



SOFTWARE DESIGN SPECIFICATION

JAVA PROGRAMMING   
Client-Server Based UTM Onboarding System

|  |  |  |  |
| --- | --- | --- | --- |
| **Created By:** | Chaitanya Katkari | **Approved By:** | <Domain Lead Name> |
| **Created On:** | 27/07/2025 | **Approved On:** | DD-MMM-YYYY |

Page left blank intentionally

**INDEX**

[**1** **PURPOSE** 2](#_Toc142418236)

[**2** **PROJECT SCOPE** 2](#_Toc142418237)

[**3** **SYSTEM OVERVIEW** 2](#_Toc142418238)

[**4** **DESIGN CONSIDERATIONS** 2](#_Toc142418239)

[4.1 Requirements 3](#_Toc142418240)

[4.2 Assumptions 3](#_Toc142418241)

[4.3 Dependencies 3](#_Toc142418242)

[**5** **SYSTEM ARCHITECTURE** 3](#_Toc142418243)

[5.1 Architectural Strategies 4](#_Toc142418244)

[5.2 Structure & Relationships 4](#_Toc142418245)

[**6** **DETAILED DESCRIPTION OF COMPONENTS** 4](#_Toc142418246)

[**7** **INTEGRATION** 5](#_Toc142418247)

[**8** **APPENDICES** 1](#_Toc142418248)

[8.1 Appendix A – Detailed Description of Components 1](#_Toc142418249)

**General Instructions for using the Live Project POC Document**

* This template and the subsequent document created using this template is a confidential document and is the intellectual property of Cloud Counselage Pvt. Ltd. Circulating it outside of the organisation without the consent of Cloud Counselage Pvt. Ltd. is the breach of company policies and will lead to legal actions
* The Design Specification of a software forms the basis of development of software
* The **text between inequality (< >) is to be replaced** by relevant text
* Please **remove the yellow highlight on the Text** between the inequality (< >). This is done to help you notice the text to be changed/replaced
* The text in *italics* highlighted in grey is just for reference and should be removed after adding the relevant text

# **PURPOSE**

This document is created based on the requirement specification document. The purpose of this Software Design Specification (SDS) Document is to break down the project into components to describe in detail what the purpose of each component is and how it will be implemented. The SDS will also serve as a tool for verification and validation of the final product.

# **PROJECT SCOPE**

The scope of the Client-Server Based UTM Onboarding System includes its distinct features, its benefits, and its limitations. The system's distinct features allow it to develop a Java-based client-server application that automates the Community Ambassador (CA) onboarding process by generating personalized UTM links and delivering them via email to new joiners, enabling tracking and performance monitoring for the Industry Academia Community (IAC) by using Java, JavaMail API, Gmail SMTP Server, Terminal, VS code. The system enables the user to automatically generating a unique UTM link and sending it via email as soon as a CA submits form.

# **SYSTEM OVERVIEW**

This section will provide an outline of the various components and subsystems of Client-Server Based UTM Onboarding System

The system is a Java-based client-server application designed to automate the onboarding of Community Ambassadors (CAs). When a new CA submits their name and email, the client sends this data to the server. The server then generates a unique UTM link based on the name and sends a personalized welcome email to the provided email address using the JavaMail API. This eliminates manual effort, ensures timely communication, and helps track individual CA performance through their UTM links.

# **DESIGN CONSIDERATIONS**

This section describes requirements, assumptions and dependencies to be addressed to devise a complete design solution.

## Requirements

The list of components  
Client Module –

Takes user input (name and email) and sends it to the server.

Server Module –

Receives data, generates UTM link, and triggers email sending.

EmailSender Module –

Uses JavaMail API to send welcome emails with UTM link.

UTM Link Generator –

Creates a unique link by formatting the user's name.

Gmail SMTP Setup –

Handles email delivery through Gmail’s SMTP server.

## Assumptions

The list of components  
Users provide a valid email address while filling the form.

The server and client are running on the same local network or machine (localhost).

The Gmail account used has App Password enabled and less secure app access configured.

The JavaMail API is properly set up and included in the project dependencies.

## Dependencies

The list of components  
Java Development Kit (JDK 8 or higher)

Required to compile and run the Java application.

JavaMail API

Used to send emails from the server to the Community Ambassador.

Gmail SMTP Server

Handles the actual email transmission (smtp.gmail.com, port 587 with TLS).

Internet Connection

Needed for the server to connect to Gmail SMTP and send the email.

App Password (Gmail)

A 16-digit app-specific password to authenticate securely without exposing the main Gmail password.

# **SYSTEM ARCHITECTURE**

The software system architecture refers to the logical organization of a distributed system into software components. It defines how components of a software system are assembled, their relationship and communication between them. It serves as a blueprint for software application and development basis for developer team. An effective architecture serves as the conceptual glue that holds every phase of the project together for all of its stakeholders, enabling agility, time and cost savings, and early identification of design risks.

The Software architecture:

* Defines structure of a system
* Defines behaviour of a system
* Defines component relationship
* Defines communication structure
* Balances stakeholder’s needs
* Influences team structure
* Focuses on significant elements
* Captures early design decisions

Below some important characteristics which are commonly considered are explained.

**Operational Architecture Characteristics:**

* Availability
* Performance
* Reliability
* Low fault tolerance
* Scalability

**Structural Architecture Characteristics:**

* Configurability
* Extensibility
* Supportability
* Portability
* Maintainability

**Cross-Cutting Architecture Characteristics:**

* Accessibility
* Security
* Usability
* Privacy
* Feasibility

## Architectural Strategies

## Client Module – Takes user input and sends to server via socket.

## Server Module – Receives data, creates UTM link, and calls email module.

## Email Module – Sends welcome email using JavaMail and Gmail SMTP.

## Networking Layer – Manages communication between client and server on port 5000.

Security Layer – Uses Gmail App Password for secure authentication

## Structure & Relationships

[User Input] --> [Client Module] --> [Server Module] --> [Generate UTM Link] --> [Email Module] --> [Send Email] --> [User Receives Email]

# **DETAILED DESCRIPTION OF COMPONENTS**

For detailed description of the components, please refer **Appendix A – Detailed Description of Components**

The below template will be used to specify the details of all the components

**Table 1: Detailed Design Specification Template**

|  |  |
| --- | --- |
| **Identification** | The unique name for the component and the location of the component in the system. |
| **Type** | A module, a subprogram, a form, a data file, a control procedure, a class, etc. |
| **Purpose** | Function and performance requirements implemented by the design component, including derived requirements. Derived requirements are not explicitly stated in the SRS - but are implied or adjunct to formally stated SDS requirements. |
| **Subordinates** | The internal structure of the component, the constituents of the component, and the functional requirements satisfied by each part. |
| **Dependencies** | How the component’s function and performance relate to other components. How this component is used by other components. The other components that use this component. Interaction details such as timing, interaction conditions (such as order of execution and data sharing), and responsibility for creation, duplication, use, storage, and elimination of components. |
| **Interfaces** | Detailed description of all external or internal interfaces as well as of any mechanism for communicating through messages, parameters, or common data areas. All error messages and error codes should be identified. All screen formats, interactive messages, and other user interface components (originally defined in the SRS) should be given here. |
| **Resources** | A complete description of all resources (hardware or software) external to the component but required to carry out its functions. |
| **Processing** | A full description of the functions presented in the Function subsection. Pseudocode can be used to document algorithms, equations, and logic. |
| **Data** | For the data internal to the component, describes the representation method, initial values, use, semantics, and format. |

# **INTEGRATIONS**

JavaMail API – Used to send welcome emails with UTM links.

Gmail SMTP Server – Sends emails securely using TLS and app password.

Java Socket Programming – Enables client-server communication for data transfer.

Scanner & I/O Streams – Captures input and sends data between client and server.

Email Clients – Recipients receive UTM links via their email apps (like Gmail, Outlook).

# **APPENDICES**

## Appendix A – Detailed Description of Components

|  |  |
| --- | --- |
| **Identification** | **Client.java** |
| **Type** | Class |
| **Purpose** | Accepts user input (name and email) and sends it to the server via socket connection. |
| **Subordinates** | Sends data to Server.java |
| **Dependencies** | Java Socket API  Scanner for input |
| **Interfaces** | Terminal-based user input |
| **Resources** | Requires connection to the server running on port 5000 |
| **Processing** | Reads user input and sends it through output stream |
| **Data** | User's name and email |

|  |  |
| --- | --- |
| **Identification** | **Server.java** |
| **Type** | Class |
| **Purpose** | Receives user data from client, generates UTM link, and triggers email sending. |
| **Subordinates** | Calls EmailSender.java |
| **Dependencies** | Java ServerSocket, Socket, and IO APIs |
| **Interfaces** | Accepts socket connections |
| **Resources** | Runs on port 5000, requires Gmail credentials for sending mail |
| **Processing** | Reads client data, builds UTM link, calls email function |
| **Data** | User's name, email, and generated UTM link |

|  |  |
| --- | --- |
| **Identification** | **EmailSender.java** |
| **Type** | Class |
| **Purpose** | Sends welcome email with UTM link to the new Community Ambassador |
| **Subordinates** | None |
| **Dependencies** | JavaMail API |
| **Interfaces** | Interacts with Gmail SMTP server |
| **Resources** | Requires internet access and valid Gmail credentials |
| **Processing** | Creates and sends email using Session, Message, and Transport |
| **Data** | Recipient’s email, name, and UTM link. |