NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

RoomieLah

[Project Plan]
Version 1.0

By: Team StrawHats

Revision History

Revision Number	Date	Primary Author(s)	Comments
1.0	15th March 2022	Aks Tayal, Srishti Arora, Pratyush Kumar Pandey, Rajagopal Iyer, Sanath Surawar, Atul Acharya, Gopal Agarwal	First version

Table of Contents

1 Introduction

1.1 Project Overview

RoomieLah is a mobile application that helps university students find roommates based on preferences and habits. It uses the user's characteristics and preferences to recommend suitable roommates and users can match with other users and initiate conversations. RoomieLah aims to help people choose their roommate to ease the fear of moving in with strangers when they enter a foreign land.

1.2 Project Description and Scope

RoomieLah is a cross platform mobile application for all university students in Singapore where the user can create an account and fill their details and roommate characteristic preferences, and the recommendation system of the application suggests potential roommates who might be compatible with the user. The user can match with these recommended users and start conversations and have the option of connecting with them on various social media. The larger aim is to integrate the RoomieLah application with current One Stop solutions for university students.

While initially RoomieLah will be limited to only university students in Singapore, it can later expand to become a generic roommate recommendation application for anyone searching for a person to move in with. A web version of the application will be developed after the initial release of the mobile application.

Users must fill their preferred qualities in a roommate for the matching algorithm (recommendation system) to suggest potential matches. Users have the option of editing their preferences at any time. If the matching algorithm cannot find any suitable matches it will notify the user that they need to edit their preferences to find new matches.

Currently, the user must answer a series of predefined questions of categorical and ordinal types. However, we plan to extend it to use Natural Language Processing solutions so that the user has the freedom to input sentences which will improve the recommendations generated by the matching algorithm.

Users get access to the chat feature once they match with other users. The system allows the user to un-match with the other user if they don't think it's a suitable match. The messages are end to end encrypted to maintain user privacy.

Further improvements may be added based on user feedback obtained by beta testing the application.

2 Project Organization

2.1 Team Structure

The following is the list of executive roles, as required by CMM level 3.

- Project Manager: Srishti Arora
- Lead Developer: Pratyush Kumar Pandey
- Frontend Developer: Sanath Surawar
- Backend Developer: Rajagopal Iyer
- Quality Assurance Manager: Atul Acharya
- Quality Assurance Engineer: Aks Tayal
- Release Engineer: Gopal Agarwal

2.2 Roles and Responsibilities

Project Manager: Srishti Arora

- Oversees project progress
- Approves and executes project plan
- Assigns tasks and reports status of project to team members
- Manages and motivates team members
- Represents the team to the outside world

Lead Developer: Pratyush Kumar Pandey

- Leads the backend and frontend developers during the implementation of the project
- Advises the Project Manager on all development and technical related decisions
- Responsible for technical aspects during product release

Frontend Developer: Sanath Surawar

- Develop the frontend interface of RoomieLah based on system and user requirements.
- Ensures the User Interface and User Experience of the application adheres to internationally approved standards.
- Participate in the entire software development life cycle, generating working product including relevant documents and source code.

Backend Developer: Rajagopal Iyer

- Develop the backend of RoomieLah, which includes the server application and database based on the system and user requirements
- Participate in the entire software development life cycle, generating working product including relevant documents and source code
- Enforces security and data protection standards in the application

Quality Assurance Manager: Atul Acharya

- Ensure acceptable product and process quality
- Ensure proper implementation of quality assurance procedures

Quality Assurance Engineer: Aks Tayal

- Ensures acceptable software quality
- Designs testing strategies
- Creates and manages test plan
- Verify software requirements
- Executes test procedures

Release Engineer: Gopal Agarwal

- Manage the release of the product
- Create baseline build and integrate changes for delivery

2.3 Team Communication

TLA communication channels include the following:

- Bi-weekly physical meetings held on Wednesday
- Weekly virtual meetings held on Monday through Microsoft Teams
- Daily stand-ups from 10:00 – 10:30
- Group messages using Slack
- File sharing using Google Drive

3 Process Definition

3.1 Lifecycle Model

The team intends to use the Waterfall model throughout the entire project. This model is more structured than other traditional methods as each phase must be completed before the next phase of development. It is relatively easier to accommodate any changes made to the software during the development process.

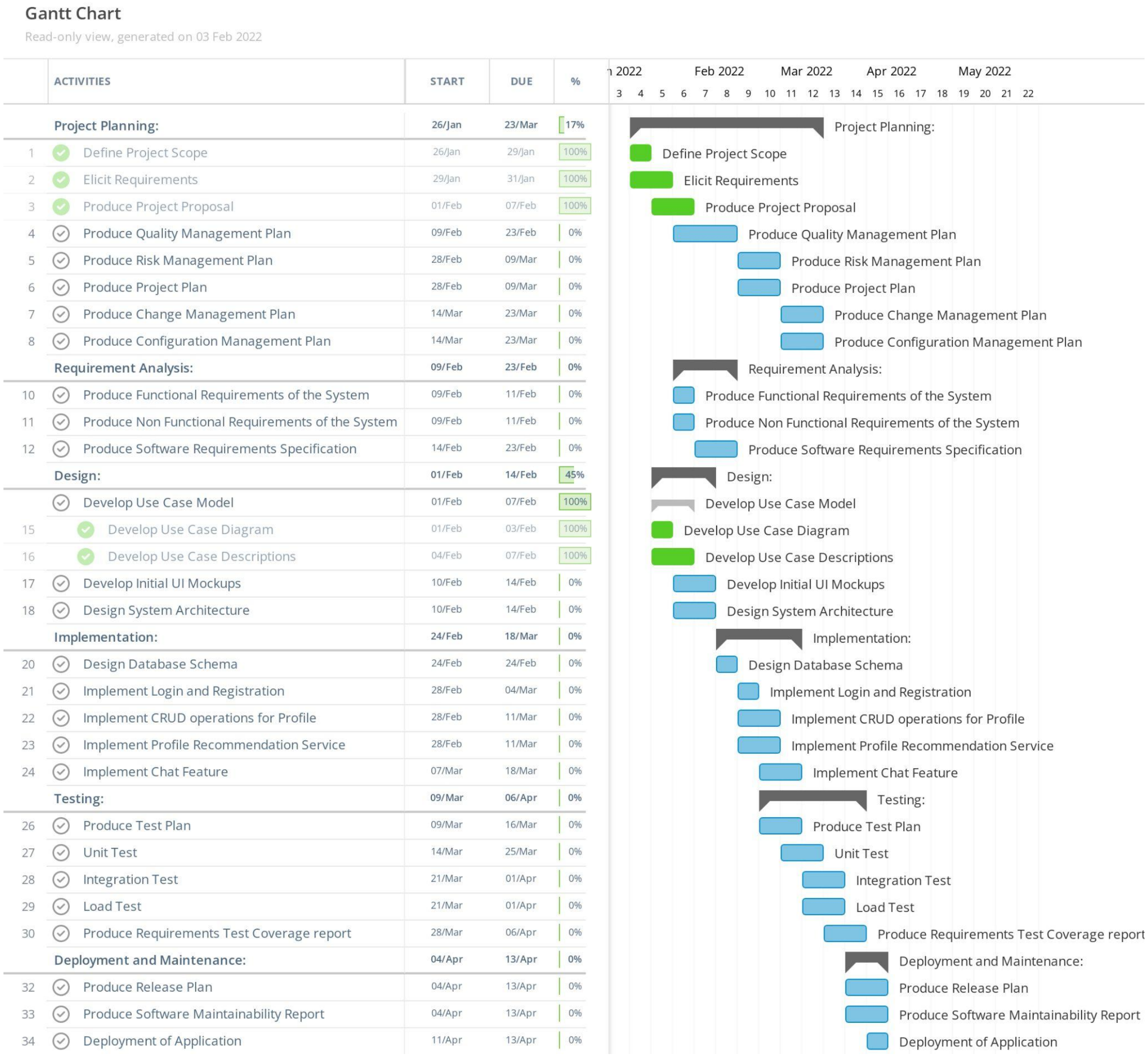
Given the financial constraints, adopting the Waterfall model will help decrease costs, with the project being completely dependent on the project team with minimal client intervention. Since this framework uses a step-by-step approach, errors are easy to detect and can be corrected early. Early detection of errors through appropriate Verification and Validation models before the completion of each stage, reduces the cost and time needed to correct them and improves the overall quality throughout the entire life cycle.

We have chosen to avoid other methods such as Agile because they are not document driven and do not involve proper planning which are necessary for RoomieLah. Proper design, quality management and development schedule are critical for RoomieLah.

We also chose to avoid Spiral lifecycle models because they are complex and often drive up the costs. Time management is also difficult using such models.

4 Schedule

4.1 Activities, Schedule and Dependencies



Gantt Chart

Read-only view, generated on 03 Feb 2022

	ACTIVITIES	START	DUE	%
	Project Planning:	**26/Jan**	**23/Mar**	**17%**
1	✓ Define Project Scope	26/Jan	29/Jan	100%
2	✓ Elicit Requirements	29/Jan	31/Jan	100%
3	✓ Produce Project Proposal	01/Feb	07/Feb	100%
4	✓ Produce Quality Management Plan	09/Feb	23/Feb	0%
5	✓ Produce Risk Management Plan	28/Feb	09/Mar	0%
6	✓ Produce Project Plan	28/Feb	09/Mar	0%
7	✓ Produce Change Management Plan	14/Mar	23/Mar	0%
8	✓ Produce Configuration Management Plan	14/Mar	23/Mar	0%
	Requirement Analysis:	**09/Feb**	**23/Feb**	**0%**
10	✓ Produce Functional Requirements of the System	09/Feb	11/Feb	0%
11	✓ Produce Non Functional Requirements of the System	09/Feb	11/Feb	0%
12	✓ Produce Software Requirements Specification	14/Feb	23/Feb	0%
	Design:	**01/Feb**	**14/Feb**	**45%**
	✓ Develop Use Case Model	01/Feb	07/Feb	100%
15	✓ Develop Use Case Diagram	01/Feb	03/Feb	100%
16	✓ Develop Use Case Descriptions	04/Feb	07/Feb	100%
17	✓ Develop Initial UI Mockups	10/Feb	14/Feb	0%
18	✓ Design System Architecture	10/Feb	14/Feb	0%
	Implementation:	**24/Feb**	**18/Mar**	**0%**
20	✓ Design Database Schema	24/Feb	24/Feb	0%
21	✓ Implement Login and Registration	28/Feb	04/Mar	0%
22	✓ Implement CRUD operations for Profile	28/Feb	11/Mar	0%
23	✓ Implement Profile Recommendation Service	28/Feb	11/Mar	0%
24	✓ Implement Chat Feature	07/Mar	18/Mar	0%
	Testing:	**09/Mar**	**06/Apr**	**0%**
26	✓ Produce Test Plan	09/Mar	16/Mar	0%
27	✓ Unit Test	14/Mar	25/Mar	0%
28	✓ Integration Test	21/Mar	01/Apr	0%
29	✓ Load Test	21/Mar	01/Apr	0%
30	✓ Produce Requirements Test Coverage report	28/Mar	06/Apr	0%
	Deployment and Maintenance:	**04/Apr**	**13/Apr**	**0%**
32	✓ Produce Release Plan	04/Apr	13/Apr	0%
33	✓ Produce Software Maintainability Report	04/Apr	13/Apr	0%
34	✓ Deployment of Application	11/Apr	13/Apr	0%

n 2022
Feb 2022
Mar 2022
Apr 2022
May 2022
3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
Project Planning:
Define Project Scope
Elicit Requirements
Produce Project Proposal
Produce Quality Management Plan
Produce Risk Management Plan
Produce Project Plan
Produce Change Management Plan
Produce Configuration Management Plan
Requirement Analysis:
Produce Functional Requirements of the System
Produce Non Functional Requirements of the System
Produce Software Requirements Specification
Design:
Develop Use Case Model
Develop Use Case Diagram
Develop Use Case Descriptions
Develop Initial UI Mockups
Design System Architecture
Implementation:
Design Database Schema
Implement Login and Registration
Implement CRUD operations for Profile
Implement Profile Recommendation Service
Implement Chat Feature
Testing:
Produce Test Plan
Unit Test
Integration Test
Load Test
Produce Requirements Test Coverage report
Deployment and Maintenance:
Produce Release Plan
Produce Software Maintainability Report
Deployment of Application

Figure 1: Gantt chart for the project

4.2 Work Breakdown Structure

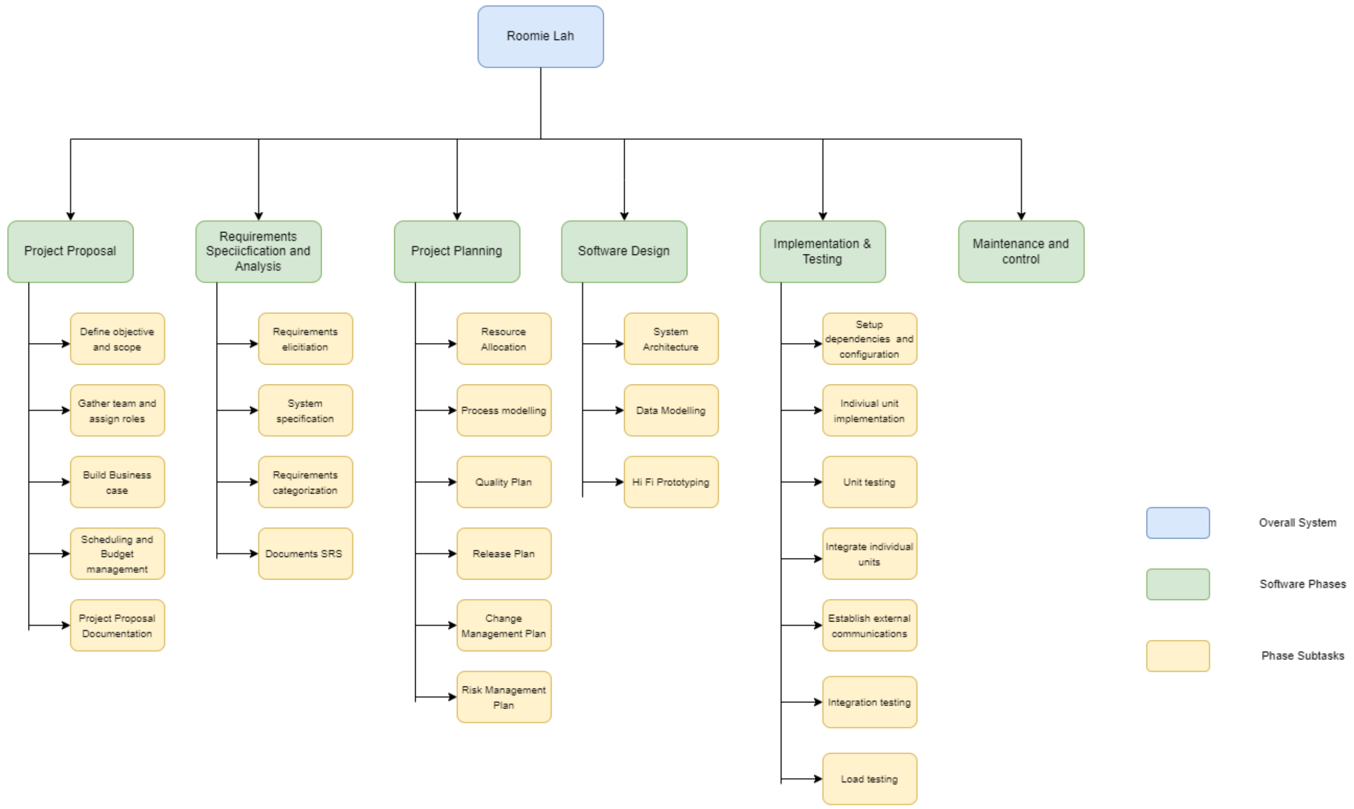


Figure 2: Work Breakdown Structure

The project work is divided based on the different phases of the SDLC.

4.3 Work Packages

The entire project work is broken down by the important phases of the software development life cycle. They include the following:

1. Project Proposal
2. Project Plan
3. Requirement Specification
4. Requirement Analysis and Documentation
5. Prototype and User Interface
6. System Architecture and Data Modeling
7. Software Implementation
8. Unit Testing
9. System Integration & Quality Assurance
10. Maintenance and Version Control

4.4 Activity Dependencies

The following table describes the dependencies of the deliverable work packages:

Work Package No.	Work Package Description	Duration	Dependencies
X01	Project Proposal	7 days	--
X02	Project Plan	7 days	X01
X03	Requirement Specification	7 days	X01
X04	Requirement Analysis and Documentation	7 days	X03
X05	Prototype and User Interface	7 days	X04
X06	System Architecture and Data Modeling	7 days	X04
X07	Software Implementation	14 days	X02, X06
X08	Unit Testing	7 days	X07
X09	System Integration & Quality Assurance	7 days	X08
X10	Maintenance and Version Control		X09

The following Activity Network Diagram describes the above in more graphical detail:

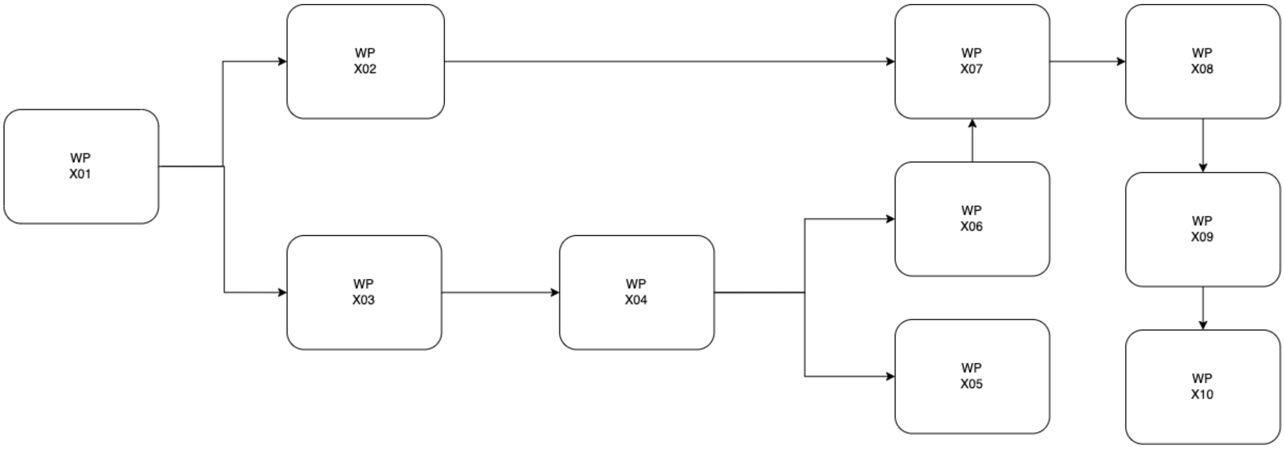


Figure 3: Activity Network Diagram

4.5 Work Package Details

Work packages are listed below. A team member, indicated in bold, has been assigned as primarily
responsible for each work package and will coordinate that package.

Table 1: Work package details for Project Proposal

Project	RoomieLah Mobile Application
Work Package	X01 - Project Proposal (1 of 10)
Assigned To	**Srishti Arora**, Pratyush Kumar Pandey, Rajgopal Iyer, Sanath Surawar, Atul Acharya, Aks Tayal, Gopal Agarwal
Effort	7 person-days
Start Date	28/01/2022
Purpose	To determine concept and overview of the project which will be refined in the following work packages.
Inputs	None
Activities	This work package consists of creating a project concept and formulating an overview of the project, its objectives, and a set of proposed deliverables throughout the software development life cycle. The assignees are also responsible for drafting the discussions from group meetings into a formal report.
Outputs	A written document of the Project Proposal and Use Case.

Table 2: Work package details for Project Plan

Project	RoomieLah Mobile Application
Work Package	X02 - Project Plan (2 of 10)
Assigned To	**Gopal Agarwal**, Srishti Arora, Pratyush Kumar Pandey, Rajagopal Iyer, Sanath Surawar, Atul Acharya, Aks Tayal
Effort	7 person-days
Start Date	04/02/2022
Purpose	To estimate project schedule, allocate project resources and conduct risk management
Inputs	Outputs from previous work package (X01)

Activities	This work package involves allocating project resources by work breakdown, estimating time to completion of various activities and prototyping the application through UI diagrams.
Outputs	A written document of the Project Plan, Risk Management Plan, Release Plan, Change Management Plan and prototype demonstration

Table 3: Work package details for Requirement Specification

Project	RoomieLah Mobile Application
Work Package	X03 - Requirement Specification (3 of 10)
Assigned To	**Pratyush Kumar Pandey,** Srishti Arora, Rajagopal Iyer, Sanath Surawar, Atul Acharya, Aks Tayal, Gopal Agarwal
Effort	7 person-days
Start Date	11/02/2022
Purpose	To establish a common understanding between the users' needs and the software project team of the customers' requirements addressed in the project.
Inputs	Issues faced by people when looking for roommates, or when living in a shared accommodation.
Activities	Identify the target audience, interview an unbiased sample of the target audience, collect data from surveys, write and build customer requirements.
Outputs	Use case diagram and description

Table 4: Work package details for Requirement Analysis and Documentation

Project	RoomieLah Mobile Application
Work Package	X04 - Requirement Analysis and Documentation (4 of 10)
Assigned To	**Pratyush Kumar Pandey**, Srishti Arora, Rajagopal Iyer, Sanath Surawar, Atul Acharya, Aks Tayal, Gopal Agarwal
Effort	7 person-days
Start Date	18/02/2022
Purpose	To analyze to be sure all product requirements accurately represent

	stakeholder needs and requirements
Inputs	Outputs from previous work packages (X01-X03)
Activities	Analyze the different actors, use cases and the requirements of the stakeholders to make decisions about the most suitable tech stack, functional and nonfunctional requirements, architecture and timeline of the project, etc.
Outputs	A written document of Software Requirements Specifications(SRS) and Quality Plan

Table 5: Work package details for Prototype and User Interface

Project	RoomieLah Mobile Application
Work Package	X05 - Prototype and User Interface (5 of 10)
Assigned To	**Surawar Sanath Sachin**, Arora Srishti, Tayal Aks, Agarwal Gopal, Acharya Atul, Pandey Pratyush Kumar, Rajagopal Iyer
Effort	7 person-days
Start Date	25/02/2022
Purpose	To allow the users evaluate developer proposals and try them out before implementation. It also helps understand the user specific requirements and may not have been considered by the developer during product design.
Inputs	Outputs from previous work packages (X01-X04)
Activities	1. The system is partially implemented before or during the analysis phase thereby giving the customers an opportunity to see the product early in the life cycle. 2. An initial prototype supporting only the basic functionality as desired by the customer is created/ implemented 3. Once the customer figures out the problems, the prototype is further refined to eliminate them. 4. The process continues until the user approves the prototype and finds the working model to be satisfactory.
Outputs	A prototype approved by the stakeholders and User Interface screens

Table 6: Work package details for Software Architecture and Data Modelling

Project	RoomieLah Mobile Application
Work Package	X06 - System Architecture and Data Modelling (6 of 10)
Assigned To	**Surawar Sanath Sachin**, Arora Srishti, Tayal Aks, Agarwal Gopal, Acharya Atul, Pandey Pratyush Kumar, Rajagopal Iyer
Effort	7 person-days
Start Date	04/03/21
Purpose	Effectuate a system architecture design and flow for various components within RoomieLah. Establish a workflow that the end-user would follow as they use the application for the use-case.
Inputs	Outputs from previous work packages (X01-X04)
Activities	The work package includes tasks: 1. A high-level design of the entire system architecture. 2. Defining the various components required for the system 3. Modeling of the data in terms of communication between these various components Deciding on the software system and hardware infrastructure required for the system such as the operating system built on, the database required for the storage. The designers must consider the various system design topics such as portability, maintainability for maintaining the quality of the system adhering to quality assurance.
Outputs	High Level System Architecture Diagram and intricate specification

Table 7: Work package details for Software Implementation

Project	RoomieLah Mobile Application
Work Package	X07 - Software Implementation (7 of 10)
Assigned To	**Rajagopal Iyer,** Srishti Arora, Pratyush Kumar Pandey, Sanath Surawar, Atul Acharya, Aks Tayal, Gopal Agarwal
Effort	14 person-days

Start Date	11/03/2022
Purpose	Setup initial dependencies and configuration. Code the application in the chosen programming language in accordance with the prototype, system design chosen and architecture defined in prior packages.
Inputs	Outputs from work packages X02 and X06.
Activities	This work package involves team members splitting, setting up initial dependencies and coding components of the application according to the design specifications defined in the system architecture.
Outputs	Source code

Table 8: Work package details for Unit Testing

Project	RoomieLah Mobile Application
Work Package	X08 - Unit Testing (8 of 10)
Assigned To	**Rajagopal Iyer**, Srishti Arora, Pratyush Kumar Pandey, Sanath Surawar, Atul Acharya, Aks Tayal, Gopal Agarwal
Effort	7 person-days
Start Date	25/03/2022
Purpose	Perform unit tests on each component constructed for reliability and performance before integrating with other application features.
Inputs	Output from work package X07
Activities	This work package involves: 1. Unit testing each component for functional and non-functional requirements.
Outputs	Source code and test files

Table 9: Work package details for System Integration & Quality Assurance

Project	RoomieLah Mobile Application

Work Package	X09 - System Integration & Quality Assurance (9 of 10)
Assigned To	**Atul Acharya**, Srishti Arora, Pratyush Kumar Pandey, Rajagopal Iyer, Sanath Surawar, Aks Tayal, Gopal Agarwal
Effort	7 person-days
Start Date	01/04/2022
Purpose	Integrate all the individual components into a single system and ensure data flow is correct between the internal as well as the external components. Identify and fix both syntactical and logical errors that might arise during integration. Evaluate the overall performance of the application to ensure it satisfies the quality standards set.
Inputs	Output from work package X08.
Activities	This work package includes: 1. Integrating the individual components into a single system by stimulating how a user will interact with the application. 2. Develop drivers in order to implement integration as well as additional code to perform integration testing. 3. Assignees will examine issues such as performance and integrity as well as establish metrics to evaluate the performance in various metrics and give strategies in order to improve.
Outputs	Source code, test files and test report

Table 10: Work package details for Maintenance & Version Control

Project	RoomieLah Mobile Application
Work Package	X10 - Maintenance & Version Control (10 of 10)
Assigned To	**Aks Tayal**, Srishti Arora, Pratyush Kumar Pandey, Rajagopal Iyer, Sanath Surawar, Atul Acharya, Gopal Agarwal
Effort	-
Start Date	08/04/2022
Purpose	To ensure that the application is maintained until its launch.
Inputs	Output from work package X09.

Activities	This work package includes 1. Assignees who unify and remove discrepancies between different versions of the application. 2. Issues raised by the customers are solved. 3. Submit the final application as well as the relevant documentation to the customer marking the end of the project.
Outputs	The final system software and documentation

5 Project Estimates

5.1 Code Size Estimation using Function Points

The Function Point method helps estimate the code size. The method involves calculating unadjusted function points based on these parameters:

- Number of inputs
- Number of outputs
- Number of logical files
- Number of external interfaces
- Number of inquiries

After assigning a complexity to the tasks in each of the above 5 categories, we get the Unadjusted Function Point.

This is followed by taking several other drivers like Data Communications, Transaction Rate etc. into account to get the influence multiplier which when multiplied with the Unadjusted Function Point gives the Adjusted Function Point. The function point obtained can be converted into lines of code based on research conducted over many years and with huge data.

RoomieLah supports the following functions:

1. User Registration
2. User Login
3. Chat Room
4. Viewing matched profiles for chat
5. Viewing recommended profiles
6. Edit Profile

The measure of unadjusted function points is based on five primary component elements of these functions: Number of Inputs, Number of Outputs, Number of Inquiries, Number of Logical Files, and Number of External Interface. For each element, the complexity is either Low, Medium, or High and accordingly, the weight of contribution to the unadjusted function point is assigned. The detailed evaluation of the complexity is as follows:

User Registration

The steps included under User Registration are as follows:

- Details input by the user like Name, Email ID, Course, Interest tags etc – 1 input
- Details of the user are stored in cloud Firestore database (available on Google Firebase) - 1 external interface

Hence, function points: 1 input (low complexity), 1 external interface (medium complexity)

User Login

The steps included under User Login are as follows:

- Input of details by the user for validation – 1 input
- Verification of input details against details stored in cloud Firestore database – 1 external interface

Hence, function points: 1 input (low complexity), 1 external interface (medium complexity)

Chat Room

The steps included under Chat Room are as follows:

- Chat with a potential matched roommate in a private chat and store the messages in the cloud database.
- The chat feature provides support for text input.

Hence, function points: 3 inputs(low complexity), 3 external interfaces in two users and one database (medium complexity)

Viewing Matched Profiles to Chat

The steps included under viewing matched profiles for the purpose of chatting are:

- Viewing matched users based on mutual matching of profiles – 1 output
- Gives users the option to chat with any profile they have matched with.

Hence, function points: 1 output (low complexity)

Viewing Recommended Profiles

The steps included under viewing recommended profiles are as follows:

- Processing of interest tags for a user and suggesting appropriate matches – 1 output
- Gives user the option to join or reject the recommendation of the system

Hence, function points: 1 output (high complexity)

Edit Profile

The steps included under Edit Profile are as follows:

- Allow the user to change profile information like name, interest tags, photo etc. – 1 input
- Store the changes in the cloud Firestore database – 1 external interface

Hence, function points: 1 input (low complexity), 1 external interface (medium complexity)

Based on the complexity (low / medium / high) of each function point, the unadjusted function point can be calculated as follows:

Characteristic	Low Priority		Medium Priority		High Priority	
Inputs	6	×3	0	× 4	0	× 6
Outputs	1	× 4	0	× 5	1	× 7
Inquiries	0	× 3	0	× 4	0	× 6
Logical Files	0	× 7	0	× 10	0	× 15
Interfaces	0	× 5	6	× 7	0	× 10

Unadjusted FP	22		42		7	
Total = Low + Medium + High	71					

Adjusted Function Points

Influence Factor	Score	Details
Data Communications	4	Application involves significant data communication between user and cloud Firestore database.
Performance	2	Response time or throughput is moderately important for the application to prevent long waiting times for the user.
Heavily used	1	The application is expected to have little load
Distributed Functions	0	The mobile application does not involve distributed processing
Transaction rate	0	Peak transaction rates are not expected
On-line data entry	3	All the data transactions are online and need to be updated continuously on the cloud database
End-user efficiency	0	End user efficiency is not a major consideration

On-line data update	0	Online update of internal files is not included
Complex processing	4	Complex processing at certain steps like providing new, personalized recommendations to users and allowing them to chat are vital.
Reusability	1	Since the application caters to a specific USP of efficient matching of roommates, reusability in different components is not targeted.
Installation Ease	0	No special considerations for ease of installation have been made
Operational Ease	1	No special consideration given
Multiple sites	0	Requirements do not include considering availability on multiple sites
Facilitate change	4	RoomieLah should be designed to facilitate change since its pre-market and entering market will cause major changes in UI and other features.
Total score	20	
Influence Multiplier = Total score × 0.01 + 0.65 = 20 × 0.01 + 0.65 = 0.85 **Adjusted FP = Unadjusted FP × Influence Multiplier = 71 × 0.85 = 60.35**		

The legend used for scoring is as follows:

Scoring (0 – 5)
0 = No influence
1 = Insignificant influence
2 = Moderate influence
3 = Average influence
4 = Significant influence
5 = Strong influence

5.1.3 Lines of Code

According to Capers Jones statistics, each Function Point requires 47.41 lines of code if the application is implemented using Dart (the Flutter framework used for the development of RoomieLah uses Dart programming language).

Therefore, we have: Lines of Code = 60.35 FP × 47.41 LOC/FP = 2861 LOC

5.2 Efforts, Duration and Team Size Estimation

To estimate the effort and duration required for the project, we use function points as the basis to calculate Effort, Duration, Team size and finally the schedule. The estimates are expanded to account for project management and extra contingency time to obtain the total average effort estimates. From these averages, the duration of each work package in working days is estimated based on the following calculations.

- Working days include 5 days in a week.
- Effort = Size / Production Rate = (2861 LOC) / (39 LOC/PD)[1] = 73.36 PD
- Duration = 3 × (Effort) $^{1/3}$ = 3 × (73.36) $^{1/3}$ = 12.55 Days
- Initial schedule = 12.55 Days / 5 days a week = 2.51 Weeks
- Team size = 73.36 PD / 12.55 D = 5.84 P = 6 Persons
- Working hours include 8 hours in a working day.
- Total person-hours (PH) = 73.36 PD × 8 hours = 586.88 PH

5.2.1 Distribution of Effort

1990's Industry Data	Work Package	Distribution	Estimates
Preliminary Design 18 %	Project Plan	5%	29.34
	Requirement Specification	9%	52.82
Detailed Design 25 %	User Interface	7%	41.08
	Technical Architecture	15%	88.03
	Data Modeling	9%	52.81
Implementation & Unit Testing 26 %	Implementation & Unit testing	21%	123.24
	Online Documentation	5%	29.34
Integration & Test 31 %	Integration & Quality Assurance	29%	170.20
	Extrapolated total effort		586.86
	2% for project management		11.73
	3% for contingency		17.61
	Total effort		616.2

These duration estimates assume that each team member works an equal amount on any given work package.

[1] Lines of code per Person Day statistics based on Industrial Benchmarks, 1997: 31 LOC/PD for United States; 62 LOC/PD for Canada

5.3 Cost Estimates

Hardware:

Developer workstations:

Item	Cost Estimate
1 - iPhone SE (to test RoomieLah on iOS) 1 - Google Pixel 6 (to test RoomieLah on Android)	$ 649 $ 500
Single Core	
1 GHz	
4 GB Ram	
Total	$ 1149

Software:

GNU, Apache, or other free license-based software:

Apache Web Server	$0.00
Perl	$0.00

Software License Provided by Third Party:

Microsoft Office 2000	$0.00
Microsoft Project 2000	$0.00
Products ESTIMATE Professional	$0.00

Other Resources:

Resource	**Estimated Cost**
7 Employees with 568.88 working hours with $18.00/hour	$10,239.84

Stationery:

Stationery Items	**Cost**
Paper, notebooks, photocopying and other miscellaneous costs	$50

The customer will supply the required hardware and software necessary to run RoomieLah. StrawHats is not responsible in any way for supplying said systems. StrawHats' hardware and software responsibilities relate only to our own development needs to accomplish the project we have been asked to complete, and which has been described in the introduction section of this document. StrawHats will also demonstrate the completed product.

6 Product Checklist

The plan is that the items listed below will be delivered on the stated deadlines.

Project Deliverable	Estimated Deadline
Project Proposal	Feb 9th, 2022
Use Case Model	Feb 9th, 2022
System Requirement Specifications	Feb 23rd, 2022
Quality Plan	Feb 23rd, 2022
Project Plan	Mar 16th, 2022
Risk Management Plan	Mar 16th, 2022
Prototype Code, Documentation and Video	Mar 16th, 2022
Prototype Visualization	Mar 16th, 2022
Design Report on Software Maintainability	Mar 30th, 2022
Configuration Management Plan	Mar 30th, 2022
Change Management Plan	Mar 30th, 2022
Release Plan	Mar 30th, 2022
Test Plan	Apr 1st, 2022
Test Cases and Test Requirement Coverage Report	Apr 1st, 2022
CMMI Level 2 Definition	Apr 1st, 2022
Documentation of Code	Apr 1st, 2022

7 Best Practice Checklist

Practice
Document every action taken by the team. All documentation must be in a standardized format. - Consistent font, font color and font sizes for headers and body. - Clear and easy to understand without unnecessary usage of technical jargon. - Standardized line spacing must be used for all documentation.
The requirement documentation must contain a complete functional specification and should follow the following best practices. - The requirements must be consistent, precise and unambiguous. - Ensure it satisfies the needs and requirements of the stakeholders. - Both functional and non-functional requirements must be clearly stated. - Constraints must be identified and also verify the viability of implementing the requirements while adhering to the constraints.
It is very important to ensure that the program does not become too complex. This is because greater complexity usually leads to greater bugs. Strive to: • Minimize interfaces between modules, procedures and data. • Minimize interfaces between people, otherwise exponential communication cost • Avoid fancy product functions, design as long as the functionality meets the

customer requirements

Require Visibility. We must see what we build otherwise we can measure the progress and take management action. This includes: the manager must have good communication with his or her employees; require developers to make code available for review; review design for appropriateness.

Plan for continuous change. We must:

- All manuals’ designs, test, source code should have revision numbers and dates revision history comments, change marks to indicate the changes
- New revisions should be approved before being made and checked for quality and compliance after being made
- Use a configuration management system and make processes
- Required maintenance

We must be careful to obtain accurate estimates for: time, effort, overhead, meeting time, and especially effort on integration, testing, documentation, and maintenance.

Code reviews are a much more efficient method to find software defects. Plan and manage code reviews between team members

The following points must be considered when conducting testing.

- The test cases must be as exhaustive as possible in covering all the scenarios.
- Every requirement must be tested by a test case.
- Test cases must cover error handling scenarios.
- Boundary conditions must be correctly identified and test cases for such scenarios must be crafted.

8 Risk Management

Besides the general risk management, the following risks have been identified for the RoomieLah project:

More changes to requirements than anticipated

Impact Severity: High

Probability: 25%

Impacts: Depending on the stage at which changes occur, could range from needing to update the requirements documentation to needing to do a complete redesign.

Risk Reduction: Be rigorous in eliciting requirements. Make stakeholders aware of potential repercussions of requirement changes.

Software components that should be reused contain defects that limit functionality

Impact Severity: High

Probability: 30%

Impact: Will lead to malfunctioning of the software.

Risk Reduction: Carry out thorough software testing before integrating any such component especially from an open-source code.

Specification Delays

Impact Severity: High

Probability: 15%

Impacts: Delay in finalizing the specification will push the schedule for all following stages of the project.

Risk Reduction: Monitor progress of specification carefully.

System size underestimated

Impact Severity: Moderate

Probability: 30%

Impacts: More work will need to be spent on design and coding; could negatively impact schedule. Risk Reduction: Update estimates often as the project progresses.

Firebase API malfunction

Impact Severity: High

Probability: 5%

Impact: The application will not be able to retrieve or post any data rendering the application useless as every functionality needs a working connection with the database.

Risk Reduction: Double check the URLs provided for the APIs of data retrieval and data posting.

The database used in the system cannot process as many transactions per second as expected

Impact Severity: High

Probability: 25%

Impact: Depending on the peak user traffic, it could lead to delayed or no response for users concurrently using the application.

Risk Reduction: Perform intensive load testing and if the database doesn't support, alternative high-performance databases should be explored and used.

Staff leaving before project complete

Impact Severity: Extreme

Probability: 5%

Impacts: There would be more work for remaining employees, and any specialized skills or knowledge would be lost.

Risk Reduction: Offer benefits and incentives to staff.

Problems coordinating within group

Impact Severity: Moderate

Probability: 40%

Impacts: Members may be unaware of what is expected of them; managers may not be able to measure progress; portions of projects not completed.

Risk Reduction: Follow communication plans as documented in section 2.3

Stakeholders cancel project Impact Severity: Super-Extreme! Probability: 1%

Impacts: All work will have been wasted.

Risk Reduction: Keep in close contact with stakeholders. Ensure that they have some market research indicating a demand for this product.

9 Quality Assurance

The project will achieve quality assurance by following an expected standard of the application. The specific procedures and details have been provided in the Quality Plan.

Specific test procedures and details shall be provided in the Module/System Test Plan.

In addition, the team will make constant use of the following testing methodologies while developing the application :

1. **Unit Testing:**
   Every module of code (function/class) will be tested as a standalone unit. Typically, a black box approach will be used. However, in case of high cyclomatic complexity i.e., complicated code like the Algorithm backend service, white box testing will be performed to achieve higher level of confidence. Before merging any code during version control, coverage of the unit tests will be calculated using open-source libraries to ensure the code is reliable. A minimum coverage of 75% will be needed to approve any pull request.

2. **Integration Testing:**
   The standard bottom-up testing approach will be followed. After the individual components of the application have been tested, we move a level up and test the following :
   1. Standalone services made up of the individual modules that were unit tested. For e.g., Backend API or microservice APIs.
   2. Intra application communication i.e., communication between internal services
   3. Inter application communication i.e., communication with external services like external API, database, or cloud-based services.
   In standard unit testing, mocks are used to create standalone tests, so we are unable to test any communication or API calls.

The team will also **stress/load test** the whole application and metrics like number of internal server errors (500/502) or MTTF will be analyzed, and fixes will be released to mitigate errors and improve user experience.

These methodologies will also be used to test two important aspects of RoomieLah:

- **System Function:** All the individual components when integrated will function properly.
- **Recommender Function:** The results of the recommender algorithm will also be analyzed, and the algorithm will be iteratively fine-tuned.

10 Monitoring & Control

Many procedures are required in order to be able to successfully monitor the progress of a software project.
Some of the most important are:

- **Quantitative measurement of resource consumption:** Estimates of RoomieLah's resource requirements, primarily in terms of human resources, can provide a quantitative measurement of project progress when compared to progress in terms of project milestones. The percentage estimates of each milestone's resource requirements provided in this document allow for easy progress tracking.

- **Project Risk Assessment and Analysis :** Early identification of major risks to the project allows for placement of preventative measures before problems can develop. Major risks have been identified in the Risk Management section of this document, along with the measures being taken to avoid them. The risk identification process will be carried out by actively considering potential threats from various intrinsic and extrinsic factors capable of adversely affecting the project. The deliverables, constraints, schedule, budget, technologies used and project dependencies will be carefully scrutinized to mitigate risks, and the identified risks will be classified into various categories based on their area of origin and impact. Each risk will also be analyzed to identify severity and probability of occurrence (classified as low, medium, and high). Quantitative assessment will also be carried out to provide an objective numerical rating, which will be clearly documented in the risk management plan.

- **Regular reviews of project progress:** Throughout the duration of the RoomieLah project, the development team shall meet weekly to review the progress of all project tasks, including management, planning, analysis, development, and testing.

- **Timeline Planning and task decomposition:** This document outlines an estimated timeline for the project. A reasonably accurate timeline can be assembled by hierarchically decomposing tasks into measurable subcomponents and estimating requirements for each. At the same time, this decomposition can assist in task assignment and balancing. Throughout the implementation phase, these subcomponents can allow for fine-grained measurement of progress. Project subcomponents and timeline estimates are included in the Estimates and Work Breakdown Structure sections of this document.

- **Risk Management Process:** SQ personnel are responsible to ensure that all risk management procedures mapped out in the ideation phase are implemented satisfactorily. This includes drafting a comprehensive risk management plan, enforcing its adoption, and relaying relevant information to other team members as and when necessary. All potential risks and scenarios must be evaluated, including the project timeline running behind schedule, the budget exceeding the allocated amount, the lack of resources to implement the project features etc, and the team should prepare for all contingencies in advance.

- **Risk Response Planning:** The identified risks will be categorized based on level of criticality into three categories- red, yellow, and green. The category will be decided based on a combination of the probability of that risk affecting the project, and the impact the risk would make if it occurred. A variety of actions can be taken for each risk, namely avoid, mitigate, accept, and transfer, and each risk will be assigned an action based on its criticality. The risks, prioritized according to their criticality, will be tracked, monitored, and reported on by the SQ personnel throughout the development lifecycle, and the management will be notified in case of any changes to the risk assessment profiles.