Lecture Notes

Requirements Identification

A new way of learning

In this module, you learnt about the first phase of Software Development Life Cycle - Requirements Identification. You were first introduced to what Experiential Learning is and then you were briefed about the SDLC process. You got to know about the web application you are going to develop in the experiential learning course and clearly understood the requirements for the same.

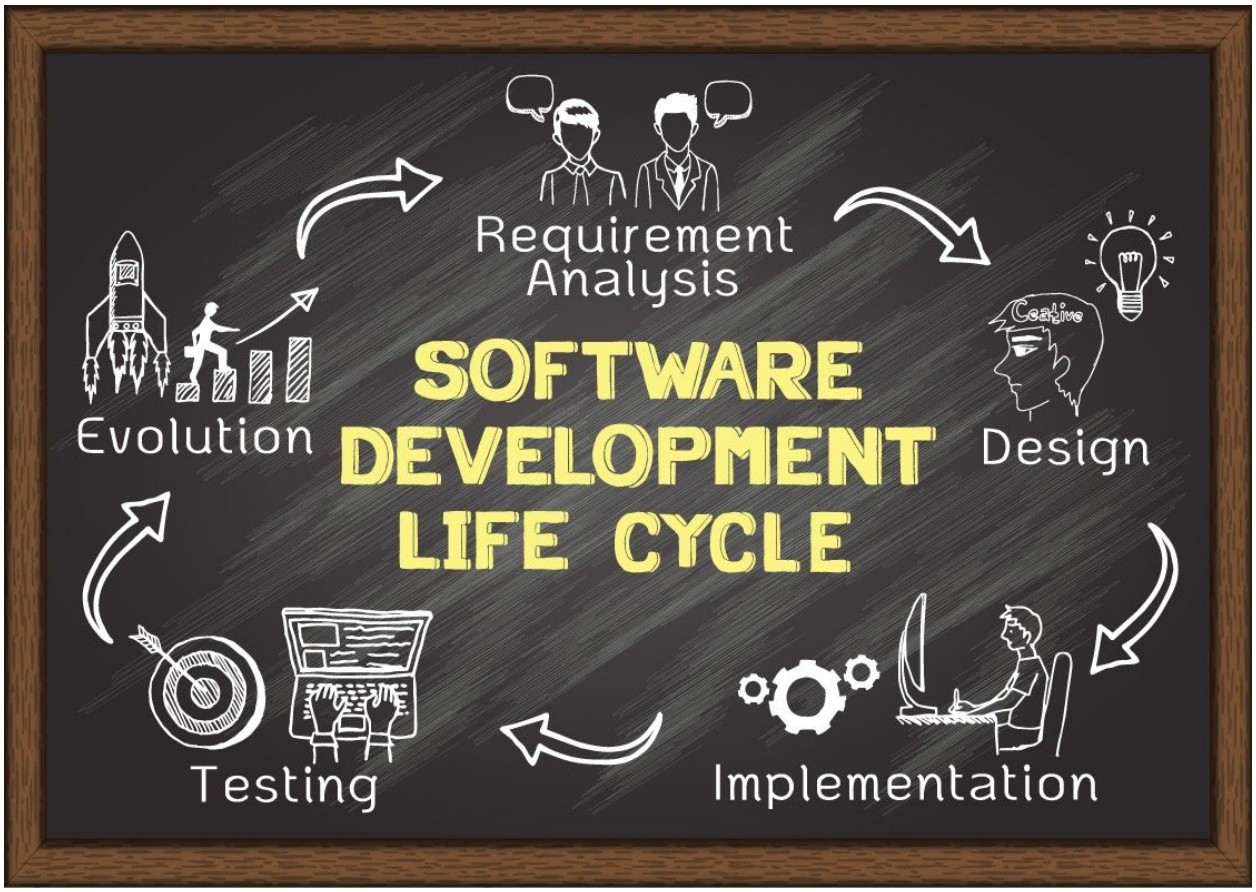
Introduction to Experiential Learning

Experiential learning is the process of learning through experience and hands-on form of learning is one such learning process. The main idea is to give you a sense on how product development happens in the industry and to make you do hands-on like a real software developer.

Requirements Identification

Introduction to Web Development and SDLC

Software development Life Cycle(SDLC) is a process used in the software industry to design, develop, test and deploy quality software applications. It is the basic process followed in any software industry to develop a software product. The following figure illustrate the basic SDLC process :



This session introduced you to the basic concepts of Requirements Identification and its importance, which is the first phase of software development life cycle. You learned about the importance of req gathering, how it is actually done, why is it beneficial % also saw some real life examples explaining requirement gathering.You were also be introduced to the various other concepts involved in the requirements gathering phase such as Use Case Diagrams, Flow Diagrams, Database Design, Designing the Mock UI and so on. Also, apart from learning these concepts, you also performed these tasks for the UPSTAC web application.

A web application is nothing but an application program stored on a remote server and delivered over the internet through a web interface like a web browser. Nowadays, software and web applications such as Twitter, Gmail, Amazon, Uber are used extensively. These applications are efficient and evolve by themselves in order to integrate new features as and when demands for new features arise. Whenever the idea for developing an application comes up, the first process performed is requirement gathering wherein the key requirements or features are gathered and further analysed or refined. This first step is important because if requirement gathering is not done prior to application development, it may lead to a failure and possible restructuring of the whole application.

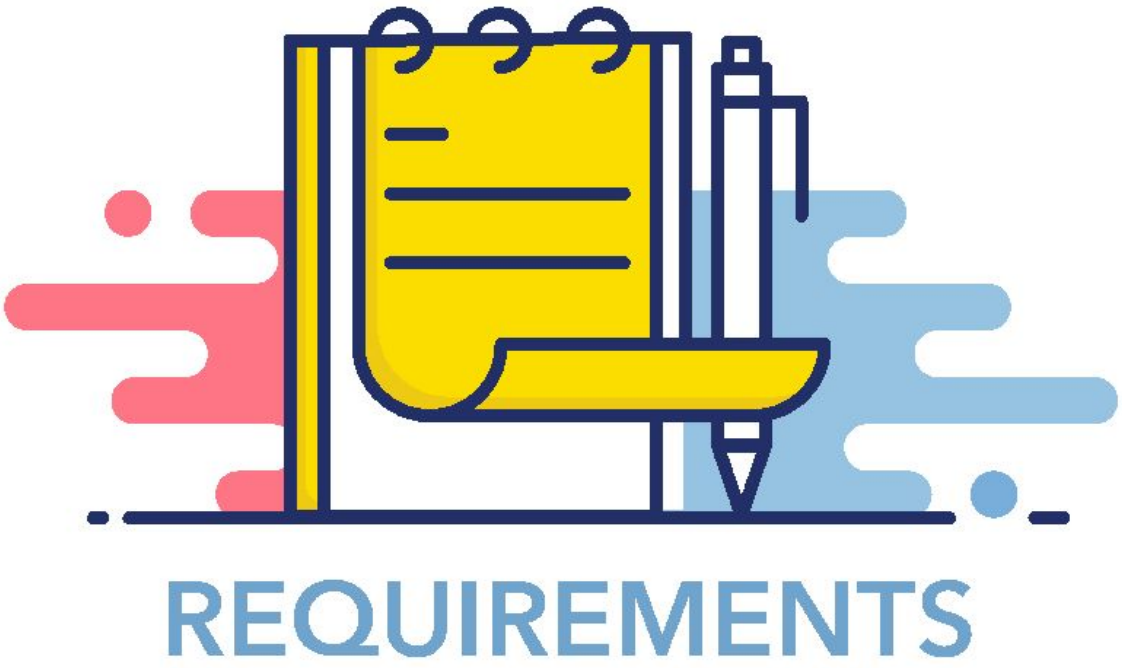
Introduction to UPSTAC application

You were given a new idea for developing a web application & analyze what is the usefulness of the application & who all will it be useful for. You also identified the major users of this application & ways in which they can get benefited from this application.

You understood that from the survey results general public, laboratory testers, and doctors were not happy with the current process for registering and performing a laboratory test request along with the diagnosis of diseases. The whole sense was the need for a unified platform where all the stakeholders can collaborate and make the process simple and easy. The UPSTAC application is built with such an idea catering a unified single platform to the general public, testers, doctors and the government authorities where the public can request for laboratory tests, the testers can view and assign these tests to themselves and the doctors will be able to view the test results and consult the patients online.

UPSTAC Application - Requirements Identification and workflow

The requirement identification phase is the first phase of the Software Development Life Cycle and is also termed as requirement gathering or requirement elicitation. This is the process of generating a list of requirements (functional, system, technical, etc.) from the various stakeholders (customers, users, vendors, IT staff, etc.) that will be used as the basis for the formal requirements document. The process is not as straightforward as just directly asking the stakeholders about the requirements but involves various meetings and is an exploratory process of researching and documenting project requirements. One of the major reasons behind most failed projects in software development is lack of proper requirement gathering. Setting the scope, deadlines and overall cost of the entire project depends on the requirement gathering and if properly done can save time, effort and money to the company.



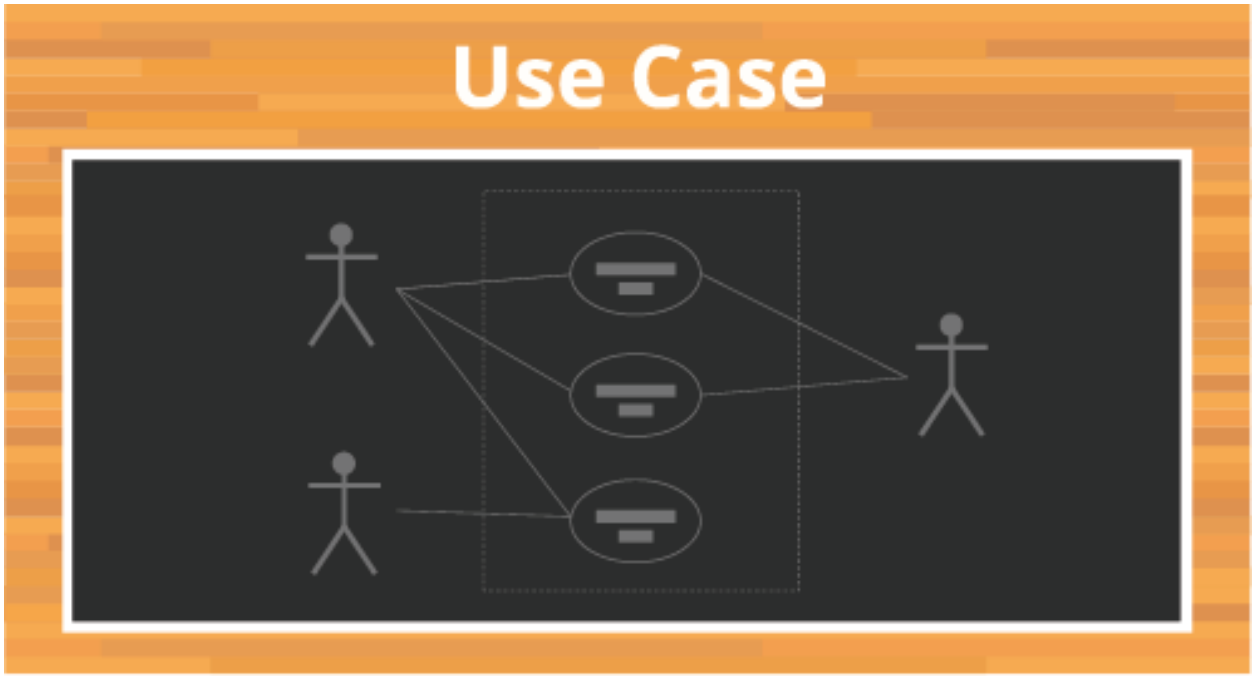
For eg, in the UPSTAC application, if the requirement for the tester accepting the test request raised by the patient is missed and if the application is developed with the feature wherein the tester makes another separate test request from his end, this will break the flow of the application. We will lose track of the request raised by the patients and this will result in re-writing the code again. One small change will cost re-writing the majority of the code since the main workflow of the application gets affected. Hence requirement gathering is a very major phase in the web development life cycle.

The major requirements of the UPSTAC application are :

- Once a patient requests for the test, the medical staff will come into the picture and they will visit the localities to measure critical health parameters such as body temp, heart rate, etc along with the health samples to be sent to the testing facility.
- Once the important parameters are recorded, these would be updated accordingly by the testers through the application itself. The testers will then go back to their labs and perform the lab tests on the sample and deduce if the person is tested positive or negative. The testers can then accordingly upload the same information on the app. These test results can be viewed by the requested patients.
- Similarly, on the doctors' end, a doctor can see the health parameters of the patient and also the test results and accordingly suggest to the patient if he/she is in need for immediate hospitalization or quarantine or can be considered safe. For example, a patient who might have very extreme health parameters such as high temperature & fluctuating breathing may be classified as "In need for immediate hospitalization".
- Finally, the health department should be able to view the details of all patients, ensuring that the required patients are immediately hospitalized. A classification algorithm may also be run to identify the lockdown/unlock protocols of the localities.

Use case diagrams

A Use Case Diagram is the primary form of requirements for a new software program under development. It is the pictorial representation of the relationship between the actors/roles of the application and does not define each and every step which the user might undertake inside an application but rather gives an overview of the high-level functionalities of how the user will interact with the application. A **use case diagram** at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. It helps us deduce the different users and their different use cases which can be further documented separately. It is a graphical representation of our requirements depicted using the users and their respective actions in a flow chart kind of format. This is a mind map of the entire application



The need for use case diagrams is important in a software development project, since this helps the developer not miss any requirement or feature to be developed and understand the whole flow even if he/she misses something while reading the software requirements document. The use case diagram also helps us reduce cost, avoid wastage in time and efforts by identifying the flaws in the flow of the application even before development. Because of this early feedback, the user/client can research and suggest any changes to the workflow and helps us avoid redeveloping the content. Not just identifying the flaws of the current workflow, but also helps us in refining the current requirements and identify possible feature additions before the development stage itself.

Some of the advantages of the use case diagrams are :

- Simple and Lucid to understand
- Avoids the usage of technical jargons
- Provides clear and concise version of the requirement document
- Not as lengthy as requirement document
- Contains no details about the internal working of the application making it easy to understand for the end-user
- Enlists the steps in order to achieve the end objectives

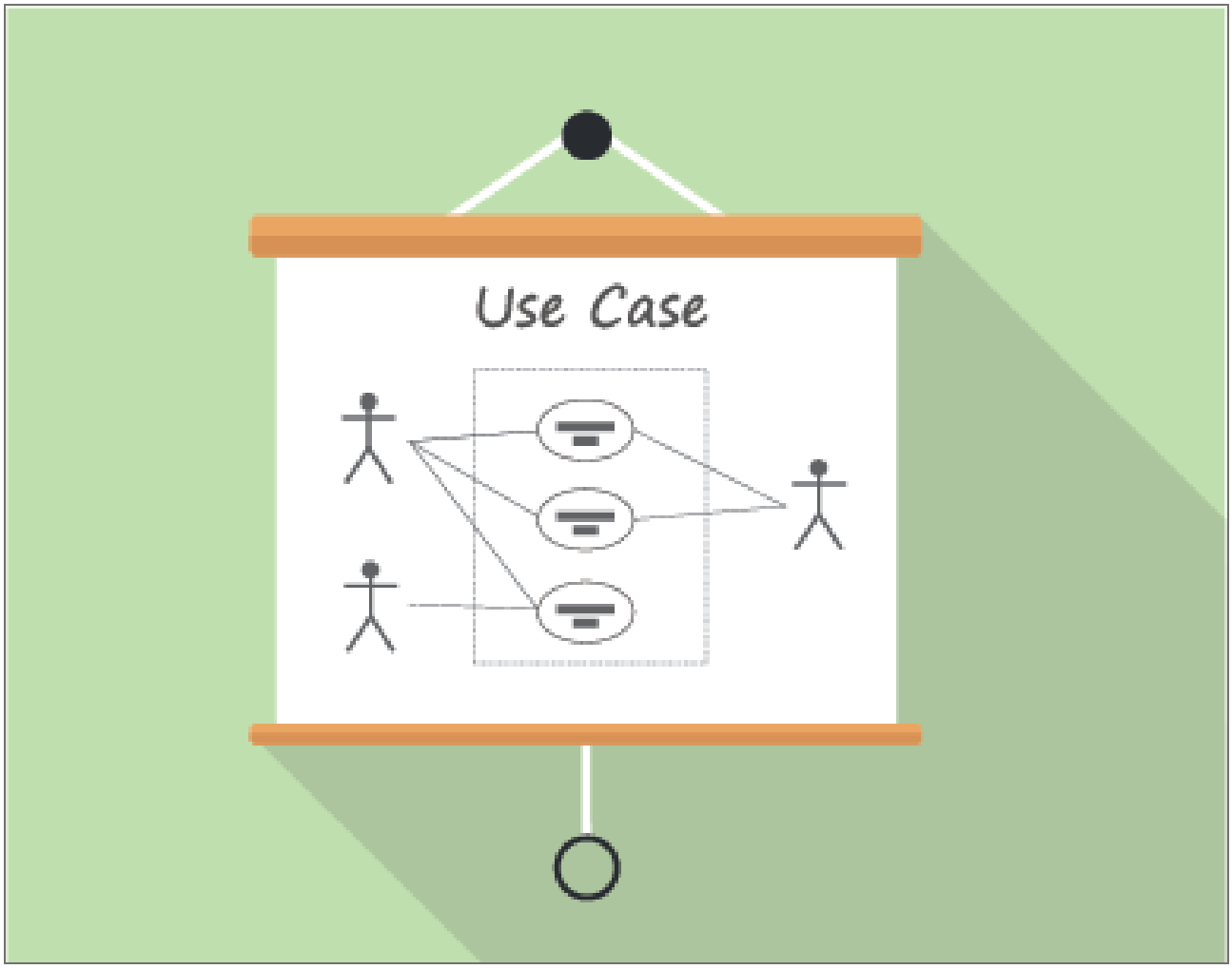
UPSTAC Application - Identification of the users and their use cases

Identifying users/actors are one of the first steps in the use case analysis. Every external entity interacting with our application can be identified as an actor. It can be a role played by a user or any other system that interacts with our application. Actors can represent roles played by human users, external hardware, or other subjects. A use case represents a function that the system performs. Alternatively, a use case can be thought of as a goal that some actor can achieve with the system.

There are mainly four major actors / players in the UPSTAC application :

1. Patient / User
2. Laboratory Tester
3. Doctor
4. Government Authority

From the use case diagram, the use cases for the above identified users can be easily figured out.



UPSTAC Application - Preliminary Object Oriented Design

Software design is the process by which an agent creates a specification of a software artifact, intended to accomplish goals, using a set of primitive components and subject to constraints. This includes both low level and high level components and algorithm design and architecture design. Object Oriented design is one of the most popular software design models. Object-oriented design is the process of planning a system of interacting objects for the purpose of solving a software problem.

There are mainly three preliminary components to our UPSTAC application.

1. **Backend Layer** - The back-end layer is where our web servers are and it forms the heart of the application.The business logic of the application is written in this layer.The code that we write in the back end is run on something called a webserver. Java is used as the backend language of the UPSTAC application.

2. **Frontend Layer** - The front-end layer is what is seen by the userThe user interacts with the application through this layer. The front end tools to be used to develop the UPSTAC application are react and javascript.
3. **Database Layer** - Database layer is where we store our persistent data.

The main step in designing the backend layer of the application involves identification of backend classes and the respective tables. Identification of tables and designing how to store the data is the next step. We make use of Tables to store, maintain and retrieve the relevant data. Identify the data that needs to be stored. Design tables according to the data to be stored. Attributes of the classes will form the base for defining the columns in a table. This whole process is termed as Database Design. The tables to be used in the UPSTAC application are :

show columns from user;

FIELD	TYPE	NULL	KEY
ID	BIGINT(19)	NO	PRI
CREATED	TIMESTAMP(26)	YES	
EMAIL	VARCHAR(255)	YES	UNI
FIRST_NAME	VARCHAR(255)	YES	
LAST_NAME	VARCHAR(255)	YES	
PASSWORD	VARCHAR(255)	YES	
PHONE_NUMBER	VARCHAR(255)	YES	UNI
PIN_CODE	INTEGER(10)	YES	
STATUS	INTEGER(10)	YES	
UPDATED	TIMESTAMP(26)	YES	
USER_NAME	VARCHAR(255)	YES	UNI

show columns from role;

FIELD	TYPE	NULL	KEY
ID	BIGINT(19)	NO	PRI
DESCRIPTION	VARCHAR(255)	YES	
NAME	VARCHAR(255)	YES	UNI

show columns from DOCUMENT ;

FIELD	TYPE	NULL	KEY
ID	BIGINT(19)	NO	PRI
CONTENT_TYPE	VARCHAR(255)	YES	
FILE_NAME	VARCHAR(255)	YES	
SIZE	BIGINT(19)	NO	
URL	VARCHAR(255)	YES	
USER_ID	BIGINT(19)	YES	

show columns from TEST_REQUEST ;

FIELD	TYPE	NULL	KEY
REQUEST_ID	BIGINT(19)	NO	PRI
AGE	INTEGER(10)	YES	
CREATED	DATE(10)	YES	
EMAIL	VARCHAR(255)	YES	
NAME	VARCHAR(255)	YES	
PHONE_NUMBER	VARCHAR(255)	YES	
PIN_CODE	INTEGER(10)	YES	
STATUS	INTEGER(10)	YES	
CREATED_BY_ID	BIGINT(19)	YES	

show columns from TEST_REQUEST_FLOW ;

FIELD	TYPE	NULL	KEY
ID	BIGINT(19)	NO	PRI
FROM_STATUS	INTEGER(10)	YES	
HAPPENED_ON	DATE(10)	YES	
TO_STATUS	INTEGER(10)	YES	
CHANGED_BY_ID	BIGINT(19)	YES	
REQUEST_REQUEST_ID	BIGINT(19)	YES	

show columns from LAB_RESULT ;

FIELD	TYPE	NULL	KEY
RESULT_ID	BIGINT(19)	NO	PRI
BLOOD_PRESSURE	VARCHAR(255)	YES	
COMMENTS	VARCHAR(255)	YES	
HEART_BEAT	VARCHAR(255)	YES	
OXYGEN_LEVEL	VARCHAR(255)	YES	
RESULT	INTEGER(10)	YES	
TEMPERATURE	VARCHAR(255)	YES	
REQUEST_REQUEST_ID	BIGINT(19)	YES	
TESTER_ID	BIGINT(19)	YES	

show columns from CONSULTATION ;

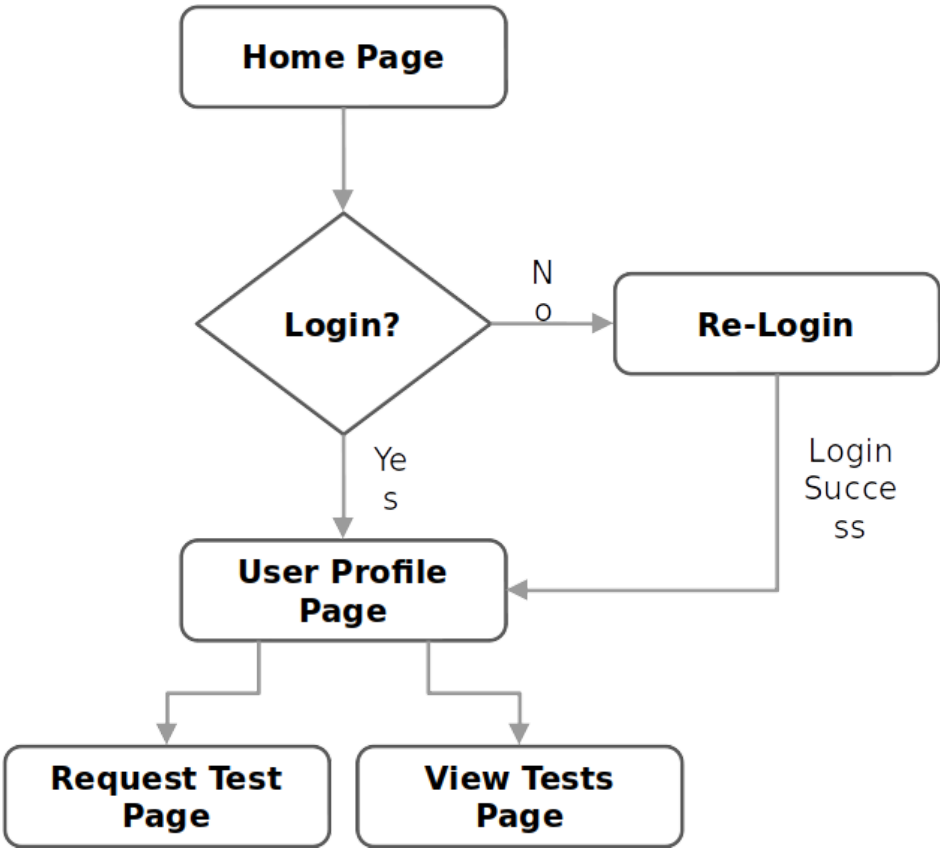
FIELD	TYPE	NULL	KEY
RESULT_ID	BIGINT(19)	NO	PRI
COMMENTS	VARCHAR(255)	YES	
SUGGESTION	INTEGER(10)	YES	
DOCTOR_ID	BIGINT(19)	YES	
REQUEST_REQUEST_ID	BIGINT(19)	YES	

show columns from COVID_COUNT_THRESHOLD

FIELD	TYPE	NULL	KEY
THRESHOLD_TYPE	INTEGER(10)	NO	PRI
MAX_LIMIT	INTEGER(10)	NO	

Control Flow Diagrams are diagrams to describe the control flow of a business process. It helps us identify the beginning and end of a process and the possible ways where it can branch off in another direction. Since this is flow graph oriented, it depicts all the paths that can be traversed during a program execution.

The control flow diagram of the patient/user actor is as follows :



Disclaimer: All content and material on the upGrad website is copyrighted material, either belonging to upGrad or its bonafide contributors and is purely for the dissemination of education. You are permitted to access, print and download extracts from this site purely for your own education only and on the following basis:

- You can download this document from the website for self-use only.
- Any copies of this document, in part or full, saved to disk or to any other storage medium may only be used for subsequent, self-viewing purposes or to print an individual extract or copy for non-commercial personal use only.
- Any further dissemination, distribution, reproduction, copying of the content of the document herein or the uploading thereof on other websites or use of content for any other commercial/unauthorised purposes in any way which could infringe the intellectual property rights of upGrad or its contributors, is strictly prohibited.
- No graphics, images or photographs from any accompanying text in this document will be used separately for unauthorised purposes.
- No material in this document will be modified, adapted or altered in any way.
- No part of this document or upGrad content may be reproduced or stored in any other web site or included in any public or private electronic retrieval system or service without upGrad's prior written permission.
- Any rights not expressly granted in these terms are reserved.