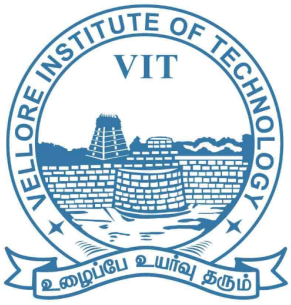


VIT®

Vellore Institute of Technology

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

Final Project Report

for: J Component of CSE3001

(Software Engineering)

Malaria detection web app

Detecting malaria from blood smears using Machine Learning

Prepared by:

Name	Reg. No.
Gandharv Sachdeva	19BCE0842
Aryaman Jaisinghaini	19BCE0853
Vikram Malani	19BCE0856

Table of Contents

1. Abstract

Malaria is a mosquito-borne disease caused by a class of parasites. These parasites are transmitted by the bites of infected female Anopheles mosquitoes. Initially if an infected mosquito bites a person, parasites carried by the mosquito get in the person's blood and start destroying their oxygen-carrying RBCs (red blood cells). Thus, early and effective testing and detection of malaria can save lives. Since the introduction of AI in the medical field, scientists and doctors have started using AI for the analysis and identification of diseases.

Some of the common methods to detect malaria are thick and thin blood smear examinations, polymerase chain reaction (PCR) and rapid diagnostic tests (RDT).

Excerpt from:

"Pre-trained convolutional neural networks as feature extractors toward improved Malaria parasite detection in thin blood smear images" by S Rajaraman et. al.

"Thick blood smears assist in detecting the presence of parasites while thin blood smears assist in identifying the species of the parasite causing the infection (Centers for Disease Control and Prevention, 2012). The diagnostic accuracy heavily depends on human expertise and can be adversely impacted by the inter-observer variability and the liability imposed by large-scale diagnoses in disease-endemic/ resource-constrained regions (Mitiku, Mengistu & Gelaw, 2003). Alternative techniques such as polymerase chain reaction (PCR) and rapid diagnostic tests (RDT) are used; however, PCR analysis is limited in its performance (Hommelsheim et al., 2014) and RDTs are less cost-effective in disease-endemic regions (Hawkes, Katsuva & Masumbuko, 2009)."

While RDTs are significantly faster than cell counting they are also much less accurate. An ideal solution would, therefore, need to combine the speed of RDTs with the accuracy of microscopy. Thus, malaria detection is an intensive manual process which can definitely benefit from the use of AI.

2. Aim

We plan on building an accessible simple web app for easy detection of malaria in an image of a blood smear. Any person with minimal medical expertise would be able to simply upload an image of a blood smear and get an accurate prediction on whether they have been infected with malaria or not.

3. Objectives

1. Determine a machine learning approach to build a good prediction model.
2. Create an api that could easily be used to query the trained neural network.
3. Create an intuitive web app to get blood smear images from users and show them results of analysis in real time.

4. Purpose and Scope

We plan to implement an intuitive web app to evaluate Malaria infection status of a patient. A machine learning approach is to be used to build a good prediction model which would be queried with an API for evaluating images of red blood cells. The app would prove to be boonful to the society as it would minimize the need for people to visit hospitals and wait hours just to get their samples analysed rather it would provide a quick and accurate result using a well defined machine learning approach.

5. Past Work and Existing software

5.1. Past Work

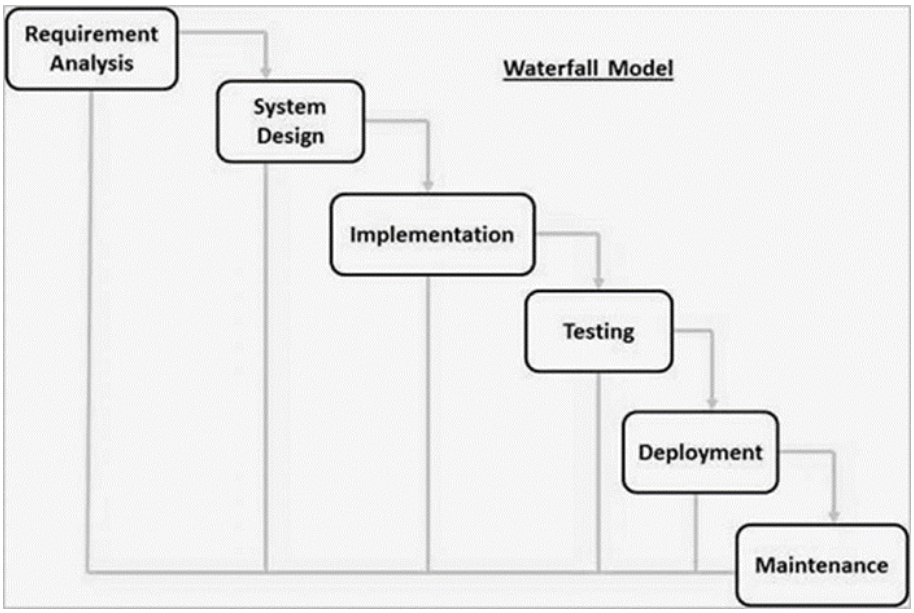
5.2. Existing Software

- Researchers at the Lister Hill National Centre for Biomedical Communications have developed a mobile application that runs on a standard Android smartphone attached to a conventional light microscope.
- Many convolutional neural networks have been trained for malaria detection as CNN's tend to be very good at computer vision tasks. **There is a serious lack of comparison between different ML approaches to classify malaria detection models.**

6. Process Model and Justification

6.1. Process Model

For this project we are going to use the “Waterfall” SDLC model. As we can imagine proper planning is a must in waterfall model. Planning is the process of establishing the scope and defining the objective and steps to attain them, the scope of our project is a well-defined and we clearly identified the steps to achieve our goal. Waterfall model is simple and easy to use, moreover works well with small-scale projects where requirements are very well understood, in waterfall model very less customer interaction is involved during the development of the product (i.e., like ours).



6.2. Shortcomings of chosen process model

The biggest shortcoming of using the waterfall model is the inflexible partitioning of the project into distinct stages. In our case this is a blessing in disguise simply because the components of our project are vastly different by nature.

The three major components of our project are:

1. Machine Learning model that predicts based on user input
2. An API (and backend) that connects the web app to the model
3. The web app itself, which is the single point of user interaction and the only thing that could be driven by the customer feedback.

Thus, all these components will have to developed in seclusion and then connected together, which would work very well with the waterfall model.

6.3. Justification

We have decided to not use other models as:

1. Use of an evolutionary model would be unfit for this project simply because the customer feedback is valuable to us, but it still only holds superficial value. The customers will only interact with our web app and thus their feedback would only guide the direction of the web app development, but much of our work would involve training the machine learning models.

2. Throw-away prototyping would be unnecessary for our project as our web app need to be robust and focussed rather than be feature rich.

3. Reuse-oriented development was a possible candidate for us, as we could have used an existing malaria detecting convolutional neural network, but we choose not to as we are not satisfied by the fitness scores of pre-existing models. We still plan on using off the shelf, existing solutions for some parts of our project ex. user authorisation on the web app could benefit from already existing solutions.

4. The iterative development model is again heavily influenced by customer feedback and thus does not work for us.

5. Incremental model, we decided against this model for similar reasons for which we decided not to use the iterative model, but we still plan on using the ideology behind the incremental model when deploying our web app, as we feel that features like user login and keeping previous patient data are our final aim but we could roll out functional software even without these features. So, during the creation of the web app, we would be following the incremental model to some extent.

6. We were not fond of using the spiral model as the biggest benefit of using the spiral model is the ability to backtrack and make changes. This would not be very useful for us simply because our requirements are very well laid out.

7. Agile methodology is again driven by customer feedback and our core requirements are unaffected by it.

7. Project Plan

7.1. Product Perspective

Our web app would allow for quick and easy Malaria detection just by uploading an image of a blood smear. For the detection of Malaria we plan on training and deploying our own Transfer Learning learning capable model. Using transfer learning algorithms such as vgg16, resnet, inception ,we will detect malaria from the blood cell image and the results of the test will be immediately conveyed to the respective patient.

7.2. Product Function

The website would allow any medical association such as a hospital or a clinic, to detect a malaria infection instantly using AI. The details of the patient which will be stored for future reference and the results will be inferred to the patient from the image of smeared blood drop. The results will be sent to the patient's email which will allow the patient to stay informed about their test results from the comfort of their home. The following functions would be available to the users.

- Secure Login and Registration
- Image Upload and Evaluation
- Intimation of test result through email

7.3. Operating Environment

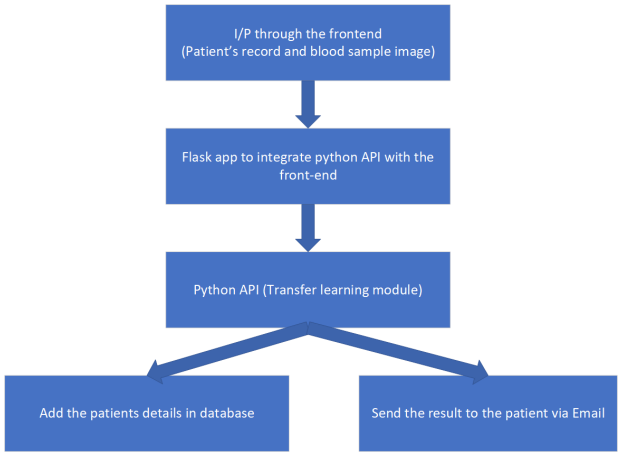
The Operating Environment is any modern web browser (preferably the latest version of Google Chrome i.e 94.0.4606.54).

7.4. Design and Implementation Constraints

1. Since we are dealing with confidential data such as user medical information we must ensure that the data we save is sanitised and is protected at all costs.
2. As our users will be providing their blood smeared rbc images, the image quality plays an important factor, so we either have to employ image enhancement policy or put constraints on the image quality being uploaded to maximise test accuracy.
3. The system SHALL be able to verify the person in case the person forgets his or her password.
4. The system SHALL be able to prevent invalid authentication and not grant access to the wrong user.
5. We also have to abide by the proposals put forth as medical software development

7.5. Assumptions made and dependencies

This project will be dependent on open-source well-curated and labeled smeared RBC datasets for training the model. Numerous such datasets are easily available officially through government websites and unofficially through private research organisations.
Once the model is trained and integrated with the web app the user would be able to access the web application and upload a picture of the blood sample for quick and accurate evaluation.
Another limitation is that to successfully detect whether a person has malaria or not our product would need a microscopic picture of the patient's blood sample.



8. Gantt Chart and Timeline

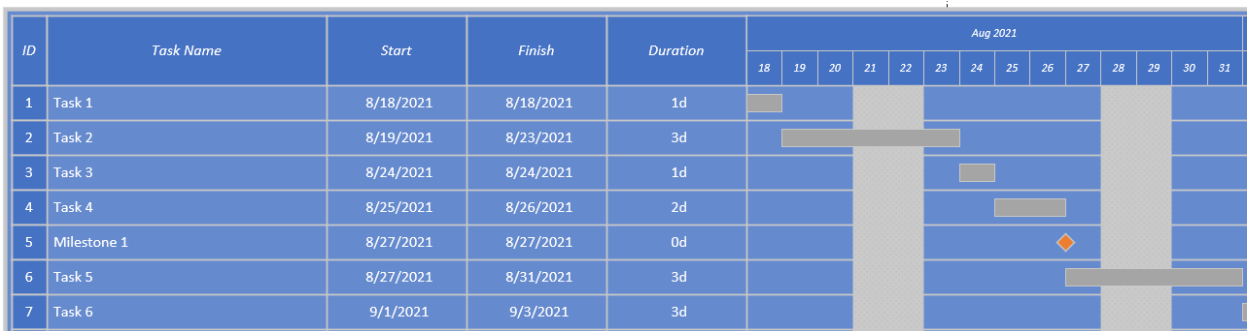
8.1. Timeline

Task / Milestone No.	Description	Date of Starting	Days	Date of Completion
Task-1 (T1)	Stakeholder Identification	18-08-21	1	18-08-21
Task-2 (T2)	Functional Requirements	19-08-21	3	23-08-21
Task-3 (T3)	System Requirements	24-08-21	1	24-08-21
Task-4 (T4)	Decide Software Process Model & Prepare Documentation	25-08-21	2	26-08-21
Milestone-1 (M1)	**Completion of Requirements Phase**	**27-08-21**	**0**	**27-08-21**
Task-5 (T5)	Task Division & Work Breakdown Structure	27-08-21	3	31-08-21

Task-6 (T6)	Pert Chart & Gantt Chart Preparation	01-09-21	3	03-09-21
Milestone-2 (M2)	**Completion of** **Analysis Phase**	**06-09-21**	**0**	**06-09-21**
Task-7 (T7)	Architecture Design & Diagram	06-09-21	3	08-09-21
Task-8 (T8)	UI Design	08-09-21	6	15-09-21
CAT 1				**13-09 to 20-09**
Task-9 (T9)	Detailed Designing using UML & ER Diagram	20-09-21	12	05-10-21
Milestone-3 (M3)	**Completion of Design Phase**	**06-10-21**	**0**	**06-10-21**
Task-10 (T10)	Machine Learning model training	7-10-21	12	22-10-21
CAT 2				**25-10 to 29-10**
Task-10 (T10) (contd.)	Machine Learning model training	1-11-21	18	19-11-21

Task-11 (T11)	Coding of User Side with all the functionalities	22-11-21	7	30-12-21
Milestone-4 (M4)	**Completion of Coding Phase**	**1-12-21**	**0**	**01-12-21**
Task-12 (T12)	Testing of Machine learning model	01-12-21	3	03-12-21
Task-13 (T13)	Testing of API and web app	06-12-21	2	07-12-21
Milestone-5 (M5)	**Completion of Testing Phase**	**08-12-21**	**0**	**08-12-21**
Task-14 (T14)	Prepare of Documentation for the project	08-12-21	2	09-12-21
Milestone-7 (M7)	**End of Project**	**10-12-21**	**0**	**10-12-21**

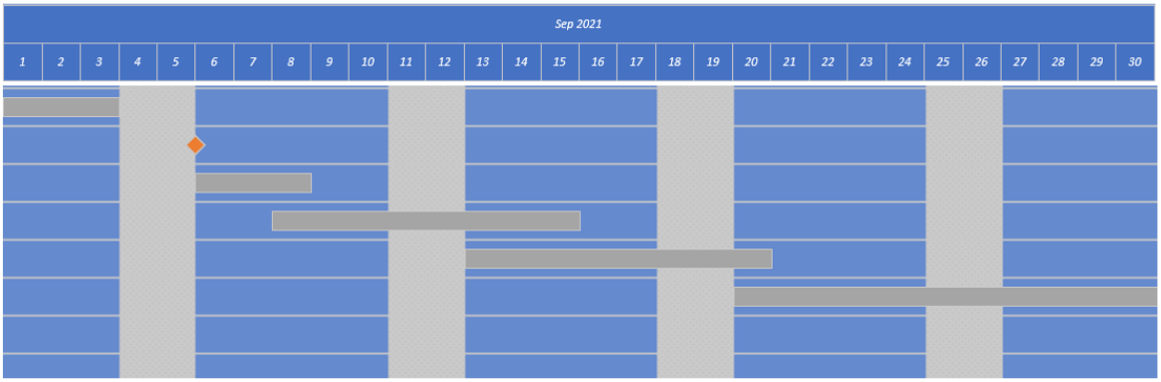
August



ID	Task Name	Start	Finish	Duration
1	Task 1	8/18/2021	8/18/2021	1d
2	Task 2	8/19/2021	8/23/2021	3d
3	Task 3	8/24/2021	8/24/2021	1d
4	Task 4	8/25/2021	8/26/2021	2d
5	Milestone 1	8/27/2021	8/27/2021	0d
6	Task 5	8/27/2021	8/31/2021	3d
7	Task 6	9/1/2021	9/3/2021	3d

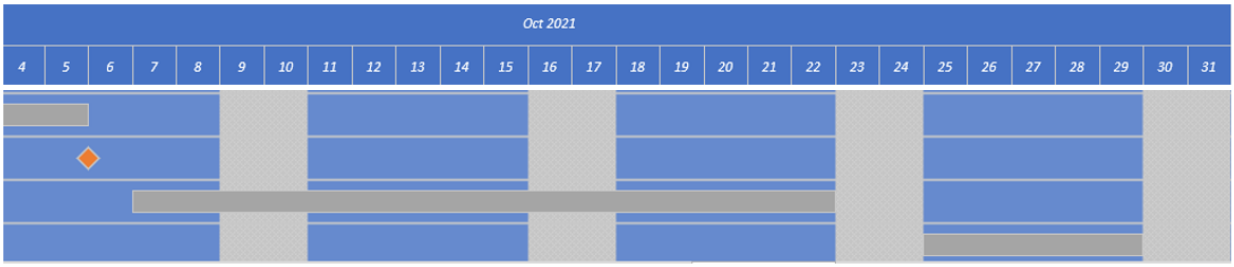
September

ID	Task Name	Start	Finish	Duration
7	Task 6	9/1/2021	9/3/2021	3d
8	Milestone 2	9/6/2021	9/6/2021	0d
9	Task 7	9/6/2021	9/8/2021	3d
10	Task 8	9/8/2021	9/15/2021	6d
11	CAT 1	9/13/2021	9/20/2021	6d
12	Task 9	9/20/2021	10/5/2021	12d



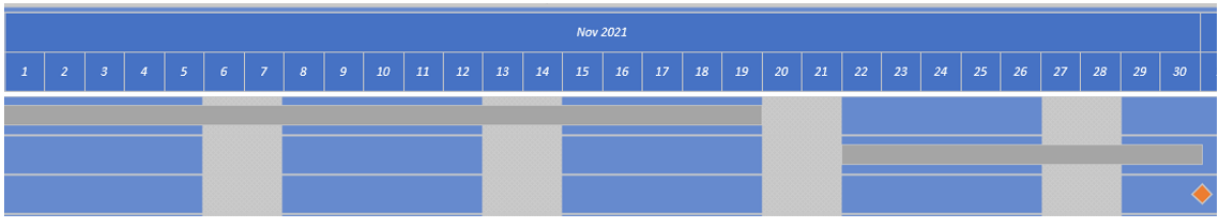
October

ID	Task Name	Start	Finish	Duration
12	Task 9	9/20/2021	10/5/2021	12d
13	Milestone 3	10/6/2021	10/6/2021	0d
14	Task 10	10/7/2021	10/22/2021	12d
15	CAT 2	10/25/2021	10/29/2021	5d



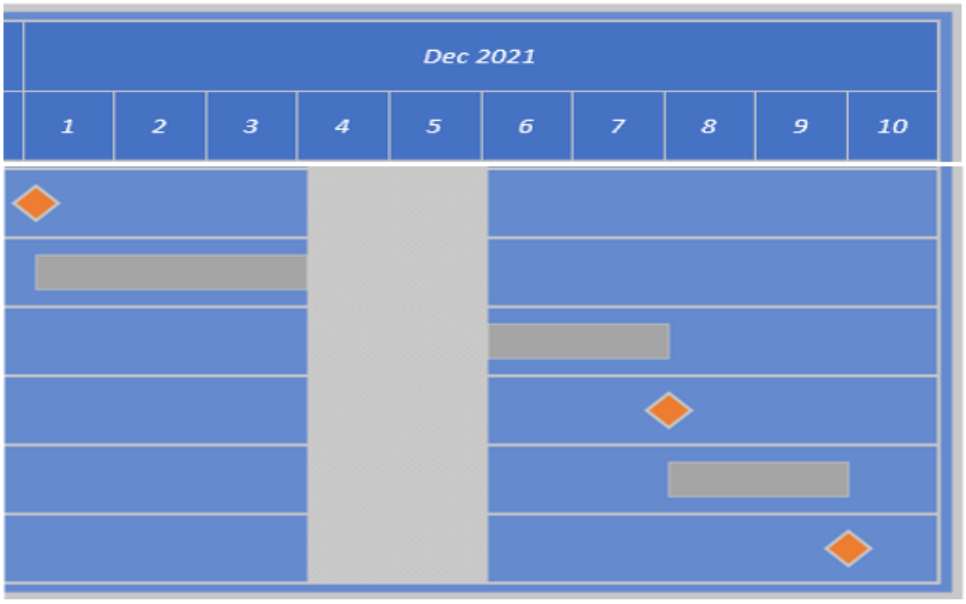
November

ID	Task Name	Start	Finish	Duration
16	Task 10 (contd.)	11/1/2021	11/19/2021	15d
17	Task 11	11/22/2021	11/30/2021	7d

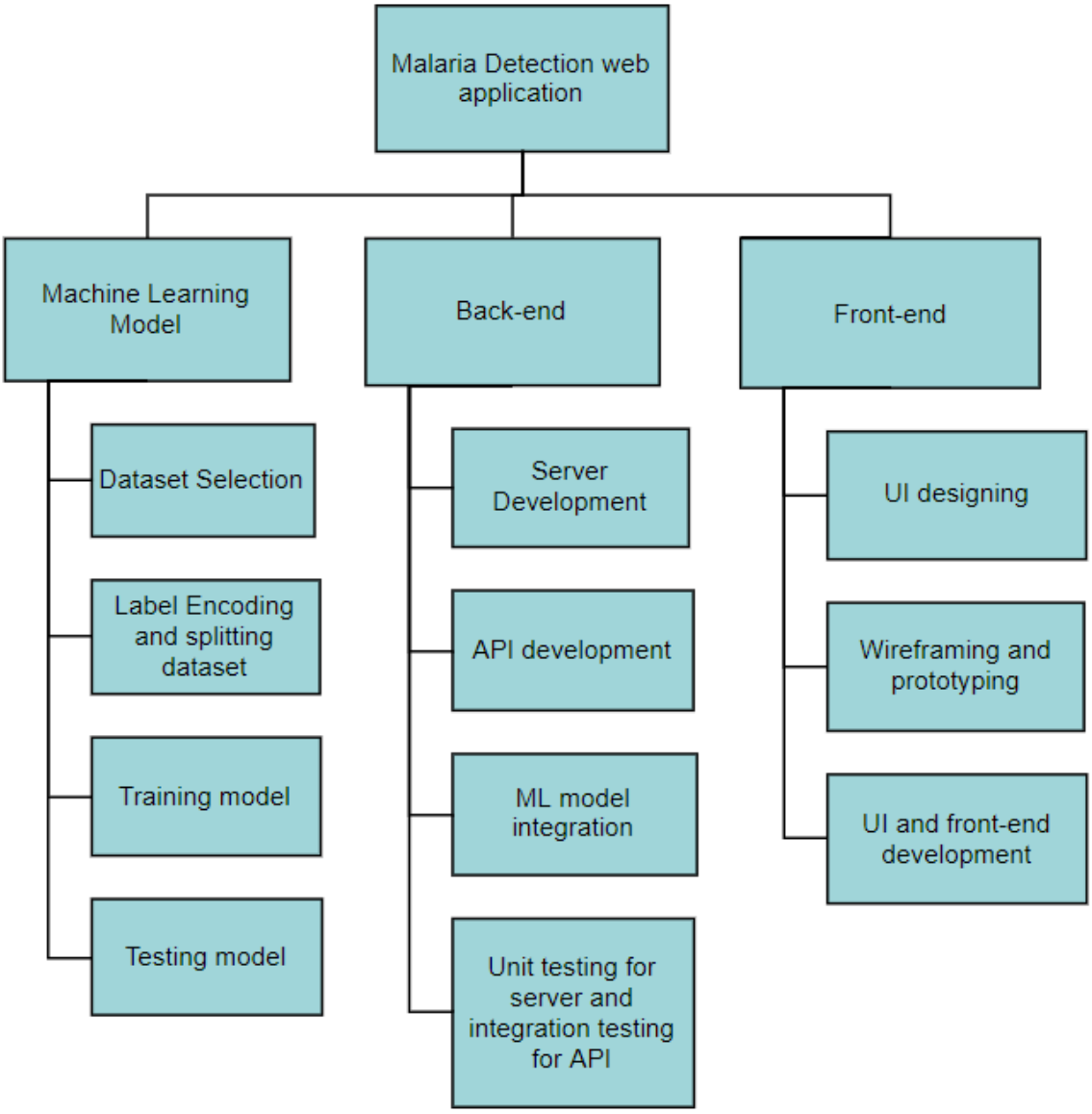


December

ID	Task Name	Start	Finish	Duration
18	Milestone 4	12/1/2021	12/1/2021	0d
19	Task 12	12/1/2021	12/3/2021	3d
20	Task 13	12/6/2021	12/7/2021	2d
21	Milestone 5	12/8/2021	12/8/2021	0d
22	Task 14	12/8/2021	12/9/2021	2d
23	Milestone 6	12/10/2021	12/10/2021	0d



9. Work Breakdown Structure



10. Requirements

10.1. Module Wise Functional Requirements

10.1.1. Secure Login and Registration

- **Description and Priority**

  Whenever the user visits our web application, he/she will be required to login and for the first time they have to register. **Priority: Medium**

- **Stimulus/Response Sequences**

  After Submission they can access to various information available in our web applications like image upload, results view, etc.

- **Functional Requirements**

  1. User's password should be in encrypted form.
  2. Should display invalid email/password combinations for unauthorized access.
  3. Should not allow multiple accounts with the same email id.
  4. Email id should be legitimate.

  Secure Login and Registration

10.1.2. Image upload and Evaluation

- **Description and Priority**

  The user must upload the image of the blood sample to request the results generated by the ML model. **Priority: High**

- **Stimulus/Response Sequences**

  Once the user has uploaded the chemical smeared image of the blood sample along with the age and the temperature the user can request results predicted by the ML model.

- **Functional Requirements**

  1. The ML model should give the results immediately.
  2. Immediate communication of blood tests to the concerned patients.
  3. Highly accessible web app design to help the users upload the scanned images.
  4. The results should be accurate.

10.1.3. Intimation of test result through Email

- **Description and priority**

  The email id entered by the user would be used to intimate the user about the result of their blood smear sample analysis at the comfort of their homes. **Priority:LOW**

- **Stimulus/Response sequences**

  After the user has entered their email information, it would be stored in a secure database and would be extracted as and when required for future intimation

- **Other Functional Requirements**

  1. Should not allow multiple accounts with the same email id.
  2. Email id should be legitimate

10.2. Non Functional Requirements

10.2.1. Performance Requirements

Since, our web app is focussed on providing essential medical information to users at the earliest, we have the following inexcusable performance requirements:

1. Average page load (from a user perspective) must be less than 500 milliseconds.
2. Slowest page load cannot take more than 4 seconds.
3. The app must be available almost 24*7.

10.2.2. Security Requirements

Secure access of confidential data (user's details). We will use a NOSQL database with data sanitization to ensure the integrity and confidentiality of data is guaranteed.

10.2.3. Safety Requirements

Since we are training our own model it is of utmost importance to make sure that the users are made aware of this fact and they should also be made aware of the accuracy and confidence score that our model possesses.

10.2.4. Software Quality Attributes

As we are training our own Transfer Learning model we have to ensure that it squares up with the existing non transfer learning models being used for Malaria Detection. Also, as we are dealing with important medical data, we must test our web app for secure transmissions.

10.3. Other Requirements

As we are developing a transfer learning capable model for Malaria detection, we can also train it on other image based datasets to reuse it for other possible applications. So we have to keep track of the reusability of the model and the way it communicates with our web app to ensure its maximal reusability.

11. General Design

11.1. Design considerations

11.1.1. Assumptions

The Blood Smear image should be of sufficient quality in order to get an accurate prediction/diagnosis. The user must have access to the internet in order to use the web page and upload an image of a blood smear of a rational size.

11.1.2. Resource Assumption:

- All project team members are available and have the necessary skills and knowledge to work on the project.
- Members working on the front end should be skilled in html, css and javascript.
- Members working on the back end should be skilled in working with Flask and MongoDB
- Electricity will be on during working hours.
- All software works flawlessly on team members' devices.

11.1.3. Scope Assumptions:

- Scope doesn't change.
- If it should; project will follow a change control approval process.

11.1.4. Constraints

Without proper internet connection, the web application, image upload and result generation, the Software would prove to be inefficient.

11.1.5. Risks and Volatile Areas

- The system will not be able to verify the person in case the person forgets his or her password.
- The system will not be able to prevent invalid authentication and grant access to the wrong user.
- Since we are dealing with confidential data such as user medical information saved must be sanitised and is protected at all costs.

11.2. Architecture

We will employ the Client Server architecture. The client-server architecture is most useful for applications that require a separation or abstraction of concerns between the client and the server; it is meant for systems with high interoperability. The client-server architectural style helps applications improve performance in scalability.

11.2.1. Overview

The client server architecture would consist of two layers, with each layer performing a specific role.

- **Web and Application layer:** The first layer would consist of the application interface. It will communicate with other layers by feeding input from users and carrying out user operations. This layer also controls the applications functionality by performing detailed processing.

- **Data layer:** This is the most critical aspect of the application, it is where the user data, operational data and metadata are stored for easy access and retrieval. All database logic and entity relationships will be defined here. It consists of Database servers.

- **Database:** A database implies a persistent and integrated storage allowing concurrent access to it by many users. It is a collection of records related by referential integrity. Thus, a database is an organized collection of structured data, to serve many applications with minimum redundancy.

Structural decomposition: The system is divided mainly into the following modules, in which each module is performing its individual building task.
▪ Module 1 is developed for Communication with ML model.
▪ Module 2 is developed Image upload and processing
▪ Module 3 is developed for Payments and Notification
The system was structurally decomposed in such a way because we have chosen the waterfall SDLC process model for our system, which is why we discussed and decided to work on module by module. Further these modules have been decomposed in such a way that they have **loose coupling and high cohesion**.

Functional decomposition:

Module 1's functionality is to make use of different API's and to allow Front-end and ML Model to communicate with each other seamlessly.

Module 2 is the interaction point for the users. The users will upload the images which will be processed and made appropriate for prediction.

Module 3 will be incharge of notifying the user of the test results and prompting payment.

11.2.2. Subcomponents

Module 1 will be used as the communication channel between the CNN model and the front-end

▪ The processed image will be provided to the ML model.
▪ The image along with user details such as age and temperature will be used to generate a diagnosis.
▪ The diagnosis will be passed onto the backend where the report generation will take place.

Module 2 is the precursor to Module 1 as it is responsible for the user interaction.

• The user will login to the web app
• User will upload the image of a blood smeared RBC
• The image will be processed and passed onto the ML model.

Module 3 is developed for handling payments and notifications

• This module is responsible for notifying the user about their test results
• The user will also be prompted for a payment for the complete medical report

11.3. Database Scheme

Collections, Fields and Relationships

We plan on using MongoDB as our database as we have a very simple database schema which does not require complex querying. A NoSQL database like MongoDB provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.

Databases

Patients: This is our main database that will hold our user details, including their information as well as their most recent diagnosis.

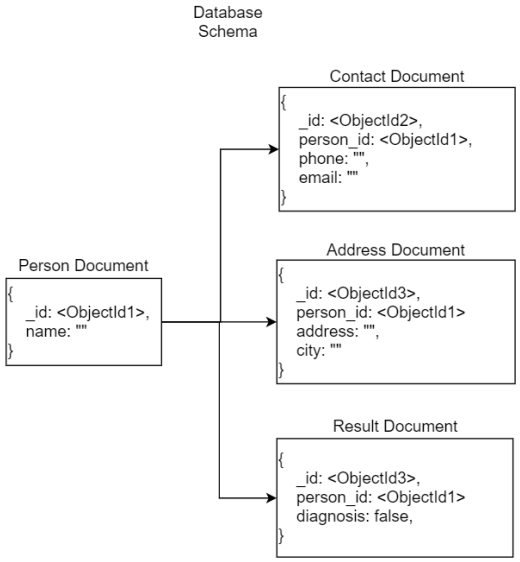
New Collections

Person: This collection will consist of the patients in our system. Every document in this collection will further be linked to corresponding Contact, Address and Result documents.

Contact: This collection will be used to store the contact details of a patient. A Contact document will primarily be referred through a corresponding document in the Person collection.

Address: This collection will be used to store the address details of a patient. An Address document will primarily be referred through a corresponding document in the Person collection.

Diagnosis: This collection will be used to store the most recent test results of patients. A Diagnosis document will primarily be referred through a corresponding document in the Person collection.

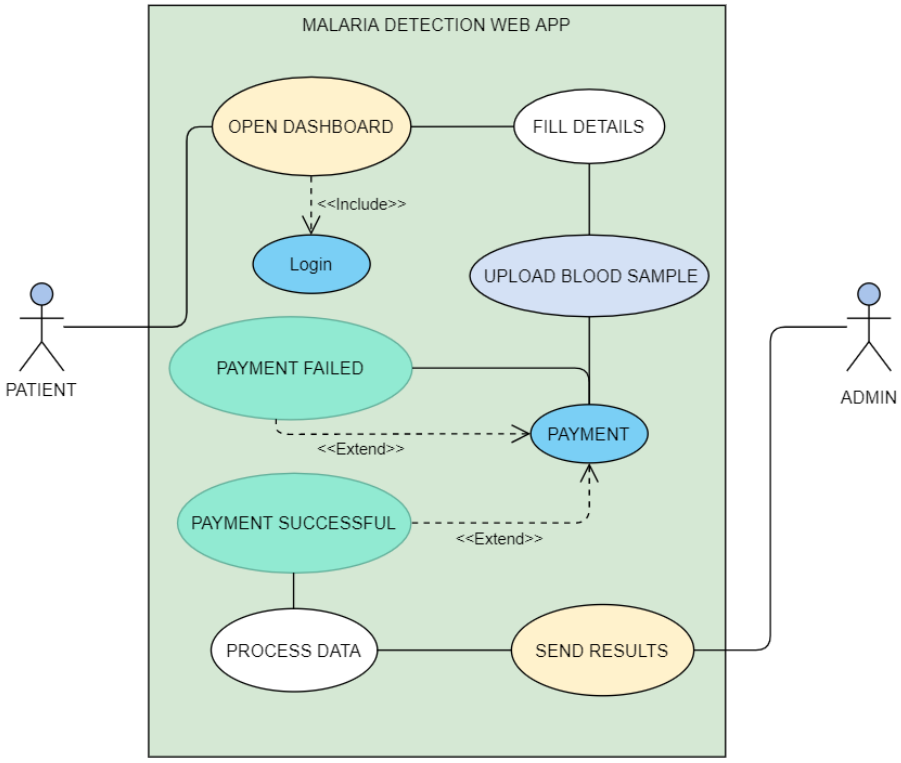


12. High Level Design

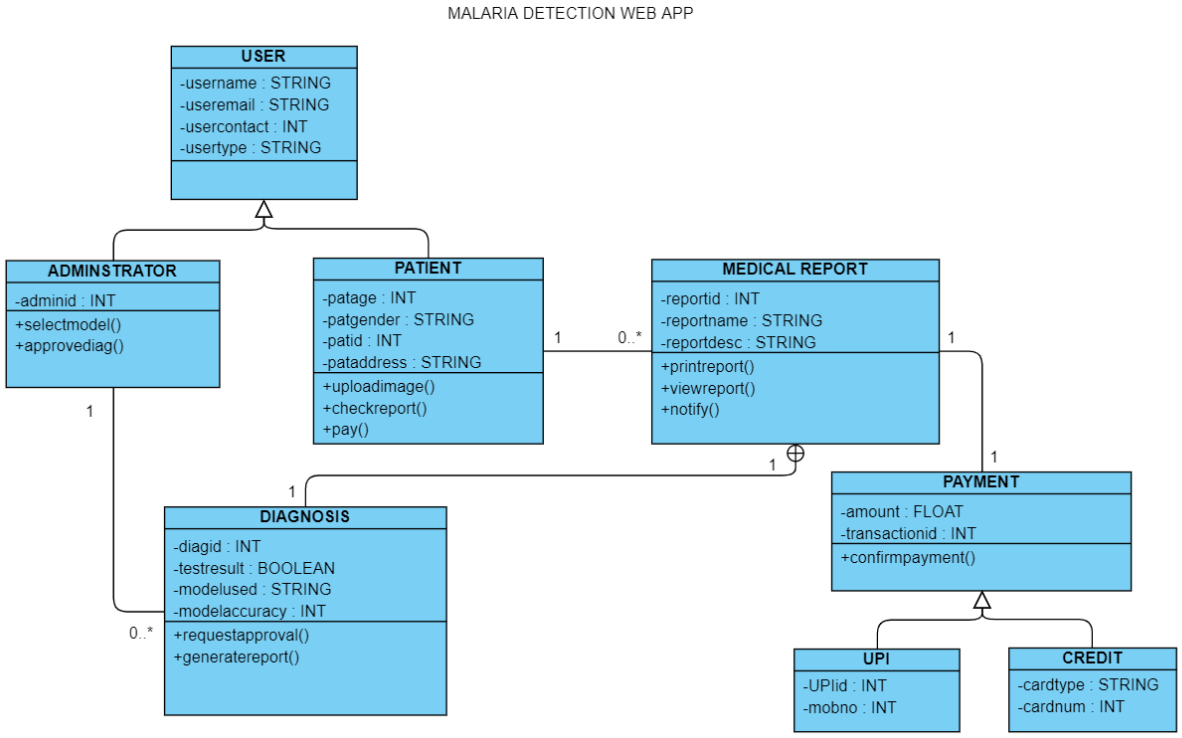
Use-case diagrams are the main analysis-level behavior modeling technique in UML. It is possible to develop many use case diagrams to represent various aspects of a system at various levels of abstraction. However, use case diagrams do not create hierarchical structures reminiscent of Data Flow Diagrams (DFDs). The real power of Use Case diagrams is in textual specifications of use cases stored in the repository. Use case specifications guide developers in most modeling tasks. They are used to develop test cases, record possible defects and identify possible future enhancements. They are actually essential for establishing maintenance and evolution tasks. The next diagram shows a schematic depiction of the use case diagrams of some aspects of the system developed in this work[5.1]. Use case represents a major piece of system functionality. Another modeling technique in UML used for modeling the information system for

managing malaria cases was the use of class diagrams. Modeling of contemporary (object oriented) systems is done in the Unified Modeling Language (UML). Thus, the UML is a language for specifying, visualizing, constructing, and documenting artifacts of software systems, as well as for modeling and other non-software systems.

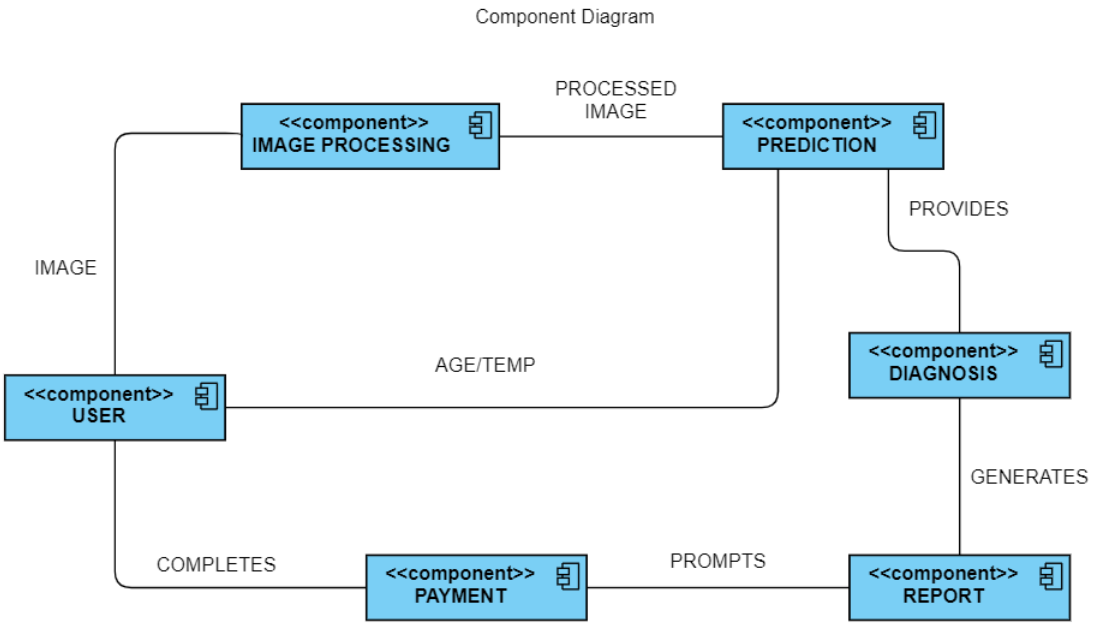
12.1. UML Diagram (Use-Case Diagram)



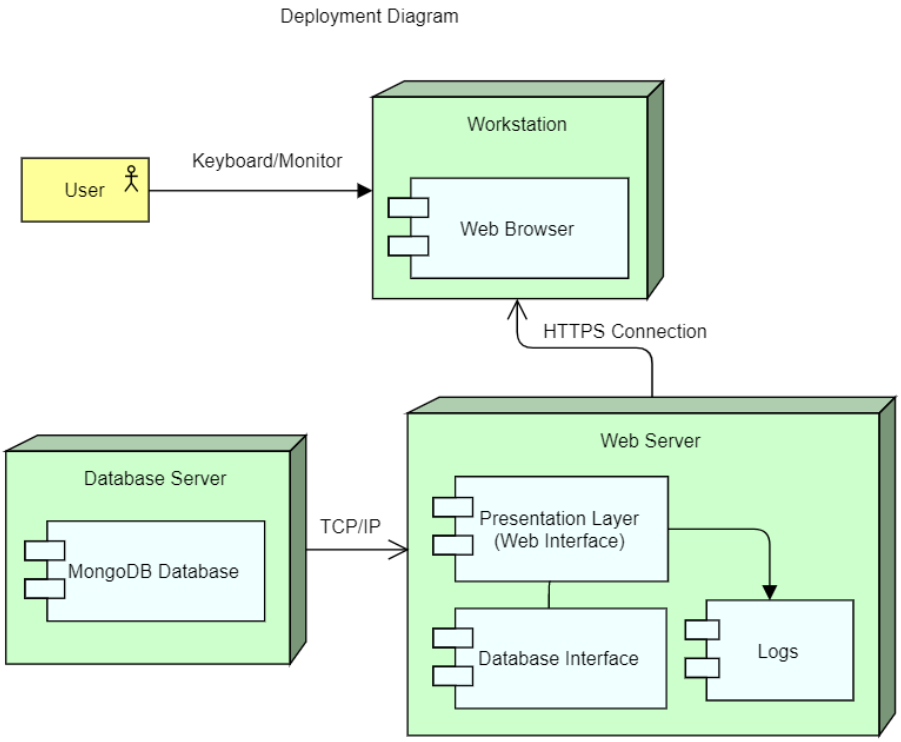
12.2. UML Diagram (Class Diagram)



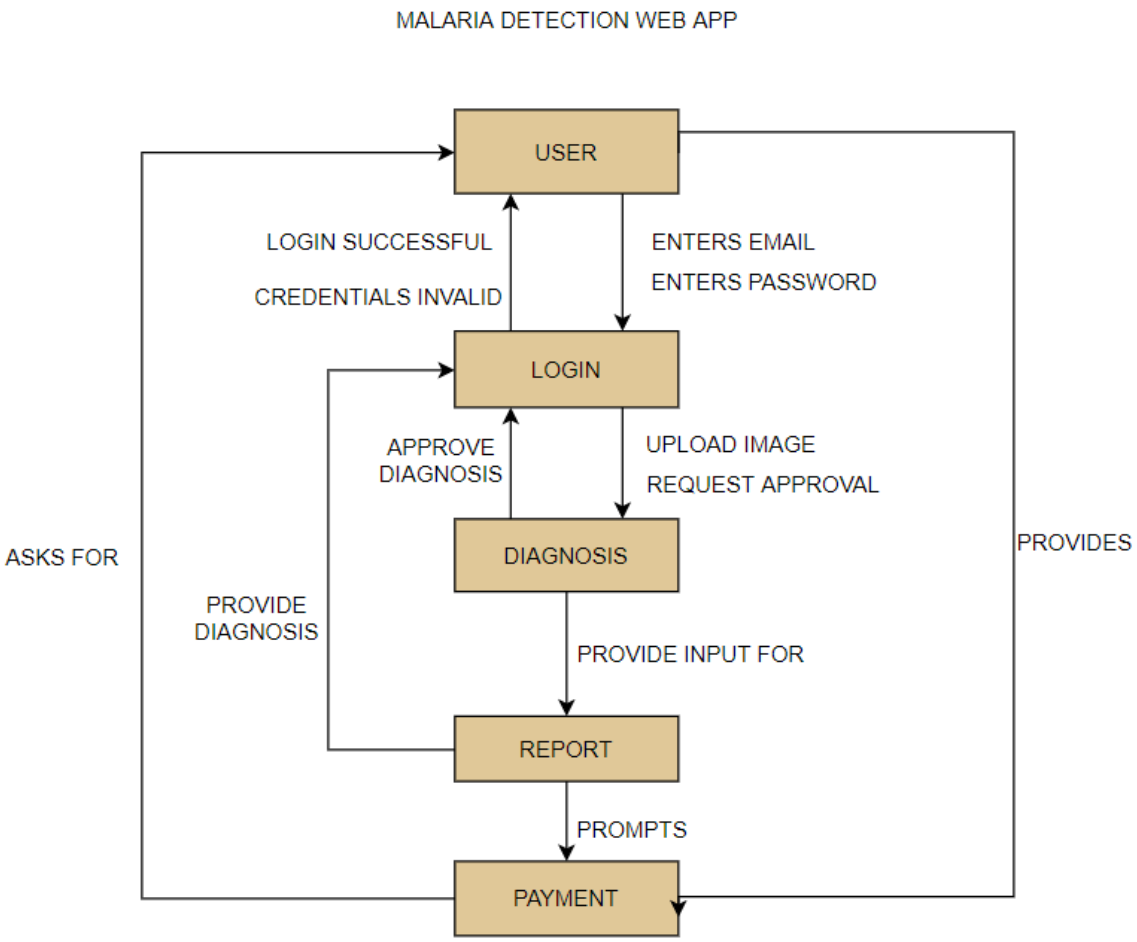
12.3. UML Diagram (Component Diagram)



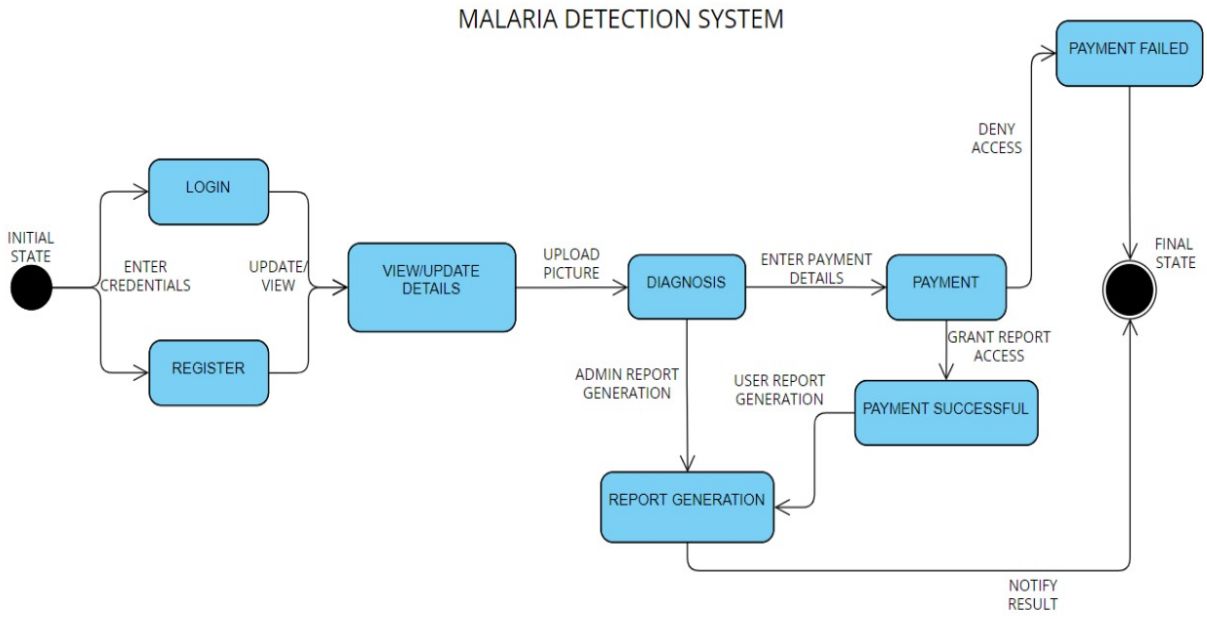
12.4. UML Diagram (Deployment Diagram)



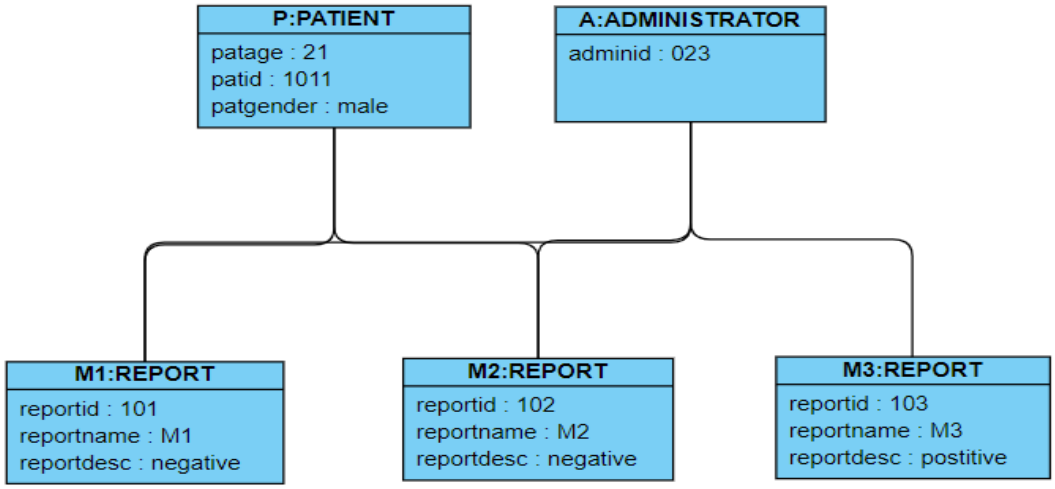
12.5. UML Diagram (Collaboration Diagram)



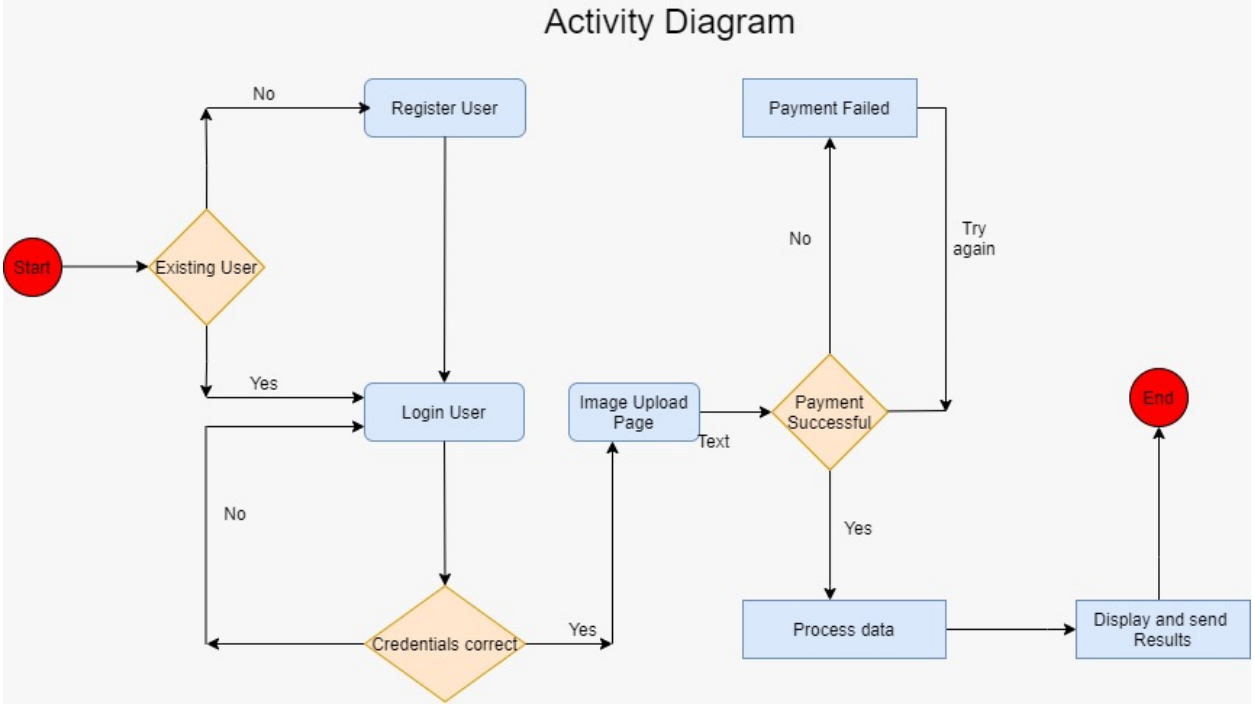
12.6. UML Diagram (State Diagram)



12.7. UML Diagram (Object Diagram)



12.8. UML Diagram (Use-Case Diagram)



13. Low Level Design

This section provides low-level design descriptions that directly support the construction of modules.

Module 1

Communication using API

The web app will communicate with the ML model using a RESTful API, the API will be implemented in Flask for Python. The API will work with a JSON request body along with a payload of the processed image.

Importing image

The ML model will access the processed image using the payload provided in the JSON POST request. The image will be stored as an upload in the Flask backend.

ML Feature Extraction

The image will be converted in a binary format which will directly be used for extracting features and generating a prediction. The features will differ from model to model. Automatic filter generation will also be supported in the Transfer learning capable ML model.

Response

The ML model will classify the image, based on this classification and user provided information a diagnosis will be generated and passed onto the web app using JSON response.

Module 2

User Login/Signup

This module will validate the users and maintain a list of active sessions. New users will be provided a signup functionality and on confirmation of their information they will be added to the database.

Image Upload

On successful sign up, the user will be taken to the dashboard, where they will be prompted for uploading the image of a blood smeared RCB.

Image Processing

The user uploaded image will be processed using 3rd party libraries to make it easier for the ML model to interpret the image and extract features from it.

POST request

A POST request will be generated consisting of the image and the user information, the JSON request body will consist of the user information, the unique identifier of the image along with its local path on the server. The JSON format is being used as a common communication format between the front end and the backend.

Module 3

Report Generation

The diagnosis provided in the JSON will be used to generate a report consisting of the diagnosis along with suggestions.

Notification

The users will be notified using email. In order to securely email a user their report SMTP protocol will be used so that the report is not intercepted over the network, leading to compromising the data integrity.

Payment

The user will be presented with payment options in order to unlock the complete medical report consisting of the diagnosis along with the complete stats about the model used, the model accuracy and further suggestions based on the diagnosis.

Payment Handling

The payments will be handled using 3rd party OAuth and Payment handlers. This is done to ensure that the payment framework is always up and running and is highly secure.

14. UI Designing

Application Controls

We have decided upon the following application controls for our app:

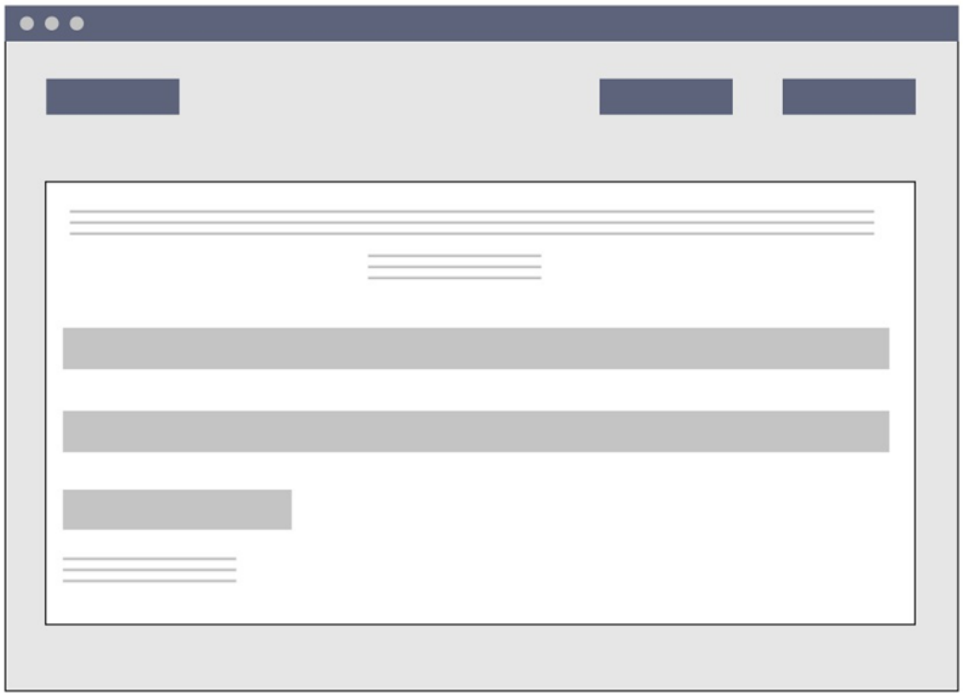
Toolbar: The toolbar element will be consistent for all the web pages, it will consist of navigation elements and app logo.

Sidebar: On smaller devices the toolbar will be replaced by a sidebar. Like the toolbar the main components of the sidebar will also be navigation elements along with app name and logo.

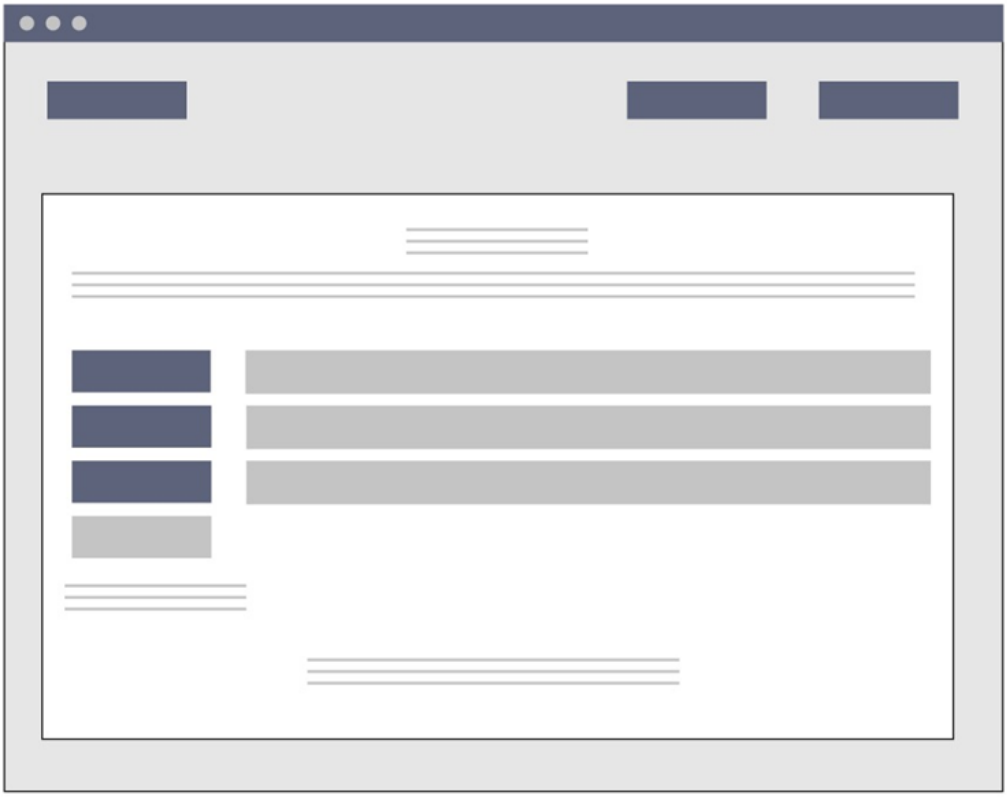
We plan on using COTS front end library- Bootstrap, to ensure consistent design among all elements.

Wireframes

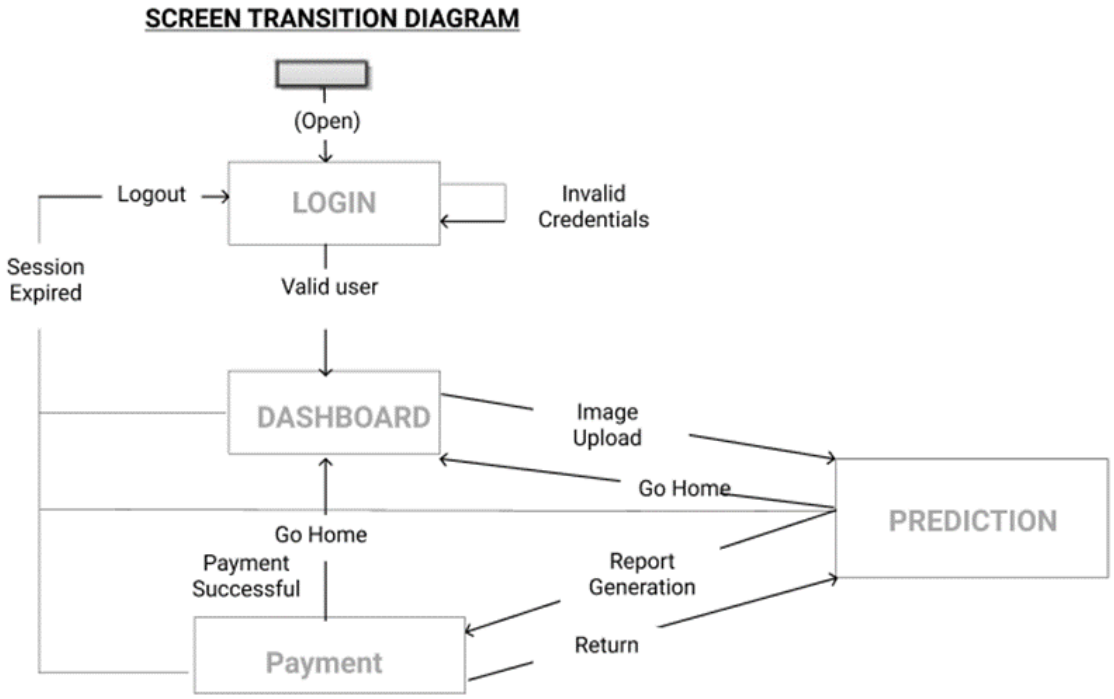
Login/Signup Wireframe



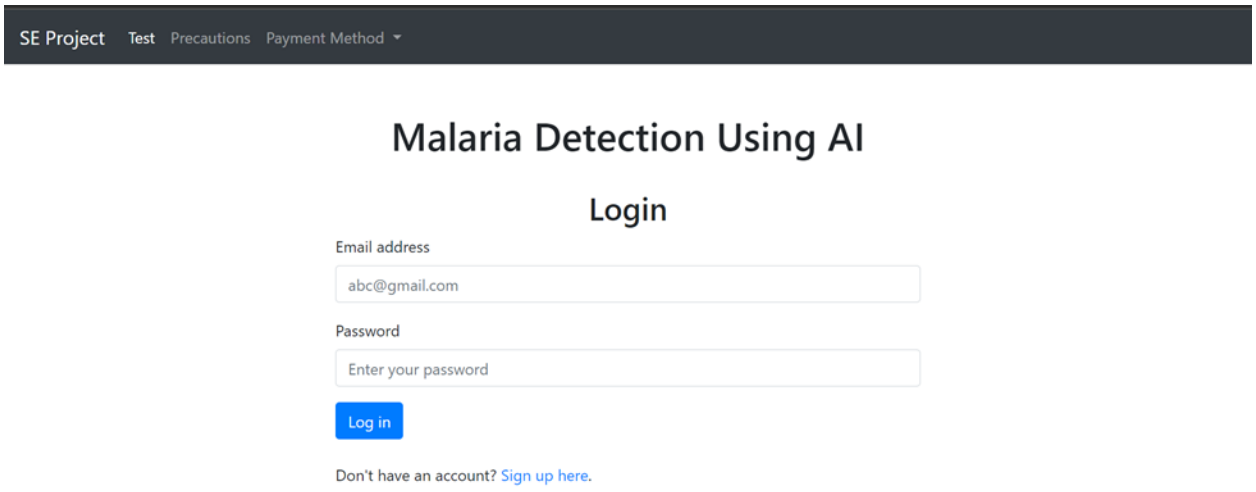
Prediction Wireframe



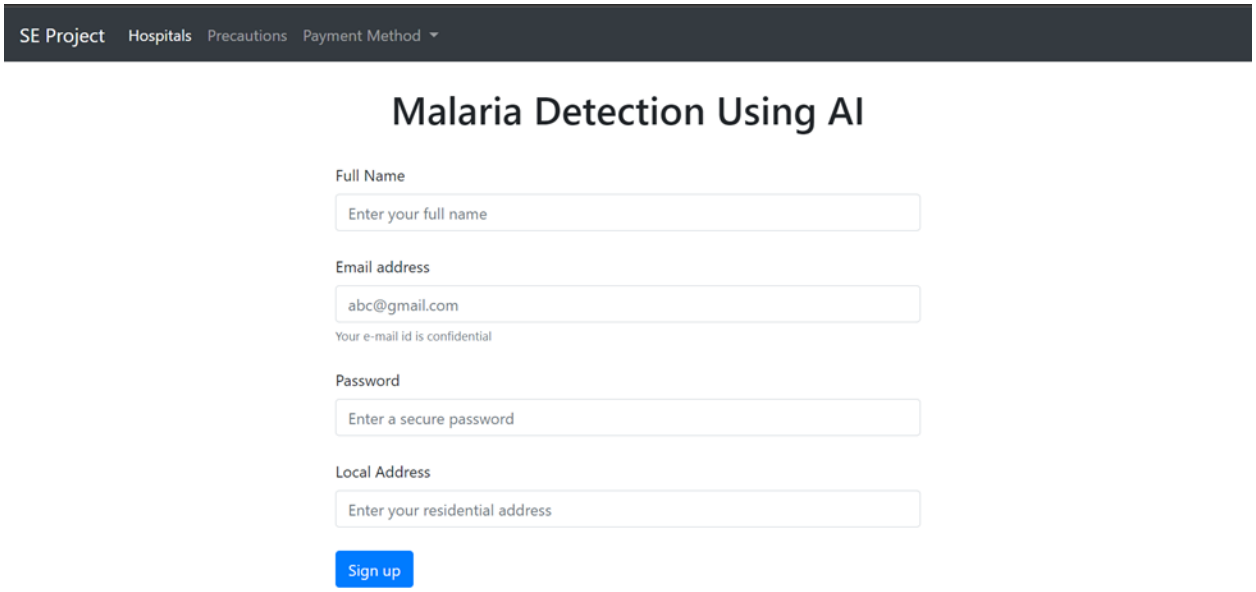
Screens



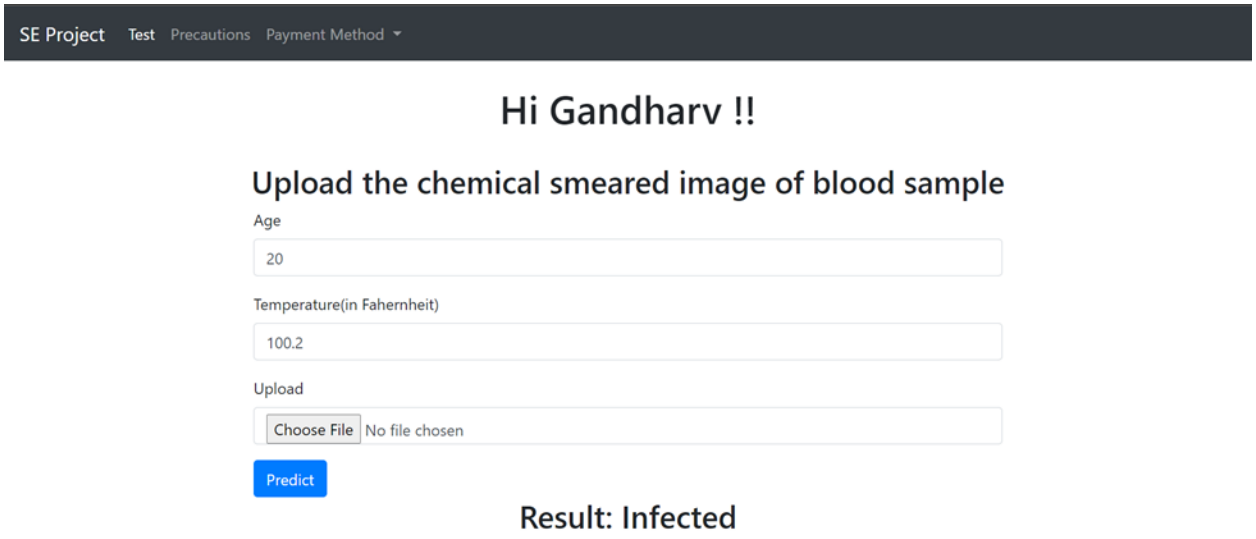
Login



Signup



Prediction Results



The following is a breakdown of the values in the above Prediction Results screen:

Label Name	Note	Source
Name		users.find_one({ "email": request.form["email"] })["name"]
Email Address	Entered by the user during login. Will be stored in session using POST request.	request.form["email"]
RBC Image	Uploaded by user. Accessed through file uploads.	request.files["file"]
Age	Will be passed to the Prediction Model.	request.form["age"]
Temperature	Will be passed to the Prediction Model.	request.form["temp"]
Diagnosis		Result of prediction. Will be serialized in JSON

15. Code

Github Link: https://github.com/gansach/SE-Project

16. Testing

16.1. Tools for Software Testing

- **Selenium**

Selenium is a testing framework to perform web application testing across various browsers and platforms like Windows, Mac, and Linux. Selenium helps the testers to write tests in various programming languages like Java, PHP, C#, Python, Groovy, Ruby, and Perl. It offers record and playback features to write tests without learning Selenium IDE.

Selenium proudly supports some of the largest, yet well-known browser vendors who make sure they have Selenium as a native part of their browser. Selenium is undoubtedly the base for most of the other software testing tools in general.

- **TestingWhiz**

TestingWhiz is a test automation tool with code-less scripting by Cygnet Infotech, a CMMi Level 3 IT solutions provider. TestingWhiz 's Enterprise edition offers a complete package of various automated testing solutions like web testing, software testing, database testing, API testing, mobile app testing, regression test suite maintenance, optimization, and automation, and cross-browser testing.

TestingWhiz offers various important features like:
Keyword-driven, data-driven testing, and distributed testing

- Browser Extension Testing
- Object Eye Internal Recorder
- SMTP Integration
- Integration with bug tracking tools like Jira, Mantis, TFS and FogBugz
- Integration with test management tools like HP Quality Center, Zephyr, TestRail, and Microsoft VSTS
- Centralized Object Repository
- Version Control System Integration
- Customized Recording Rule

TestComplete

TestComplete is a functional testing platform that offers various solutions to automate testing for desktop, web, and mobile applications by SmartBear Software.

TestComplete offers the following features:

- GUI testing
- Scripting Language Support – JavaScript, Python, VBScript, JScript, DelphiScript, etc.
- Test visualizer
- Scripted testing
- Test recording and playback

- **Ranorex**

Ranorex Studio offers various testing automation tools that cover testing all desktop, web, and mobile applications.

Ranorex offers the following features:

- GUI recognition
- Reusable test codes
- Bug detection
- Integration with various tools
- Record and playback

16.2. Test Cases (Tool used: Ranorex)

Test cases for Web App

Test Case	Details	Test Data	Expected Results	State Reached
1	Open website to goto signup page	URL	Website should open	Site Opened
2	Fill Signup Page	Name: Gandharv Email: sachdevagandharv@gmail.com Password: test Location: Gurgaon	Validate all fields if specifications don't meet, error is thrown	Details Entered
3	Page Signup	Pressing Button	Validate and redirect to the Login Page.	User Registered
4	Fill Login Page	Email: sachdevagandharv@gmail.com	Authenticate and Redirect to Image Upload Page	User Logged In

Test Cases for ML Prediction and Notification

Test Case	Details	Test Data	Expected Results	State Reached
1	Redirect to image upload	URL	Website should open	Site Opened
2	Uploading the image	Upload the image of the Blood Smear	Image should be stored as file upload in Flask backend	Image Upload is successful
3	Prediction button is clicked	Blood Smear image	Model predicts infection status	Diagnosis is produces
4	Notification via email	Diagnosis	Result should be received as email	Email Received