**Automated Email Generation.**

**Chapter 1:**

Introduction

The Scope of this system allow the administration to generate automatically email to all university student and university faculty staff member for any notification in case of any pandemic condition or any other notification regarded to university or country. Every student and faculty member inform by email for any notification from university and also report share on key performance.

## Background

The background of this Web application is reduce human work or efforts in system whenever university wants to deliver some message or any kind of news to students or faculty member university face many difficulties by this only by some click university deliver its message to every stakeholder. University also share report on key performance.

## Motivations and Challenges

In this Web Application we will reduce human efforts. We will maintain a strong system which automatically generate email according university calendar. Information is already being stored in calendar and message is auto generated also university can share report on key performance.

## Goals and Objectives

We are going to generate best automatically email generated system using university calendar also report sharing on key performance. It will be efficient to use and less time consuming. Team work is a key to success, so we work as a team and we will develop this system in given duration of time.

## Literature Review/Existing Solutions

[Paragraph Text 12 pt, Calibri, 1.5 Line Spacing, Justified]

## Gap Analysis

Sometimes university face problem to send email to every student and faculty staff individually and university also face problem to share report on key performance university ma face too much problem in this and also waste lot of time in this.

## Proposed Solution

We will purposed automatically email generated system using university calendar also report sharing on key performance. Email is generated automatically according to university calendar.

Report also share automatically on key performance.

## Project Plan

In this we write about our system. We are going to make a form through which faculty gives feedback after every month.

* Requirement gathering.

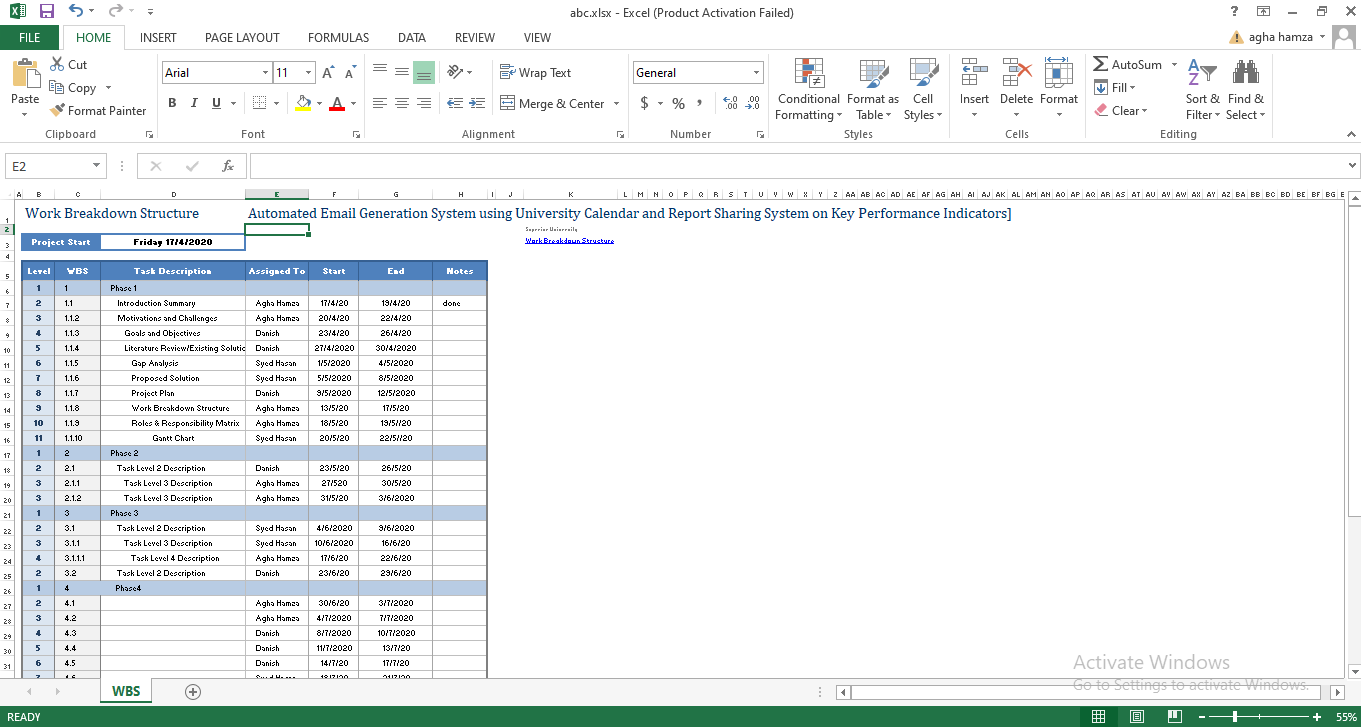
↓

Home

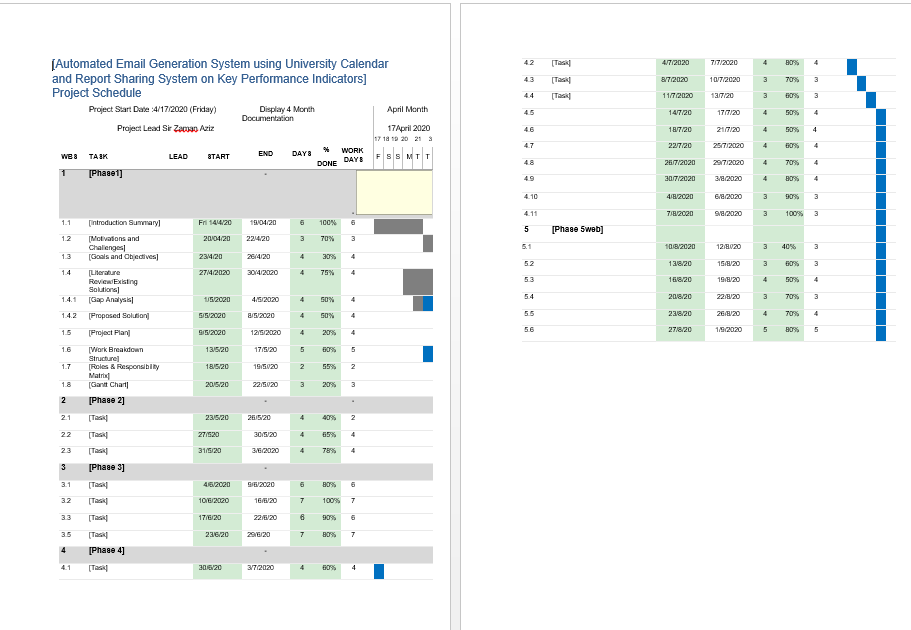
Etc.

* System Design.
* Implementation
* Testing

## Work Breakdown Structure



## Gantt Chart



## Chapter 2

## Software Requirement Specifications

**Chapter 2:** Software Requirement Specifications



## Introduction

## 

## Purpose

The purpose of SRS document is to provide a detailed overview of our software product also its parameters and goals. This document provides internal working of software how its work or interact with users of the system. This document describes the projects target audience and its user interface, hardware and software requirements. SRS document helps any designer and developer to assist in software delivery lifecycle (SDLC) processes.

Moreover, this document also has software system constraints, interface and interactions with other external applications and data maintain in database or flow of data.

## Document Conventions

The font style used in this documentation is **Calibri** and the font size is **12.**

## 

## Intended Audience and Reading Suggestions

The intended audience of this document includes:

**Project Manager**

* Which use SRS to evaluate the project in all phases of SDLC to evaluate to check project progress going according to Software System requirements or not.
* **Designer**
* To make appropriate design flow of the product according to SRS document and architecture of the software system. The software system meets its design according to this SRS document
* **Project Testers**
  + Will use this document for testing the modules of this software system and for testing strategies as some errors are easier to find using a requirement document.
* **Developers**
  + Can review the project requirements, its functionalities and try to understand the design and improve its features and functions. It can also act as guidelines for future used.

## Product Scope

Email automation is a way to create emails that reach the right people with the right message at the right moment without doing the work every time. The scope of our project is very useful. We can send mails students, teacher, faculty and workers at anytime and anywhere. Just we set the date and time it will automatically send to those whom we selected. We can use this in offices, factories anywhere to tell them any important announcement, holiday etc. Now a day’s usage of emails is very common, so Automated Email Generation System is very useful in every field.

## Overall Description

## 

## Product Perspective

The system allow the administration to generate automatically email to all university student and university faculty staff member for any notification in case of any pandemic condition or any other notification regarded to university or country. Every student and faculty member inform by email for any notification from university and also report share on key performance.

## 

## Product Functions

* The application must allow it’s users to let themselves register and login, logout.
* Admin should set the date and time.
* Admin type a mail and send it to the server.
* Further server will manage the mail to send students, teachers.

## 

## Operating Environment

It is Web-Base application. It will run on windows, mac-book. We use JavaScript, noteJS for making this application.

## 

## Design and Implementation Constraints

Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).

## 

## User Documentation

It is very easy to use its base on the email so anyone can easily use this application but we attach the tutorial how to use it.

## 

## Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

## External Interface Requirements

## 

## User Interfaces

## The list below shows the user interfaces that are comprised in Web Pages respectively.

**Screens for website**

* Sign in
* Login
* Logout
* Select participant
* Set calendar
* Set mail
* Mail to server
* Open Mail form
* Check Delivery reports

## 

## Software Interfaces

**Operating System** Windows / MAC book

**User / Web Interface** HTML/CSS/JavaScript/Node js.

## 

## Communications Interfaces

It is a communication type project, in which admin send mail to others to share new notification. It communicate through emails. Admin can send mails to each student and faculty member.

## 

## System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## 

## System Feature 1

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

## Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

## Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

## Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-SF1-1: <Write your requirement here>

REQ-SF1-2:

REQ-SF1-3:

## 

## System Feature 2

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

## Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

## Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

## Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-SF2-1:

REQ-SF2-2:

REQ-SF2-3:

## 

## System Feature 3 (and so on)

## Other Nonfunctional Requirements

## Performance Requirements

Performance of system is also high as it will response to user request within seconds. The system will not take time on performing any task.

## Security Requirements

Regular checks would be applied to the data that would make sure that data is clean in terms of its reliability. Data can be modified on regular basis and progress can also be updated.

## Software Quality Attributes

The software will be easy to use for people because of the user friendly interface and the data of user can easily store in it they can easily access and generate mail on the server.

## Business Rules

The business rules are described briefly in the functional requirements section, all the users like Schools, Colleges, Universities, Offices etc. employees register their and operate the server and select date and time through calendar and send mails.

## Chapter 3

## Use Case Analysis

**Chapter 3:**

System Analysis

In software and systems engineering, a use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language (UML) as an actor) and a system to achieve a goal. The actor can be a human or other external system.

## Use Case Model



## Fully Dressed Use Cases

