

**ADDIS ABABA UNIVERSITY**

**ADDIS ABABA INSTITUTE OF TECHNOLOGY**

**CENTER OF INFORMATION TECHNOLOGY AND SCIENTIFIC COMPUTING**

**DEPARTMENT OF SOFTWARE ENGINEERING**

**SOFTWARE ENGINEERING II**

**Individual Assignment I Documentation**

Name: ANDUAMLAK TEMESGEN

ID: ATE/9658/08

Section: 1 (day time extension)

**ADVISOR**: Mr. Natnael **Date: March 24 2018 G.C**

**INTRODUCTION**

The main purpose of this document is to explain the applications purpose and the architectures that these two systems are designed.

**Application I**

The first application I submitted for you is called “Ethio Tele Solution”

**Objective of the application**

The main goal of this application is to send a picture for Ethiopian telecommunication center when everyone see an accident on Ethio tele wires, pols, and soon.

When someone looks this accident he/she can take a picture by him/her phone and sent that picture for the company.

I used a **client – server architecture** for develop this application, and the system is designed by using HTML5 for front end and PHP for back end purpose.

**Why Client – SERVER Architecture?**

A Client-Server network is designed for end-users, called Clients, to access resources such as files or some other services from a central computer called a Server.

The main reason to choose this architecture is to access the file that stored on the server or database by end-users easily, and also to retrieve that files.

Since the system store a file or picture that sent by users and retrieve that file or picture by administrators I thought Client-Serve architecture is better and easy for this system and I used it.

**Application II**

The second application I submitted for you is called “Registration System”

**Objective of the application**

The main goal of this application is to register any users by their android phone and allows to access the system.

I used a **MVC** **architecture** for develop this application, and the system is designed by using android studio.

**Why MVC Architecture?**

The model-view-controller pattern proposes three main components or objects to be used in software development:

A **Model**, which represents the underlying, logical structure of data in a software application and the high-level class associated with it. This object model does not contain any information about the user interface.

A **View**, which is a collection of classes representing the elements in the user interface (all of the things the user can see and respond to on the screen, such as buttons, display boxes, and so forth)

A **Controller**, which represents the classes connecting the model and the view, and is used to communicate between classes in the model and view.

The main reason to choose this architecture is when someone try’s to access the system he/she enter him/her username and password by using the view component, and the controller takes username and password of the user and compares with the username and password of the user that stored in the model component, and is that the same the user can login successfully, but not same the user can not login but he/she can register as a new users.