnel Continuity	Nominal	Platform Volatility	Nominal
Developed for Reusability	Very High	Application Experience	Nominal	**Project**	
Documentation Match to Lifecycle Needs	High	Platform Experience	Nominal	Use of Software Tools	High
		Language and Toolset Experience	Nominal	Multisite Development	High
				Required Development Schedule	Nominal

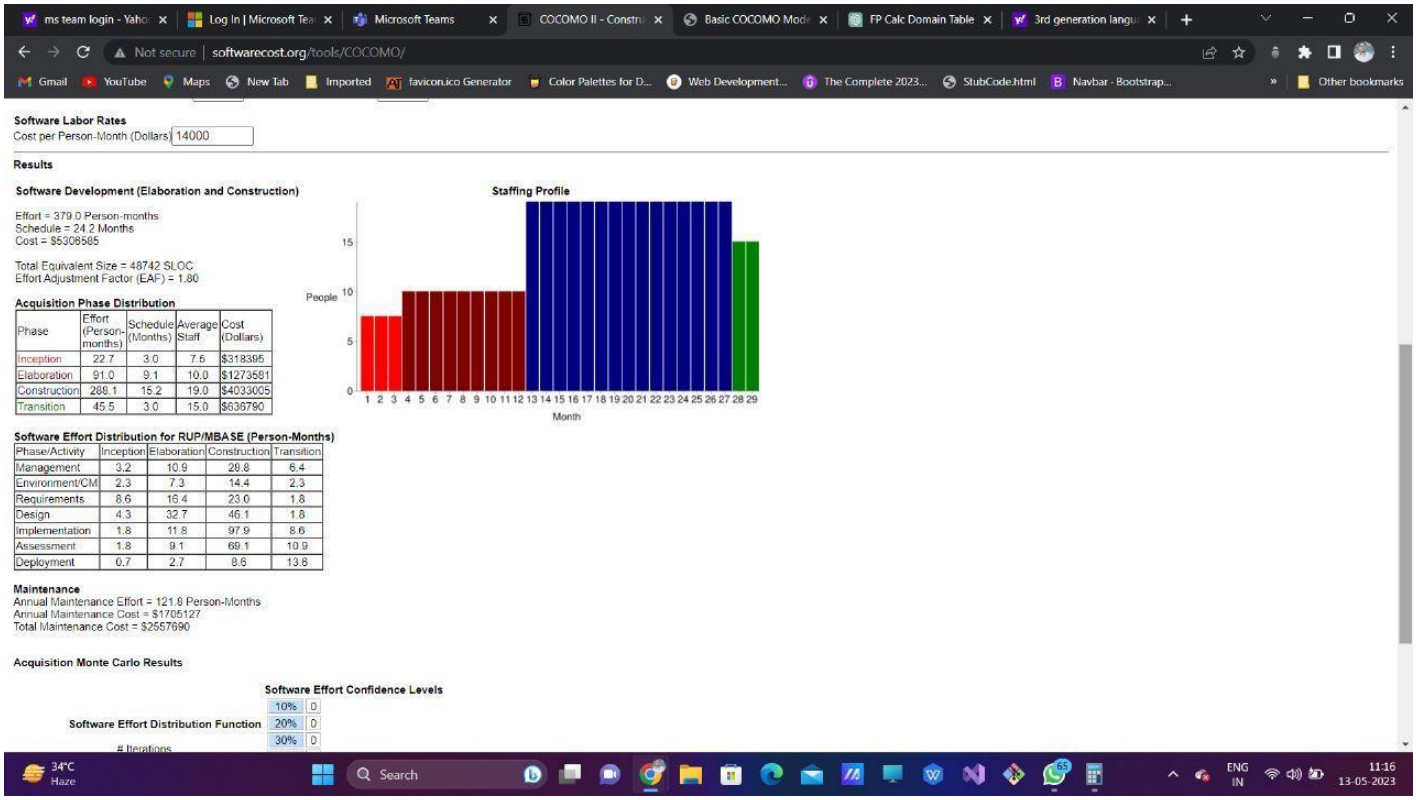
Maintenance On
Annual Change Size (ESLOC) 14000 Maintenance Duration (Years) 1.5
Software Understanding (0%-50%) 30% Unfamiliarity (0-1) 0.8

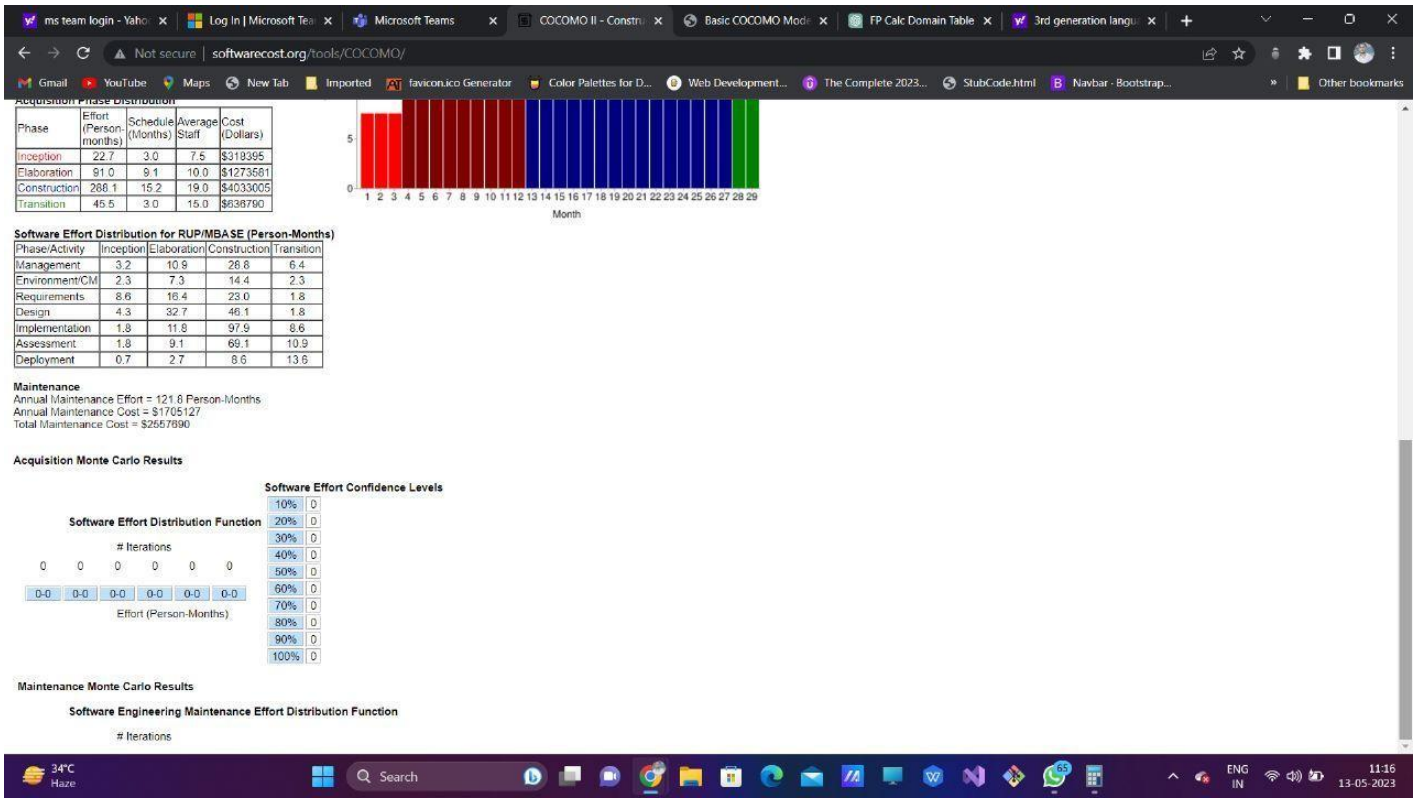
Software Labor Rates
Cost per Person-Month (Dollars) 14000

Results

Software Development (Elaboration and Construction)
Effort = 379.0 Person-months
Schedule = 24.2 Months
Cost = $5306585

Staffing Profile





Result:

Thus the COCOMO based cost Estimation is done for planning the timeline of various activities for the project " Smart Home Automation System App" .

NAME- AMISHA TRIPATHI

REGISTRATION NUMBER- 21BCE1735

EXPERIMENT- 3

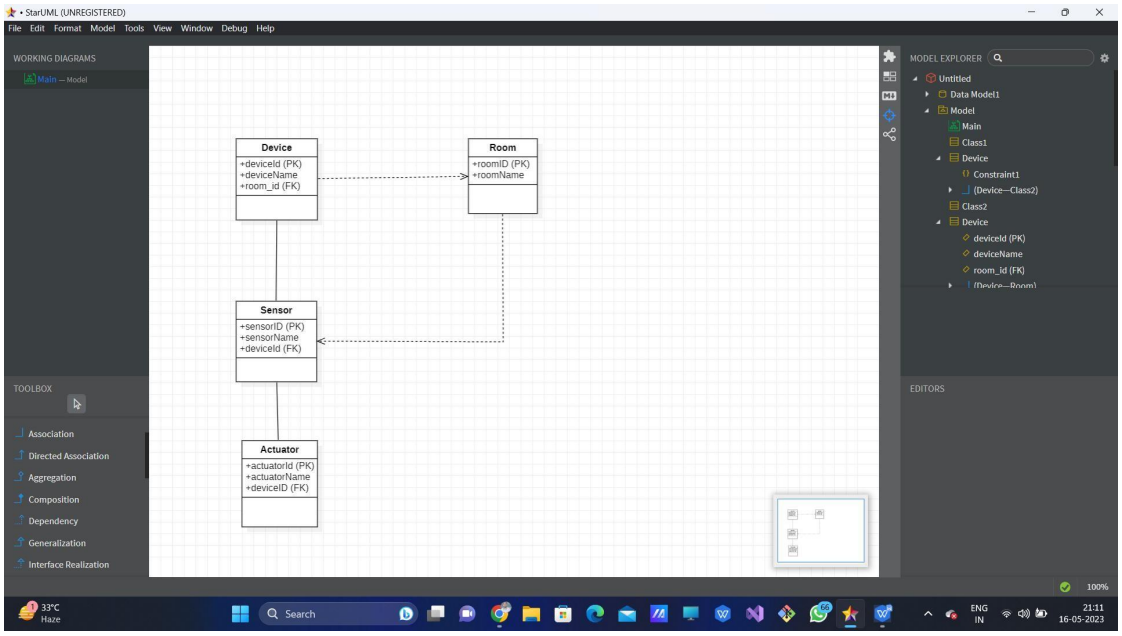
TITLE- ER DIAGRAM FOR OUR PROJECT

AIM- Creating a ER- Diagram for our Project "SMART HOME AUTOMATION SYSTEM".

DESCRIPTION

An Entity-Relationship (ER) diagram is a visual representation of the entities (objects or concepts) in a system or domain, their attributes (properties or characteristics), and the relationships between them. It is a widely used modeling technique in software engineering and database design.

OUTPUT:



In this ER diagram, we have four main entities:

- Device: Represents a smart device in the system. It has attributes such as deviceId (primary key), deviceName, and room_id (foreign key) to associate it with a specific room.

- Room: Represents a room in the house. It has attributes such as roomId (primary key) and roomName to identify the room.

- Sensor: Represents a sensor attached to a device. It has attributes such as sensorId (primary key), sensorName, and deviceId (foreign key) to associate it with a specific device.

- Actuator: Represents an actuator attached to a device. It has attributes such as actuatorId (primary key), actuatorName, and deviceId (foreign key) to associate it with a specific device.

The relationships in the diagram are as follows:

1. Each Device is associated with one Room (represented by the room_id foreign key).
2. Each Sensor is associated with one Device (represented by the deviceId foreign key).
3. Each Actuator is associated with one Device (represented by the deviceId foreign key).

RESULT:

THUS THE ENTITY-RELATIONSHIP MODEL SHOWS THE ENTITIES AND RELATIONSHIP BETWEEN THEM .This ER diagram provides a basic structure for modeling a **Smart Home Automation System.**

NAME- AMISHA TRIPATHI

REGISTRATION NUMBER- 21BCE1735

EXPERIMENT- 4

TITLE- CREATING A DATA FLOW DIAGRAM

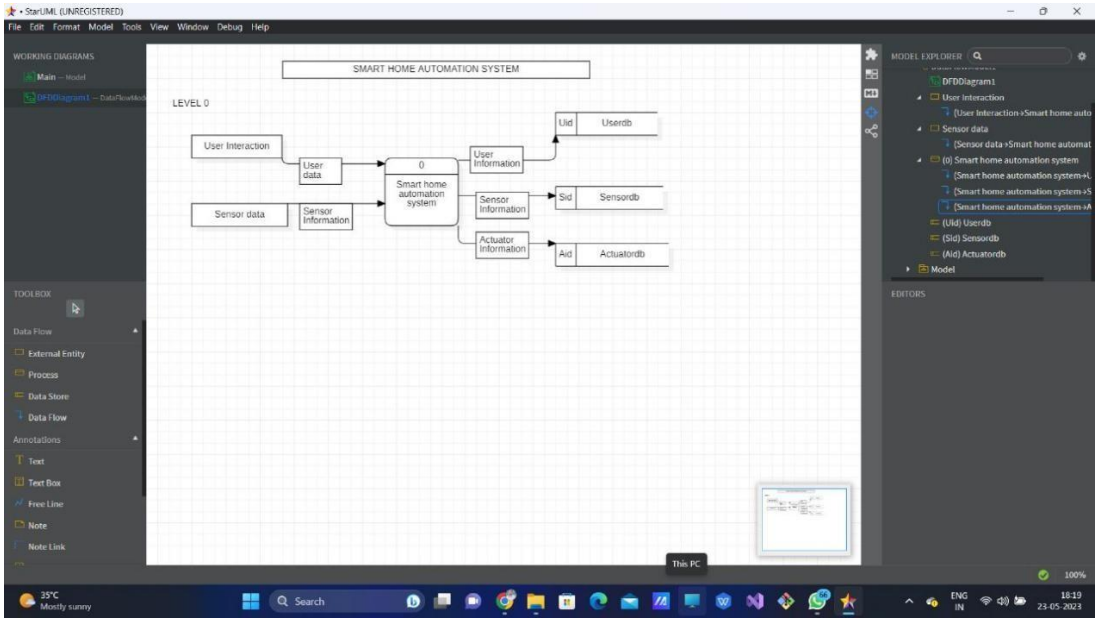
AIM- Creating a Data Flow Diagram for our Project **"SMART HOME AUTOMATION SYSTEM"**.

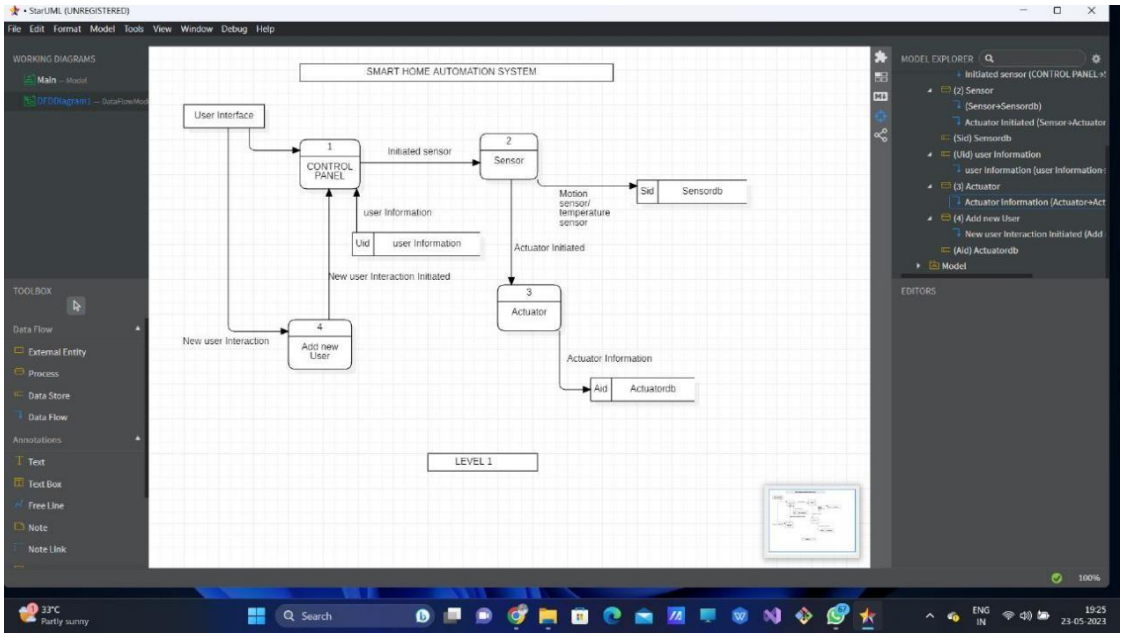
DESCRIPTION

A Data Flow Diagram (DFD) is a graphical representation of the flow of data within a system or process. It illustrates the movement of data between various components, processes, and external entities involved in a system. DFDs are commonly used in system analysis and design to visualize and communicate the structure and behavior of a system.

In a DFD, data flows are represented by arrows, which indicate the direction of data movement. The components or processes that interact with the data are represented by rectangles, and external entities (such as users, systems, or databases) are represented by rounded rectangles. Data stores, where data is temporarily or permanently stored, are represented by parallel lines.

OUTPUT:





RESULT:

The DFD provides a simplified overview of the data flows and processes within a **"Smart home automation system"**.

NAME- AMISHA TRIPATHI

REGISTRATION NUMBER- 21BCE1735

EXPERIMENT- 5

TITLE- Use case and Class Modeling

AIM- Creating a Use case and Class Modeling Diagram for our Project "SMART HOME AUTOMATION SYSTEM".

DESCRIPTION

- **Use Case Diagram**

A use case diagram is a visual representation of the functional requirements of a system, showcasing the interactions between various actors (users, external systems, or other entities) and the system itself. It illustrates the specific actions or use cases that can be performed by the actors within the system.

Use case diagrams are widely used in software development to capture and document the system's behavior and the relationships between different actors and use cases. They provide a high-level overview of the system's functionality and help in identifying the system's boundaries and scope.

Key components of a use case diagram include:

Actors: Actors represent the entities (users, external systems, or other systems) that interact with the system and perform specific roles. Actors are depicted as stick figures or labeled blocks.

Use Cases: Use cases represent the specific actions or functionalities that the system provides. They describe the behavior of the system from the perspective of the actors. Use cases are depicted as ovals or ellipses and are labeled with descriptive names.

Relationships: Relationships depict the associations and dependencies between actors and use cases. The primary relationship in a use case diagram is the association between an actor and a use case, indicating that the actor interacts with or uses that particular functionality.

System Boundary: The system boundary is a box or rectangle that encloses all the actors and use cases within the diagram, representing the scope and boundaries of the system.

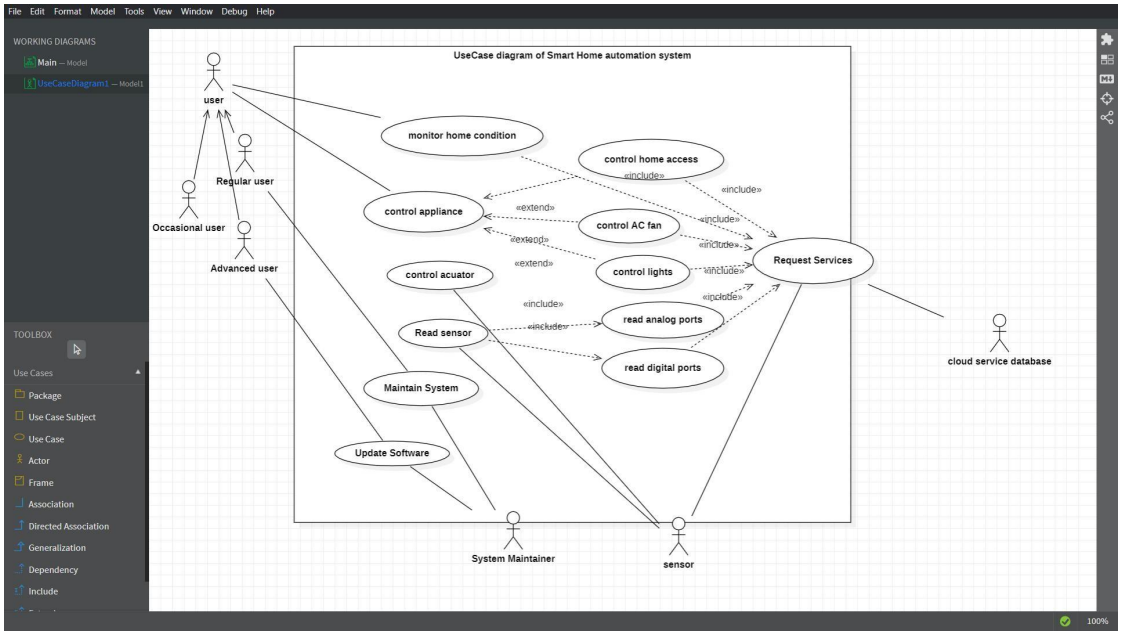
- **Class Modelling Diagram**

A class modeling diagram, also known as a class diagram, is a visual representation of the static structure of a system, showcasing the classes, their attributes, methods, relationships, and associations. It provides a high-level overview of the classes and their interactions in the system.

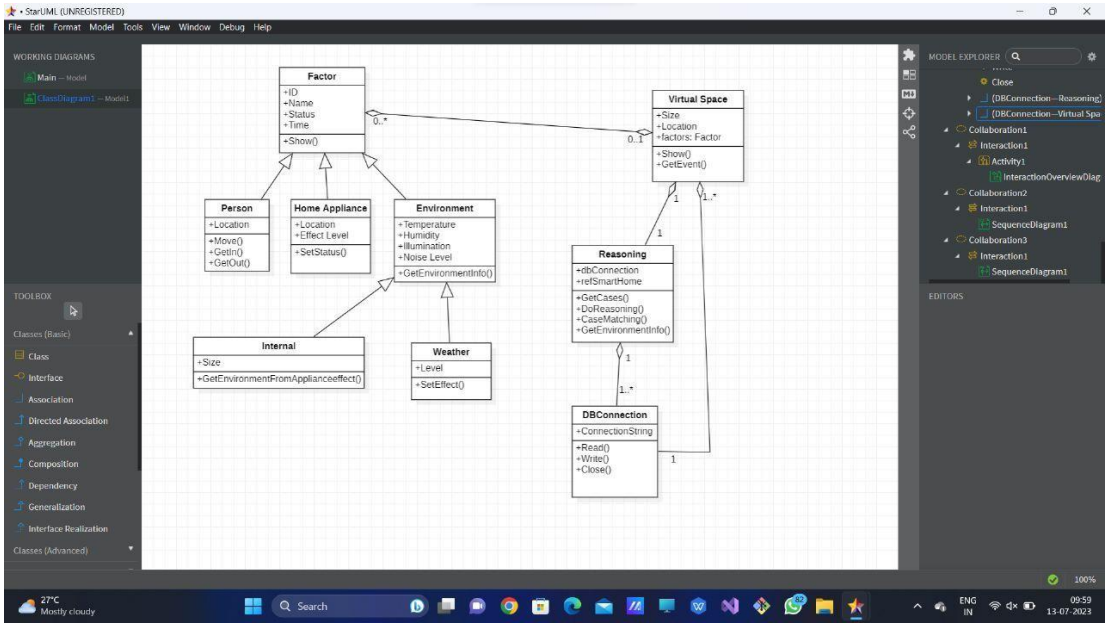
Class diagrams are widely used in object-oriented analysis and design to illustrate the relationships between classes, inheritance hierarchies, and the overall structure of the system.

OUTPUT:

Use Case Diagram



Class Diagram



RESULT:

Use case diagrams and Class Modelling Diagram provide a clear understanding of the system's requirements, facilitating communication between stakeholders, analysts, and developers for Smart Home Automation System.

NAME- AMISHA TRIPATHI

REGISTRATION NUMBER- 21BCE1735

EXPERIMENT- 6

TITLE- Interaction Diagram

AIM- Creating a Interaction Diagram(Smart Home Automation System) for our Project "SMART HOME AUTOMATION SYSTEM".

DESCRIPTION

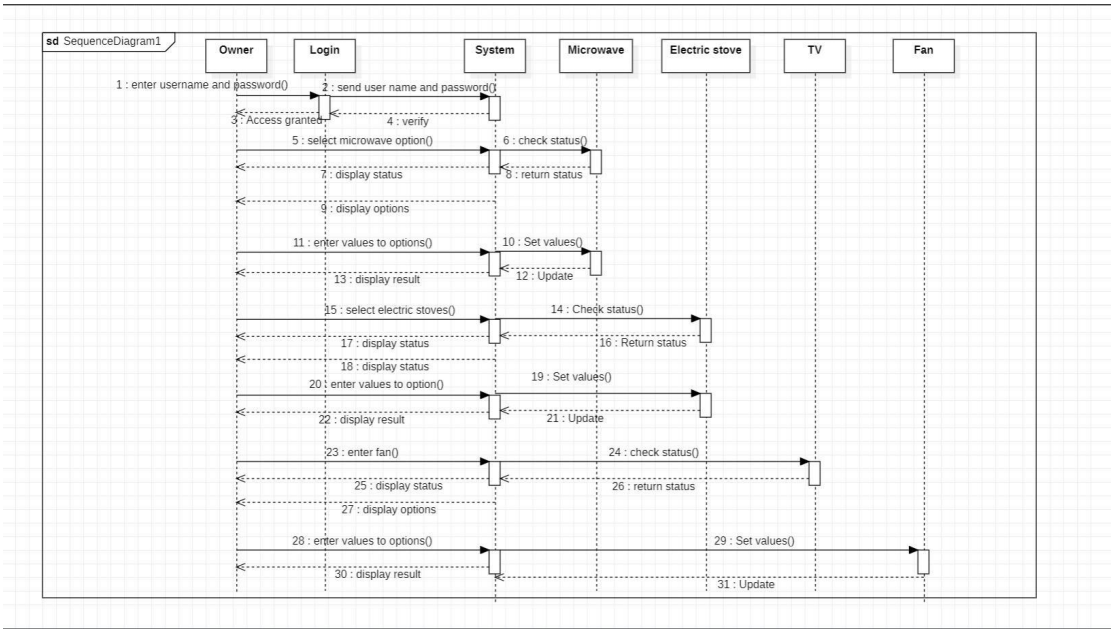
Interaction Diagram:

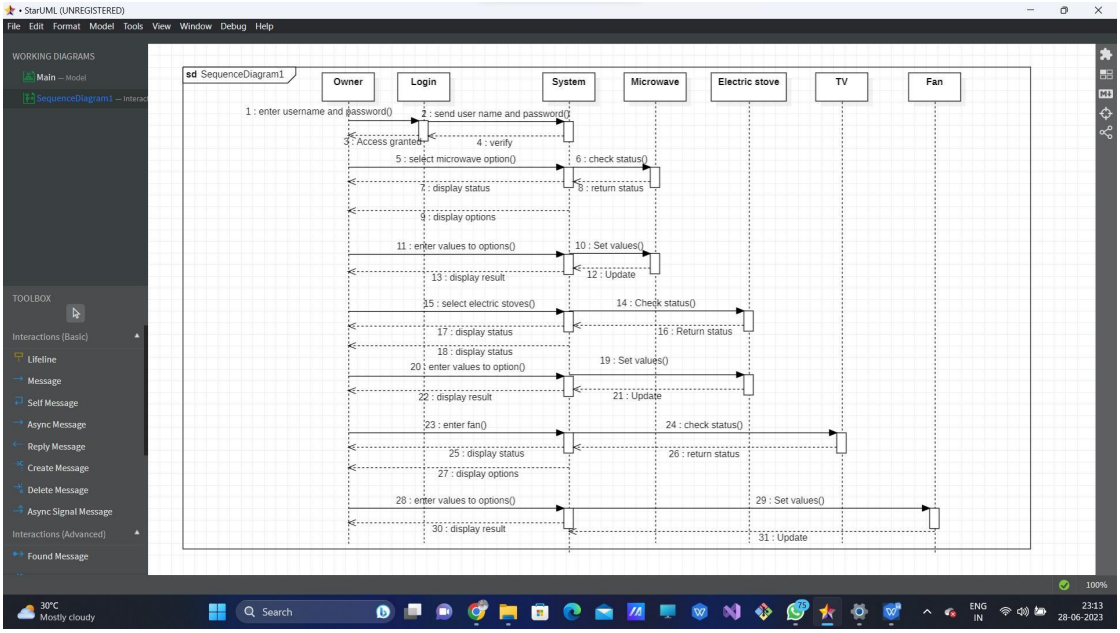
An interaction diagram is a type of UML (Unified Modeling Language) diagram that depicts the interactions and communications between objects or components within a system. It illustrates how different elements of a system collaborate and exchange messages to accomplish specific tasks or achieve a particular behavior.

Both sequence and communication diagrams are effective tools for modeling and understanding the interactions in a system, but they offer slightly different perspectives. Sequence diagrams excel at showing the time-dependent flow of messages, while communication diagrams highlight the structural relationships between objects and the overall system architecture.

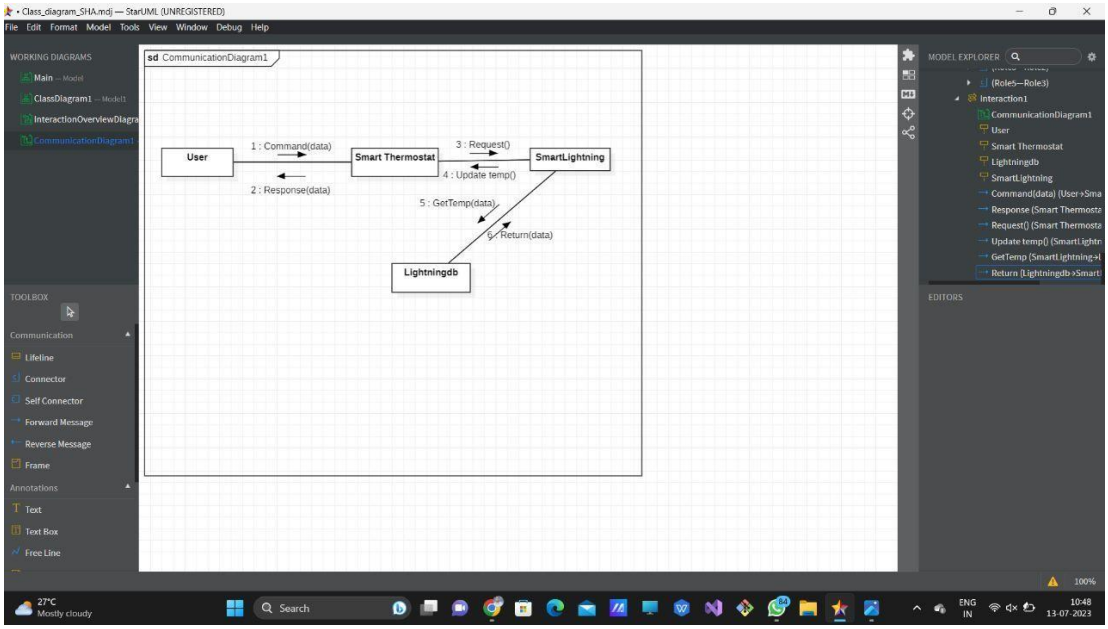
OUTPUT:

Sequence Diagram





Communication Diagram



RESULT:

Sequence and Communication diagram for Smart Home Automation System illustrates the flow of messages and interactions between the user, mobile application, and the smart home system, showing how commands are initiated, processed, and executed. It provides a high-level overview of the system's behavior during the automation process.

NAME- AMISHA TRIPATHI

REGISTRATION NUMBER- 21BCE1735

EXPERIMENT- 7

TITLE- State Transition Diagram

AIM- Creating a State Transition Diagram(Smart Home Automation System) for our Project "SMART HOME AUTOMATION SYSTEM".

DESCRIPTION

Interaction Diagram:

A state transition diagram, also known as a state machine diagram or state chart diagram, represents the various states and transitions of a system. In the case of a smart home automation system, the states and transitions can be defined as follows:

States:

Idle: The system is in a standby mode, waiting for user input or triggering events.
Active: The system is actively monitoring and controlling devices in the home.
Alert: The system has detected an abnormality or an emergency situation.
Maintenance: The system is undergoing maintenance or configuration changes.
Transitions:

User Interaction:

Idle -> Active: The user initiates the system by providing input or activating a control.
Active -> Idle: The user chooses to stop the automation and put the system into standby mode.
Sensor/Event Triggered:

Idle -> Active: A sensor or event triggers the system to start automation.
Active -> Alert: An abnormal condition or emergency event is detected.
Active -> Idle: The system stops automation due to a lack of sensor or event triggers.
Emergency Handling:

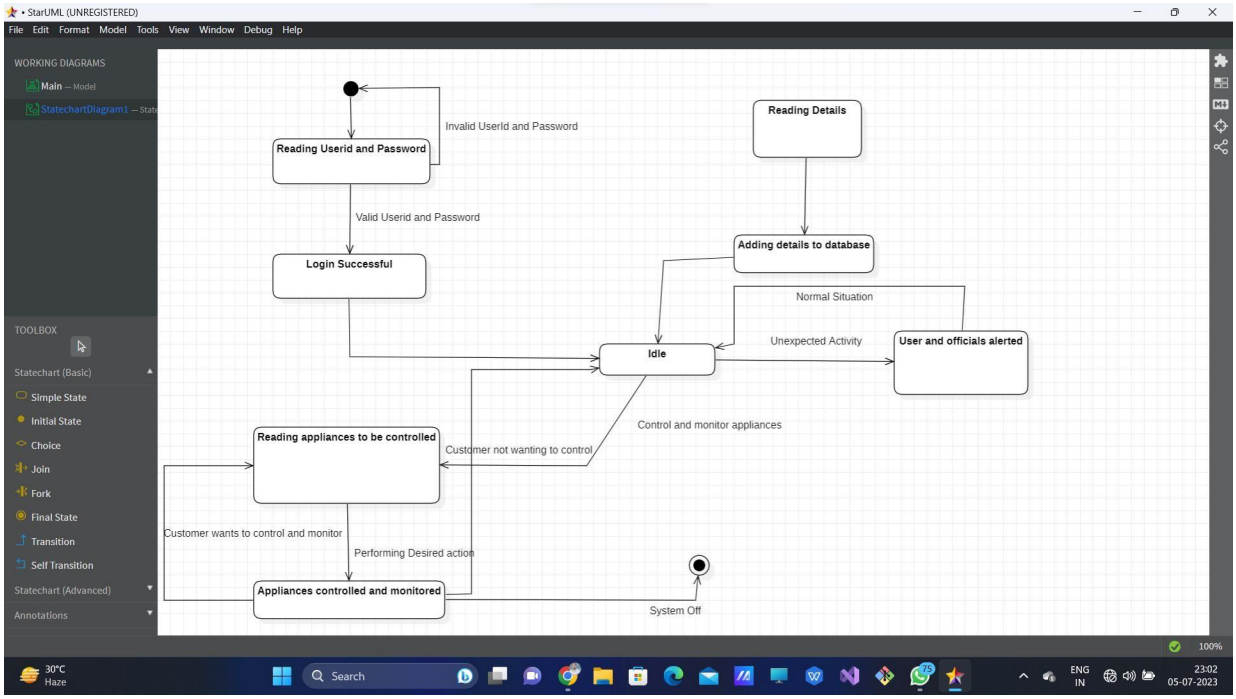
Alert -> Active: The system resolves the emergency situation and returns to normal operation.
Alert -> Idle: The user acknowledges the alert and chooses to disable the automation.
Maintenance:

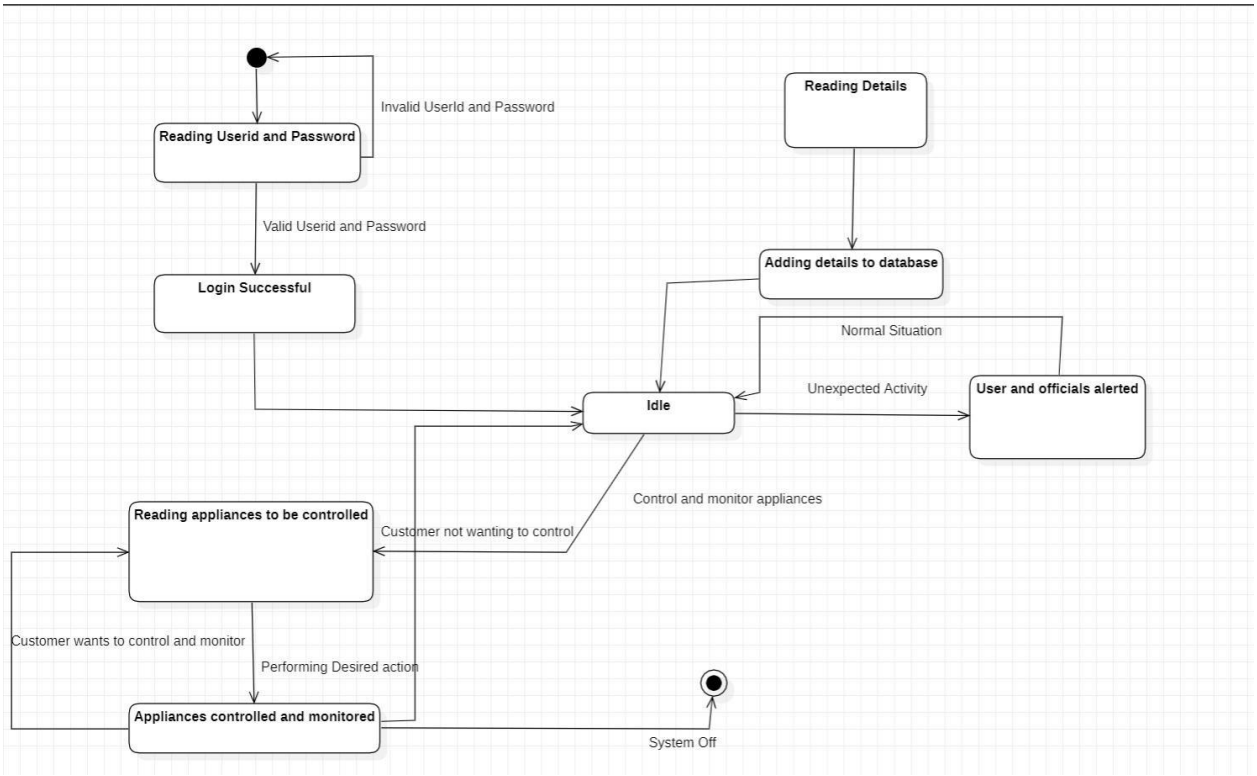
Idle -> Maintenance: The system enters a maintenance mode for configuration changes or updates.

Maintenance -> Idle: The maintenance or configuration changes are completed, and the system returns to standby mode.
These are general states and transitions that can be adapted and expanded based on the specific features and functionalities of the smart home automation system you have in mind.

OUTPUT:

State Transition Diagram





RESULT:

The State Transition of Smart Home Automation System represents the various states and transitions of a Smart Home.

Name of Student- Amisha Tripathi
Reg. No- 21BCE1735
Ex. No- 08

PACKAGE, COMPONENT AND DEPLOYEMENT MODELS

Aim

To prepare a package component and deployment diagram for **SMART HOME AUTOMATION SYSTEM** to show the organization of system components into cohesive packages, to show the internal structure of these components and their relationships, and to show how these components are deployed and interconnected across a distributed system respectively.

Tools Used

Star UML software

Description

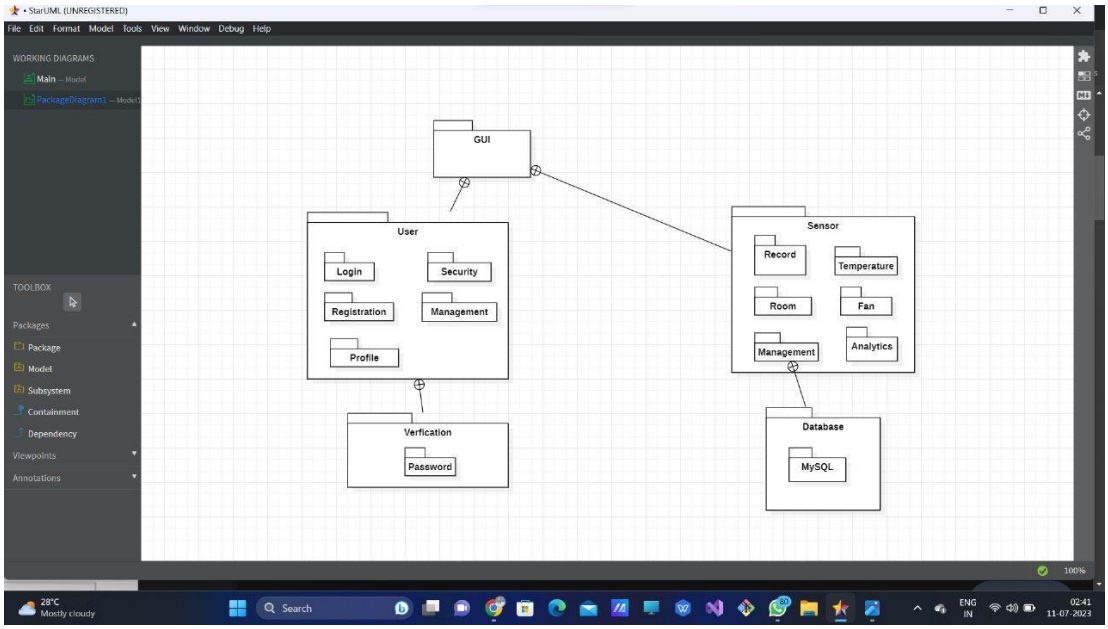
A package diagram is a type of UML diagram that represents the organization and structure of a software system by grouping related elements into cohesive packages. It shows the dependencies and relationships between these packages and their contents, and can be used to provide a high-level overview of the system's architecture and design. Package diagrams can be used to help stakeholders understand the overall structure of a software system, identify areas of cohesion and coupling, and support modular design and development practices. They can also be used to help manage complexity by breaking down a large system into smaller, more manageable components.

A component diagram is a type of UML diagram that shows the internal structure and composition of a software system by decomposing it into smaller, self-contained components. It focuses on the implementation aspect of the system and shows how its various components are connected and interact with each other to achieve a specific functionality.

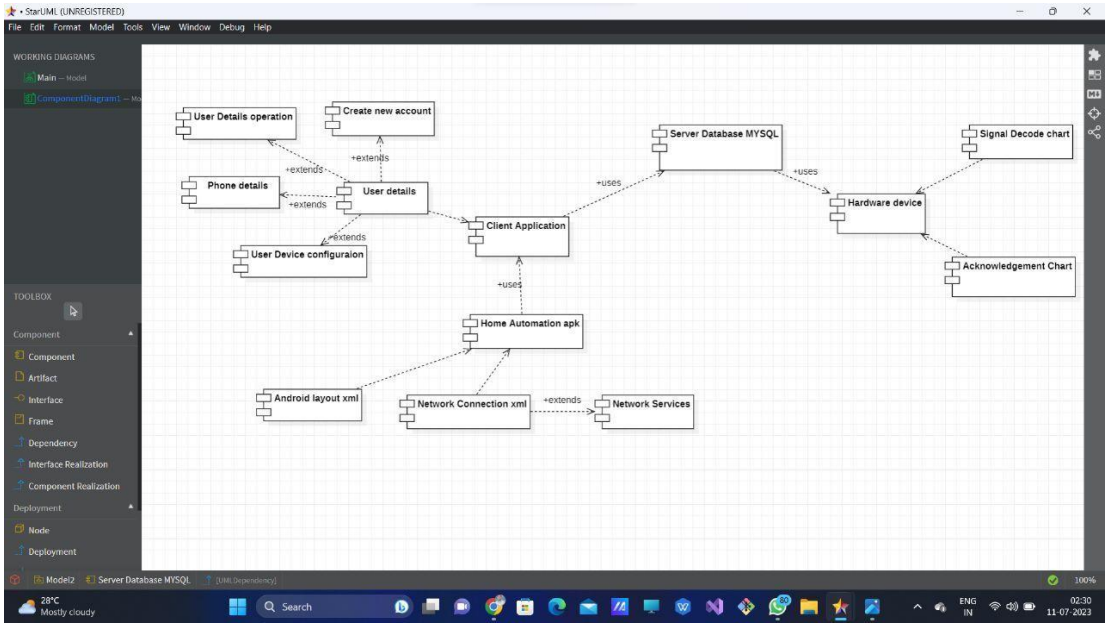
Deployment diagram represents the deployment view of a system. It is related to the component diagram because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware used to deploy the application.

Output Screenshot:

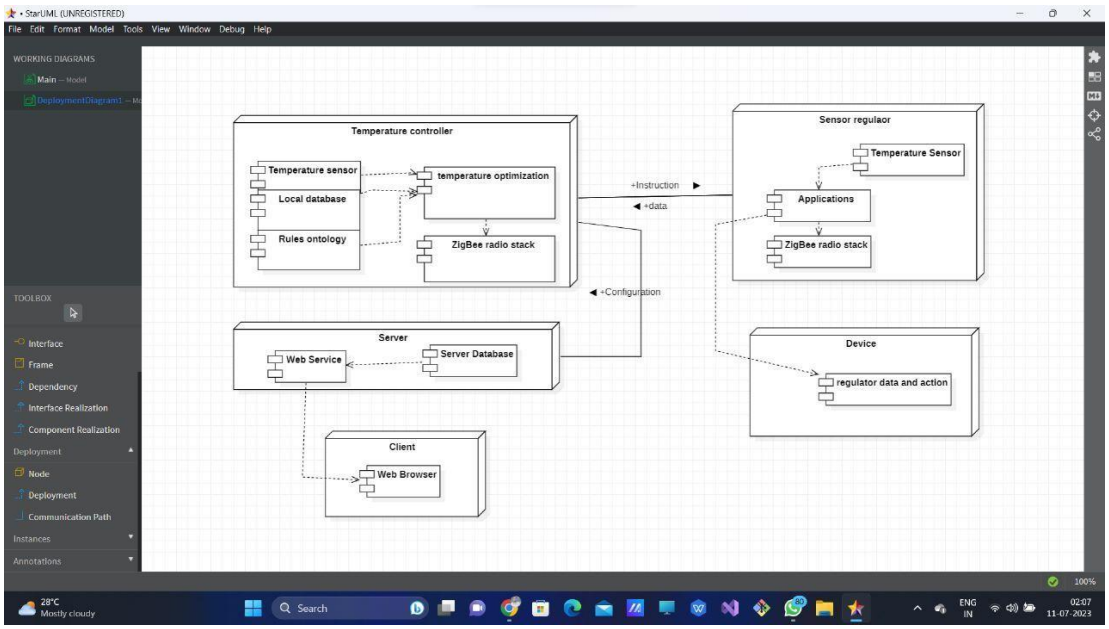
1) Package Diagram



2) Component Diagram



3) Deployment Diagram



Result

Thus, the Package, Component and Deployment diagram is prepared for the application - SMART HOME AUTOMATION SYSTEM

NAME - AMISHA TRIPATHI

REGISTRATION NUMBER- 21BCE1735

Experiment- 09

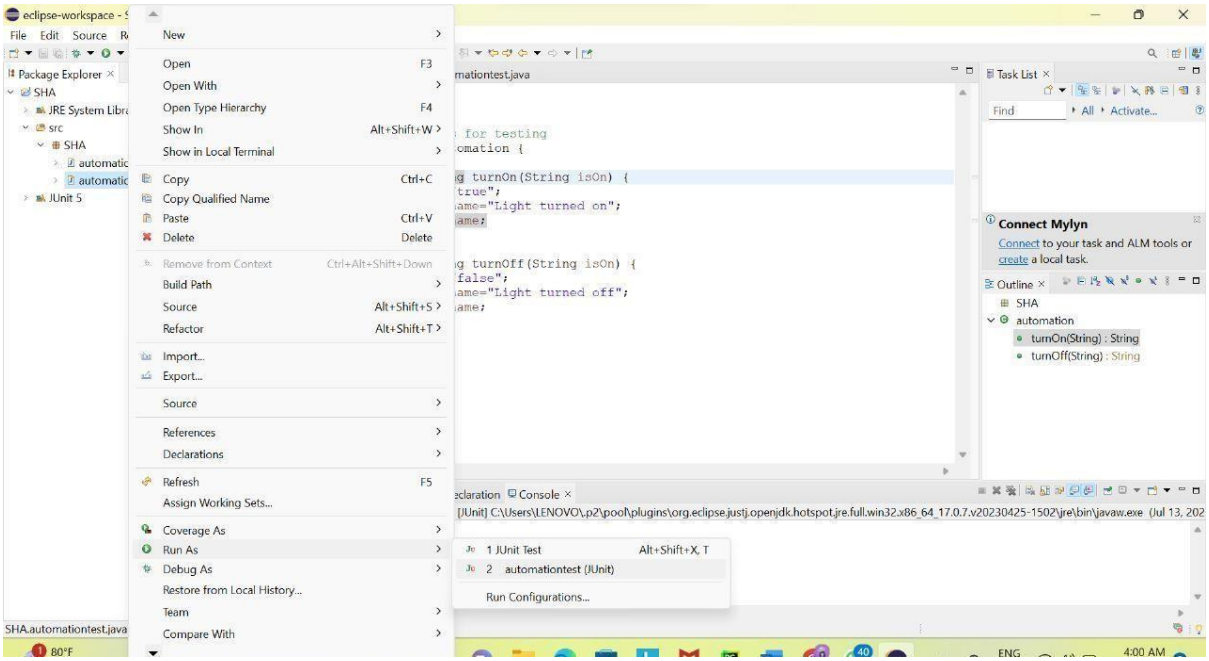
Perform Unit Testing Using JUnit

Aim: To develop JUnit test cases for a given Java Class and generate unit test report.

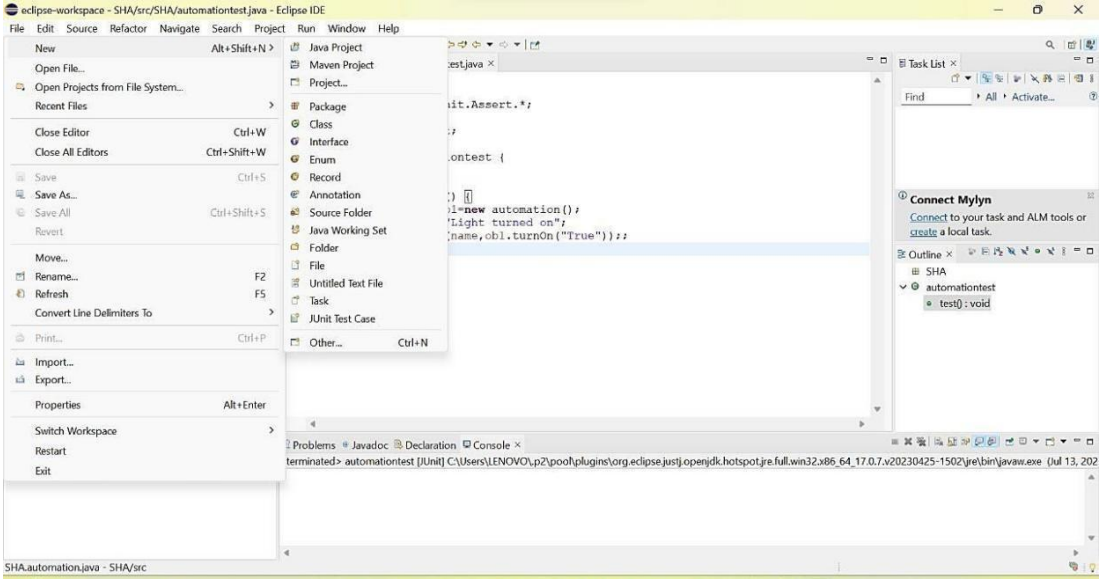
Steps Involved:

Steps in creating a JUnit Test case and Executing it in NetBeans and Eclipse Enviornment
A unit test is to test a smaller unit of code, e.g. methods. Writing a unit test to test the individual unit of code is one of the best development practices and helps to find bug earlier in the

development cycle. Though there is another unit testing framework available in Java e.g. TestNG, JUnit has its own place among Java developers
The main difference between JUnit 4 and JUnit 3 is that, JUnit4 is based on annotation feature of Java 1.5 and easy to write, while JUnit 3 uses "test" keyword, to identify test methods.



How to write JUnit tests in Eclipse
1. Create a New Java Project called JUnitExample.
2. Create a Java class Calculator in a project which should have to add() and multiply() method.
3. Right click on Java class and click on create Junit testcase



How to execute JUnit tests in Eclipse

Now you have your Junit test created, you can execute it just like you run any Java application:
Right Click --> Run As --> Junit Test
This will run all the *JUnit tests* declared in this class and will pass if all the tests run successfully and pass the condition tested by various assert statements and fail if any of the JUnit tests failed. The eclipse will print stack trace and hyperlink to the failed test and you can go and fix the problem.

An error is when some other Exception occurs due to programming mistakes, such as a NullPointerException or an ArrayIndexOutOfBoundsException. Both errors and failures are not good for your code. A good Junit test case should be able to bring failure and error alike. You should also ensure that your JUnit test case should pass always and doesn't throw any errors or failures.

How to create a JUnit Test suit in Eclipse? Example

Like individual unit tests, you can also create a test suite for creating tests for more than one classes in Java. In order to create a JUnit test suite in Eclipse

Go to File → New → Other... → Java → JUnit → TestSuite, and click Next>. Select all the classes, and click Finish. Once created, You can run this test suite the same way you run other JUnit tests. The result of the JUnit test suite will also show in JUnit console-like the previous run.

How to write JUnit tests in Netbeans

JUnit support in Netbeans is also great and seamless. Here is the steps to create JUnit test in Netbeans
1. Create a New Java Project called JUnitExample.
2. Create a Java Class Calculator in project which should have add() and multiply() method.
3. Now Select a Java Class --> Right click --> Tools --> Create Junit tests
this will create Junit test class for all the methods of selected Java class.

How to execute Junit tests in Netbeans

Executing JUnit tests in Netbeans is much simpler than it was in Eclipse. Go to your Junit test class and right-click and select run File option. This will execute all the JUnit tests on File and show the result in console. As earlier test will be pass if all test method passes otherwise it will fail. Netbeans also shows complete stack trace and hyperlink of failed test cases.

Code

IMPLEMENTATION OF OUR PROJECT (CODING PART)

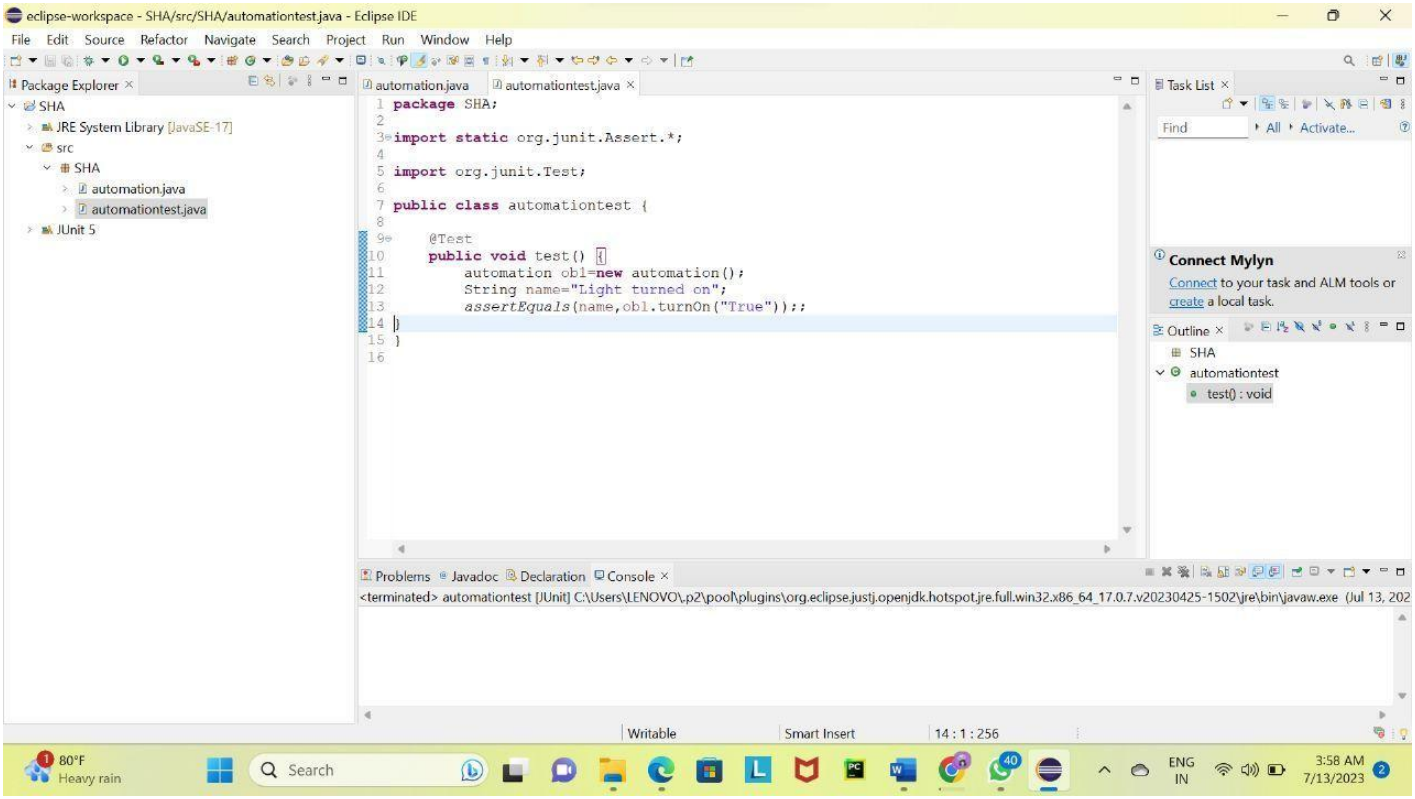
(VERIFIED RESULTS THROUGH IMPLEMENTATION OF JAVA CODE IN ECLIPSE)

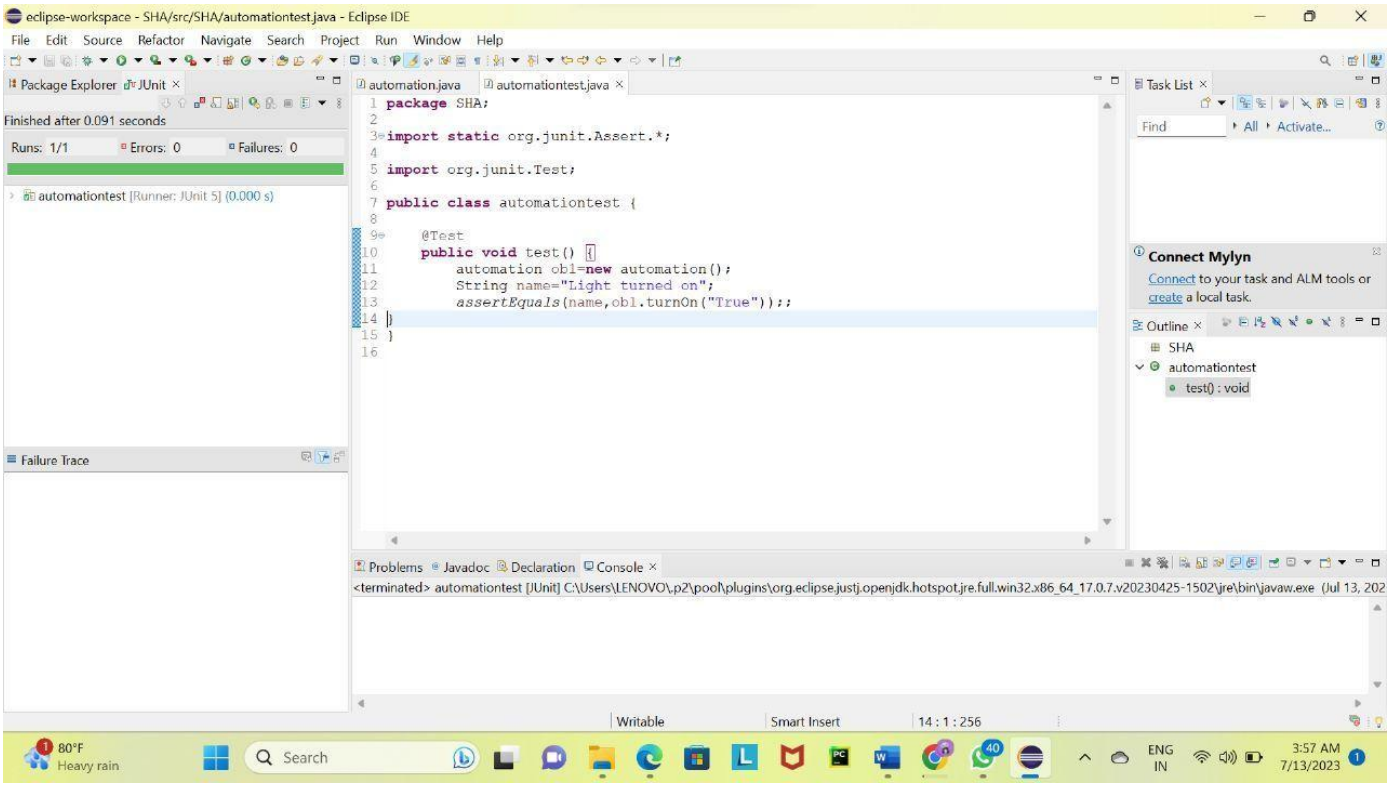
DOMAIN – SMART LIGHT DEVICE

TEST CASES

Unit Testing

```
package SHA;
import static org.junit.Assert.*;
import org.junit.Test;

public class automationtest
        {@Test
        public void test() {
                automation ob1=new automation();
                String name="Light turned on";
                assertEquals(name,ob1.turnOn("True"));
}
}
```




eclipse-workspace - SHA/src/SHA/automationtest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer JUnit
Finished after 0.091 seconds
Runs: 1/1
Errors: 0
Failures: 0
automationtest [Runner: JUnit 5] (0.000 s)
Failure Trace
automation.java
automationtest.java
1 package SHA;
2
3 import static org.junit.Assert.*;
4
5 import org.junit.Test;
6
7 public class automationtest {
8
9 @Test
10 public void test() {
11 automation obl=new automation();
12 String name="Light turned on";
13 assertEquals(name,obl.turnOn("True"));;
14 }
15 }
16
Task List
Find
All
Activate...
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Outline
SHA
automationtest
test() : void
Problems Javadoc Declaration Console
<terminated> automationtest [JUnit] C:\Users\LENOVO\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (Jul 13, 202
Writable
Smart Insert
14 : 1 : 256
80°F
Heavy rain
Search
ENG
IN
3:57 AM
7/13/2023

Perform Integration Testing using Junit

Aim

To perform Integration testing using JUnit

Procedure

Step 1: Open NetBeans IDE 8.2
Step 2: File-> NewProject -> Java Application
Step 3: From the created project, right click and New-> Java Class
Step 4: Type the Java Code to be tested:

```
package SHA;

//Main class for testing
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

// Lighting domain
class Light {
```

```
    private String  name;
    private boolean isOn;
    public Light(String name) {

        this.name = name;
    }

    public void turnOn()
        {isOn = true;
        System.out.println("Light " + name + " turned on");
    }

    public void turnOff()
        {isOn = false;
        System.out.println("Light " + name + " turned off");
    }

    public boolean isOn()
        {return isOn;
    }
}

// Integration test class
class SmartHomeIntegrationTest
    {@Test
    void testSmartHomeAutomationSystem() {
        // Create smart devices
        Light livingRoomLight = new Light("Living Room Light");
        Light bedroomLight = new Light("Bedroom Light");

        // Perform integration testing
        livingRoomLight.turnOn();
        assertTrue(livingRoomLight.isOn(), "Living room light should be turned on");

        bedroomLight.turnOn();
        assertTrue(bedroomLight.isOn(), "Bedroom light should be turned on");

        livingRoomLight.turnOff();
        assertFalse(livingRoomLight.isOn(), "Living room light should be turned off");

        bedroomLight.turnOff();
        assertFalse(bedroomLight.isOn(), "Bedroom light should be turned off");
    }
}

// Main class for running the smart home automation system
public class automation {
    public static void main(String[] args) {
        // Create smart devices
        Light livingRoomLight = new Light("Living Room Light");
        Light bedroomLight = new Light("Bedroom Light");

        // Turn on lights
        livingRoomLight.turnOn();
        bedroomLight.turnOn();
```

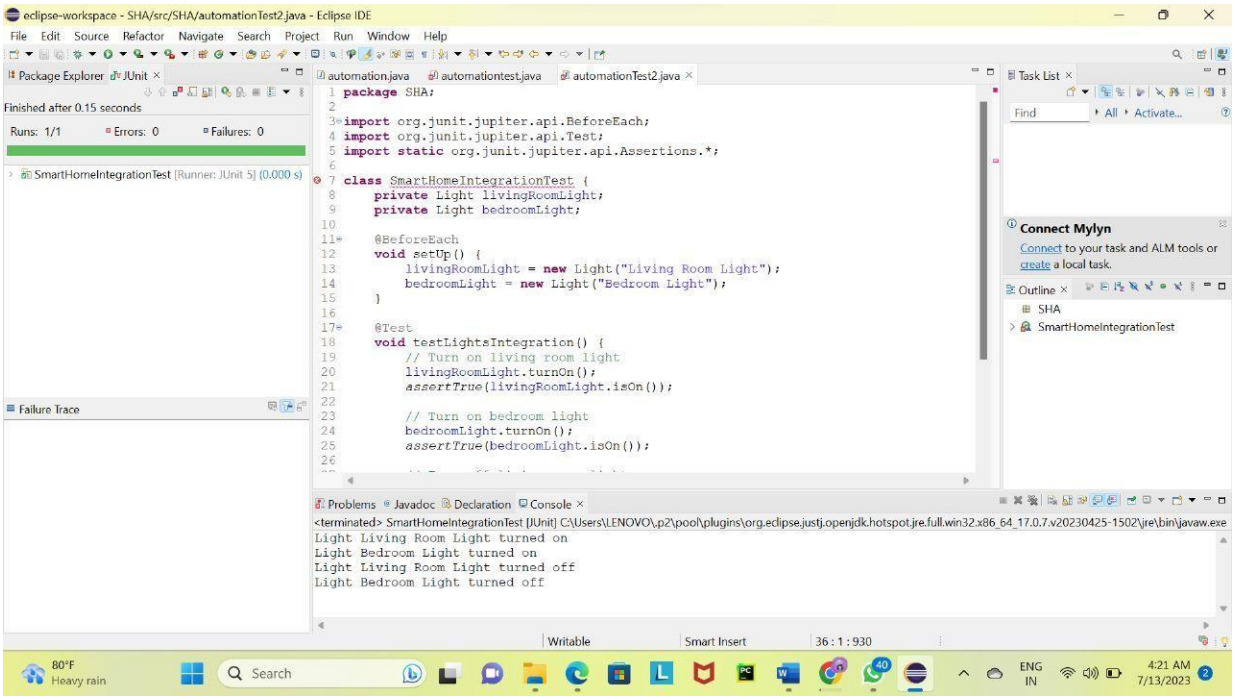
```
        // Turn off lights
        livingRoomLight.turnOff();

        bedroomLight.turnOff();

        // Run the integration test
        SmartHomeIntegrationTest integrationTest = new SmartHomeIntegrationTest();
        integrationTest.testSmartHomeAutomationSystem();
    }
}
```

Step 5: Create Test with a Test Name with Integration Tests Checked
Step 6: Create JUnit test cases for the above two units to perform Integration testing

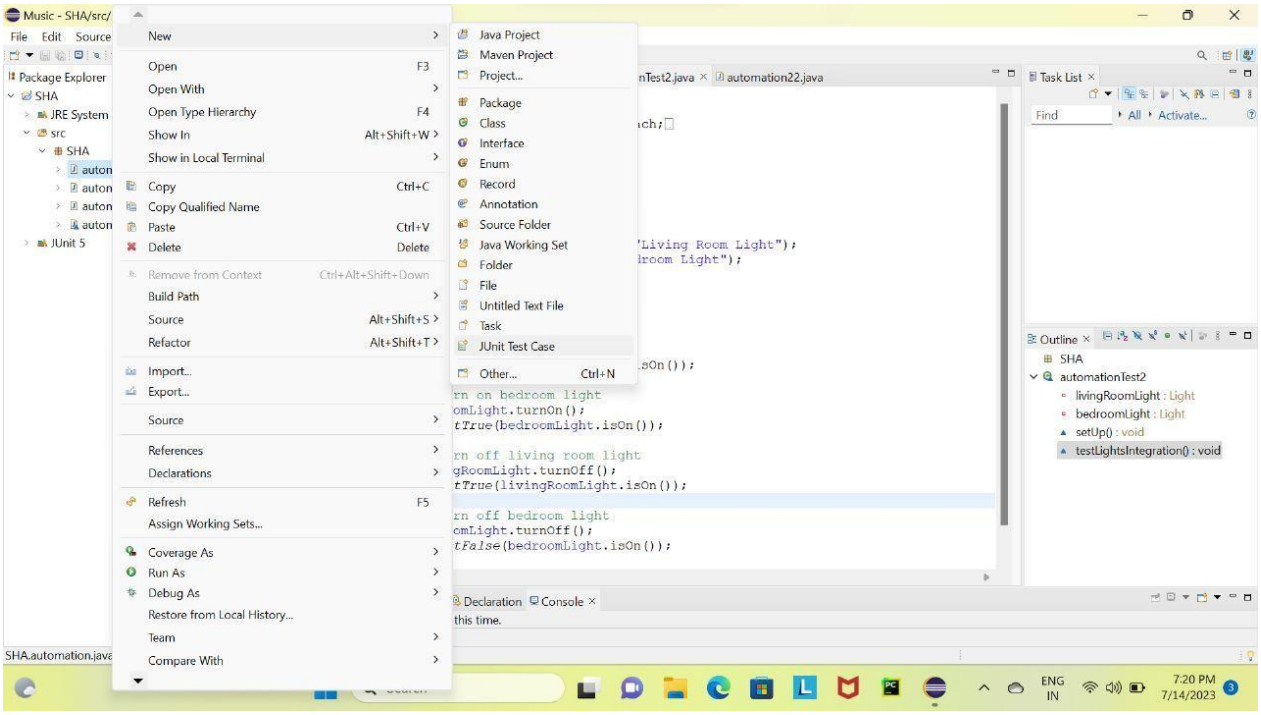
```
package SHA;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

class automationTest2 {
    private Light livingRoomLight;
    private Light bedroomLight;

    @BeforeEach
    void setUp() {
        livingRoomLight = new Light("Living Room Light");
        bedroomLight = new Light("Bedroom Light");
    }

    @Test
    void testLightsIntegration() {
        // Turn on living room light
        livingRoomLight.turnOn();
        assertTrue(livingRoomLight.isOn());
```

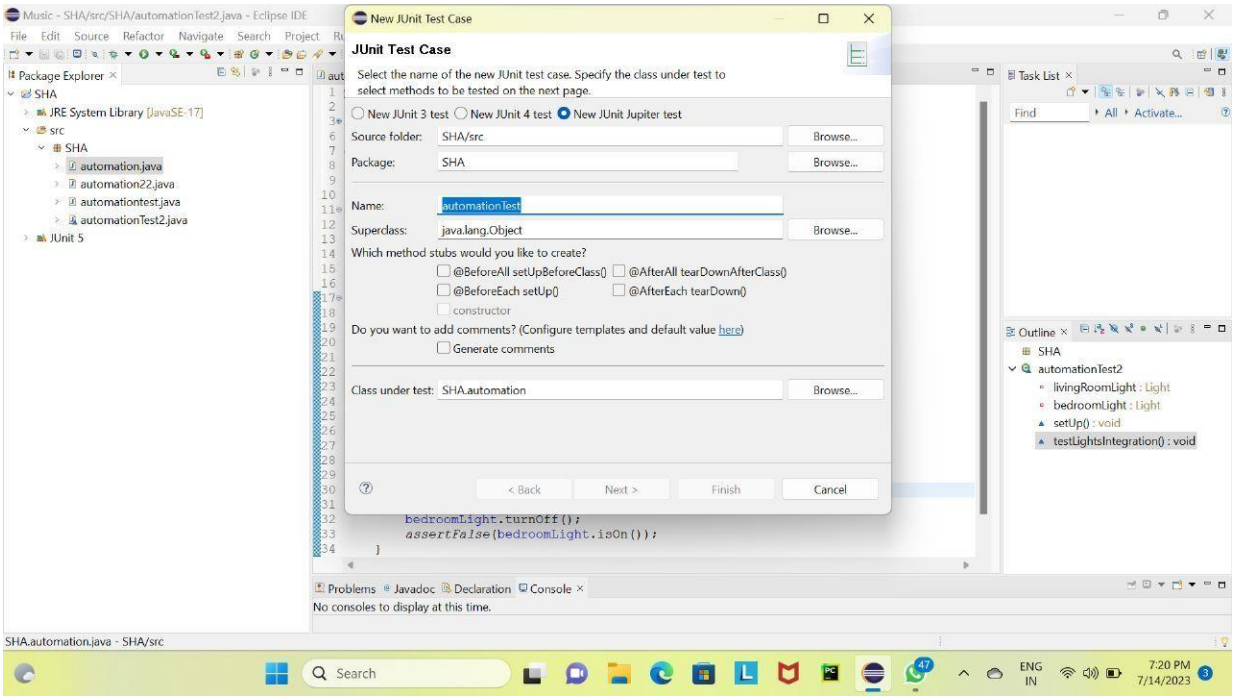
```
        // Turn on
        bedroom light
        bedroomLight.
        turnOn();
        assertTrue(be
        droomLight.is
        On());

        // Turn off living room light
        livingRoomLight.turnOff();
        assertFalse(livingRoomLigh
        t.isOn());

        // Turn off bedroom light
        bedroomLight.turnOff();
        assertFalse(bedroomLight.is
        On());
    }
}
```

Step 7: Right click Test code and Click RunTest, the tests will be executed.
Step 8: Take both pass and fail test results
Step 9: Close the project
Screenshots

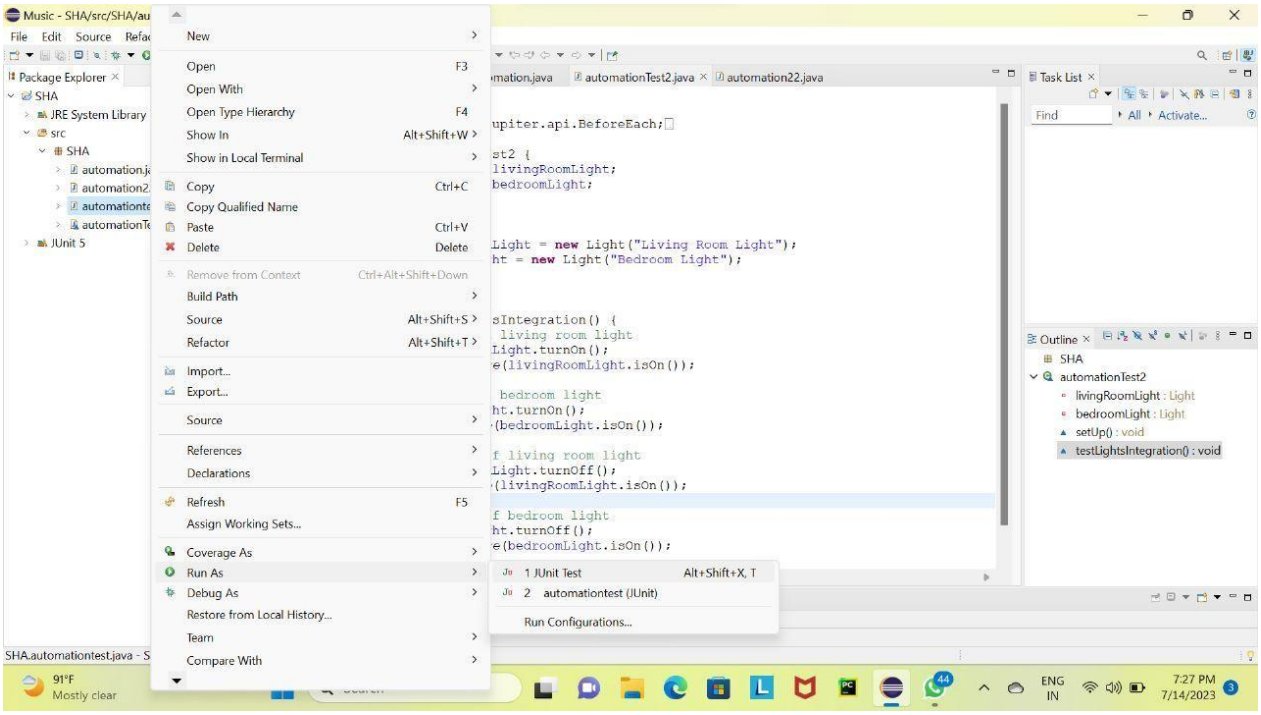


Create Tests

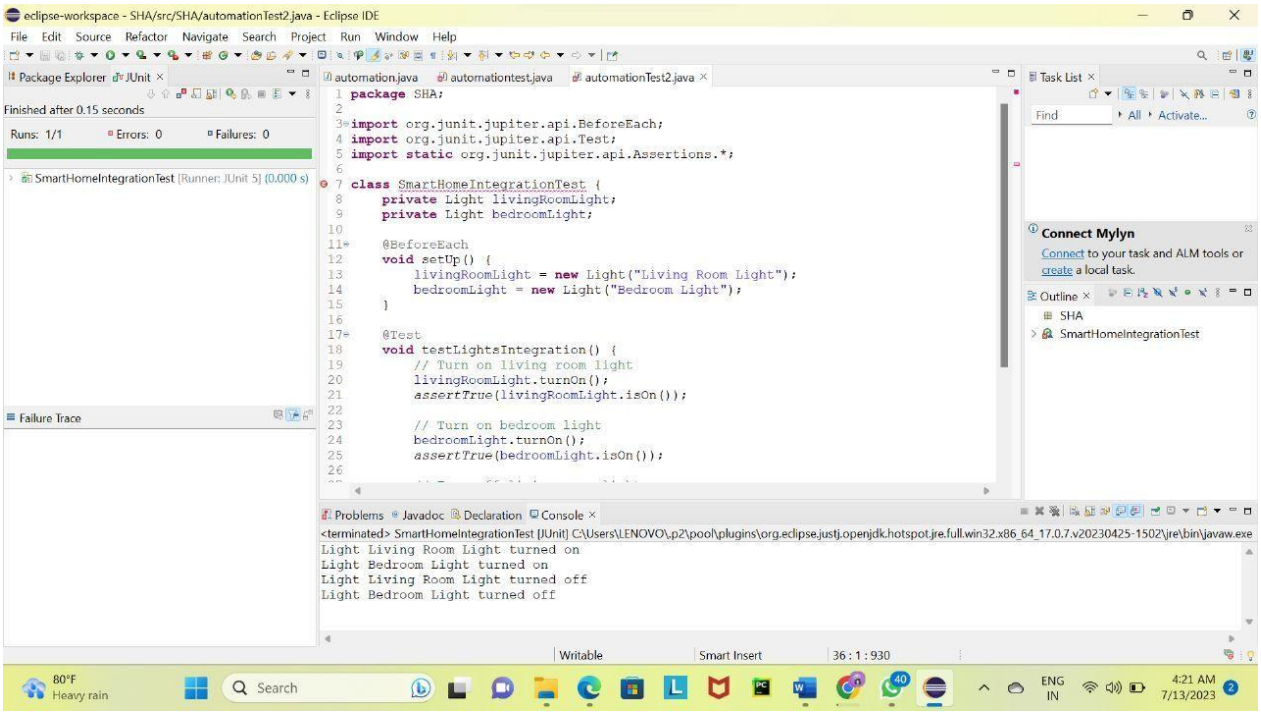




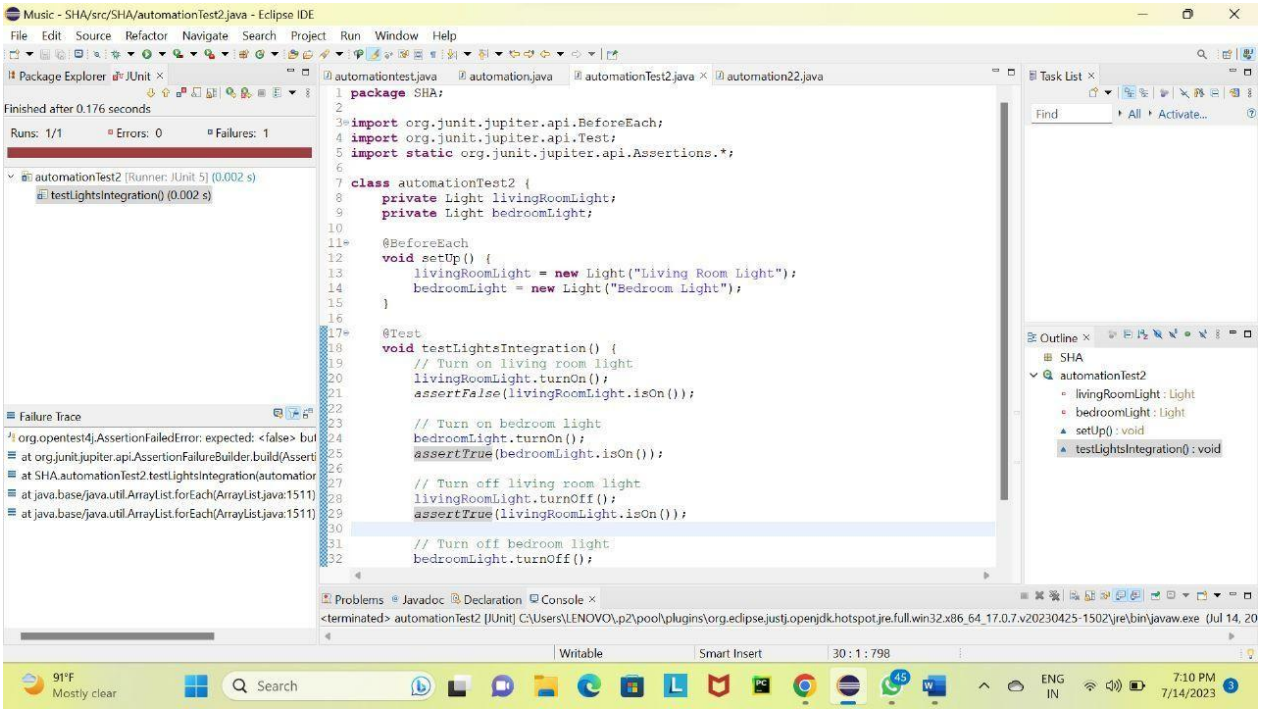
Execute Tests



Positive Result



Negative Result



Result

Thus integration testing is conducted for the given application.

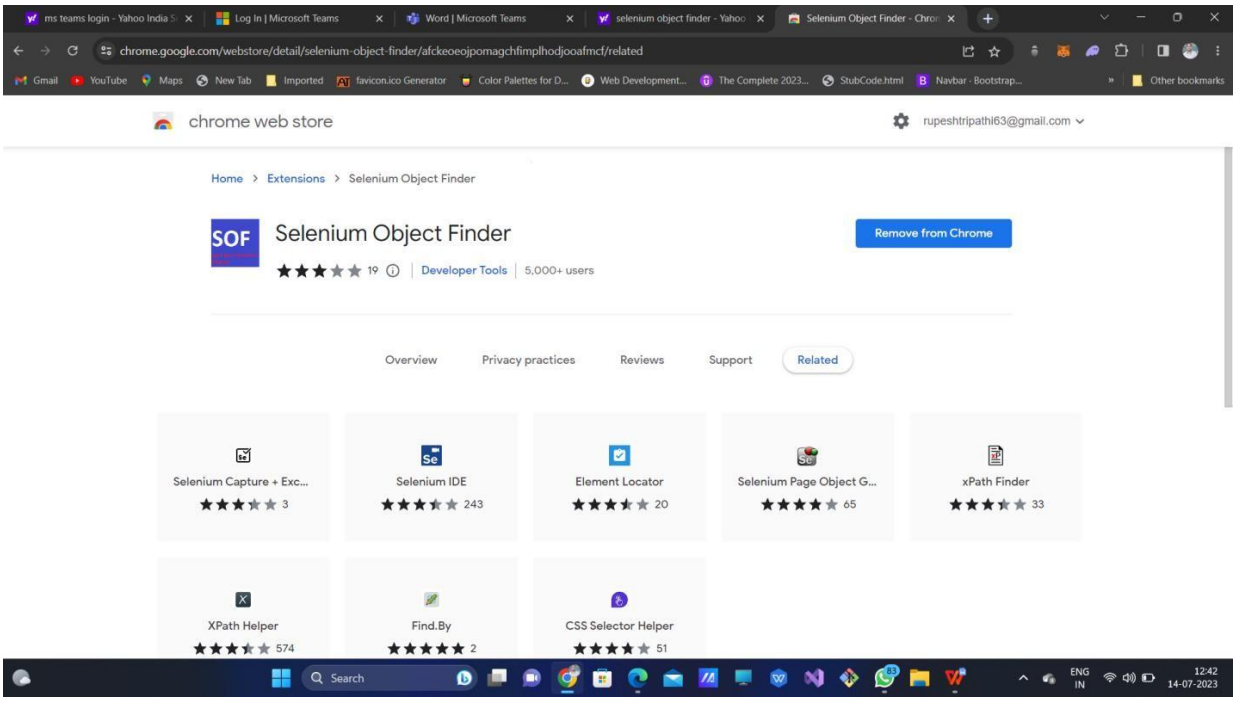
Implement Regression testing using Selenium-Object Finder

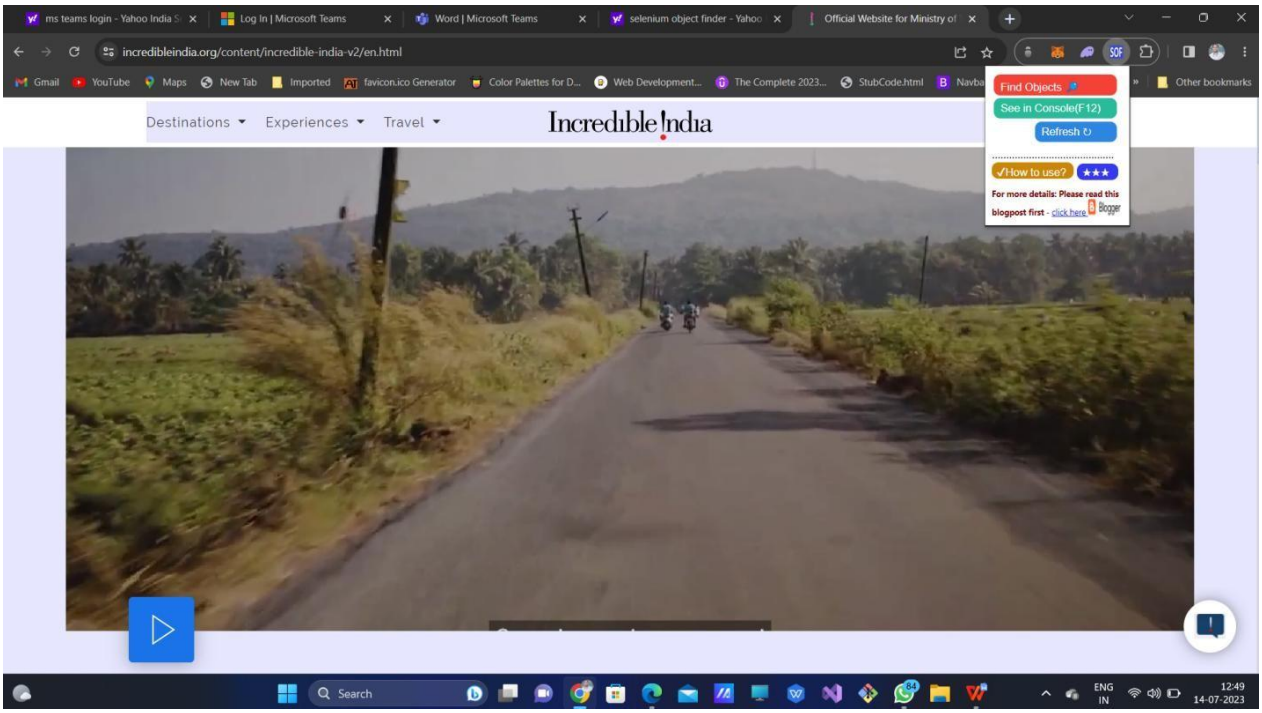
Aim:

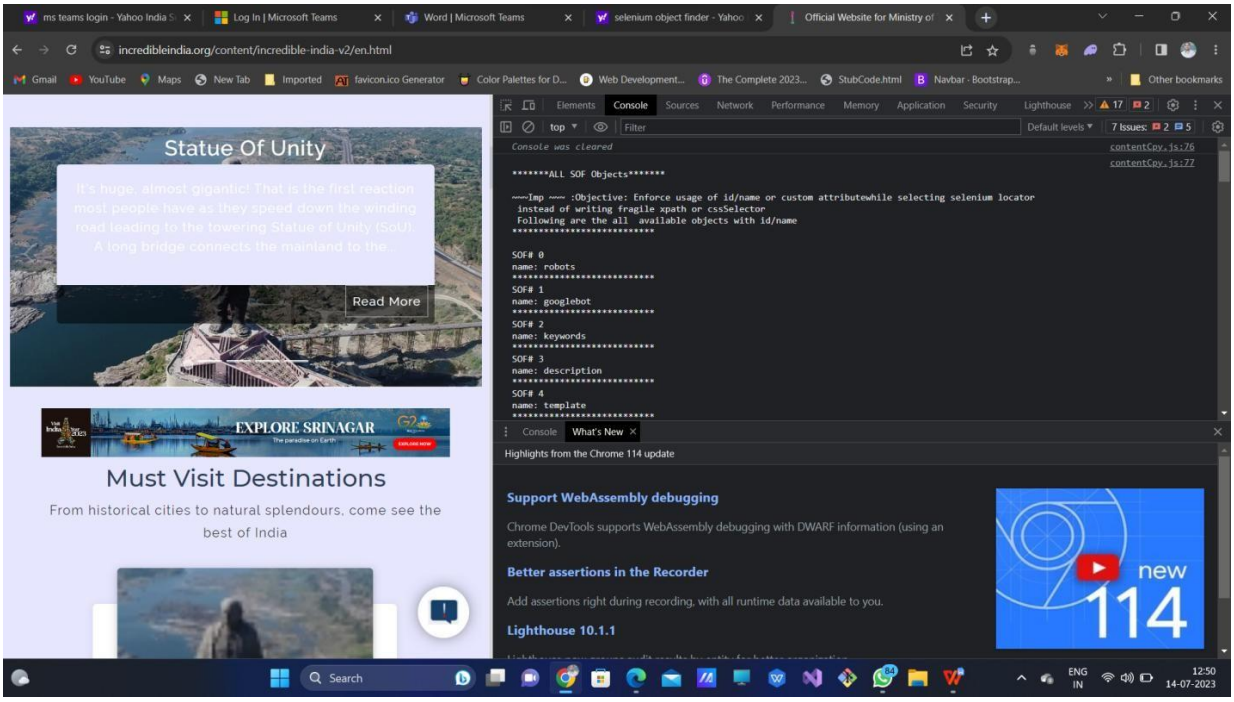
To implement automation testing using Selenium Object Finder

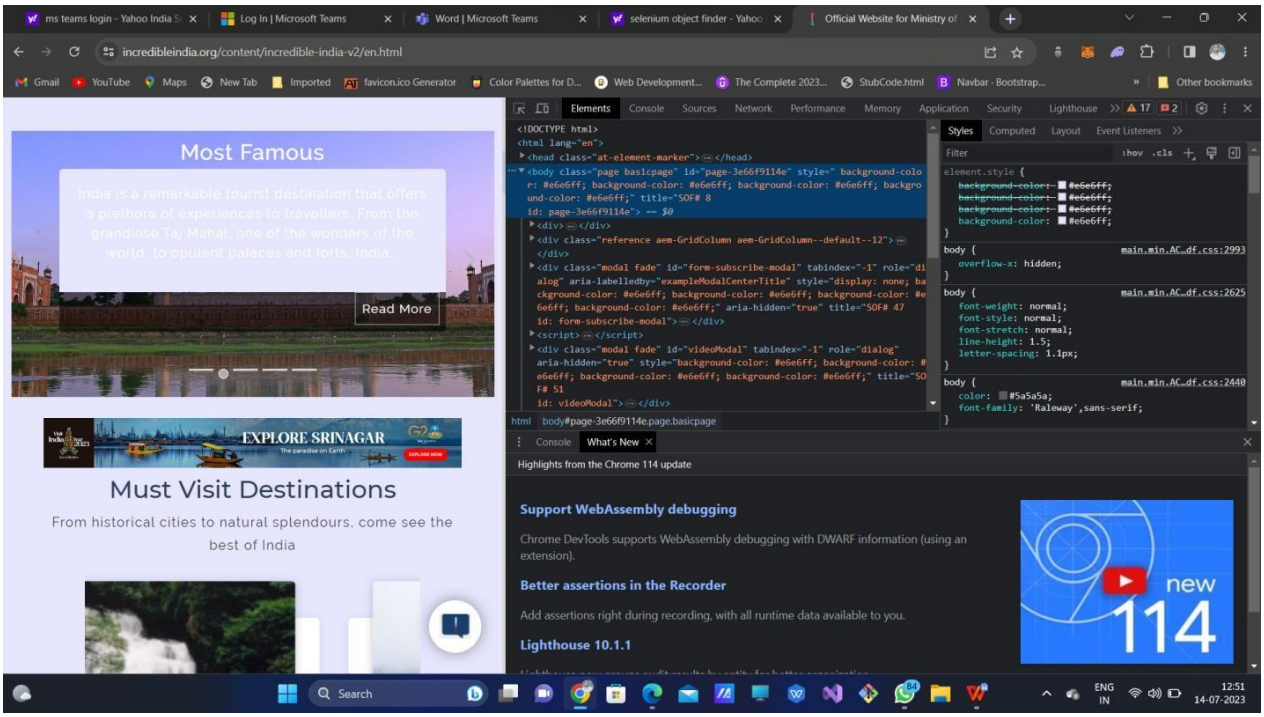
Procedure

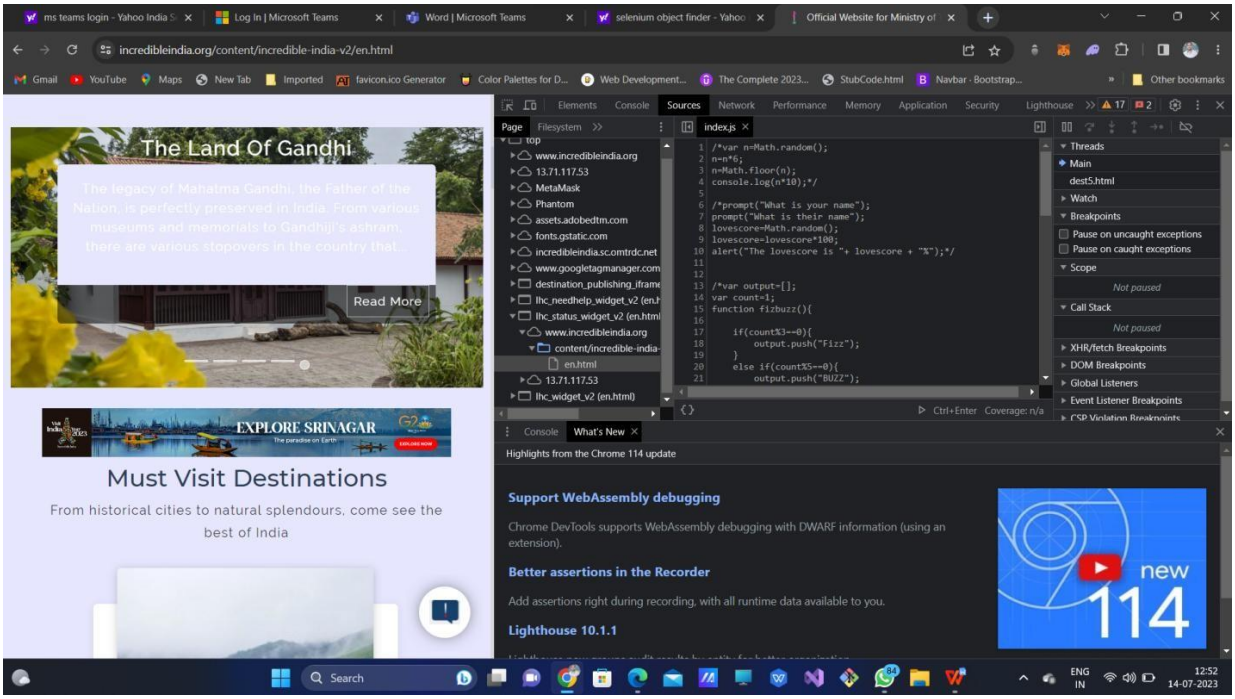
Step 1 – Open Google Chrome -> Search for Chrome Extension for Selenium-Object Finder
Step 2- Add it to the chrome
Step 3 – Double click it and do the testing on the various objects.

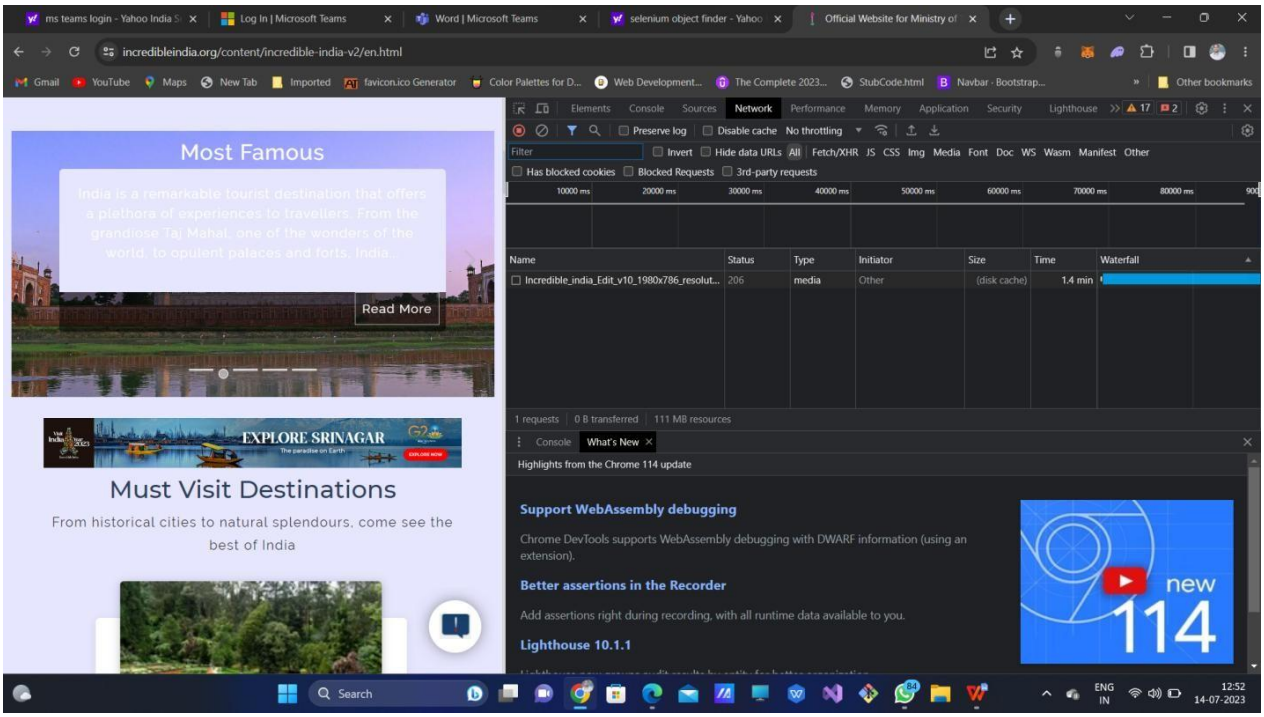


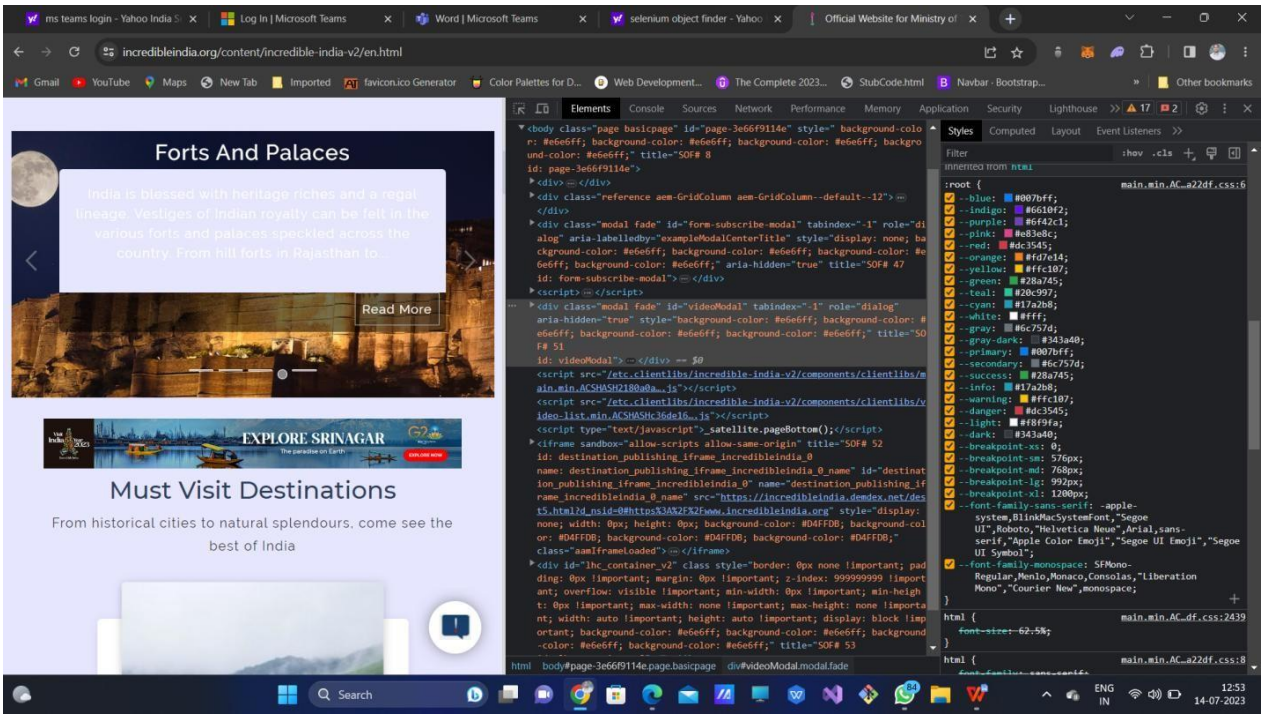

incredibleindia.org/content/incredible-india-v2/en.html
Destinations
Experiences
Travel
Incredible India
Find Objects
See in Console(F12)
Refresh
How to use?
For more details: Please read this blogpost first - click here
Search
ENG
IN
12:49
14-07-2023


incredibleindia.org/content/incredible-india-v2/en.html
Statue Of Unity
It's huge, almost gigantic! That is the first reaction most people have as they speed down the winding road leading to the towering Statue of Unity (SoU). A long bridge connects the mainland to the
Read More
EXPLORE SRINAGAR
Must Visit Destinations
From historical cities to natural splendours, come see the best of India
Elements
Console
Sources
Network
Performance
Memory
Application
Security
Lighthouse
Console was cleared
*******ALL SOF Objects*******
~~~Imp ~~~ :Objective: Enforce usage of id/name or custom attributewhile selecting selenium locator
instead of writing fragile xpath or cssSelector
Following are the all available objects with id/name
SOF# 0
name: robots
SOF# 1
name: googlebot
SOF# 2
name: keywords
SOF# 3
name: description
SOF# 4
name: template
What's New
Highlights from the Chrome 114 update
Support WebAssembly debugging
Chrome DevTools supports WebAssembly debugging with DWARF information (using an extension).
Better assertions in the Recorder
Add assertions right during recording, with all runtime data available to you.
Lighthouse 10.1.1
new
114
Search
ENG
IN
12:50
14-07-2023
~~~


Most Famous
India is a remarkable tourist destination that offers a plethora of experiences to travellers. From the grandiose Taj Mahal, one of the wonders of the world, to opulent palaces and forts, India...
Read More
EXPLORE SRINAGAR
Must Visit Destinations
From historical cities to natural splendours, come see the best of India
Support WebAssembly debugging
Chrome DevTools supports WebAssembly debugging with DWARF information (using an extension).
Better assertions in the Recorder
Add assertions right during recording, with all runtime data available to you.
Lighthouse 10.1.1
new 114


The Land Of Gandhi
The legacy of Mahatma Gandhi, the Father of the Nation, is perfectly preserved in India. From various museums and memorials to Gandhiji's ashram, there are various stopovers in the country that...
Read More
EXPLORE SRINAGAR
Must Visit Destinations
From historical cities to natural splendours, come see the best of India
Support WebAssembly debugging
Chrome DevTools supports WebAssembly debugging with DWARF information (using an extension).
Better assertions in the Recorder
Add assertions right during recording, with all runtime data available to you.
Lighthouse 10.1.1
new 114





Result:

The automation testing using Selenium Object Finder is done and the processes are analyzed.

Web application testing with Capture and Playback using Selenium IDE

Aim

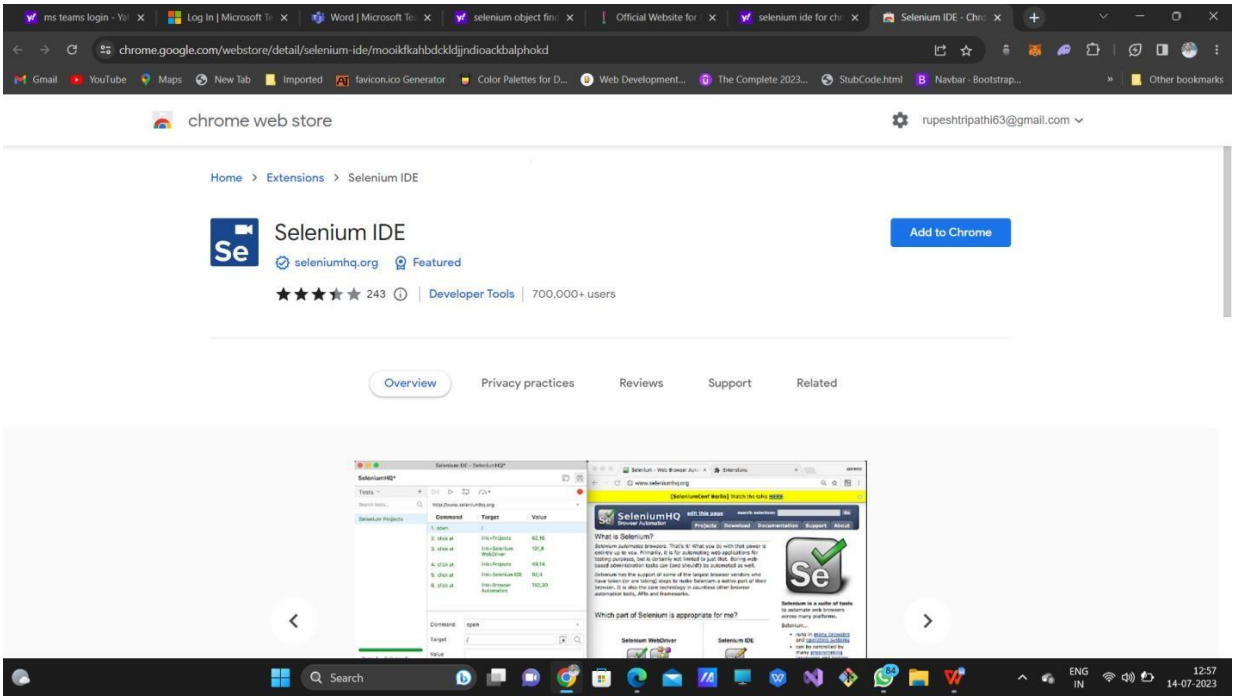
To perform web application testing with Capture and Playback using Selenium IDE.

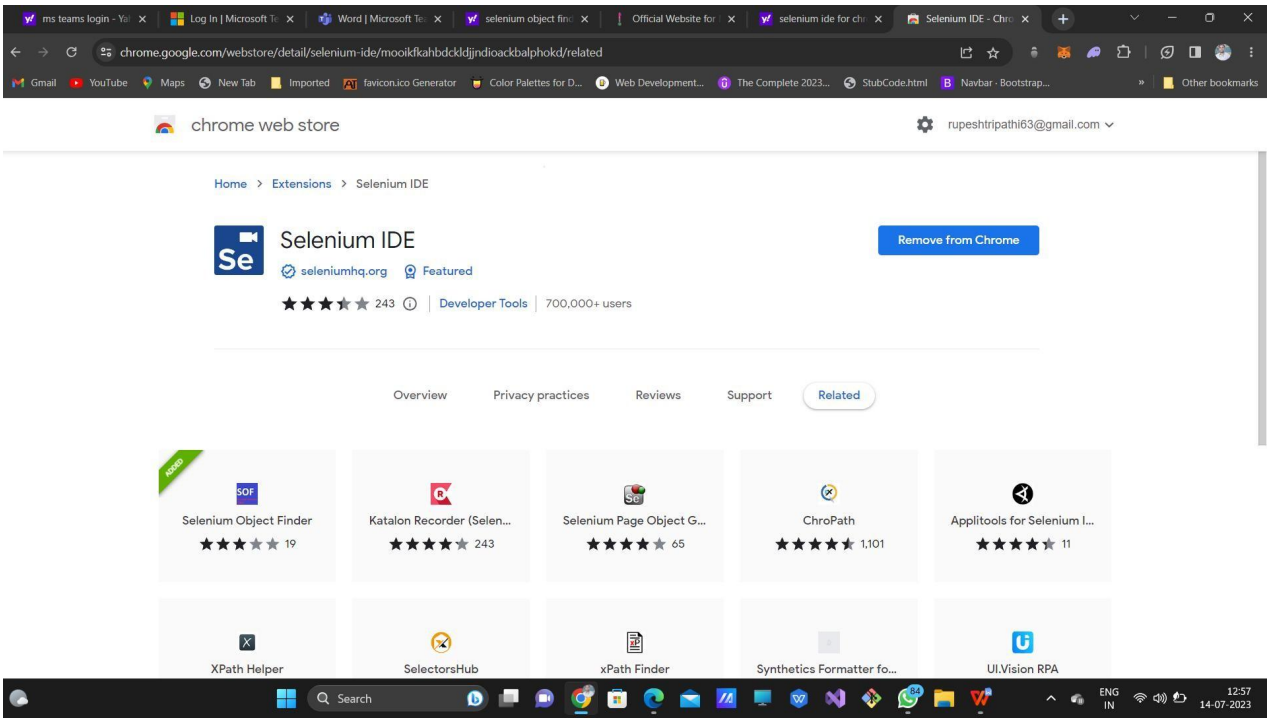
Procedure

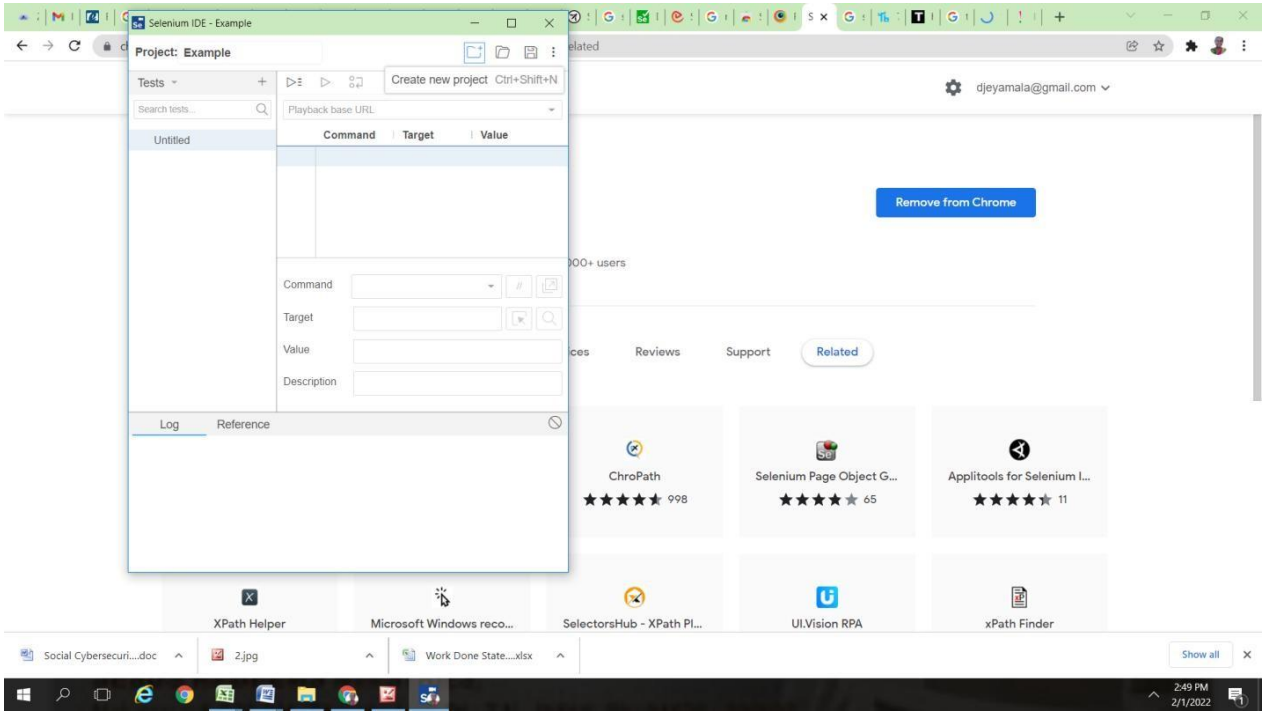
Step 1: Open Google Chrome, search for Chrome Extention on “Selenium IDE”

Step 2: Add it into Google Chrome.

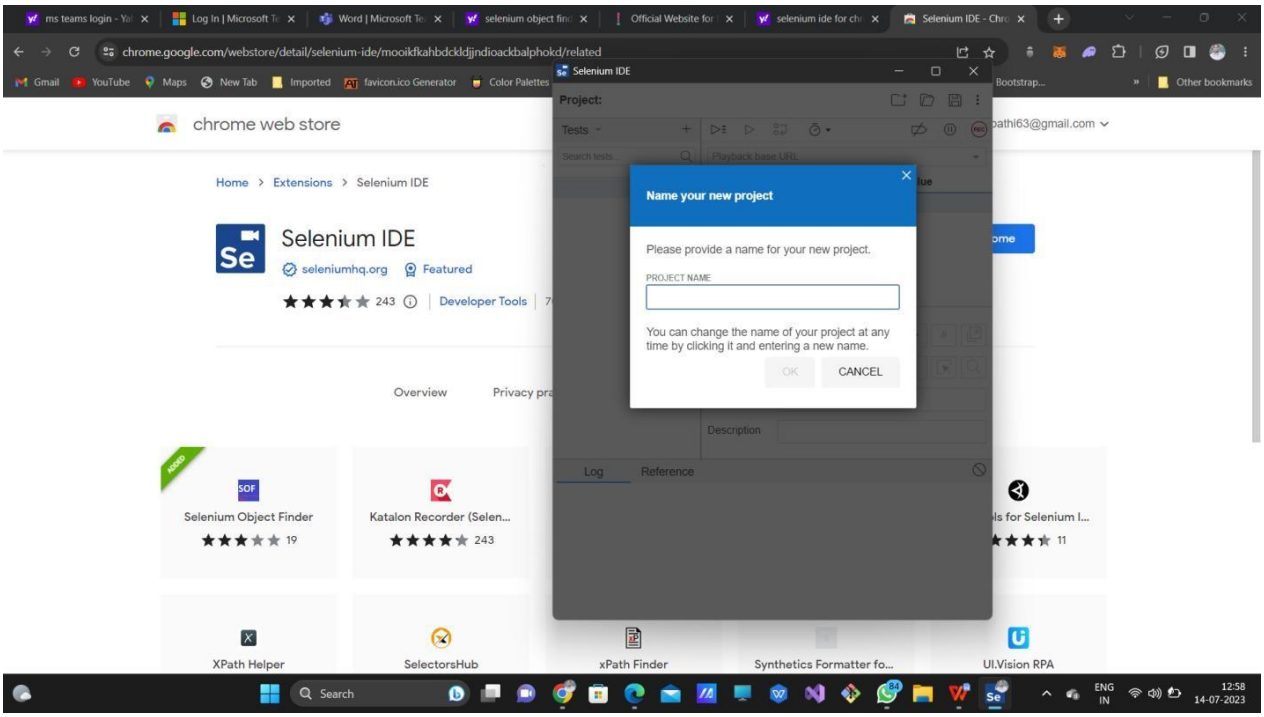
Step 3 – Then perform the steps as below:





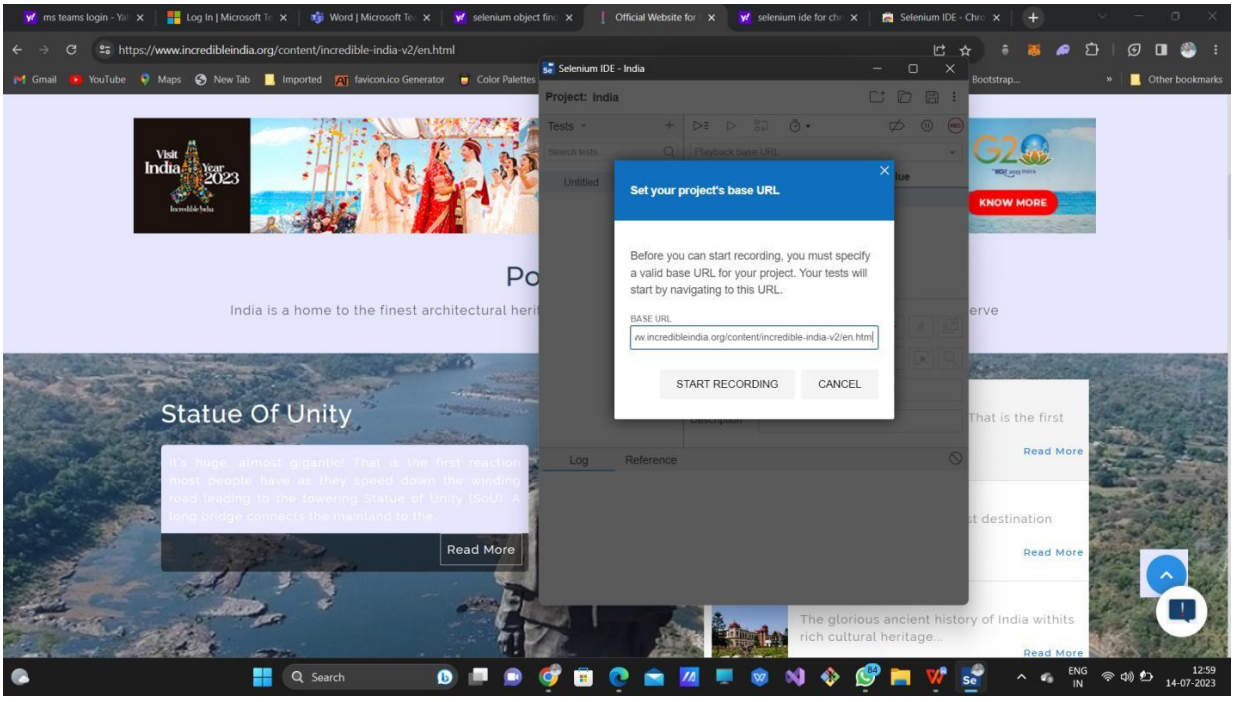


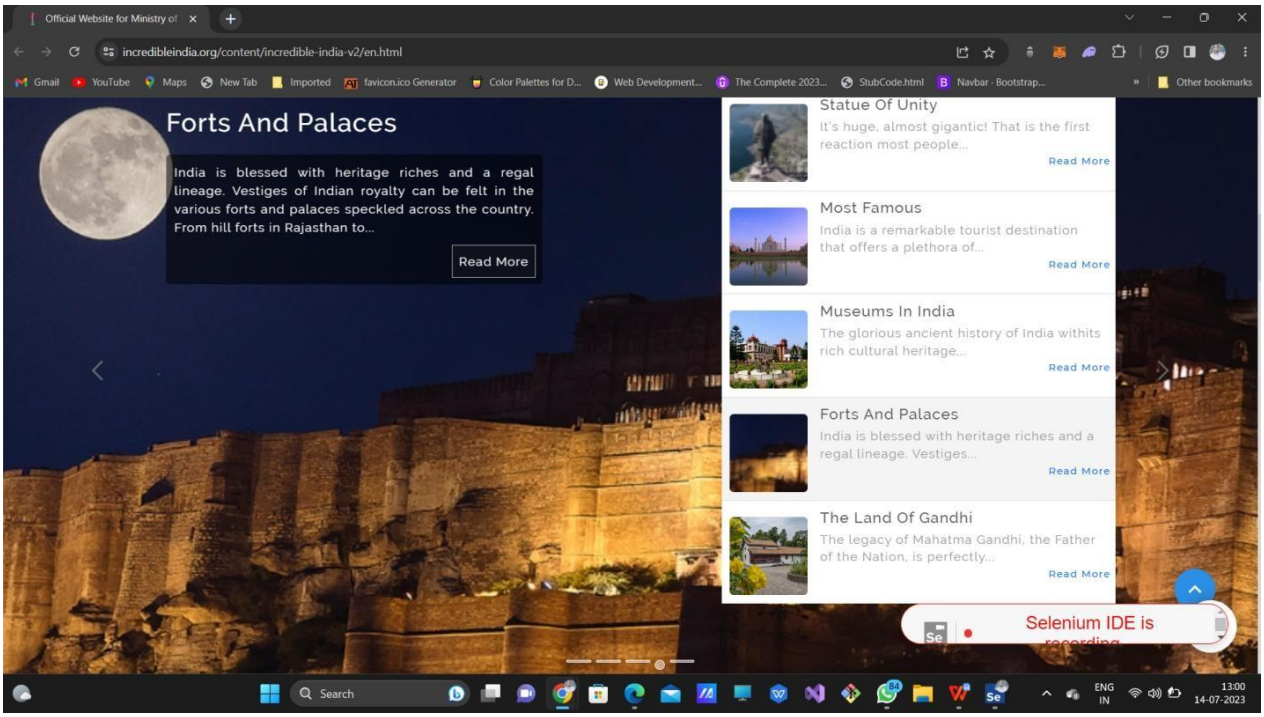
Create a new test case in a new project

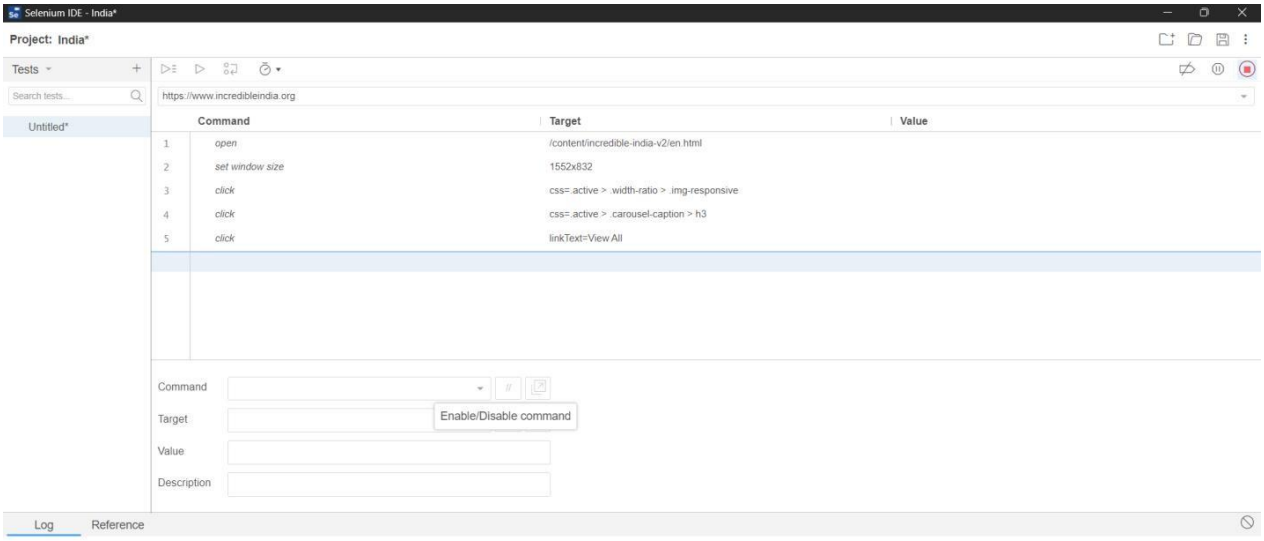


Give name of the project

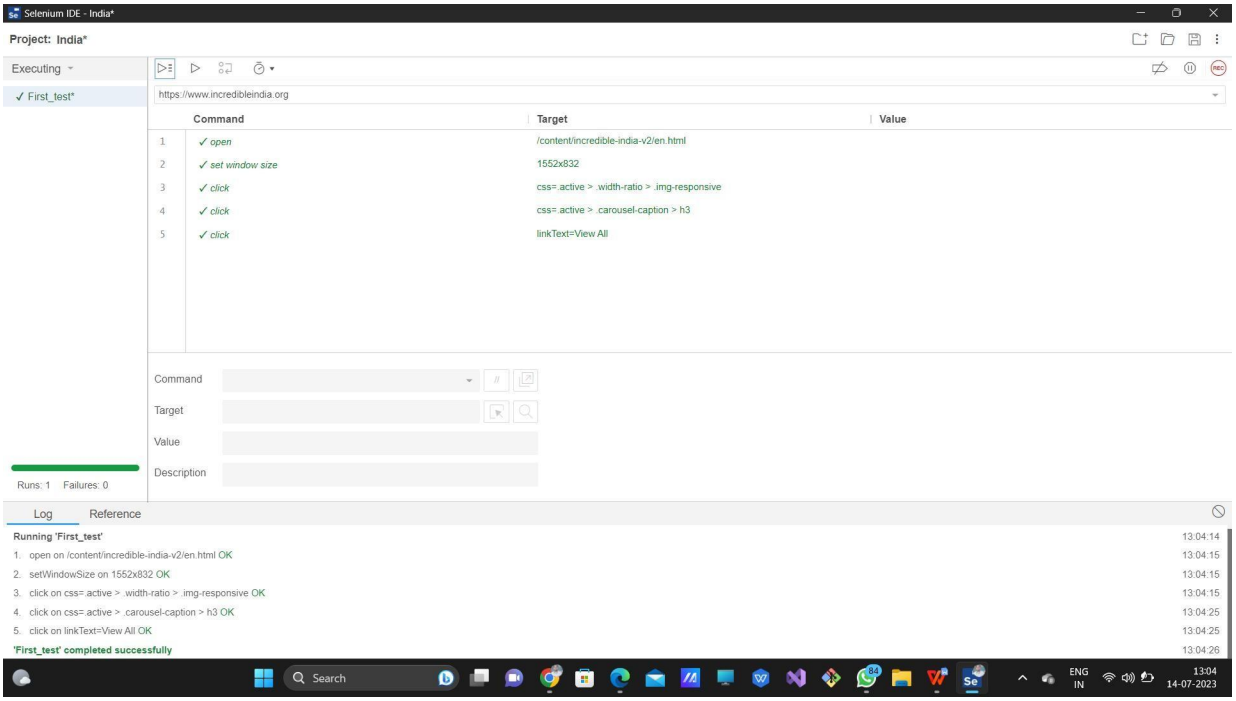
Enter the URL of the website to be tested



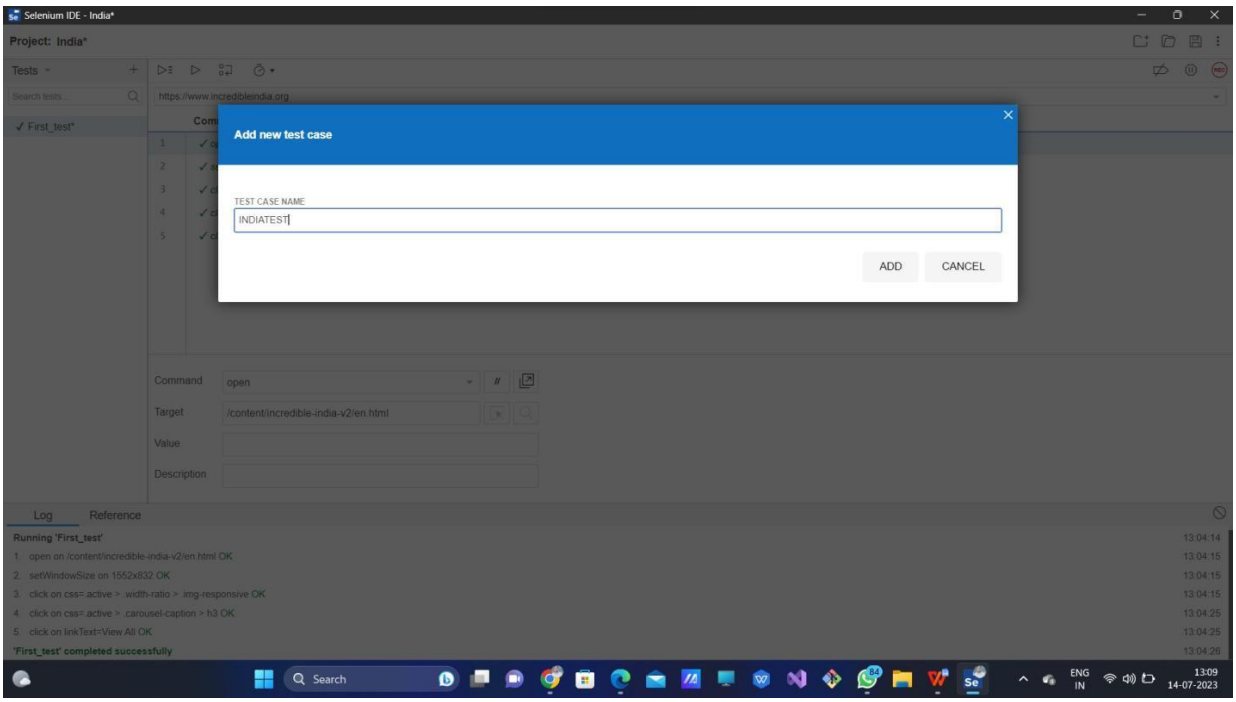


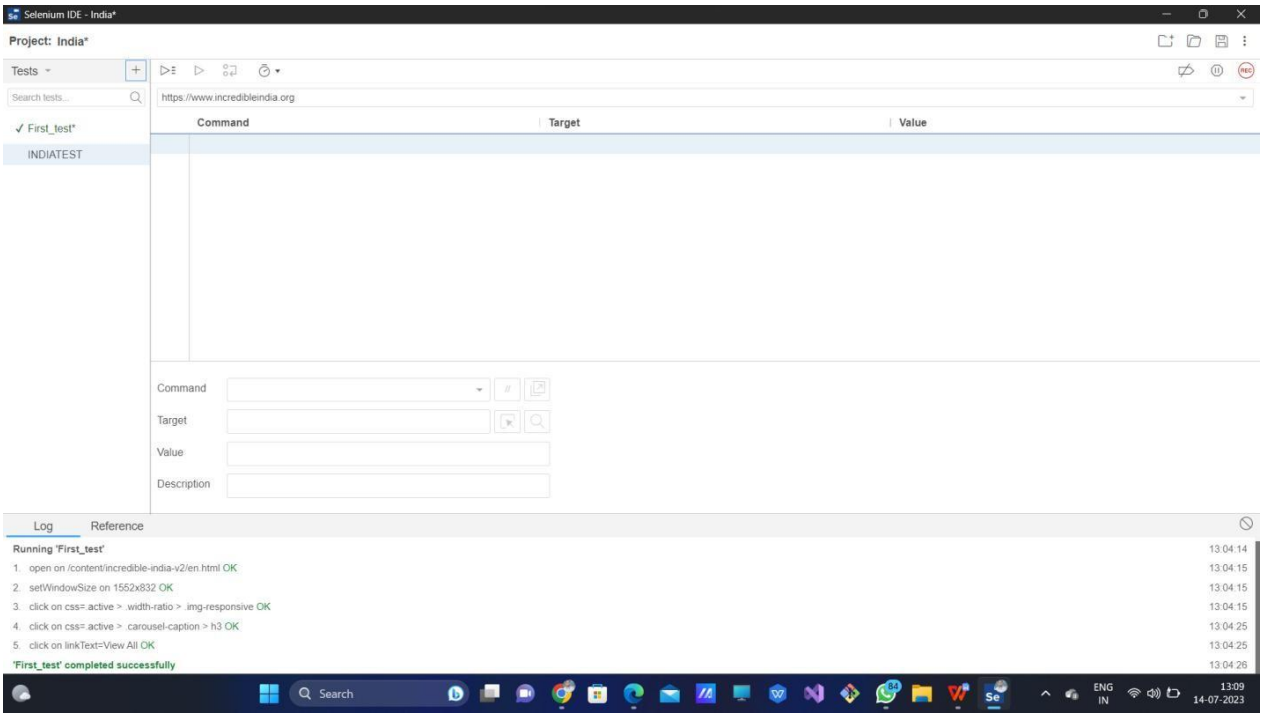


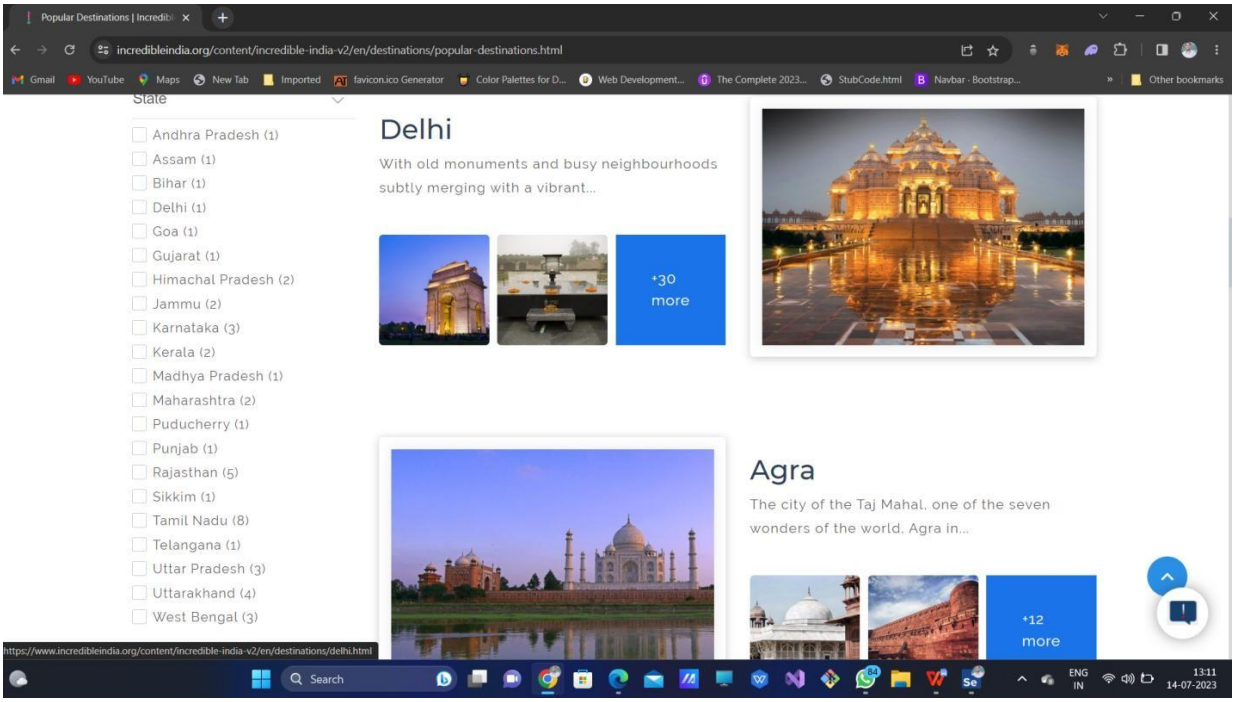
The test pass report

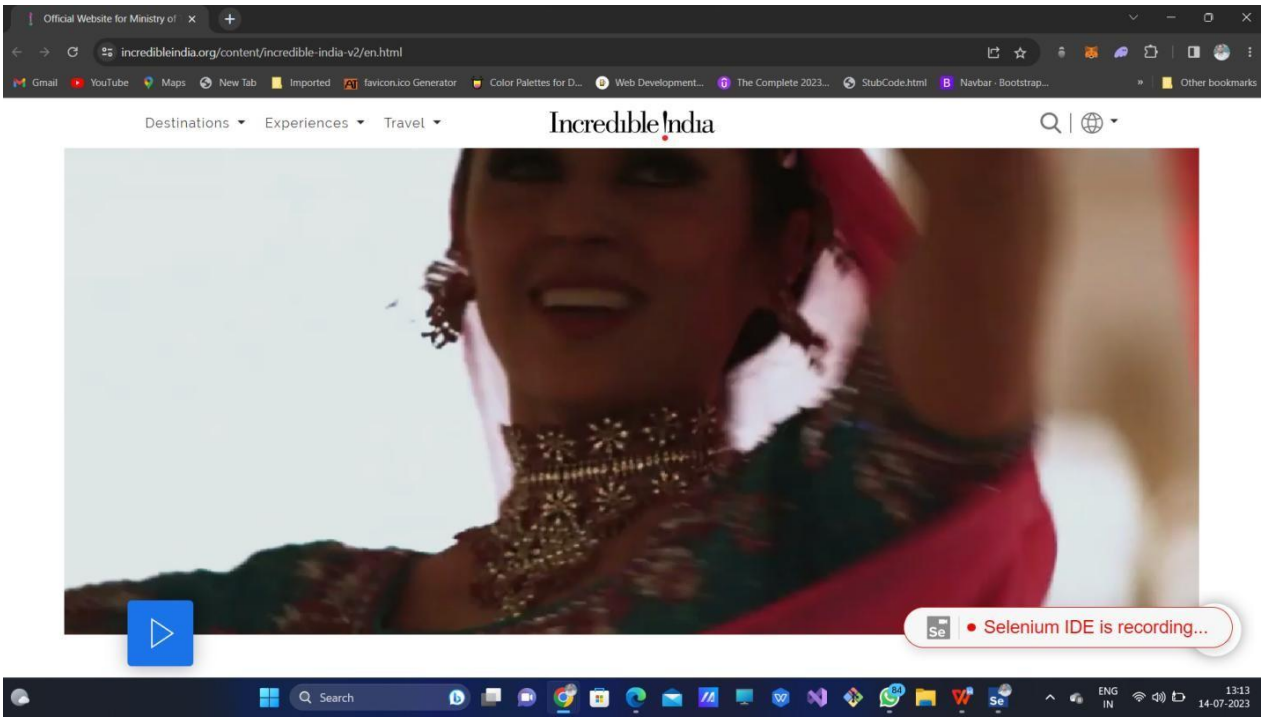


Save the test suite with a name

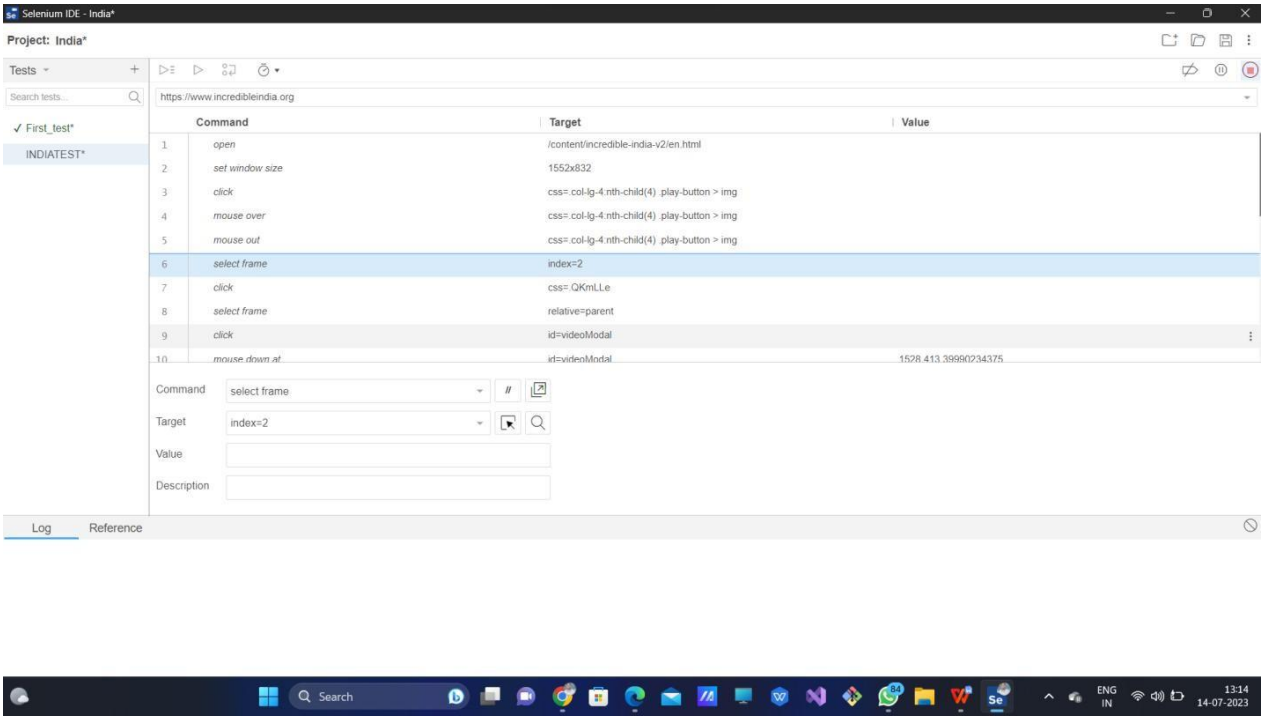



Selenium IDE - India*
Project: India*
Tests
Search tests...
First_test*
INDIATEST
https://www.incredibleindia.org
Command
Target
Value
Command
Target
Value
Description
Log
Reference
Running 'First_test'
1. open on /content/incredible-india-v2/en.html OK
2. setWindowSize on 1552x832 OK
3. click on css=.active > .width-ratio > .img-responsive OK
4. click on css=.active > .carousel-caption > h3 OK
5. click on linkText=View All OK
'First_test' completed successfully
13:04:14
13:04:15
13:04:15
13:04:15
13:04:25
13:04:25
13:04:26
Search
ENG
IN
13:09
14-07-2023


Popular Destinations | Incredib
incredibleindia.org/content/incredible-india-v2/en/destinations/popular-destinations.html
Gmail
YouTube
Maps
New Tab
Imported
favicon.ico Generator
Color Palettes for D...
Web Development...
The Complete 2023...
StubCode.html
Navbar · Bootstrap...
Other bookmarks
State
Andhra Pradesh (1)
Assam (1)
Bihar (1)
Delhi (1)
Goa (1)
Gujarat (1)
Himachal Pradesh (2)
Jammu (2)
Karnataka (3)
Kerala (2)
Madhya Pradesh (1)
Maharashtra (2)
Puducherry (1)
Punjab (1)
Rajasthan (5)
Sikkim (1)
Tamil Nadu (8)
Telangana (1)
Uttar Pradesh (3)
Uttarakhand (4)
West Bengal (3)
Delhi
With old monuments and busy neighbourhoods subtly merging with a vibrant...
+30 more
Agra
The city of the Taj Mahal, one of the seven wonders of the world. Agra in...
+12 more
https://www.incredibleindia.org/content/incredible-india-v2/en/destinations/delhi.html
Search
ENG
IN
13:11
14-07-2023

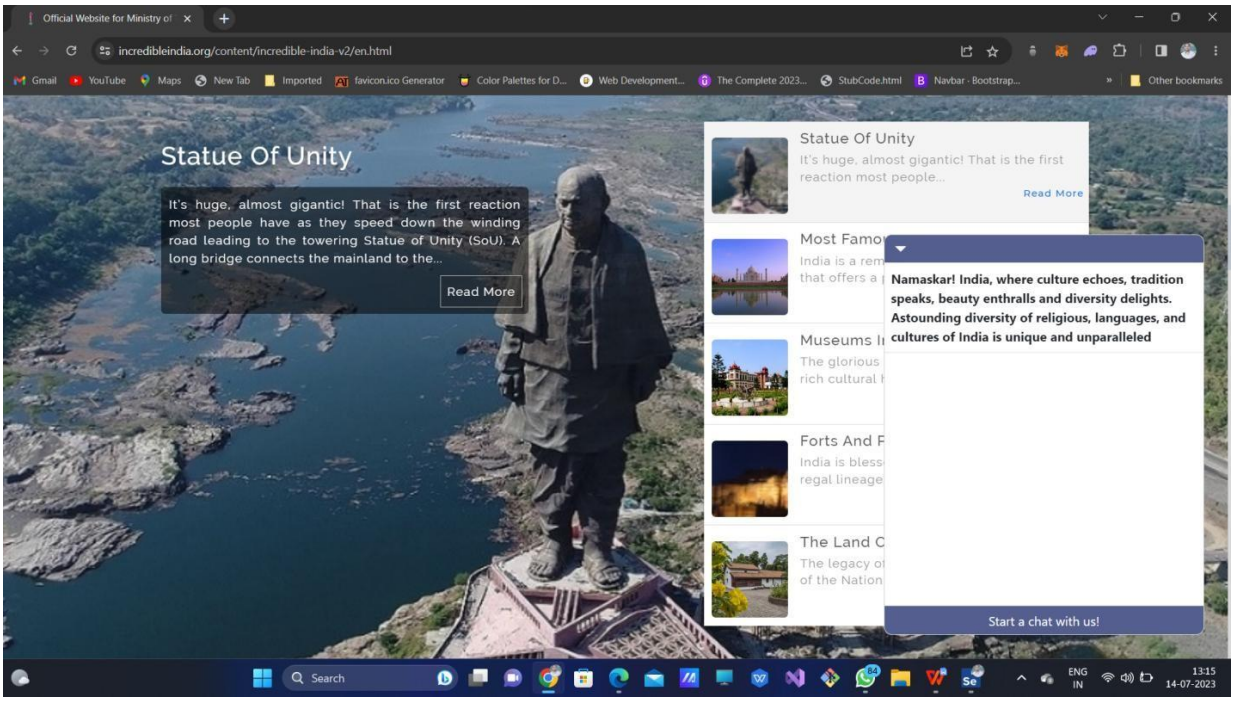


Test cases are recorded in Selenium



Test cases captured are playback in Selenium IDE

Test cases are executed on the same website as playback



Result:

Thus the capture and playback of test cases are done for web application testing as a regression test suite.

Software Requirements Specification

for

Smart Home Automation System

Version 1.0 approved

Prepared by Amisha Tripathi

21BCE1735

Vellore Institute of Technology, Chennai

10-06-2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this document is to define the requirements for the development of The SHA(Smart Home Automation) System. It outlines the functionality, performance, and constraints of the system to ensure a clear understanding between the stakeholders and the development team.
The general purpose of developing this application is to enable the homeowners to manage the home environment through the integration and automation of various devices and systems. The system aims to provide users with greater opportunity with greater control and flexibility in managing their homes while optimizing consumption and improving the overall living experience.

1.2 Document Conventions

SHA - Smart Home automation
SRS - Software Requirement Specification
GUI - Graphical User Interface
UI - User Interface

1.3 Intended Audience and Reading Suggestions

Intended Audience:

- Homeowners: Individuals who own or plan to own a residential property and are interested in enhancing their living spaces with smart home automation systems.
- Technology Enthusiasts: People who have a keen interest in emerging technologies and enjoy exploring the possibilities of integrating technology into their daily lives.
- DIYers (Do-It-Yourselfers): Individuals who enjoy building and customizing their own projects, and are interested in creating their own smart home automation systems.
- Home Builders and Architects: Professionals involved in the construction and design of residential properties, who may want to incorporate smart home automation systems into their *projects.*

Reading Suggestions:

- "Home Automation For Dummies" by Dwight Spivey: This book provides a beginner-friendly introduction to home automation, covering various aspects including smart home technologies, devices, and systems.
- "Smart Home Automation with Linux and Raspberry Pi" by Steven Goodwin: This book explores the implementation of smart home automation using Linux-based platforms like Raspberry Pi. It covers different technologies and protocols used in smart home systems.

- "Smart Home Automation with Alexa" by Mark Diamond: This book focuses on integrating Amazon Alexa into smart home automation systems, providing step-by-step instructions and examples.

- "Make: Bluetooth: Bluetooth LE Projects with Arduino, Raspberry Pi, and Smartphones" by Alasdair Allan: This book delves into the world of Bluetooth Low Energy (LE) and its applications in smart home automation. It provides practical projects and code examples using Arduino, Raspberry Pi, and smartphones.

Online Resources and Communities:

- Home Assistant Community (https://community.home-assistant.io/): Home Assistant is an open-source home automation platform. The community forum provides a wealth of information, tutorials, and user experiences.

- Reddit's r/smarthome (https://www.reddit.com/r/smarthome/): A subreddit dedicated to discussing and sharing ideas, news, and experiences related to smart home automation.

- Blogs and Websites: Explore popular smart home automation blogs and websites like CNET's Smart Home (https://www.cnet.com/smart-home/) and SmartThings Blog (https://blog.smartthings.com/) for articles, reviews, and guides.

1.4 Product Scope

The Smart Home Automation System aims to provide homeowners with a centralized platform to control and automate various aspects of their home environment. This includes managing lighting, HVAC systems, security systems, appliances, and other connected devices. The system will allow users to monitor and control these devices remotely through a user-friendly interface.

1.5 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

2. Overall Description

2.1 Product Perspective

The Smart Home Automation System is a revolutionary product that aims to transform traditional homes into intelligent, automated environments. It provides homeowners with a centralized control platform to manage and monitor various aspects of their homes, including security, energy

efficiency, and entertainment. The system integrates with a wide range of smart devices and employs an intuitive user interface to simplify interaction and customization.

2.2 Product Functions

- Control and Monitor Devices: The system should provide the ability to control and monitor various devices in the home, such as lights, thermostats, locks, cameras, and appliances, remotely or through voice commands.
- Automate Routine Tasks: Users should be able to set up automated routines or schedules for their devices, enabling tasks to be performed automatically at specific times or under certain conditions.
- Energy Management: The system should offer energy management features, allowing users to monitor and optimize energy usage in their home, such as adjusting thermostats based on occupancy or turning off lights when no one is in the room.
- Security and Safety: The product should include security and safety features like motion detection, door/window sensors, smoke detectors, and surveillance cameras, allowing users to monitor and secure their home remotely.
- Integration and Interoperability: The system should be capable of integrating with a wide range of devices, protocols, and platforms, enabling interoperability and providing a seamless user experience.
- Voice Control and Virtual Assistants: Users should be able to control the smart home system and its connected devices using voice commands through popular virtual assistants like Amazon Alexa, Google Assistant, or Apple Siri.
- Remote Access and Notifications: The system should allow users to remotely access and control their smart home devices through mobile apps or web interfaces. It should also provide notifications or alerts for events like security breaches, device malfunctions, or unusual activity.

2.3 User Classes and Characteristics

The Smart Home Automation System caters to different user classes, each with unique characteristics and requirements. The following are the typical user classes and their corresponding characteristics:

- Homeowners:

Characteristics: Homeowners are the primary users of the smart home automation system. They own and reside in the homes where the system is installed.
Requirements: Homeowners seek convenience, security, energy efficiency, and customization in their smart homes. They desire control over devices, easy-to-use interfaces, and the ability to personalize automation based on their preferences.

- Family Members:

Characteristics: Family members, including spouses, children, or elderly individuals, may also interact with the smart home automation system.
Requirements: Family members require user profiles and permissions to access and control the system according to their needs. They may have specific preferences for device settings, scenes, or security features.

- Guests:

Characteristics: Guests are temporary users who visit the homeowner's smart home.
Requirements: Guests may require limited access to the smart home system, such as temporary access codes for door locks or restricted control over specific devices. The system should provide a guest mode or temporary access options to accommodate their needs.

- Service Providers:

Characteristics: Service providers include technicians, maintenance personnel, or customer support representatives who may interact with the smart home system for installation, troubleshooting, or assistance.
Requirements: Service providers need access to diagnostic tools, configuration interfaces, or remote support capabilities to perform system installation, maintenance, or troubleshooting tasks efficiently.

- System Administrators:

Characteristics: System administrators are responsible for managing and maintaining the smart home automation system.
Requirements: System administrators require advanced access privileges and tools to configure system settings, manage user accounts, monitor system health, and perform software updates or maintenance tasks.

- Third-Party Integrators:

Characteristics: Third-party integrators are professionals or companies that develop or integrate external devices, services, or applications with the smart home system.
Requirements: Third-party integrators require well-documented APIs, development tools, and technical resources to seamlessly integrate their products or services with the smart home automation system.

2.4 Operating Environment

The most recent versions of Chrome, Firefox, Safari, and Microsoft Edge are recommended for use with our website.
Particulars - Operating System, Processor, Hard Disk, RAM
Client System – Operating system used is: Windows, Linux, Android, or iOS and for server system Operating system used is: Linux.
Hard Disk for Client system is 1GB and server system is 1TB.
RAM for client system is: 256 and server system is: 8GB.
Processor for client system and server system is: Intel or AMD

2.5 Design and Implementation Constraints

Any tool or software package, such as a Java Applet, Microsoft Front Page, Enterprise JavaBeans (EJB), etc. on the following languages, must be used to implement the user interface.

->Availability will degrade as a result of the backup's inability to be maintained.

-> Supports just HTTP/HTTPS.

-> Neither a banking system nor a credit card validation mechanism exists in the actual world.

->Client-Side Programming Language: HTML5, CSS, JavaScript

2.6 User Documentation

The following user documentation components are typically included:

User Manual:

A detailed user manual provides step-by-step instructions on how to install, set up, and operate the smart home automation system. It covers topics such as hardware requirements, system configuration, device pairing, user account management, and troubleshooting procedures.
The user manual should also include explanations of key concepts, terminology, and best practices to help users make the most of the system's features and functionalities.

On-boarding Guide:

An on-boarding guide helps new users get started with the smart home automation system. It provides a concise overview of the system's core features, user interface navigation, and initial setup steps. This guide aims to familiarize users with the basic functionality quickly and efficiently.

Online Help:

Online help is a context-sensitive resource accessible from within the system's user interface. It provides users with assistance and guidance specific to the task or feature they are currently using. Online help may include tooltips, pop-up explanations, and clickable links to relevant sections of the user manual or knowledge base.

Tutorials and How-to Guides:

Tutorials and how-to guides offer detailed instructions and examples on specific tasks or scenarios within the smart home automation system. These guides can include setting up automation routines, creating custom scenes, managing user profiles, integrating third-party devices, and optimizing energy efficiency. They provide users with practical guidance to accomplish specific objectives.

Troubleshooting Guide:

A troubleshooting guide assists users in diagnosing and resolving common issues they may encounter while using the smart home automation system. It provides a comprehensive list of potential problems, error messages, and suggested solutions. The troubleshooting guide may also include troubleshooting steps for specific devices or integrations.

Frequently Asked Questions (FAQs):

FAQs address common questions and concerns that users may have about the smart home automation system. It covers topics ranging from basic system functionality to advanced features, troubleshooting, and best practices. FAQs provide quick answers to common inquiries, improving the overall user experience.
Glossary of Terms:

A glossary of terms provides definitions and explanations of key terms and concepts used within the smart home automation system. It helps users understand technical jargon and ensures consistent communication between the documentation and the system's user interface.

2.7 Assumptions and Dependencies

Some assumed factors that could affect the requirements in the SRS for a smart home automation system include:

Third-Party Components:

Assumed Integration: Assuming seamless integration with specific third-party components, such as smart devices, voice assistants, or cloud services. Any changes or limitations in these components can affect the system's functionality and requirements.
Assumed Availability: Assuming the availability and continued support of specific third-party components. Changes in availability or discontinuation of these components may require alternative solutions or impact the system's requirements.

Development and Operating Environment:

Assumed Development Tools/Frameworks: Assuming the use of specific development tools or frameworks to build the system. Changes in the chosen tools or frameworks may impact the development process, resource requirements, and system performance.
Assumed Operating System(s): Assuming the system will run on specific operating systems, such as Windows, Linux, or mobile platforms. Changes in the supported operating systems may require adjustments to the system design and functionality.
Assumed Hardware Constraints: Assuming certain hardware constraints, such as processor capabilities, memory requirements, or connectivity options. Any discrepancies in the actual hardware environment may affect the system's performance and resource utilization.

Constraints and Limitations:

Assumed Performance Requirements: Assuming specific performance thresholds, such as response times, processing speed, or data transfer rates. Deviations from these assumptions may require performance optimizations or adjustments to meet the system's performance objectives.

Assumed Scalability: Assuming the system will accommodate a certain number of devices or users. Changes in scalability requirements may impact the design and architecture of the system.
Assumed Regulatory Compliance: Assuming compliance with specific regulations or industry standards. Changes in regulatory requirements may necessitate updates to the system's security measures, data handling practices, or user privacy considerations.

External Dependencies:

Assumed Reusability of Components: Assuming the reuse of software components from other projects. Dependencies on external chomponents or software from other projects may require coordination and integration efforts.
Assumed Availability of APIs: Assuming the availability and stability of APIs provided by third-party services or platforms. Changes in the APIs or limitations in their functionality may impact the system's integration capabilities.

3. External Interface Requirements

3.1 User Interfaces

Landing Page:

The landing page serves as the entry point to the website and should provide an overview of the smart home automation system. It may include a visually appealing layout, key features, and benefits of the system, and call-to-action buttons to encourage users to sign up or learn more.

User Registration and Login:

User registration and login interfaces allow users to create an account and securely access their smart home automation system. These interfaces typically include input fields for entering login credentials, password recovery options, and account creation forms. They should prioritize user security by implementing encryption, password strength requirements, and multi-factor authentication.

Dashboard:

The dashboard interface is the central hub of the smart home automation system, providing users with an overview of their connected devices, automation settings, and system status. It may include widgets, cards, or tiles displaying real-time data, such as device statuses, energy consumption, and alerts. The dashboard allows users to control devices, view activity logs, and manage automation rules.

Device Control:

Device control interfaces enable users to interact with individual smart devices connected to their home automation system. They typically provide controls for turning devices on/off, adjusting settings, and setting schedules or timers. Device control interfaces may include sliders, buttons, dropdown menus, or toggles, depending on the type of device and its supported functionalities.

Automation Management:

Automation management interfaces allow users to create, customize, and manage automation rules or routines. These interfaces should provide an intuitive and visual way for users to set up automation triggers, define actions, and specify conditions. Users can create rules based on time, sensor inputs, device states, or other events to automate their smart home system.

Energy Monitoring and Management:

Energy monitoring interfaces display real-time and historical energy consumption data to users. They may include graphs, charts, or meters to visualize energy usage patterns and help users identify opportunities for energy savings. These interfaces may also include features like setting energy goals, receiving energy efficiency recommendations, or integrating with utility providers for billing information.

User Profile and Settings:

User profile and settings interfaces allow users to personalize their smart home automation system according to their preferences. Users can modify their account information, notification preferences, privacy settings, and language preferences. These interfaces should be user-friendly and provide clear options for managing user profiles and system configurations.

Help and Support:

Help and support interfaces provide users with access to documentation, frequently asked questions (FAQs), troubleshooting guides, and customer support channels. They should be easily accessible from the website and offer self-service resources to address common user queries or issues. Additionally, users should have the option to contact customer support or submit support tickets through the website interface.

Mobile Responsiveness:

The website interface should be responsive and adaptable to different screen sizes, ensuring optimal user experience on various devices, including desktop computers, laptops, tablets, and smartphones. The interface should adjust its layout, elements, and navigation to provide a seamless user experience across different screen resolutions and orientations.

3.2 Hardware Interfaces

Here are some hardware interfaces that can be included in the website of a smart home automation system:

Device Compatibility Lists:

The website can provide a list of compatible smart home devices that users can integrate into their automation system. This list should include various categories of devices, such as lights, thermostats, door locks, cameras, and sensors. Each device entry should provide details about the device model, supported features, and any specific requirements or recommendations.

Integration Guides and APIs:

The website can offer integration guides and application programming interfaces (APIs) for developers or advanced users interested in integrating their own hardware devices with the smart home automation system. These guides should provide documentation, code samples, and

instructions for utilizing the provided APIs to enable communication and control between the custom hardware and the automation system.
Communication Protocols and Standards:

The website can provide information about the supported communication protocols and standards for integrating third-party hardware devices. This includes details about protocols like Wi-Fi, Bluetooth, Zigbee, Z-Wave, or Thread, as well as compatibility with standards such as HomeKit, SmartThings, or Works with Alexa/Google Assistant. Users can refer to this information to ensure their desired hardware devices align with the supported interfaces.

Connectivity Options:

The website can outline the connectivity options available for users to connect their smart home devices to the automation system. This can include wired connections like Ethernet or powerline networking, as well as wireless connections like Wi-Fi or Bluetooth. Users can understand the available connectivity options and their implications when selecting and configuring their smart home devices.

Hub or Gateway Requirements:

If the smart home automation system utilizes a central hub or gateway for device communication, the website can provide information about the hardware requirements and compatibility for such hubs. This can include details about specific hub models or hardware specifications required for seamless integration and optimal system performance.

Power and Electrical Requirements:

The website can provide guidance on power and electrical requirements for the smart home automation system. This includes information on voltage compatibility, power supply recommendations, and any specific electrical considerations users should be aware of when installing and operating their hardware devices.

Expansion and Scalability:

The website can highlight the system's capacity for expansion and scalability in terms of adding new hardware devices. This can include information about the maximum number of devices the system can support, suggestions for system architecture and organization, and any limitations or recommendations for optimizing system performance as the number of devices increases.

3.3 Software Interfaces

Java Development Kit (JDK)
Operating system: at least Windows 7 Ultimate with networking support.

3.4 Communications Interfaces

- **Email Communication:**

Requirement: The system should support email communication to send notifications, alerts, and system updates to users.
Message Formatting: Emails should be formatted in a user-friendly and informative manner, including relevant device information, event details, and actionable links if necessary.

- **Web Browser Communication:**

Requirement: The system's web-based user interface should be compatible with major web browsers (e.g., Chrome, Firefox, Safari) to ensure access from various devices.
Communication Protocol: HTTP (Hypertext Transfer Protocol) or HTTPS (HTTP Secure) can be used for communication between the web browser and the system's server.
Message Formatting: The system's web pages should be HTML/CSS compliant and responsive to different screen sizes, providing an optimal user experience.

- **Network Server Communication Protocols:**

Requirement: The system's components, such as the smart home hub/gateway and connected devices, should communicate with the network server securely and efficiently.
Communication Protocols: Commonly used protocols like TCP/IP, MQTT (Message Queuing Telemetry Transport), or CoAP (Constrained Application Protocol) can be employed for device-to-server communication.
Message Formatting: The communication messages should adhere to the specified protocol's message formatting guidelines, ensuring reliable transmission and proper interpretation by the server.

- **Electronic Forms:**

Requirement: The system may utilize electronic forms to collect user preferences, configuration settings, or feedback.
Form Structure: The electronic forms should be designed with clear labels, input fields, validation checks, and appropriate error handling to ensure accurate and user-friendly data entry.

- **Communication Standards:**

Requirement: The system should adhere to widely accepted communication standards to ensure compatibility with external services, protocols, or devices.
Standards: Common standards like HTTP, MQTT, REST (Representational State Transfer), or WebSockets can be employed for interoperability and integration with external systems.

- **Communication Security and Encryption:**

Requirement: The system should prioritize communication security to protect user data, prevent unauthorized access, and ensure the integrity of transmitted information.
Security Measures: Implementation of encryption mechanisms, such as SSL/TLS (Secure Sockets Layer/Transport Layer Security), can be employed to secure data transmission. User authentication and access control mechanisms should also be implemented.

- **Data Transfer Rates:**

Requirement: The system should ensure efficient data transfer rates, minimizing delays and ensuring real-time or near-real-time communication.
Transfer Rate Considerations: The system should be designed to handle the data transfer requirements of different devices and services, considering factors like network bandwidth, latency, and the nature of the transmitted data (e.g., sensor readings, command signals).

- **Synchronization Mechanisms:**

Requirement: The system should support synchronization mechanisms to ensure data consistency and coordination between different system components.
Synchronization Methods: Techniques such as periodic polling, event-driven updates, or publish-subscribe patterns can be employed to synchronize data across devices, user interfaces, and external services.

4. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 System Feature 1- DEVICE CONTROL

4.1.1 Description and Priority-

The ability to remotely control and manage connected devices within the home, such as lights, thermostats, locks, and appliances, through a central interface or mobile app.

4.1.2 Stimulus/Response Sequences

- On/Off control of lights, appliances, and electronic devices.
- Adjusting settings and configurations of connected devices.
- Monitoring the status and availability of devices.

4.1.3 Functional Requirements

REQ-DC-001: The system shall provide the ability to turn devices on or off remotely.

REQ-DC-002: The system shall support individual control of each connected device.

REQ-DC-003: The system shall allow users to adjust device settings, such as brightness, volume, or temperature.

REQ-DC-004: The system shall provide a centralized interface to control multiple devices simultaneously.

REQ-DC-005: The system shall support grouping of devices to enable synchronized control.

REQ-DC-006: The system shall display the status (on/off) of each device in real-time.

REQ-DC-007: The system shall provide a device discovery feature to detect and add new devices automatically.

REQ-DC-008: The system shall allow users to remove or unregister devices from the system.
REQ-DC-009: The system shall provide access control mechanisms to restrict device control based on user permissions.
REQ-DC-010: The system shall support device control through voice commands via integration with virtual assistants.
REQ-DC-011: The system shall allow users to create and manage presets or scenes for specific device configurations.
REQ-DC-012: The system shall provide a history log of device control actions for reference and auditing purposes.
REQ-DC-013: The system shall ensure secure and encrypted communication between the control interface and devices.
REQ-DC-014: The system shall provide options for users to customize device control interfaces according to their preferences.
REQ-DC-015: The system shall support scheduling of device control actions at specific times or recurring events.

4.2 System Feature 2- Automation Routines

4.1.1 Description and Priority-

The capability to create customized automation routines based on predefined triggers, conditions, and actions. This allows for seamless and automatic control of devices and scenarios, such as scheduling lights to turn on/off or adjusting thermostat settings based on occupancy.

4.1.2 Stimulus/Response Sequences

- Creating and managing customized automation routines.
- Setting triggers, conditions, and actions for automated functions.
- Scheduling and time-based automation for various devices.

4.1.3 Functional Requirements

REQ-AR-001: The system shall provide the ability to create and manage automation routines.

REQ-AR-002: The system shall support defining triggers for automation routines, such as time-based triggers, sensor inputs, or device states.

REQ-AR-003: The system shall allow users to set conditions or rules that must be met for an automation routine to execute.

REQ-AR-004: The system shall provide a wide range of available actions that can be executed as part of an automation routine.

REQ-AR-005: The system shall support chaining multiple actions within a single automation routine.

REQ-AR-006: The system shall allow users to specify the order of actions within an automation routine.

REQ-AR-007: The system shall provide a user-friendly interface for configuring and editing automation routines.

REQ-AR-008: The system shall support enabling or disabling automation routines as per user preference.

REQ-AR-009: The system shall provide a preview or simulation mode to test the execution of automation routines before activating them.

REQ-AR-010: The system shall allow users to create automation routines based on a combination of triggers and conditions.

REQ-AR-011: The system shall support the option to trigger automation routines remotely through the mobile app or web interface.

REQ-AR-012: The system shall provide the ability to assign automation routines to specific users or user groups.

REQ-AR-013: The system shall support the import and export of automation routines for backup and sharing purposes.

REQ-AR-014: The system shall provide a log or history of executed automation routines for troubleshooting and auditing purposes.

REQ-AR-015: The system shall ensure reliable and accurate execution of automation routines, minimizing delays or failures.

4.3 System Feature 3 -Energy Management

4.1.1 Description and Priority-

Features focused on energy efficiency and consumption monitoring, including energy usage analytics, smart meter integration, and automated energy-saving routines.

4.1.2 Stimulus/Response Sequences

- Monitoring energy consumption of devices.
- Providing energy usage analytics and insights.
- Implementing energy-saving recommendations and automation.

4.1.3 Functional Requirements

REQ-EM-001: The system shall provide real-time monitoring of energy consumption for connected devices.

REQ-EM-002: The system shall track and display historical energy usage data

for analysis and comparison.
REQ-EM-003: The system shall offer insights and recommendations for optimizing energy usage based on user behavior and device patterns.
REQ-EM-004: The system shall support the creation of personalized energy saving goals and targets.
REQ-EM-005: The system shall provide the ability to set schedules for device operations to maximize energy efficiency.
REQ-EM-006: The system shall allow users to receive alerts or notifications for abnormal energy consumption or high usage periods.
REQ-EM-007: The system shall integrate with smart energy meters or utility providers to obtain accurate energy consumption data.
REQ-EM-008: The system shall support the integration of renewable energy sources, such as solar panels or wind turbines, for monitoring and optimization.
REQ-EM-009: The system shall provide suggestions for energy-efficient device replacements or upgrades based on usage patterns.
REQ-EM-010: The system shall offer the ability to remotely control and manage energy-consuming devices, allowing users to turn off or reduce power when not in use.
REQ-EM-011: The system shall enable users to set budget limits for energy usage and receive notifications when approaching or exceeding the defined thresholds.
REQ-EM-012: The system shall support the integration of smart plugs or power strips to monitor and control energy usage for individual devices or groups of devices.
REQ-EM-013: The system shall calculate and display estimated energy costs based on current rates and device usage.
REQ-EM-014: The system shall provide visual representations, such as graphs or charts, to visualize energy consumption patterns and trends.
REQ-EM-015: The system shall ensure data security and privacy for energy consumption information, adhering to relevant regulations and standards.

4.4 System Feature 4 -Security and Surveillance

4.1.1 Description and Priority-

Integration with security systems, sensors, cameras, and alarms for monitoring and securing the home. This includes features like real-time video monitoring,

motion detection, door/window sensors, and alerts for security breaches.

4.1.2 Stimulus/Response Sequences

- Integration with security systems, sensors, and alarms.
- Real-time monitoring and video surveillance.
- Intrusion detection and alert notifications.

4.1.3 Functional Requirements

REQ-SS-001: The system shall integrate with security cameras to provide real-time video monitoring of the home.

REQ-SS-002: The system shall support motion detection and alert notifications when unauthorized movement is detected.

REQ-SS-003: The system shall provide the ability to record video footage and store it locally or in the cloud.

REQ-SS-004: The system shall allow users to access live video feeds from security cameras through a mobile app or web interface.

REQ-SS-005: The system shall support remote pan, tilt, and zoom controls for security cameras.

REQ-SS-006: The system shall allow users to schedule and customize motion detection zones for security cameras.

REQ-SS-007: The system shall integrate with door/window sensors to detect and notify users of unauthorized access or breaches.

REQ-SS-008: The system shall provide a log or history of security-related events and notifications.

REQ-SS-009: The system shall support the integration of smart locks for remote locking and unlocking of doors.

REQ-SS-010: The system shall allow users to create and manage personalized security modes, such as home, away, or vacation mode.

REQ-SS-011: The system shall provide the ability to arm or disarm security systems remotely.

REQ-SS-012: The system shall support integration with third-party monitoring services or emergency response systems.

REQ-SS-013: The system shall allow users to receive alerts and notifications for security-related events, such as fire or carbon monoxide detection.

REQ-SS-014: The system shall provide options for privacy settings, including the ability to disable cameras or sensors when desired.

REQ-SS-015: The system shall ensure secure communication and encryption for all security-related data and interactions.

4.5 System Feature 4 - Integration with Third-Party Devices and Services

4.1.1 Description and Priority-

Compatibility and integration with a wide range of smart devices, protocols, and services, enabling seamless control and coordination between different brands and technologies.

4.1.2 Stimulus/Response Sequences

- Integration with security systems, sensors, and alarms.
- Real-time monitoring and video surveillance.
- Intrusion detection and alert notifications.

4.1.3 Functional Requirements

REQ-INT-001: The system shall support seamless integration with a wide range of third-party smart devices, including lights, thermostats, locks, and sensors.

REQ-INT-002: The system shall provide compatibility with popular smart home protocols such as Zigbee, Z-Wave, Wi-Fi, and Bluetooth.

REQ-INT-003: The system shall allow users to discover and add third-party devices to the system with ease.

REQ-INT-004: The system shall provide a unified interface to control and manage all integrated third-party devices.

REQ-INT-005: The system shall support bidirectional communication with third party devices for status updates and control commands.

REQ-INT-006: The system shall allow users to create automation routines or scenarios involving third-party devices and services.

REQ-INT-007: The system shall enable users to access and control third-party devices remotely through the mobile app or web interface.

REQ-INT-008: The system shall support voice control integration with popular virtual assistants such as Amazon Alexa or Google Assistant for third-party devices.

REQ-INT-009: The system shall provide the ability to authenticate and securely connect with third-party services or cloud platforms.

REQ-INT-010: The system shall ensure compatibility and interoperability with certified third-party devices and ecosystems.

REQ-INT-011: The system shall allow users to manage and update the firmware of integrated third-party devices.

REQ-INT-012: The system shall provide an interface or API for developers to integrate custom or specialized third-party devices.

REQ-INT-013: The system shall support data synchronization and sharing between

the smart home system and third-party applications or platforms.

REQ-INT-014: The system shall offer documentation and resources for developers to facilitate the integration of new third-party devices or services.

REQ-INT-015: The system shall provide a mechanism for users to rate and review the compatibility and performance of integrated third-party devices.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

- Responsiveness: The system should exhibit quick response times when executing user commands or triggering automated actions. It should minimize delays to ensure a seamless user experience.
- Reliability: The system should be reliable and operate consistently without frequent failures or downtime. It should be able to handle simultaneous device control and automation tasks without compromising performance.
- Scalability: The system should be scalable, allowing for the addition of new devices and functionalities as the smart home network expands. It should handle increased device connections and data processing without significant performance degradation.
- Compatibility: The system should be compatible with a wide range of devices, protocols, and platforms commonly used in smart homes. It should support interoperability to ensure smooth integration and communication between different devices.
- Network Performance: The system should optimize network performance to handle the data traffic generated by connected devices. It should minimize latency, ensure stable connections, and support reliable data transmission.
- Energy Efficiency: The system should be designed to optimize energy consumption, both in terms of the system itself and the connected devices. It should have low power requirements and provide energy management features to help users monitor and reduce energy usage.
- Security: The system should prioritize security to protect user privacy and prevent unauthorized access. It should implement robust encryption, authentication mechanisms, and secure communication protocols to ensure the integrity and confidentiality of data.
- User-Friendly Interface: The system should have an intuitive and user-friendly interface that makes it easy for users to interact with and control their smart home devices. The interface should provide clear feedback and guidance to enhance user experience.

- Data Processing and Analytics: If the system includes data processing and analytics capabilities, it should be able to handle large volumes of data efficiently. It should perform data analysis in a timely manner and provide relevant insights and recommendations to users.

- Compatibility with Virtual Assistants: If the system integrates with virtual assistants like Amazon Alexa or Google Assistant, it should ensure smooth and reliable communication with these platforms. Voice commands and interactions should be accurately interpreted and executed.

5.2 Safety Requirements

- Electrical Safety: The system should comply with relevant electrical safety standards and regulations. It should be designed to prevent electrical hazards such as short circuits, overloading, and electrical fires.

- Fire Safety: The system should incorporate fire safety measures, including smoke detectors, heat sensors, and fire suppression systems where applicable. These components should be properly installed, regularly maintained, and tested to ensure their effectiveness.

- Physical Safety: The system should not pose any physical risks to users or occupants of the home. Devices and installations should be securely mounted, tamper-resistant, and designed to prevent injuries or accidents, especially when dealing with high-voltage components or moving parts.

- Secure Access Control: The system should implement robust access control mechanisms to prevent unauthorized access and tampering. This includes secure user authentication, encryption, and protection against hacking or cyber-attacks that could compromise user safety.

- Emergency Shutoff: The system should provide an emergency shutoff mechanism or a clearly designated manual override to quickly disable automated functions in case of emergencies or system malfunctions. This ensures that users can regain control and mitigate potential risks.

- Device Safety Compliance: All devices connected to the smart home automation system should comply with relevant safety standards and certifications. This includes electrical safety, electromagnetic compatibility, and environmental safety requirements.

- Privacy Protection: The system should prioritize user privacy by implementing strong security measures to protect personal data and sensitive information. It should adhere to applicable data protection regulations and provide clear privacy policies to users.

- User Education and Documentation: The system should come with comprehensive user documentation, including safety guidelines, installation instructions, and troubleshooting procedures. Users should be educated on the safe operation, maintenance, and potential risks associated with the system.

- Monitoring and Alerts: The system should be equipped with monitoring capabilities to detect abnormal conditions, faults, or safety hazards. It should provide timely alerts to users, allowing them to take appropriate actions to ensure safety.

- Regular Maintenance and Updates: The system should be designed with regular maintenance in mind, including software updates, security patches, and device inspections. Regular maintenance helps ensure that the system remains safe, secure, and functioning optimally.

5.3 Security Requirements

- Authentication and Authorization: The system should implement robust authentication mechanisms to ensure that only authorized users can access and control the system. This includes strong password policies, two-factor authentication, or biometric authentication where applicable.

- Secure Communication: The system should utilize secure communication protocols, such as Transport Layer Security (TLS), to encrypt data transmitted between devices, the system's hub, and remote access interfaces. This ensures the confidentiality and integrity of sensitive information.

- Secure Device Integration: When integrating third-party devices or services, the system should verify their security features, adherence to standards, and potential vulnerabilities. It should prioritize devices that have undergone security testing and follow best practices.

- Secure Configuration: The system should encourage users to follow secure configuration practices, such as changing default passwords, enabling automatic software updates, and disabling unnecessary features or services that may introduce security risks.

- Privacy Protection: The system should prioritize user privacy by implementing privacy-enhancing features. This includes data anonymization or pseudonymization, minimizing data collection and storage, and providing clear privacy policies to users.

- Intrusion Detection and Prevention: The system should incorporate intrusion detection and prevention mechanisms to identify and respond to potential unauthorized access attempts or malicious activities. This may involve network monitoring, anomaly detection, or behavior-based analysis.

- Secure Remote Access: If the system allows remote access, it should provide secure access methods, such as encrypted VPN connections or secure cloud-based services, to protect against unauthorized access or interception of sensitive data.

- Regular Security Updates: The system should have a mechanism to receive and apply security updates and patches promptly. This includes both the system's software and firmware updates for connected devices to address known vulnerabilities.

- User Access Management: The system should provide robust user access management capabilities, allowing administrators to assign different access levels and permissions to users. This helps ensure that users only have access to the functions and devices they are authorized to control.

- Incident Response and Recovery: The system should have a well-defined incident response plan to address security breaches or incidents effectively. This includes mechanisms to detect and respond to security events, as well as data backup and recovery procedures.

5.4 Software Quality Attributes

- Adaptability: The system should be able to accommodate new devices and technologies with minimal effort. A quantitative measure could be the time required to integrate a new device into the system, aiming for a specific timeframe, such as within 10 minutes.

- Availability: The system should be accessible and operational for users whenever needed. A quantitative measure could be the system's uptime, aiming for at least 99% availability over a specified period, such as a month.

- Correctness: The system should accurately perform its intended functions without errors or incorrect outputs. A quantitative measure could be the percentage of commands executed correctly, aiming for a 99% success rate.

- Flexibility: The system should offer customization options to adapt to different user preferences and scenarios. A quantitative measure could be the number of configurable parameters or options available, aiming for a minimum of 20 customizable settings.

- Interoperability: The system should seamlessly integrate with various devices, protocols, and platforms. A quantitative measure could be the number of supported protocols or devices, aiming for compatibility with at least five commonly used protocols (e.g., Zigbee, Z-Wave, Wi-Fi).

- Maintainability: The system should be easy to maintain, update, and troubleshoot. A quantitative measure could be the average time to perform a system update or apply a software patch, aiming for less than 5 minutes.

- Portability: The system should be easily transferable to different hardware or software environments. A quantitative measure could be the time required to migrate the system to a new platform, aiming for a maximum of 1 hour.

- Reliability: The system should consistently perform its functions without failures or disruptions. A quantitative measure could be the mean time between failures (MTBF), aiming for an MTBF of at least 1,000 hours.

- Reusability: The system should have reusable components or modules that can be utilized in different contexts. A quantitative measure could be the percentage of reusable code or modules, aiming for a minimum of 80% code reusability.

- Usability: The system should be intuitive and easy to use for various user groups. A quantitative measure could be the average time required for a new user to set up and configure the system, aiming for a maximum of 30 minutes.

5.5 Business Rules

1. User Roles and Permissions: The system can define different user roles with varying levels of access and permissions. For example:
- Administrators: Have full control over the system, including adding and removing devices, creating automation routines, and managing user accounts.
- Standard Users: Can control devices, create personal automation routines, and customize settings within their assigned permissions.
- Guests: Have limited access and can only control specific devices or perform predefined actions.

2. Device Ownership: The system can assign device ownership to specific users. The device owner has primary control and configuration rights over the device, including granting or revoking access for other users.

3. Remote Access: The system may allow authorized users to access and control their smart home devices remotely, typically through a mobile app or web interface. Remote access can be granted to users with appropriate authentication and authorization.

4. Automated Actions: The system can perform automated actions based on predefined rules or triggers. For example:

- Motion Detection: When motion is detected, specific lights turn on and a notification is sent to the device owner.
- Time-based Automation: Devices can be programmed to turn on or off at specific times of the day.
- Voice Control: The system may support voice control through virtual assistants like Amazon Alexa or Google Assistant. Users can issue voice commands to control their devices or trigger automation routines.

5. Override and Emergency Control: The system should provide mechanisms for overriding automated actions or emergency control. For example:

6. Manual Controls: Users can manually operate devices or disable automation routines. Emergency Shutdown: In case of emergencies, a designated emergency shutdown button or voice command can disable all automated functions.

7. User Feedback and Notifications: The system can provide feedback and notifications to users. This includes alerts for device malfunctions, security breaches, or system updates. Notifications can be delivered through various channels, such as push notifications, emails, or SMS.

6. Other Requirements

The tools you're utilising to create your content are a further factor to take into account throughout the internationalisation step. Although there are many excellent content management systems (CMS) available, how well they support global markets depends on their differences. Make sure it is appropriate for localization if you use one. It must offer a method for producing content that can be translated in a manner that translation software can comprehend. Alternatives include XLIFF (XML Localization Interchange File Format) and other XML-based formats. Remerging the translated data into the appropriate locations in the localised material is another capability that the tool must have.

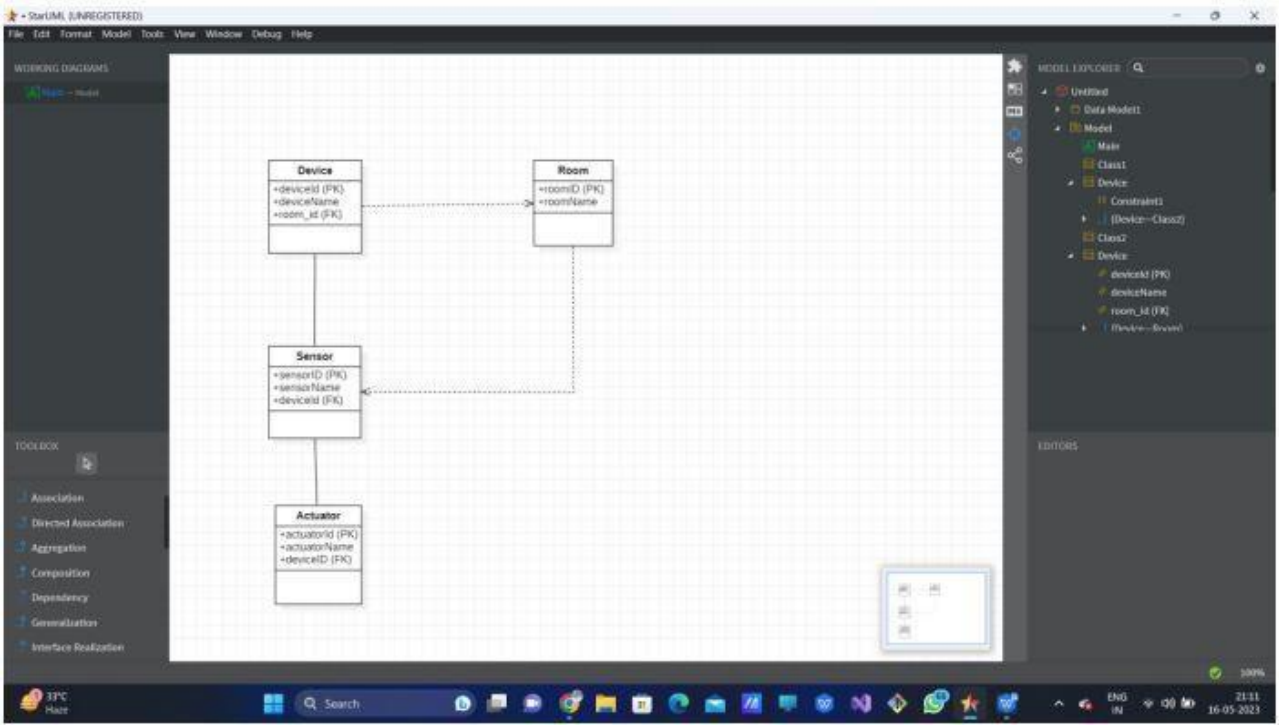
Appendix A: Glossary

GUI- Graphical User Interface
DNS - Domain Name Space
SHA- Smart Home Automation
SRS- Software Requirement Specification
SA-Stock is available
NIL-Stock is not available

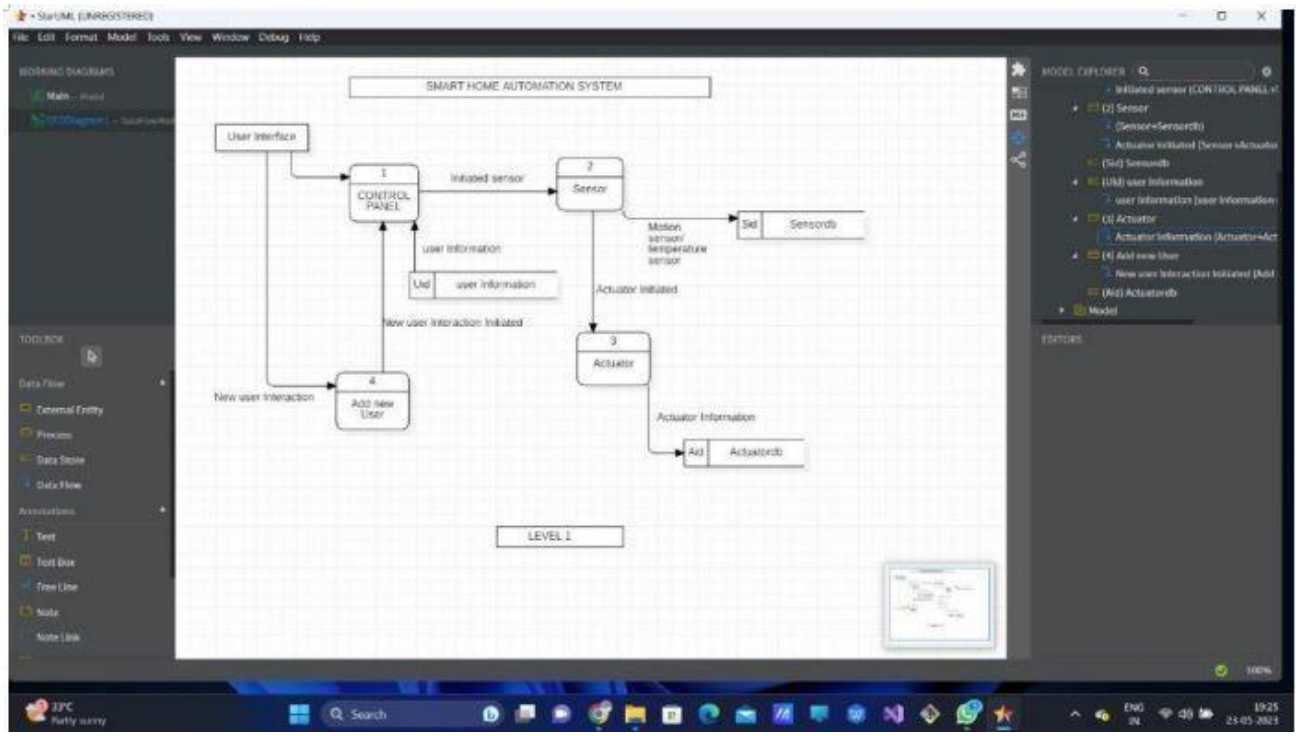
Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

ER DIGRAM



DATA FLOW DIAGRAM



Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>

EXPERIMENT

TOPIC : AGILE STORY BOARDING

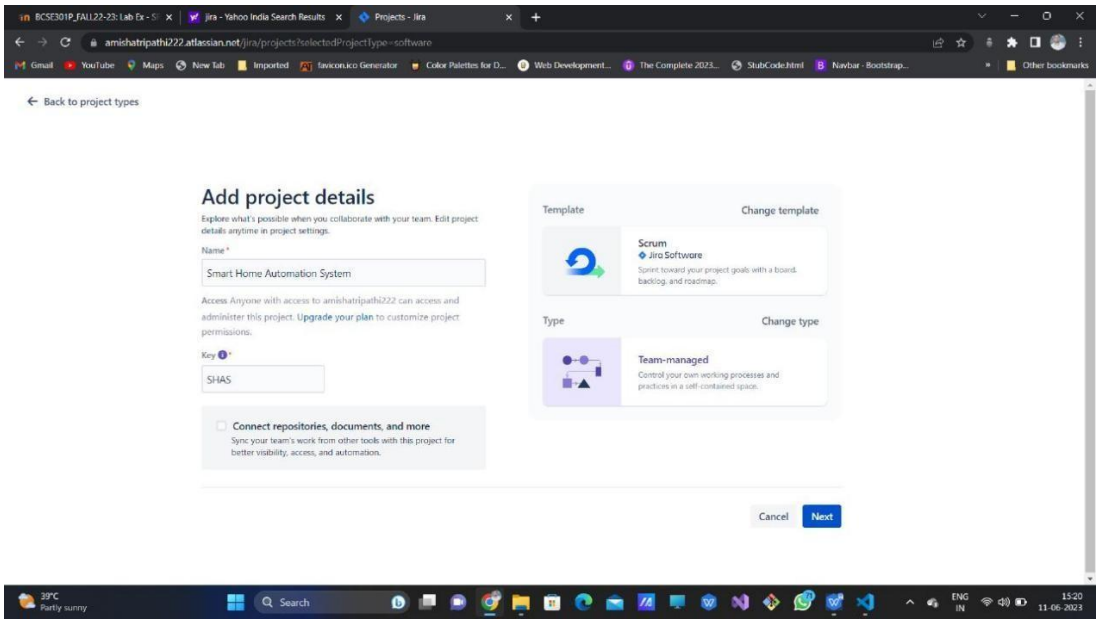
Aim:

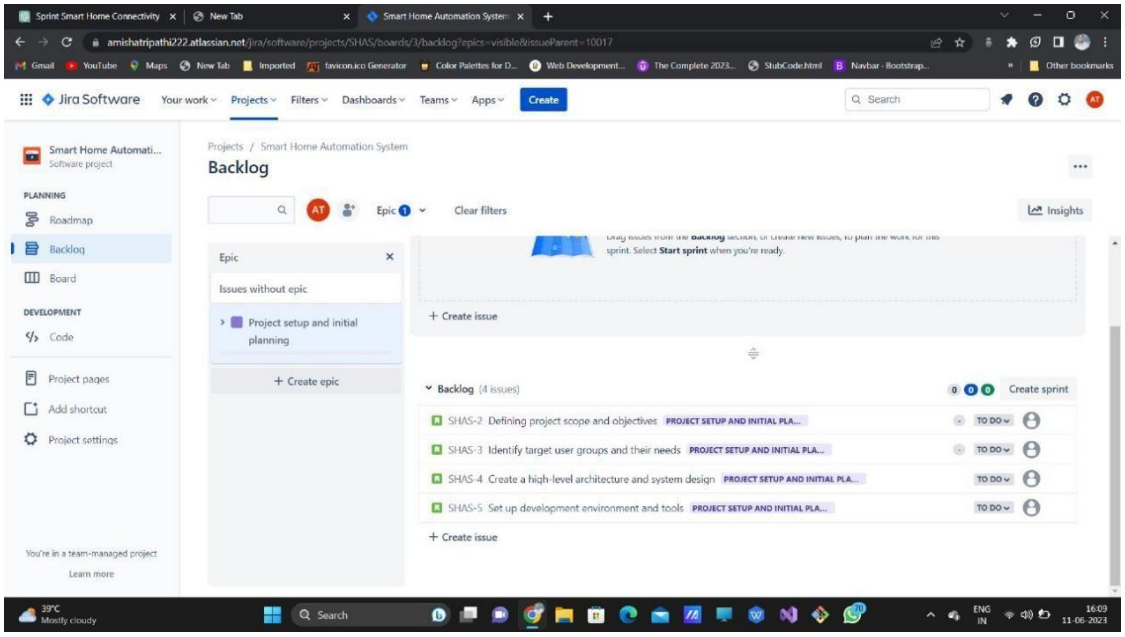
To create user stories for “SMART HOME AUTOMATION SYSTEM” and demonstrate test cases for the same.

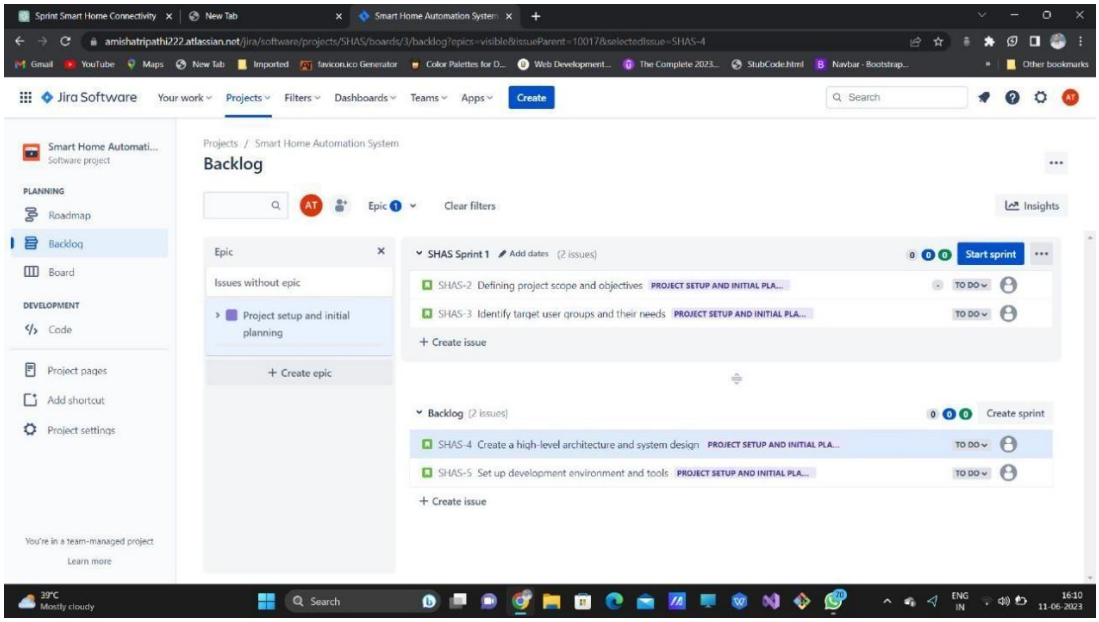
Description:

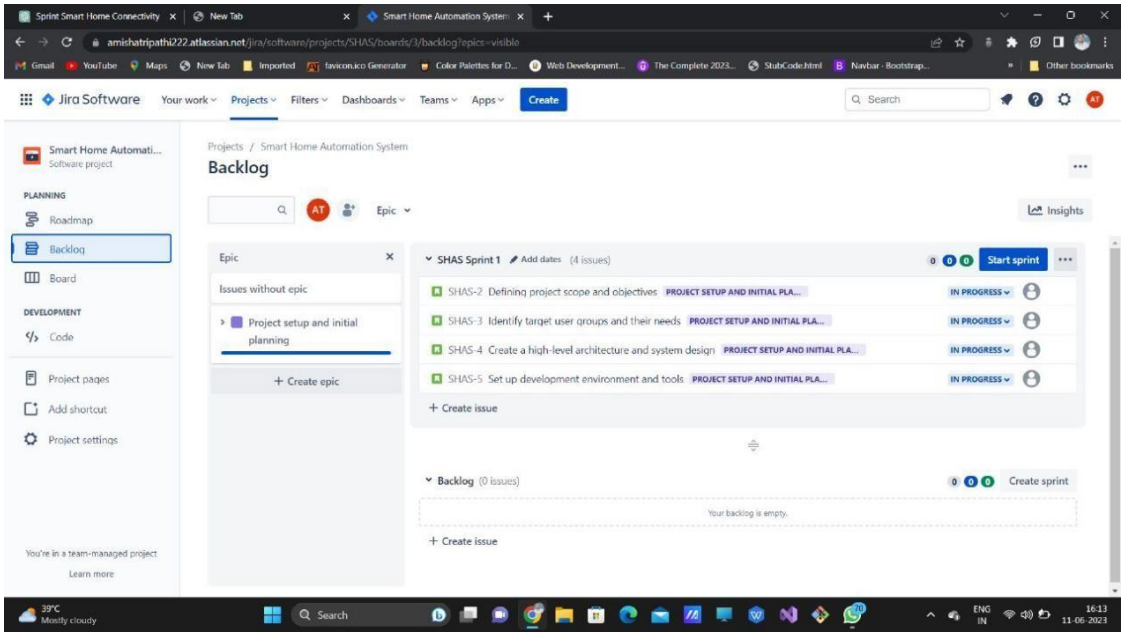
A user story is a short and simple description of a feature or a requirement from the perspective of the end user or customer. A scrum is a framework for managing complex projects by breaking them down into smaller chunks called epics, which are further divided into user stories. Scrum helps teams collaborate and deliver value to the customer in short iterations called sprints.

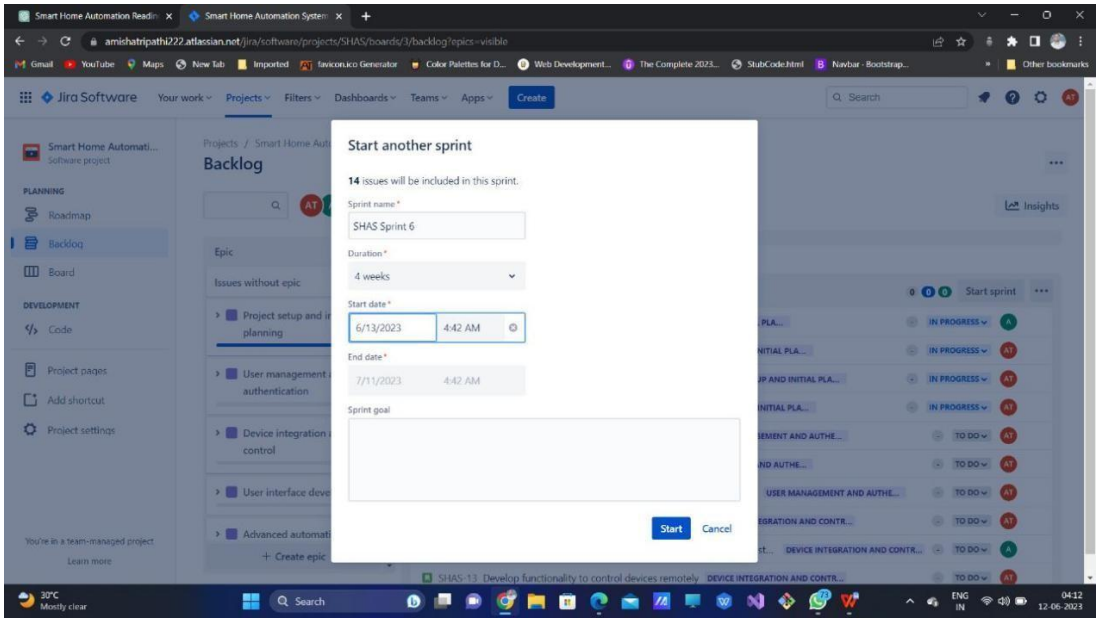
OUTPUT:

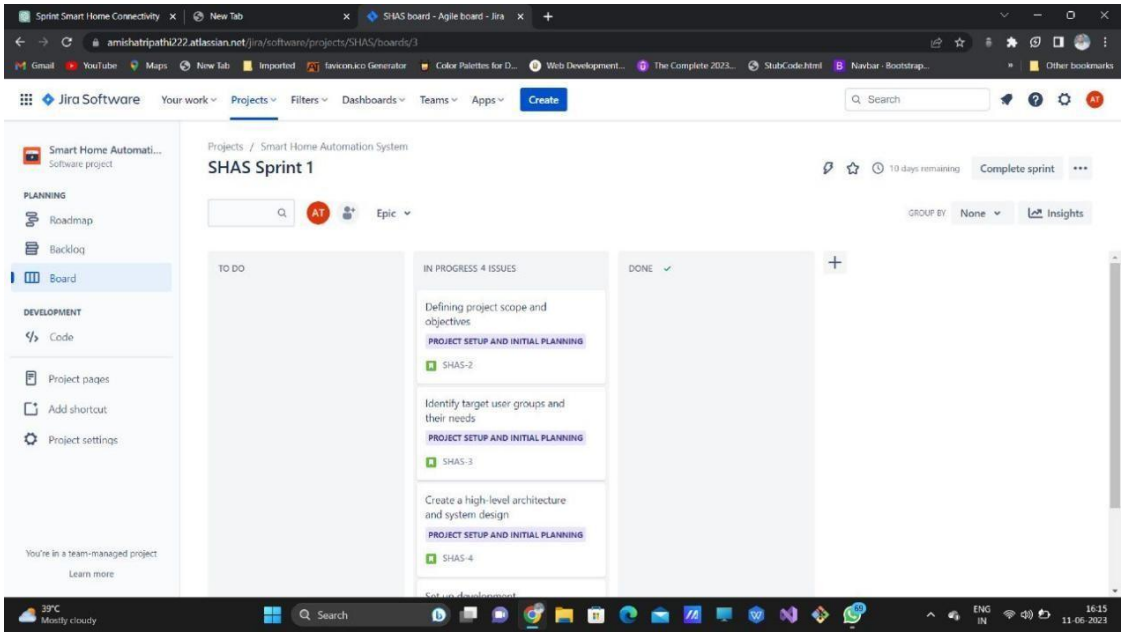


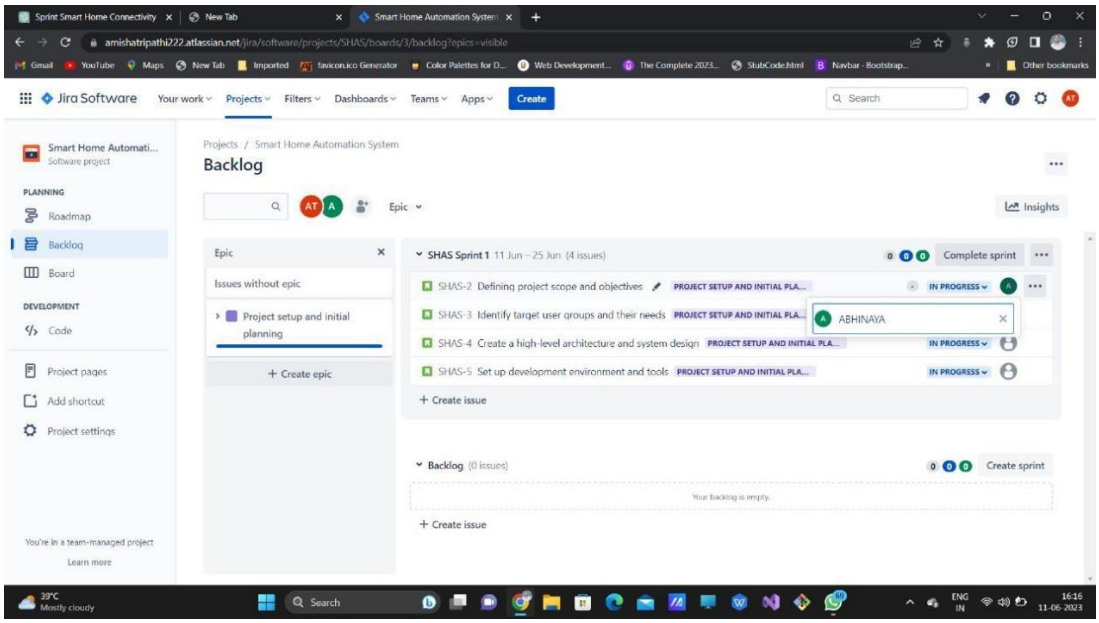

Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Smart Home Automati...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Smart Home Automation System
Backlog
Epic
Clear filters
Insights
Issues without epic
Project setup and initial planning
+ Create epic
+ Create issue
Backlog (4 issues)
Create sprint
SHAS-2 Defining project scope and objectives
SHAS-3 Identify target user groups and their needs
SHAS-4 Create a high-level architecture and system design
SHAS-5 Set up development environment and tools
PROJECT SETUP AND INITIAL PLA...
TO DO

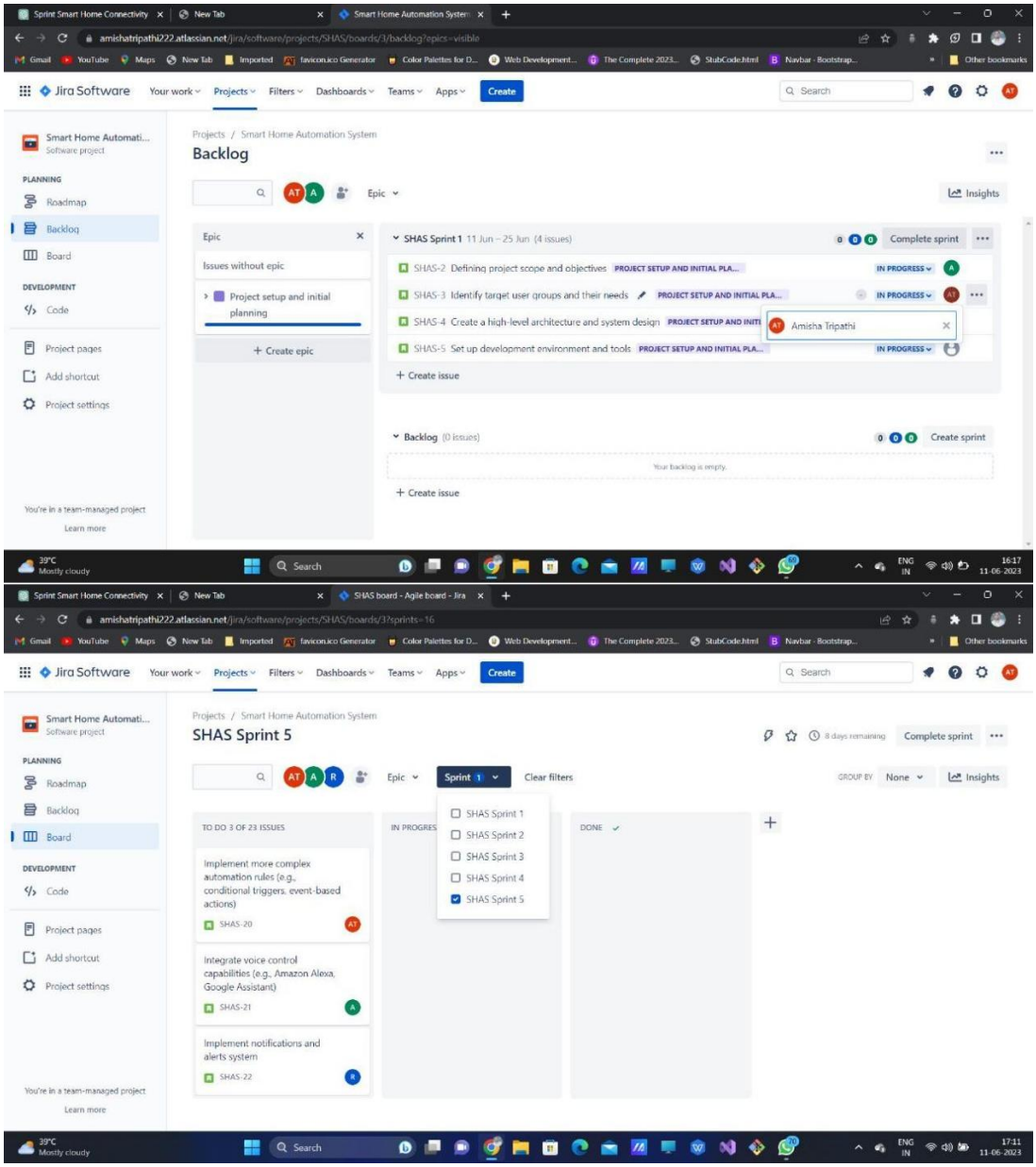

Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Smart Home Automati...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Smart Home Automation System
Backlog
Epic
Clear filters
Insights
Issues without epic
Project setup and initial planning
+ Create epic
SHAS Sprint 1
Add dates
(2 issues)
Start sprint
SHAS-2 Defining project scope and objectives
SHAS-3 Identify target user groups and their needs
+ Create issue
Backlog (2 issues)
Create sprint
SHAS-4 Create a high-level architecture and system design
SHAS-5 Set up development environment and tools
PROJECT SETUP AND INITIAL PLA...
TO DO

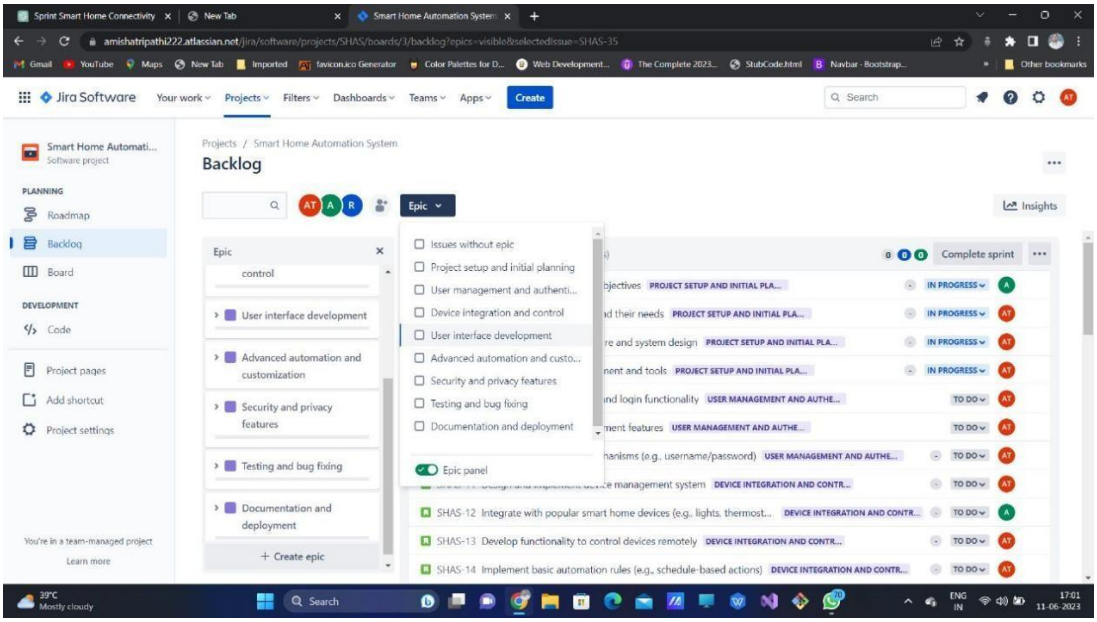

Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Smart Home Automati...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Smart Home Automation System
Backlog
Epic
Insights
Issues without epic
Project setup and initial planning
+ Create epic
SHAS Sprint 1
Add dates
(4 issues)
Start sprint
SHAS-2 Defining project scope and objectives
SHAS-3 Identify target user groups and their needs
SHAS-4 Create a high-level architecture and system design
SHAS-5 Set up development environment and tools
PROJECT SETUP AND INITIAL PLA...
IN PROGRESS
+ Create issue
Backlog (0 issues)
Create sprint
Your backlog is empty.
+ Create issue

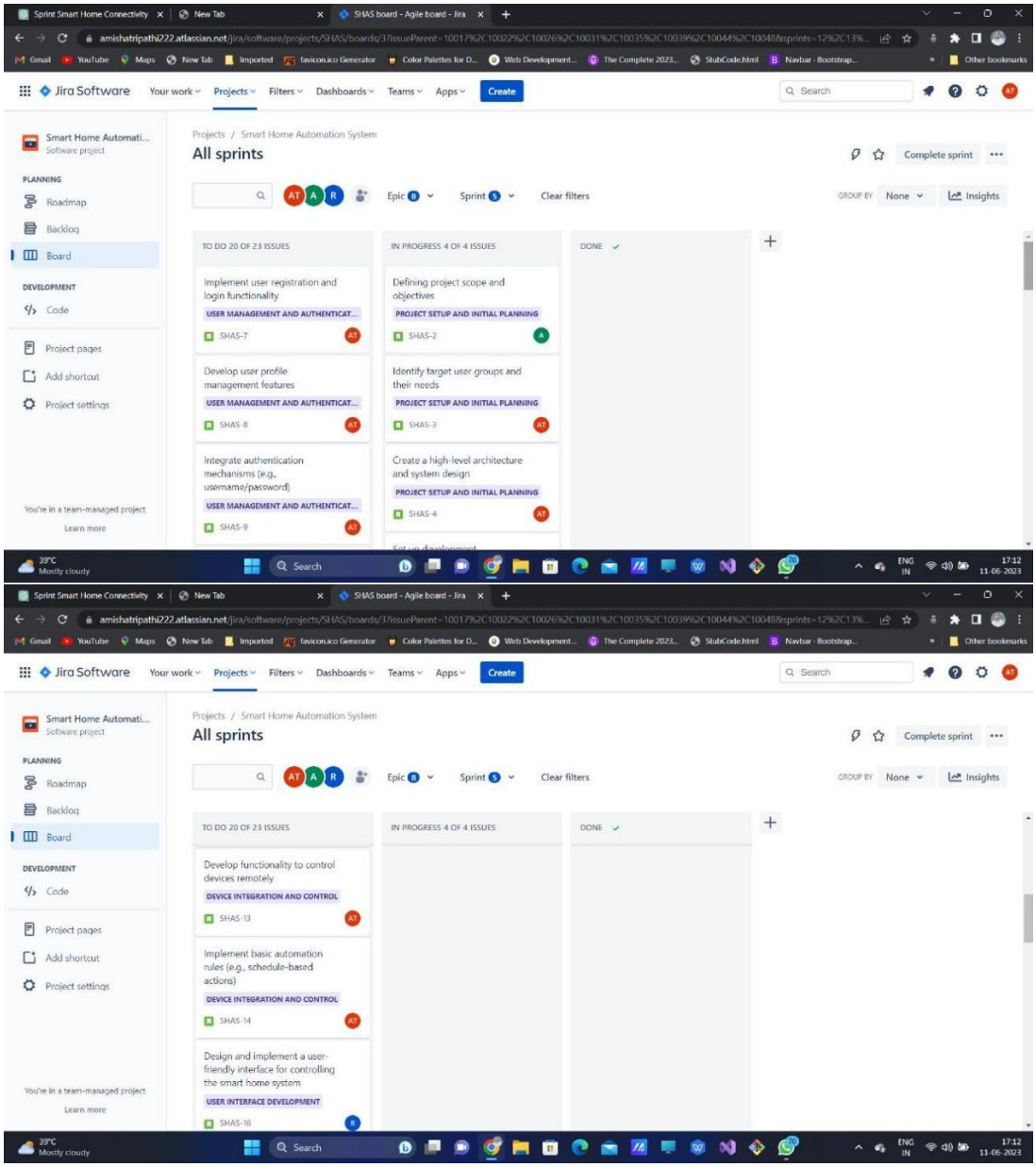

Jira Software
Backlog
Start another sprint
14 issues will be included in this sprint.
Sprint name *
SHAS Sprint 6
Duration *
4 weeks
Start date *
6/13/2023
4:42 AM
End date *
7/11/2023
4:42 AM
Sprint goal
Start
Cancel


Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Smart Home Automati...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Smart Home Automation System
SHAS Sprint 1
10 days remaining
Complete sprint
Epic
GROUP BY
None
Insights
TO DO
IN PROGRESS 4 ISSUES
DONE
Defining project scope and objectives
PROJECT SETUP AND INITIAL PLANNING
SHAS-2
Identify target user groups and their needs
PROJECT SETUP AND INITIAL PLANNING
SHAS-3
Create a high-level architecture and system design
PROJECT SETUP AND INITIAL PLANNING
SHAS-4
39°C
Mostly cloudy
Search
ENG
IN
16:15
11-06-2023

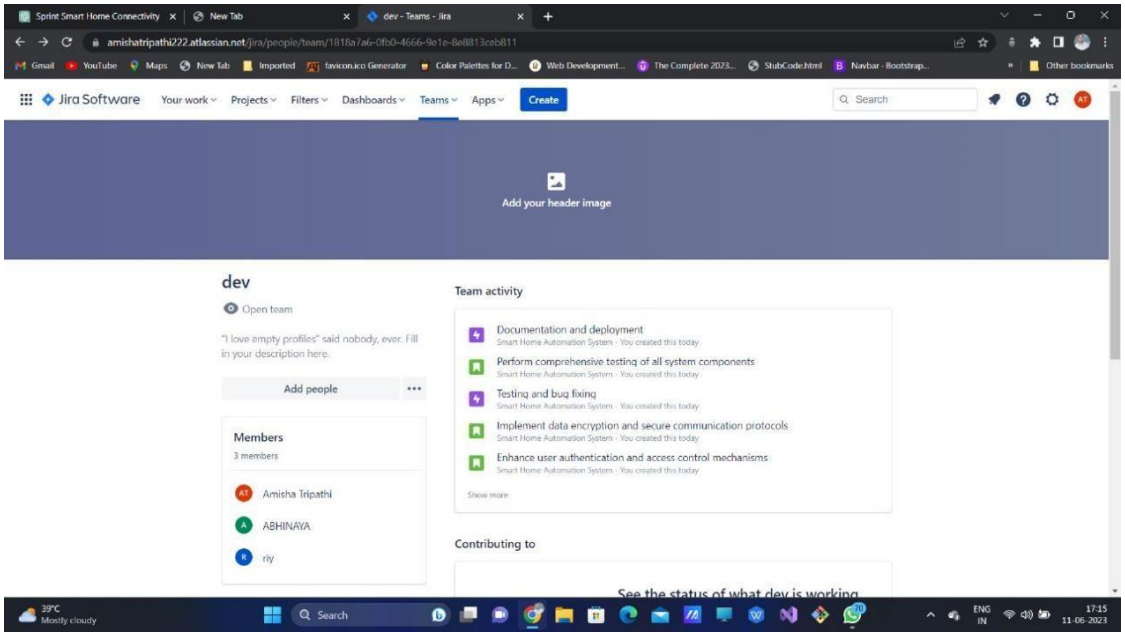

Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Smart Home Automati...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Smart Home Automation System
Backlog
Epic
Insights
Issues without epic
Project setup and initial planning
+ Create epic
SHAS Sprint 1 11 Jun – 25 Jun (4 issues)
Complete sprint
SHAS-2 Defining project scope and objectives
SHAS-3 Identify target user groups and their needs
ABHINAYA
SHAS-4 Create a high-level architecture and system design
SHAS-5 Set up development environment and tools
PROJECT SETUP AND INITIAL PLA...
IN PROGRESS
+ Create issue
Backlog (0 issues)
Create sprint
Your backlog is empty.
+ Create issue
39°C
Mostly cloudy
Search
ENG
IN
16:16
11-06-2023

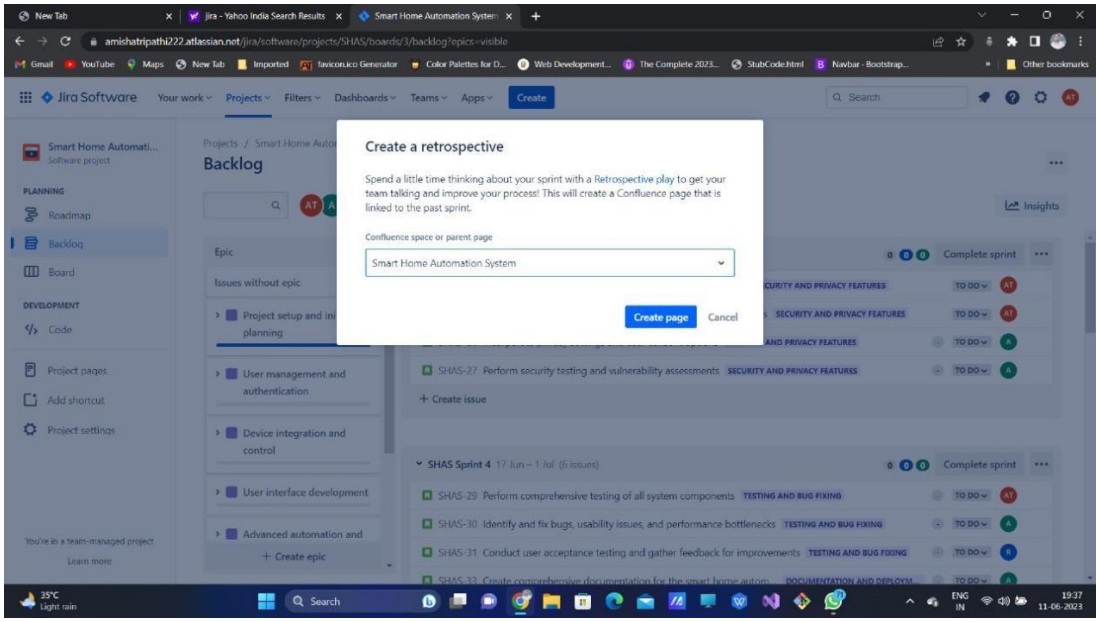

Jira Software
Backlog
SHAS Sprint 1
Create sprint
Create issue
SHAS Sprint 5
Complete sprint
Clear filters
Implement more complex automation rules (e.g., conditional triggers, event-based actions)
Integrate voice control capabilities (e.g., Amazon Alexa, Google Assistant)
Implement notifications and alerts system

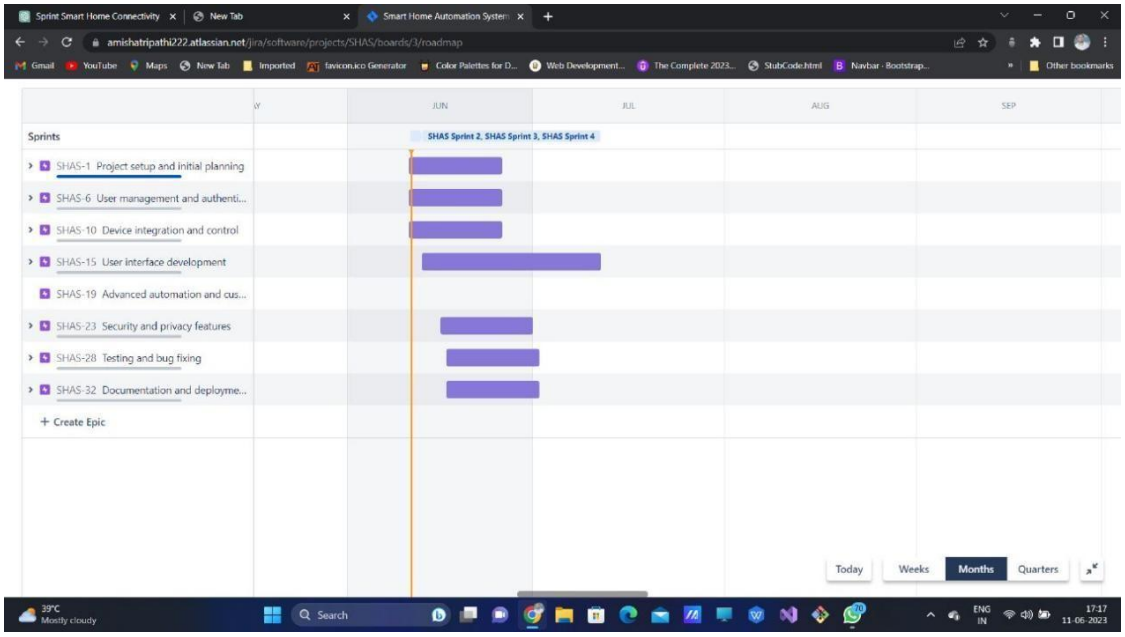

Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Smart Home Automati...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Smart Home Automation System
Backlog
Epic
Insights
Complete sprint
Issues without epic
Project setup and initial planning
User management and authenti...
Device integration and control
User interface development
Advanced automation and custo...
Security and privacy features
Testing and bug fixing
Documentation and deployment
Epic panel
User interface development
Advanced automation and customization
Security and privacy features
Testing and bug fixing
Documentation and deployment
+ Create epic
SHAS-12 Integrate with popular smart home devices (e.g., lights, thermost...
SHAS-13 Develop functionality to control devices remotely
SHAS-14 Implement basic automation rules (e.g., schedule-based actions)
39°C
Mostly cloudy
Search
ENG
IN
17:01
11-06-2023

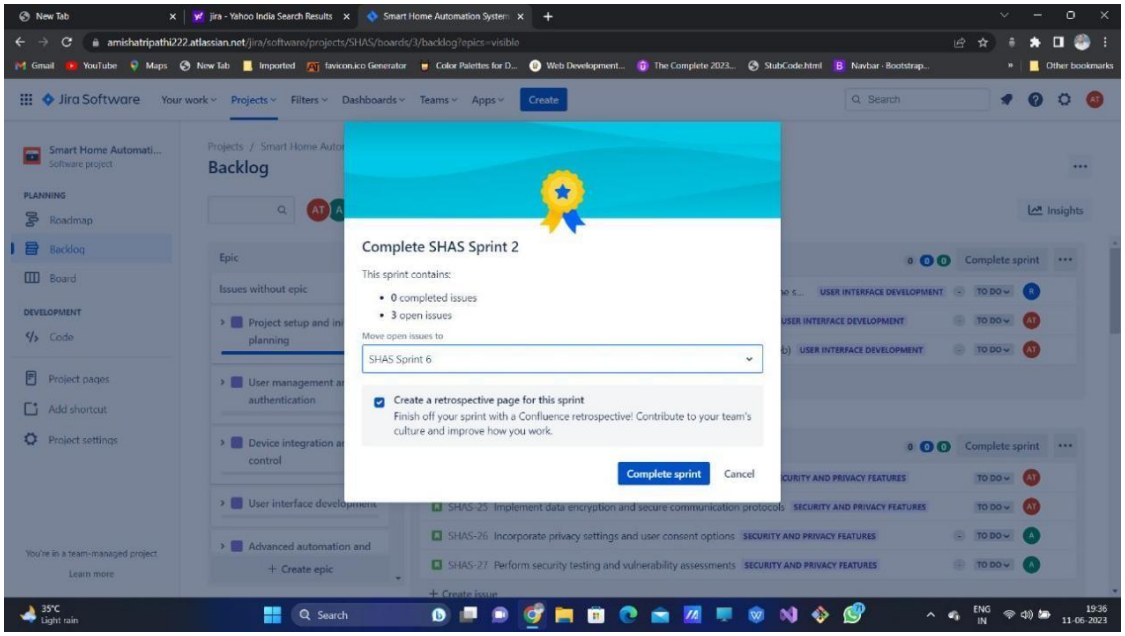

Sprint Smart Home Connectivity
New Tab
SHAS board - Agile board - Jira
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Smart Home Automati...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Smart Home Automation System
All sprints
Complete sprint
Epic
Sprint
Clear filters
GROUP BY
None
Insights
TO DO 20 OF 23 ISSUES
IN PROGRESS 4 OF 4 ISSUES
DONE
Implement user registration and login functionality
USER MANAGEMENT AND AUTHENTICAT...
SHAS-7
Defining project scope and objectives
PROJECT SETUP AND INITIAL PLANNING
SHAS-2
Develop user profile management features
USER MANAGEMENT AND AUTHENTICAT...
SHAS-8
Identify target user groups and their needs
PROJECT SETUP AND INITIAL PLANNING
SHAS-3
Integrate authentication mechanisms (e.g., username/password)
USER MANAGEMENT AND AUTHENTICAT...
SHAS-9
Create a high-level architecture and system design
PROJECT SETUP AND INITIAL PLANNING
SHAS-4
39°C
Mostly cloudy
Search
ENG
IN
17:12
11-06-2023

Sprint Smart Home Connectivity
New Tab
SHAS board - Agile board - Jira
Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Smart Home Automati...
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Smart Home Automation System
All sprints
Complete sprint
Epic
Sprint
Clear filters
GROUP BY
None
Insights
TO DO 20 OF 23 ISSUES
IN PROGRESS 4 OF 4 ISSUES
DONE
Develop functionality to control devices remotely
DEVICE INTEGRATION AND CONTROL
SHAS-13
Implement basic automation rules (e.g., schedule-based actions)
DEVICE INTEGRATION AND CONTROL
SHAS-14
Design and implement a user-friendly interface for controlling the smart home system
USER INTERFACE DEVELOPMENT
SHAS-16
39°C
Mostly cloudy
Search
ENG
IN
17:12
11-06-2023


Jira Software
Your work
Projects
Filters
Dashboards
Teams
Apps
Create
Search
Add your header image
dev
Open team
"I love empty profiles" said nobody, ever. Fill in your description here.
Add people
Members
3 members
Amisha Tripathi
ABHINAYA
riy
Team activity
Documentation and deployment
Perform comprehensive testing of all system components
Testing and bug fixing
Implement data encryption and secure communication protocols
Enhance user authentication and access control mechanisms
Contributing to


Jira Software
Backlog
Create a retrospective
Spend a little time thinking about your sprint with a Retrospective play to get your team talking and improve your process! This will create a Confluence page that is linked to the past sprint.
Confluence space or parent page
Smart Home Automation System
Create page
Cancel
Complete sprint
Create issue
SHAS Sprint 4





Result:

A scrum sprint was created with user stories and it was completed when all the tasks were done.