A logo with black text

Description automatically generated

NIT6150 Advanced Project

**System Analysis and Design Report**

**Healthcare Chatbot System**

**Team Leader:** Jitendra Shrestha (s8104215)

**Team Member:** Pranish Acharya (s8100698)

**Client:** Holroyd Private Hospital

**Supervisor:** Fakhra Jabeen

VU Sydney

**Coordinator**: Dr. Alex Wenjie Ye

Table of Contents

[Introduction 3](#_Toc175172942)

[Objective 3](#_Toc175172943)

[Technology Stack 3](#_Toc175172944)

[System Development Approach 5](#_Toc175172945)

[Functional Decomposition Diagram (FDD) 6](#_Toc175172946)

[System Requirement Specifications (SRS) 6](#_Toc175172947)

[Functional Requirements 7](#_Toc175172948)

[Non-Functional Requirements 7](#_Toc175172949)

[1. Look and Feel Requirements 7](#_Toc175172950)

[2. Usability and Humanity Requirements 7](#_Toc175172951)

[3. Performance Requirements 7](#_Toc175172952)

[4. Operational Requirements 7](#_Toc175172953)

[5. Maintainability and Support Requirements 7](#_Toc175172954)

[6. Security Requirements 8](#_Toc175172955)

[7. Cultural and Political Requirements 8](#_Toc175172956)

[8. Legal Requirements 8](#_Toc175172957)

[Usability Requirements 8](#_Toc175172958)

[SRS Table for User Management System (S1) 8](#_Toc175172959)

[Use Case Diagram for UMS 10](#_Toc175172960)

[ER Diagram of UMS 11](#_Toc175172961)

[WireFrame of UMS Register Page 12](#_Toc175172962)

[WireFrame of UMS Login page 13](#_Toc175172963)

[SRS Table of Personal Health Management System (S2) 14](#_Toc175172964)

[Use Case Diagram of PHMS 15](#_Toc175172965)

[ER Diagram of PHMS 16](#_Toc175172966)

[Wireframe of PHMS 17](#_Toc175172967)

[SRS Table of Chatbot System (S3) 17](#_Toc175172968)

[Chatbot Architecture 18](#_Toc175172969)

[Flow Chart Diagram for Chatbot 19](#_Toc175172970)

[Wireframe for Chatbot Interface 20](#_Toc175172971)

[Wireframe for Chatbot Homepage 21](#_Toc175172972)

[Wireframe for Chatbot About Us page 22](#_Toc175172973)

[**Project Management** 23](#_Toc175172974)

[Project Cost Estimate 23](#_Toc175172975)

[Project Detail Costs 23](#_Toc175172976)

[Project Schedule 25](#_Toc175172977)

[Resources Used 28](#_Toc175172978)

[Issues in Project 28](#_Toc175172979)

[Project Risks 28](#_Toc175172980)

[Cutover Issues 28](#_Toc175172981)

[Quality Assurance and Testing Issues 28](#_Toc175172982)

[Potential Involvements of human factors 29](#_Toc175172983)

[Team Dynamics: 29](#_Toc175172984)

[User Interactions: 29](#_Toc175172985)

[User Feedback and Adaptation: 29](#_Toc175172986)

[**Potential Privacy Issues** 30](#_Toc175172987)

[**Professional ethics Issues** 30](#_Toc175172988)

[Data Privacy and Confidentiality: 30](#_Toc175172989)

[References 31](#_Toc175172990)

**List of Figures**

[Figure 1: FDD of the System 11](#_Toc175173682)

[Figure 2: Use case Diagram of UMS 15](#_Toc175173683)

[Figure 3: Detailed ER Diagram of UMS 16](#_Toc175173684)

[Figure 4: Wireframe of User Registration Form 17](#_Toc175173685)

[Figure 5 : Wireframe of User Login System 18](#_Toc175173686)

[Figure 6: Use Case Diagram of PHMS 20](#_Toc175173687)

[Figure 7: Detailed ER Diagram of PHMS 21](#_Toc175173688)

[Figure 8: Wireframe of PHMS 22](#_Toc175173689)

[Figure 9: Chatbot Architecture 23](#_Toc175173690)

[Figure 10: Flow chart Diagram of Chatbot System 24](#_Toc175173691)

[Figure 11: Wireframe Diagram of Chatbot Interface 25](#_Toc175173692)

[Figure 12: Wireframe of chatbot Homepage 26](#_Toc175173693)

[Figure 13: Wireframe of chatbot About us page 27](#_Toc175173694)

[Figure 14: Work Break Down for Project Schedule 30](#_Toc175173695)

[Figure 15: Gantt chart for Work Break Down showing the date 32](#_Toc175173696)

[Figure 16: Timeline of project scheduleResources Used 32](#_Toc175173697)

[Figure 17: Resource Used for Project 33](#_Toc175173698)

**List of Tables**

[Table 1: SRS for S1 11](#_Toc175173926)

[Table 2: SRS for S2 17](#_Toc175173927)

[Table 3: SRS for S3 20](#_Toc175173928)

[Table 4: Miscellanous Cost Breakdown 27](#_Toc175173929)

# Introduction

The way people interact with one other has been completely transformed by internet-connected devices. As a result of technological advancements, "people are demanding more intelligent self-service options and experiences answers within seconds, not minutes." Machine learning (ML) and artificial intelligence (AI) are being adopted by several sectors to improve customer service. Chatbots are employed in various fields these days, including e-commerce to answer product-related queries from customers and banking systems for customer support. A chatbot is a computer software that receives human input in natural language and responds to it with a perceptive and pertinent response before sending it back to the user. Hospitals are now permitting patients to engage with technology, such chatbots, to learn more about disease, doctor or hospital speciality. This technology will assist in offering the user support around-the-clock. (Ayanouz, S., Abdelhakim, B.A. and Benhmed, M.) also explains that Physicians, nurses, patients, and their families could benefit from a chatbot. Chatbots may assist medical personnel in several scenarios, such as better patient information organization, medication administration, emergency assistance, first aid, and providing a solution for minor medical problems.

# Objective

The objectives of the system are as follows:

* To provide response on health queries
* To diagnose disease based on symptoms provided based on yes/no statement
* To provide way to maintain patient health record

# Technology Stack

Various tools and techniques were used for the development of web applications in different programming languages such as java, python etc. Some of the tools and techniques that I used for my project are described here.

**Python Programming Language**

A wide variety of programming languages are available for use in the creation of web applications. I have decided to use Python as my programming language for the project development. It is simple to read and write using simple programming syntax. Python programming was created in 1991 by Guido Van Rossum at the National Research Institute. Since its inception, this programming language has grown in popularity among developers on a daily basis. One of my other favorite things about Python for my project is how many web application development frameworks it offers, such Django, Flask, and others. Python programming is particularly well-known for its ability to support artificial intelligence and machine learning, making it suitable for projects involving AI like chatbots. Since I created the chatbot with the RASA framework, it has also used python in backend. (Kumar, 2018).

**Django Framework**

The open source web application development platform Django is used. For projects that are mid-range scalable, Django is excellent and helpful. The simplicity, flexibility, dependability, and scalability of Django helped my project get off to a good start. Django offers a wealth of built-in features, including flexible code rearranging. Thus, it was the ideal option for my project's online application. Django makes it simple for developers to handle databases by supporting a variety of database types. Additionally, Django extends passwords using the SHA256 hash of the NIST-recommended PBKDF2 technique. This password encryption technique is reliable and secure. A large amount of processing time will be needed to crack this encryption technique. We will only keep the necessary personal data, and it will be kept in a secure manner with restricted access. Different vulnerabilities attacks may be used to make an application more secure. (Django, 2020)

**Rasa Framework**

The open-source Rasa framework is used to create chatbots using artificial intelligence. Rasa is a natural language processing framework that can comprehend spoken language and provide users with answers to their queries. One advantage of utilizing the rasa framework is that it is built on Python programming. Rasa is an open-source NLP toolkit based on Python that has a pre-built architecture that we may modify to our specifications. One suggestion is to start filling up this framework with words to establish a dataset for training your bot. Next, construct the NLU model, create dialogue patterns, and develop a skeleton for the dialogues. Rasa Stack is meant for brief messages and can do tasks. (Rasa Technologies, 2019)

**Visual Studio Code Editor**

Better Community support is compatible with all operating systems, and it has excellent code highlighting and a clear indentation level for Python programming, which is better for my project. Visual Studio Code combines sophisticated development tools like IntelliSense code completion and debugging with the user-friendliness of a source code editor. (Visual Studio Documentation , 2020)

**SQlite**

An excellent, free, and visual tool for building, developing, and modifying SQLite database files is called DB Browser for SQLite (DB4S). Data manipulation and maintenance are made simple by Django's inclusion of SQlite as an integrated database.

# System Development Approach

It can therefore be argued that a project succeeds when it is well managed. Development team or the manager of the development must well choose a software development methodology which is best suited to the project. Every methodology has its own method, advantage and disadvantage and exist for a purpose.

**Scrum** is one of the examples of agile software development methodology. Agile is the way that most software is developed in the 21st century. Repetition of development and testing at some organized intervals within the complete life cycle of the particular project is the key consideration of the agile software development method. The above method of the development is a parallel process with testing different from the waterfall method where development and testing are sequential. (Guru99, 2020)

Thus, there are several things to take into account while choosing a software development process, the most crucial of which is how to handle changes in end-user requirements. The primary advantage of Agile approaches is that modifications and communication techniques are followed in a clear-cut and efficient way. Frequent meetings with the supervisor are feasible now that the Agile system is in place.

For our project, the Agile Scrum framework would be the most suitable methodology. During the project's operating phase, requirements that were not found during the beginning phase can be resolved. Scrum is particularly adaptable to requirements changes. It can be used for projects involving small teams, like ours.

# Functional Decomposition Diagram (FDD)

**A diagram of health care system

Description automatically generated**   
 Figure 1: FDD of the System

# System Requirement Specifications (SRS)

This system requirement describes the software system that is being developed for the Health Care Chatbot System. Drawing on the department's and customers' transactions, it delineates the main functional, non-functional, and usability requirements.

## Functional Requirements

These specifications define the functions the system must perform, such as how it must process outputs, react to inputs, and act in a specific manner under certain conditions.

## Non-Functional Requirements

These prerequisites are the non-functional services that the system makes use of to support enhanced system functionality.

Here’s a brief explanation of each of the non-functional requirement’s categories:

### 1. Look and Feel Requirements

The system should have a unified, eye-catching interface that complements the company's logo. This guarantees a positive user experience and encourages system familiarity throughout.

### 2. Usability and Humanity Requirements

The navigation should be simple, and the directions should be straightforward for users to understand. It should be able to support several languages as necessary and be usable by a wide range of users, including those with impairments.

### 3. Performance Requirements

The system needs to react rapidly, have short load times, and function flawlessly. It must manage large numbers of concurrent users without experiencing performance deterioration.

### 4. Operational Requirements

With little downtime, the system should be dependable and accessible around-the-clock. It should be compatible with a range of platforms and devices, allowing consumers to access it easily.

### 5. Maintainability and Support Requirements

The system should have modular code, comprehensive documentation, and be simple to maintain. Ensuring that regular updates and bug fixes are simple will help minimize user interruption.

### 6. Security Requirements

User data must be safeguarded by the system using robust authentication, encryption, and frequent security audits. It should be safe from online attacks and adhere to all applicable data protection regulations.

### 7. **Cultural and Political Requirements**

Avoiding potentially harmful or politically charged information, the system should be impartial and sensitive to cultural differences. It needs to be flexible enough to accommodate varying regional laws and cultural norms.

### 8. Legal Requirements

The system needs to abide by all applicable laws and rules, including those pertaining to accessibility and data security. This entails getting the required permits and abiding by the laws that are particular to your sector.

## Usability Requirements

These requirements guarantee that users will find the produced system easy to use and that it will fulfil all necessary objectives while operating flawlessly.

**Legend**

UMS: User Management System

PHMS: Personal Health Management System

CS: Chatbot System

F: Functional Requirement

NF: Non-functional Requirement

UR: Usability Requirement

## SRS Table for User Management System (S1)

Table 1: SRS for S1

|  |  |  |
| --- | --- | --- |
| **Requirement Code** | **Requirement Description** | **MoSCoW** |
| UMS-F-01 | The system should allow normal user to sign up and login to the system. | Must have |
| UMS-F-02 | The system should allow user to reset the password. | Must have |
| UMS-UR-03 | System should suggest adding numeric  and special characters in password. | Must have |
| UMS-UR-04 | System should give the option of making the credentials visible while login. | Should have |
| UMS-NF-05 | System should hide the login credentials of the user. | Must Have |
| UMS-F-06 | The system should allow user to chat with CHATBOT system without logged in. | Must have |
| UMS-F-07 | System should allow doctor to login to the system. | Must Have |
| UMS-F-08 | System should allow admin to Add Doctor. | Must Have |
| UMS-F-09 | System should allow admin to delete doctor. | Must have |

### Use Case Diagram for UMS

A diagram of a network

Description automatically generated

Figure 2: Use case Diagram of UMS

### ER Diagram of UMS

A diagram of a user

Description automatically generated  
 Figure 3: Detailed ER Diagram of UMS

### WireFrame of UMS Register Page

A screenshot of a computer

Description automatically generated

Figure 4: Wireframe of User Registration Form

### WireFrame of UMS Login page

Graphical user interface

Description automatically generated

Figure 5 : Wireframe of User Login System

## SRS Table of Personal Health Management System (S2)

Table 2: SRS for S2

|  |  |  |
| --- | --- | --- |
| **Requirement Code** | **Requirement Description** | **MoSCoW** |
| PHMS-F-01 | Allow user (patient) to book appointment. | Must have |
| PHMS-F-02 | Allow user to delete appointment. | Must have |
| PHMS-F-03 | Allow Doctor to view appointment of users. | Must have |
| PHMS-F-04 | Allow Doctor to add prescription in the booked appointment. | Must have |
| PHMS-F-05 | Allow user to see doctor’s prescription report. | Must have |
| PHMS-F-06 | Allow user to know about follow up date. | Should have |
| PHMS-NF-07 | Allow user to download the prescription report. | Could have |
| PHMS-UR-08 | Allow user to Know about doctors Speciality. | Could have |
| PHMS-UR-09 | Allow user to delete health records from the system. | Could have |

### Use Case Diagram of PHMS

A diagram of a diagram

Description automatically generated

Figure 6: Use Case Diagram of PHMS

### ER Diagram of PHMS

A diagram of a doctor

Description automatically generated

Figure 7: Detailed ER Diagram of PHMS

### Wireframe of PHMS

Table

Description automatically generated  
Figure 8: Wireframe of PHMS

## SRS Table of Chatbot System (S3)

Table 3: SRS for S3

|  |  |  |
| --- | --- | --- |
| **Requirement Code** | **Requirement Description** | **MoSCoW** |
| CS-F-01 | Should allow user to interact with chatbot interface without registration. | Must have |
| CS-F-02 | Should greet user. | Must have |
| CS-F-03 | Allow user to Ask question regarding health. | Must have |
| CS-UR-04 | User could use voice chat to ask questions. | Could have |

### Chatbot Architecture

A diagram of a machine learning process

Description automatically generated

Figure 9: Chatbot Architecture

### Flow Chart Diagram for Chatbot

A diagram of a flowchart

Description automatically generated

Figure 10: Flow chart Diagram of Chatbot System

### Wireframe for Chatbot Interface

Graphical user interface, text, application

Description automatically generated

Figure 11: Wireframe Diagram of Chatbot Interface

### Wireframe for Chatbot Homepage

Diagram

Description automatically generatedFigure 12: Wireframe of chatbot Homepage

### Wireframe for Chatbot About Us page

Graphical user interface, text

Description automatically generatedFigure 13: Wireframe of chatbot About us page

# **Project Management**

## Project Cost Estimate

A project cost estimate accounts for personnel, software, and contingencies for unforeseen charges in addition to any other costs necessary to finish the project. It supports efficient budgeting and planning to guarantee the project stays within allocated funds.

## Project Detail Costs

The project's below-average cost is intended to provide a realistic assessment of the undertaking, treating it as though it were an endeavour in professional growth. The breakdown shows the time, effort, and resources that would normally be needed if this project were sponsored or managed by university.

1. **Project Planning and Management Tools**:

* Project Management: Jira, Trello
* Communication: Microsoft Teams
* Version Control: Bitbucket
* Resources: Scrum Master @ $90/hr
* **Budget**: $20,600

1. **Development Costs**

**Budget:** $8,960

**Frontend Development**

* + Technologies: HTML, CSS, JavaScript
  + Tools: Webpack, Babel, npm
  + Resources: Frontend Developer @ $60/hr

**Backend Development**

* + Technologies: Django will handle backend, API Creation
  + Tools: Django REST Framework
  + Resources: Django Developer @ $60/hr

**Chatbot Development**

* + Technologies: Rasa Framework, NLP
  + Tools: RASA SDK, RASA CLI
  + NLP: SpacCY, TensorFlow
  + Resources: Python Developer @ $80/hr

**Database**

* + Technologies: Django ORM

1. **Design and User Experience (UX)**

**UI/UX Design**

* Wireframes: Balsamiq
* Resource: UI/UX Developer @ $50/h
* **Budget**: $10,000

1. **Testing and Quality Assurance**

**Functional, Performance, and Security Testing**

* CI/CD: Jenkins
* Functional Testing: Selenium
* Performance Testing: Apache
* Resource: Quality Control Team @ $40/hr, DevOps Engineer @ $60/hr
* **Budget**: $6,000

1. **Deployment and Hosting**:

**Tools and Technologies**

* Hosting: AWS
* Deployment: Docker
* Budget: $15,000

1. **Maintenance and Support**: ($30,000)
2. **Contingency** - ($20,000)

As the cost for other purposes which will be used while developing the project from beginning to end phase are given below.

Table 4: Miscellanous Cost Breakdown

|  |  |  |  |
| --- | --- | --- | --- |
| **S.N.** | **Name** | **Used For** | **Cost** |
| 1 | Microsoft Teams | Communication | $20,000 |
| 2 | Bitbucket | For Source | $20,000 |
| 3 | Jenkins | For Pipeline, DevOps | $15, 000 |
| 4 | Laptops | For development and testing purpose | $20,000($2,000 pp) |
| 5 | Miscellaneous |  | $20,000 |
|  |  | **Total** | **$95,000** |

**Total Budget**: $200,000(Approx.)

## Project Schedule

A project schedule is a comprehensive timetable that outlines the activities, deadlines, and dependencies needed to finish a project on time. It facilitates resource management, progress monitoring, and on-time project delivery.

The following figure shows the project schedule with the duration, completion of the project and cost related to only development phase.

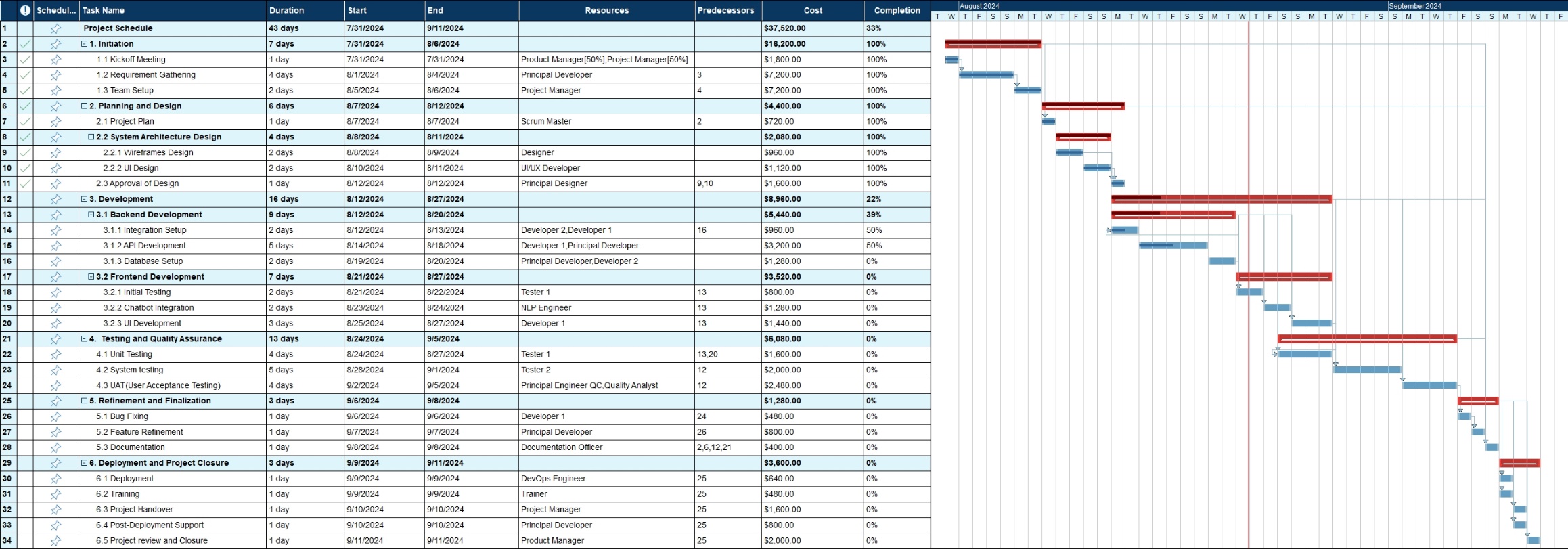


Figure 14: Work Break Down for Project Schedule

1. **Initiation**: Specify the goals, parameters, and deliverables of the project. Determine the important parties and compile the healthcare chatbot's first specifications. To describe the project's objectives and vision, draft a project charter.
2. **Planning and Design**: Create a thorough project plan with deadlines, goals, and resource allocation. Wireframes and prototypes should be made to design the chatbot's user interface and experience.
3. **Development**: Construct the chatbot's front end, back end, and components. Django is used to construct the backend while HTML, CSS, and JavaScript are used to implement the user interface. Set up the database using Django ORM and integrate the Rasa Framework for the chatbot.
4. **Testing and Quality Assurance**: To make sure the chatbot functions properly and satisfies quality requirements, do functional, performance, and security testing. Find and address any defects or problems found during testing.
5. **Refinement and Finalization**: Examine and improve the chatbot in light of user input and testing results. Complete the functionality and design to make sure the application satisfies all project criteria and is prepared for deployment.
6. **Deployment and Project Closure**: Install and run the chatbot program in a live setting. Make sure users can access and the application is hosted correctly. Finalize project documentation, carry out a project review, and get input from relevant parties. Ascertain that all deliverables are fulfilled and offer any training or resources that may be required. Archive the project's documentation, then declare it closed.

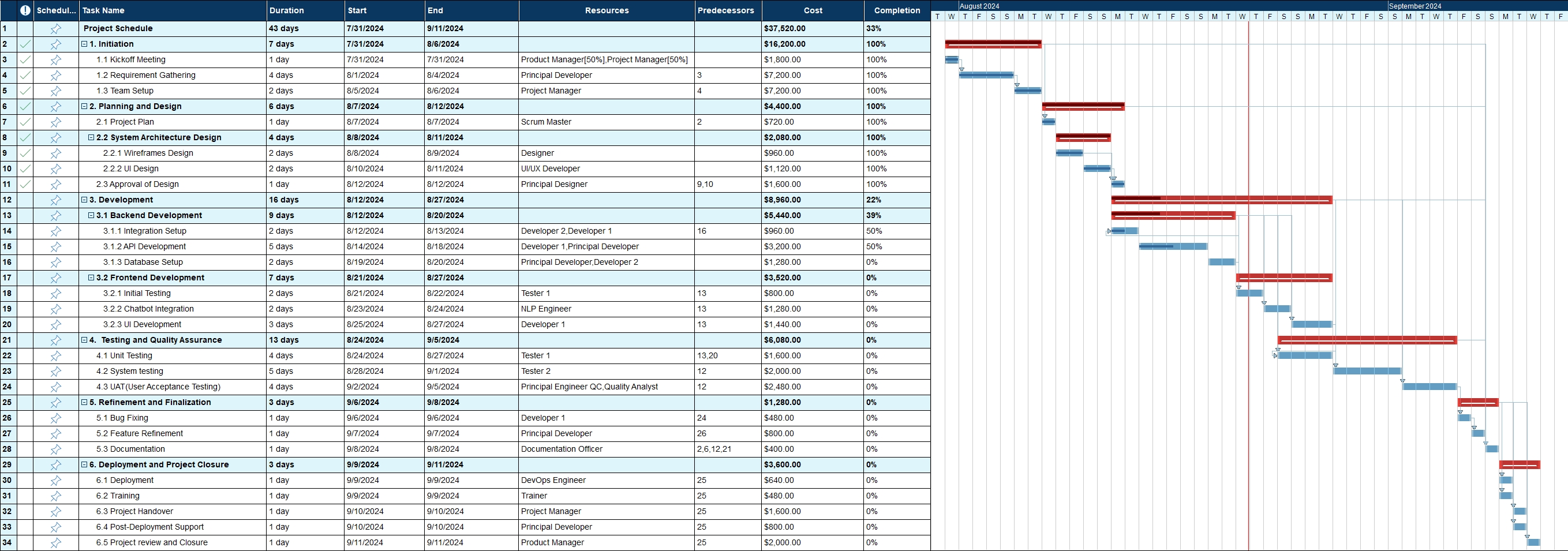


Figure 15: Gantt chart for Work Break Down showing the date

The below figure shows the overall timeline of the project schedule

A computer screen shot of a diagram

Description automatically generated

Figure 16: Timeline of project scheduleResources Used

A screenshot of a computer

Description automatically generated

Figure 17: Resource Used for Project

## Issues in Project

Project Risks**:** Overcoming unanticipated technological obstacles and integration problems is essential to the healthcare chatbot project's success. Risks might include unforeseen restrictions in the tools and frameworks employed, or incompatibilities with emerging technology. It is crucial to carry out in-depth risk assessments early in the project and to put strong testing procedures in place as it develops to mitigate these risks. This entails establishing trial periods to spot any problems and making sure technological fixes are thoroughly tested and documented.

Cutover Issues: Cutting edge problems like data transfer mistakes or service outages are common during the development to production phase. These problems may interfere with the chatbot's availability and operation. Create a thorough cutover strategy that outlines data transfer procedures, backup plans, and a failsafe rollback plan to reduce cutover risks. To further ensure a seamless transition and reduce interruptions, thorough pre-deployment testing as well as user training and support are recommended.

Quality Assurance and Testing Issues**:** Undiscovered defects or incomplete testing might have an impact on the chatbot's dependability and functionality. A less-than-ideal user experience may be caused by problems like insufficient test coverage or overlooked bugs. Create a thorough testing plan that incorporates security, performance, and functional testing to allay these worries. Use continuous integration techniques to make sure problems are found and fixed as soon as possible throughout the development process.

## Potential Involvements of human factors

Project success, including the development of healthcare chatbots, is greatly influenced by human factors. Here are the some of the human factors explained:

Team Dynamics: Controlling team dynamics is essential to the healthcare chatbot project's success. Variations in communication styles, working methods, and experience levels can impact team cohesiveness and productivity. Promoting a collaborative atmosphere through frequent team meetings and the use of efficient communication technologies like Microsoft Teams are crucial to overcome these problems. To keep team morale up and conflicts resolved in a constructive manner, it may also be helpful to promote open communication and offer conflict resolution training. This will assist to keep the team united and motivated.

User Interactions: The degree to which the chatbot satisfies user requirements and expectations determines its efficacy. User discontent and low adoption might result from misunderstandings or unfulfilled demands. It's critical to build the chatbot with a user-centric perspective to handle these interactions, which entails performing user research and collecting feedback across the development process. A smoother deployment and improved user experience may be achieved by offering thorough training and support materials to users, who will need them to handle any problems they may have adjusting to the new system.

User Feedback and Adaptation: The efficacy and relevancy of the chatbot depend heavily on ongoing user input and change. User input might call for system updates or modifications, which would affect project budgets and schedules. It is possible to incorporate user feedback and make essential adjustments with flexibility when using agile techniques and an iterative development strategy. This guarantees that the chatbot adapts to user demands and maintains its effectiveness over time.

# **Potential Privacy Issues**

Since personal health information is sensitive, controlling privacy issues is essential in healthcare chatbot projects. Unauthorized access to data, data breaches, regulatory noncompliance, incorrect data retention, insufficient user permission, and insufficient data anonymization are some major hazards. To solve them, put strong access restrictions and encryption in place to stop unwanted access, carry out frequent security assessments to find weaknesses, and make sure that laws like HIPAA and GDPR are being followed by conducting regular audits and training. To safely handle information, establish explicit regulations for the preservation and deletion of data, offer clear permission procedures that educate users about the use of their data, and employ data anonymization techniques for any study or analysis to safeguard user privacy. This may protect sensitive data, uphold regulatory compliance, and foster user confidence by proactively addressing these threats.

# **Professional ethics Issues**

A healthcare chatbot must consider several ethical considerations, including protecting user privacy by disclosing the chatbot's limitations, maintaining accuracy and dependability through evidence-based information, preventing bias through the use of diverse data, and handling data ownership and usage transparently.

Among all, the data privacy and confidentiality seem to be the major one so described below clearly.

Data Privacy and Confidentiality: For a healthcare chatbot, maintaining data privacy and confidentiality is essential. To comply with laws like HIPAA and GDPR, this entails protecting data with robust encryption, enforcing stringent access restrictions, getting explicit user consent, and performing frequent security assessments. By taking these precautions, users' confidence is preserved, and sensitive data is protected.

# References