**SOFTWARE DESIGN SPECIFICATION**

**FOR**

**E**-**V**oting **S**ystem

Prepared By :

Taha Asif (15k-2855)

Shahiq Qureshi (15k-2822)

Muhammad Ibrahim (15k-2131)

**VERSION: REVISION – 1.0**

**April 23rd, 2018**

**REVISION CHART**

*.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Version** |  | **Primary Author(s)** |  | **Description of Version** | **Date** |  |
|  |  |  |  |  | **Completed** |  |
| Draft | Taha Asif | |  | Initial draft created for distribution and | 23/04/2018 |  |
|  |  | Shahiq Qureshi |  | review comments |  |  |
|  |  |  |  |  |  |
|  |  | Muhammad Ibrahim |  |  |  |  |
|  |  | |  |  |  |  |
| Revision – 1.0 | Taha Asif | |  | First Release | 23/04/2018 |  |
|  |  | Muhammad Ibrahim |  |  |  |  |
|  |  | Shahiq Qureshi |  |  |  |  |
|  |  | |  |  |  |  |
|  |  |  |  |  |  |  |

**CONTENTS**

Contents

[1 INTRODUCTION 5](#_Toc512158360)

[1.1 Purpose 5](#_Toc512158361)

[1.2 Scope 5](#_Toc512158362)

[1.3 Objective 5](#_Toc512158363)

[2 SYSTEM OVERVIEW 6](#_Toc512158364)

[2.1 Product Perspective 6](#_Toc512158365)

[2.1.1 Design Method 6](#_Toc512158366)

[2.1.1 User Interfaces 6](#_Toc512158367)

[2.1.2 Hardware Interfaces 6](#_Toc512158368)

[2.1.3 Software Interfaces 6](#_Toc512158369)

[2.1.4 Memory Constraints 6](#_Toc512158370)

[2.2 Product Functions 7](#_Toc512158371)

[2.3 User Characteristics 7](#_Toc512158372)

[2.4 Constraints 7](#_Toc512158373)

[3 DESIGN CONSIDERATIONS 8](#_Toc512158374)

[3.1 Operating Environment 8](#_Toc512158375)

[3.2 Fault Tolerant Design 8](#_Toc512158376)

[3.3 Design Conventions 8](#_Toc512158377)

[3.4 Architectural Design 8](#_Toc512158378)

[3.5 User Interface: 8](#_Toc512158379)

[4 SYSTEM ARCHITECTURE 9](#_Toc512158380)

[4.1 View of E-Voting System Classes 9](#_Toc512158381)

[**4.2** Individual Classes of System 10](#_Toc512158382)

[4.2.1 Admin: 10](#_Toc512158383)

[4.2.1.1 Attributes of Admin class 10](#_Toc512158384)

[4.2.1.2 Functions available in Admin Class 10](#_Toc512158385)

[4.2.2 Candidate 10](#_Toc512158386)

[4.2.2.1 Functions available in Candidate Class 10](#_Toc512158387)

[*4.2.2.2.1 SetName ()* 10](#_Toc512158388)

[4.2.2.2.2 SetID () 11](#_Toc512158389)

[4.2.2.2.3 SetProgfilePic () 11](#_Toc512158390)

[4.2.3 User 11](#_Toc512158391)

[4.2.3.1 Functions available in User Class 11](#_Toc512158392)

[4.2.3.1.1 setVote () 11](#_Toc512158393)

[4.2.3.1.2 setCNIC () 11](#_Toc512158394)

[4.2.4 VotingEvent 11](#_Toc512158395)

[4.2.4.1 Functions available in VotingEvent class 11](#_Toc512158396)

[4.2.4.1.1 updateVote () 11](#_Toc512158397)

[4.2.4.1.2 FunctionName HEre ( ) 11](#_Toc512158398)

[4.2.5 MailerSystem 12](#_Toc512158399)

[4.2.5.1 Functions available in MailerSystemClass Class 12](#_Toc512158400)

[4.2.5.2.1 MailALL () 12](#_Toc512158401)

[4.2.5.2.2 MailSend () 12](#_Toc512158402)

[5. FIGURES 13](#_Toc512158403)

[5.1 Use Cases 13](#_Toc512158404)

[5.2 Election Event Upload 14](#_Toc512158405)

[5.2.1 Election Event Upload Sequence (Figure 3) 14](#_Toc512158406)

[5.2.2 Election Event Collaboration (Figure 4) 15](#_Toc512158407)

[5.3 Admin Register Candidates 16](#_Toc512158408)

[**5.3.1** Admin Register Candidates **(Figure 5)** 16](#_Toc512158409)

[5.3.2 Admin Register Candidates Collaboration (Figure 6) 17](#_Toc512158410)

[5.4 Vote Casting 18](#_Toc512158411)

[5.4.1 Vote Casting Sequence (Figure 7) 18](#_Toc512158412)

[5.4.2 Vote Casting Collaboration (Figure 8) 19](#_Toc512158413)

[5.5 Signup 21](#_Toc512158414)

[5.5.1 Signup Sequence (Figure 9) 21](#_Toc512158415)

[5.5.2 Signup Collaboration (Figure 10) 22](#_Toc512158416)

[5.6 Setting Panel Modify 23](#_Toc512158417)

[5.6.1 Setting Panel Modify Sequence (Figure 11) 23](#_Toc512158418)

[5.6.3 Setting Panel Modify Collaboration (Figure 12) 24](#_Toc512158419)

[6 REFERENCES 25](#_Toc512158420)

[6.1 References 25](#_Toc512158421)

# 1 INTRODUCTION

## 1.1 Purpose

*This document will define the design of the E-Voting System. It contains specific information about the process of voting, classes, and functions. The document also contains the Use Cases, Sequence Diagrams and Collaboration diagram for E-Voting System.*

## 1.2 Scope

*This SDS(Software Design Specification) for E-Voting System is to be used by our Teachers and Evaluators as a definition of the design to be used to implement the E-Voting System.*

## 1.3 Objective

*The E-Voting System is developed to replace the paper work in Voting System to more reliable and accurate paperless work i.e. Android Application. Its purpose is to make the voting phenomenon rig free. It records the Voters list, Candidates List, Number of Votes and processes the final result of Voting.*

# 2 SYSTEM OVERVIEW

*.*

## 2.1 Product Perspective

*This product provides an efficient way of Holding elections and Vote Casting. It records the number of planes processed and their average time spent in waiting. It records the Voters list, Candidates List, Number of Votes and processes the final result of Voting.*

## 2.1.1 Design Method

*The design of this product utilizes an object-oriented approach.*

## 2.1.1 User Interfaces

*The user of E-Voting System will be interfacing with the Android Application where User can see the Candidates and select any one of them appropriately. The product allows the user to cast the vote without actually having the responsibility of going to polling station.*

## 2.1.2 Hardware Interfaces

*E-Voting system is capable to run on Android Devices and Its admin panel will be executed on website.*

## 2.1.3 Software Interfaces

*E-Voting System will execute on Android phones having Android Version greater than 4. 2 (jelly beans).*

*.*

## 2.1.4 Memory Constraints

*5MB/10MB Internal Storage space is required.*

## 2.2 Product Functions

*E-Voting System allows Administrator to conduct an election of any type and the Candidates can register with their National Identity Card. This application allows Voters to cast vote and calculates the total votes against each Candidate. It then Displays the Final Result of Election.*

## 2.3 User Characteristics

*The general characteristics of the intended users, include*

* *Candidates must satisfy the requirement of Administrator who conducts the elections.*
* *experience- Must know how to operate an Android Phone and have the basic English Knowledge to handle the application.*
* *Voters will be able to be view the ongoing updates of election.*
* *One Voter will only be allowed to vote once*
* *Voter can check their vote but not change once given.*

## 2.4 Constraints

*This application can only run on Android phones having android version greater than 4.2.*

### 2.4.2 End-user environment

*Administrator, candidate and voter should have network and internet connectivity. They will need to login with their user credentials to be able to use the e-voting system.*

### 2.4.3 Internet

*The internet connection of the user’s Android Phone to be able to use the e-voting online Application.*

# 

# 3 DESIGN CONSIDERATIONS

## 3.1 Operating Environment

*The E-Voting System is intended to be operated on Android Phones*.

## 3.2 Fault Tolerant Design

*Application errors will be handled by common fault detection services (e.g. common JAVA exception handling, and error checking on task processing).*

## 3.3 Design Conventions

*The E-Voting Systemstrictly follows the Object-Oriented methodology.*

## 3.4 Architectural Design

*There are no special architectural designs to be followed by E-Voting System. Its basic requirement is just Android Devices having Android version greater than 4.2.*

## 3.5 User Interface:

*User Interface is first provided with the Login/Signup Panel.*

*User can then Select Candidate from the provided options.*

*Administrator will be provided with the web support to conduct the Elections. On the*

*Web, Administrator will be provided with the interface to start the election period, set its constraints*

# 4 SYSTEM ARCHITECTURE

## 4.1 View of E-Voting System Classes



## **4.2** Individual Classes of System

In this part the main classes, their attributes and their respective important functions are highlighted.

### 4.2.1 Admin:

*The admin class is where the data about the information of administrators of E-Voting System is kept. Admin class shares very crucial information with respect to security.*

### 4.2.1.1 Attributes of Admin class

#### 4.2.1.1.1 Email

*The Email is a string type variable that holds the email as primary key of administrators*

*.*

#### 4.2.1.1.2 Password

*The Password field holds the encrypted password for the administrators.*

#### 4.2.1.1.3 Full\_Name

*The Full\_Name field holds the name for the administrator in order to point him with that name at various places on Website. This will provide easiness for admin to check whether he is logged in or not.*

### 4.2.1.2 Functions available in Admin Class

4.2.1.2.1 set\_Email()

*Function: Sets the email to desired email and the email must be unique and saves the email in Database.*

*.*

4.2.1.2.2 set\_Password( )

*Function: Sets password of the Admin and saves it to the database after encryption.*

4.2.1.2.3 set\_Fullname()

*Function: Sets the Fullname to desired fullname field in the database.*

*.*

4.2.1.2.4 get\_Password( )

*Function: Returns the password.*

4.2.1.2.5 get\_email( )

*Function: Returns the email.*

4.2.1.2.6 get\_Fullname( )

*Function: Returns the name of the admin.*

# 4.2.2 Candidate

*The candidate class is for the representation of Candidates who will compete in an election. This candidate class holds the details about the Candidate like his Name, Email, Address, Picture and etc.*

## 4.2.2.1 Functions available in Candidate Class

### 4.2.2.2.1 SetName()

*Function: Sets the User Name of the Candidate and updates it to the database. It is the duty of admin that he must enter candidate name correctly*

### 4.2.2.2.2 SetID()

*Function: Automatically sets the ID of the Candidate at the time of the registration to the next available ID and saves it to the Database. Every candidate is assigned a unique id.*

### 4.2.2.2.3 SetProfilePic()

*Function: Allows the admin to set candidates profile picture name only so that at android (client side) picture of candidate can be fetch as well. It is the admin duty that he must put candidate picture in images folder and put the name of image in database in order to get candidate picture at client side.*

*.*

### 4.2.2.2.4 SetElectionID()

*Function: Since each candidate belongs to a particular election event. Therefore, this function sets election id with candidate id. This will help in referencing the candidate standing in particular election.*

### 4.2.2.2.1 GetName()

*Function: Returns the name of candidate. It depends upon the object that called this function. Whenever the object is made, it stores candidate information. Therefore, this function is associated with candidates object and returns name of the object bind with candidate name.*

### 4.2.2.2.2 GetID()

*Function: Returns the id of candidate. It depends upon the object that called this function. Whenever the object is made, it stores candidate information. Therefore, this function is associated with candidates object and returns id of the object bind with candidate id.*

### 4.2.2.2.3 GetProfilePic()

*Function: Returns the profile picture name of candidate. It depends upon the object that called this function. Whenever the object is made, it stores candidate information. Therefore, this function is associated with candidates object and returns profile picture name of the object bind with candidate profile picture name.*

*.*

### 4.2.2.2.4 GetElectionID()

*Function: Returns the election id in which candidate participated. It depends upon the object that called this function. Whenever the object is made, it stores candidate information. Therefore, this function is associated with candidates object and returns election id of the object bind with candidate’s election id.*

# 4.2.3 User

*The User class is the representation of Voters in the E-Voting System. This class holds the*

*general information of the Voters with a unique CNIC number as a primary key.*

## 4.2.3.1 Functions available in User Class

### 4.2.3.1.1 set\_Code()

*Function: This function sets the code of voter(user). This code is unique for every android device. This code is user to send firebase notification in order to notify user.*

*Precondition: Voter has logged in to our android app.*

*Postcondition: the database code field is filled with android dependent unique code key. Whenever admin will send notification, Notification Server will use this code to refer individual android device*

### 4.2.3.1.2 set\_CNIC()

*Function: Takes the input from the user about his CNIC. Verifies the CNIC and adds the record to Database.*

*Precondition: The user has a CNIC.*

*Postcondition: Record must have been added to the Database.*

4.2.3.1.3 set\_Email()

*Function: Sets the email to desired email and the email must be unique and saves the email in Database. This function is also used in singleton designed patter in order to save the session of user.*

*.*

4.2.3.1.4 set\_Password( )

*Function: Sets password of the user and saves it to the database after encryption. This function is also used in singleton designed patter in order to save the session of user.*

4.2.3.1.5 set\_Fullname()

*Function: Sets the Fullname to desired fullname field in the database. This function is also used in singleton designed patter in order to save the session of user.*

*.*

4.2.3.1.6 get\_Password( )

*Function: Returns the password. This function is also used in singleton designed patter in order to save the session of user.*

4.2.3.1.7 get\_email( )

*Function: Returns the email. This function is also used in singleton designed patter in order to save the session of user.*

4.2.3.1.8 get\_Fullname( )

*Function: Returns the name of the user. This function is also used in singleton designed patter in order to save the session of user.*

# 4.2.4 VotingEvent

*This class represents the scenario of a voting event. It contains the Election Id, Name, Description of the Voting Event, Image of the Poster and etc.*

## 4.2.4.1 Functions available in VotingEvent class

### *4.2.4.1.1 updateVote()*

*Function: Insert in the votes table to the Database. Checks for duplicates and updates the vote.*

4.2.4.1.2 setName()

*Function: Sets the name of event to desired name and saves the name in Database. This function is also used in singleton designed patter in order to save the session of voting events.*

*.*

4.2.4.1.3 setDescription( )

*Function: Sets the description about event. This function is also used in singleton designed patter in order to save the session of voting events.*

4.2.4.1.4 setElectionID()

*Function: Sets the election id to desired election id field in the database. This function is also used in singleton designed patter in order to save the session of user.*

*.*

4.2.4.1.5 get\_Description( )

*Function: Returns the description of event. This function is also used in singleton designed patter in order to save the session of user.*

4.2.4.1.6 get\_ElectionID ( )

*Function: Returns the unique election id associated with each voting event. This function is also used in singleton designed patter in order to save the session of user.*

4.2.4.1.7 get\_Name( )

*Function: Returns the name of the voting event. This function is also used in singleton designed patter in order to save the session of user.*

4.2.4.1.6 set\_Institute ( )

*Function: sets the name of institute if the voting event is associated with any institute. This filed could be null in database. This feature of voting in the institute can be extended in future*

4.2.4.1.7 get\_Institute ( )

*Function: Returns the name of institute if the voting event is associated with any institute. This filed could be null in database. This feature of voting in the institute can be extended in future*

*.*

# 4.2.5 MailerSystem

## 4.2.5.1 Functions available in MailerSystem Class

### *4.2.5.2.1 MailALL ()*

*Function: Accepts the text for the email generation and send the email to all users.*

*Precondition: Mailer is ready and is configured with SMTP server.*

*Postcondition: Email has been successfully sent to all the users.*

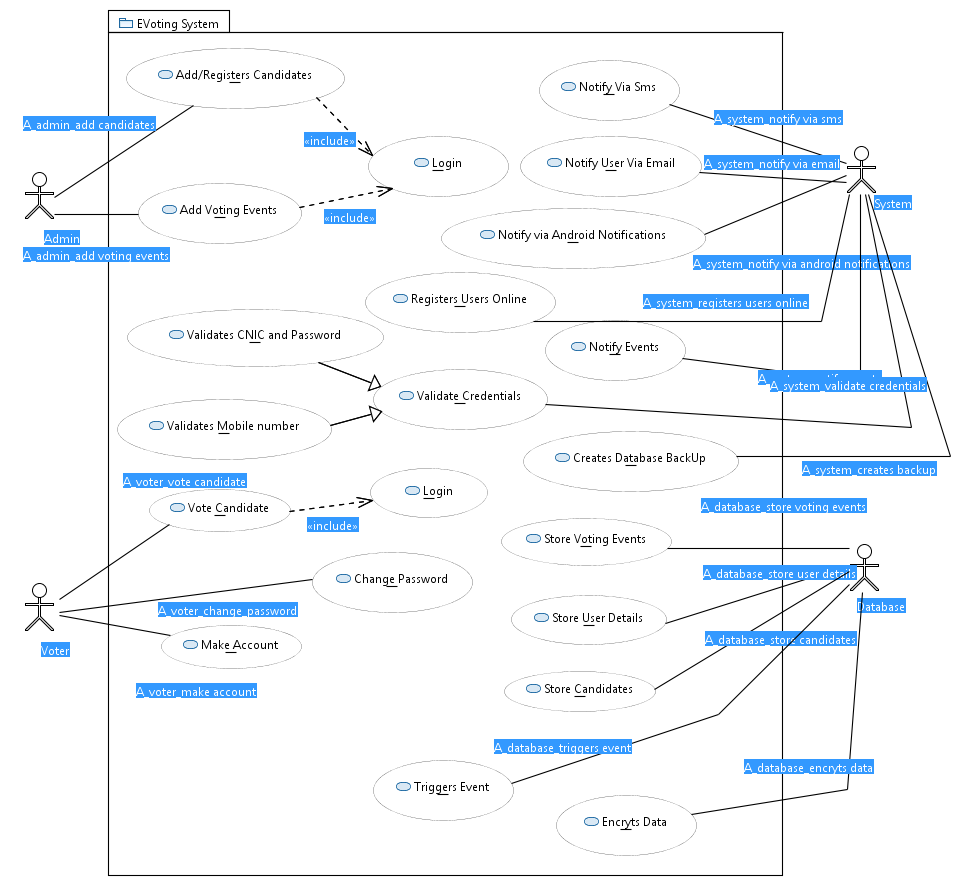
### *4.2.5.2.2 MailSend()*

*Function: Sends mail to specific members specified.*

*Precondition: Mailer is ready and is configured with SMTP server.*

*Postcondition: Email has been successfully sent to all the users.*

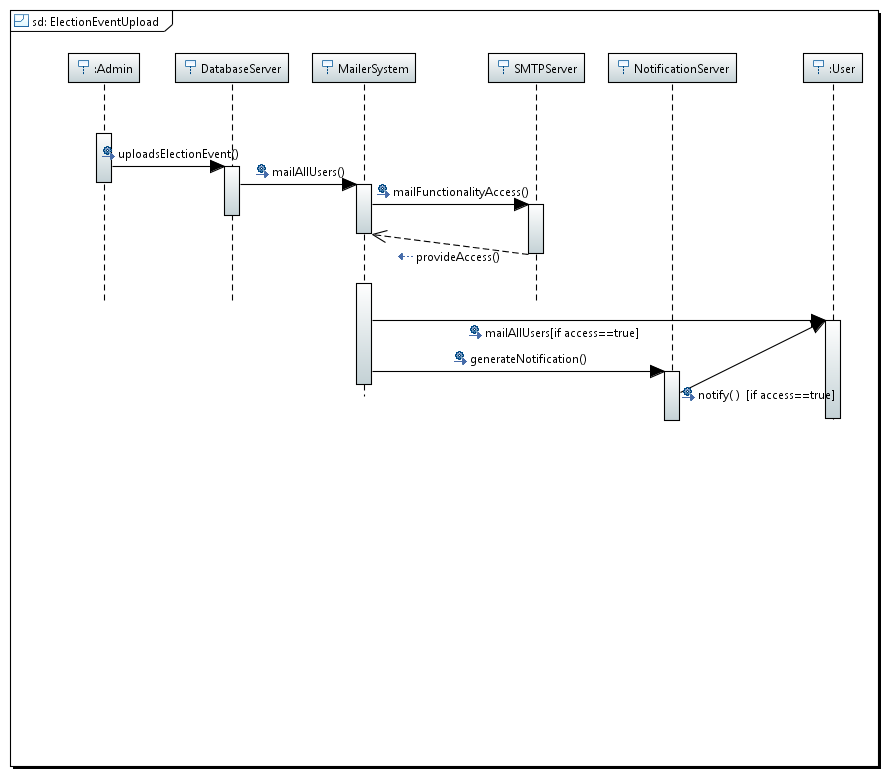
# 5. FIGURES

5.1 Use Cases****

**USE CASE (Figure** 1)

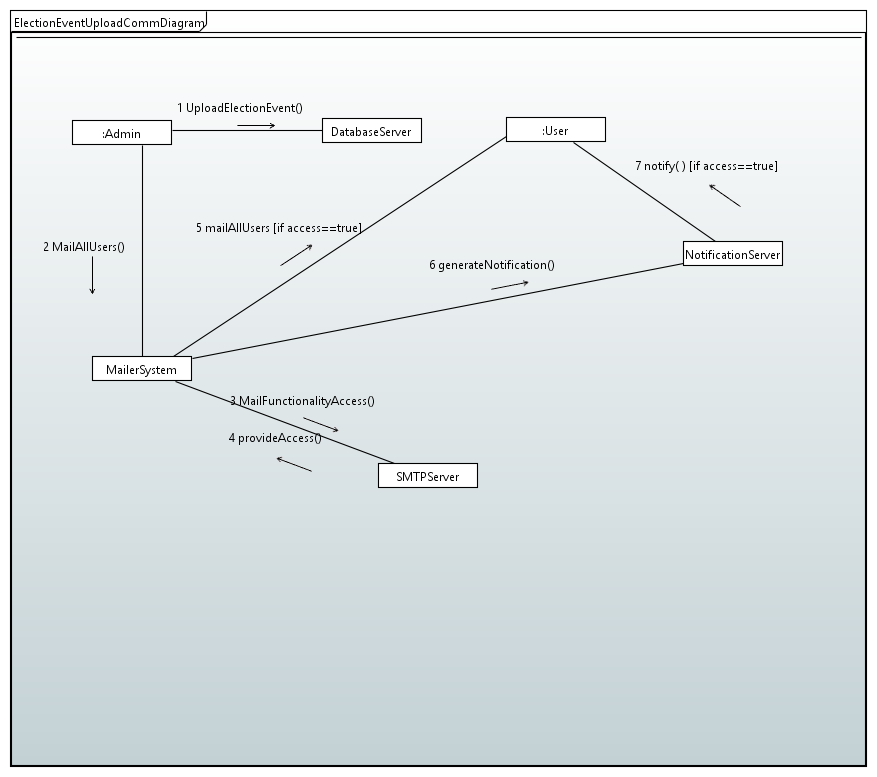
## 5.2 Election Event Upload

### 5.2.1 Election Event Upload Sequence (Figure 3)



***Figure 3- Election Event Upload Sequence***

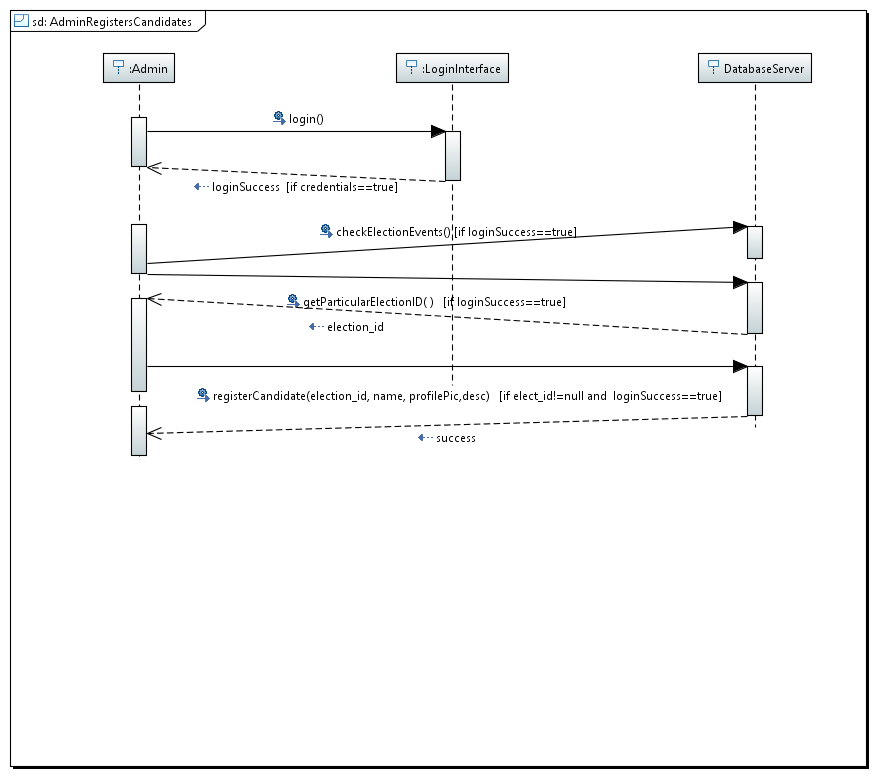
### 5.2.2 Election Event Collaboration (Figure 4)



***Figure 4- Election Event Upload Collaboration***

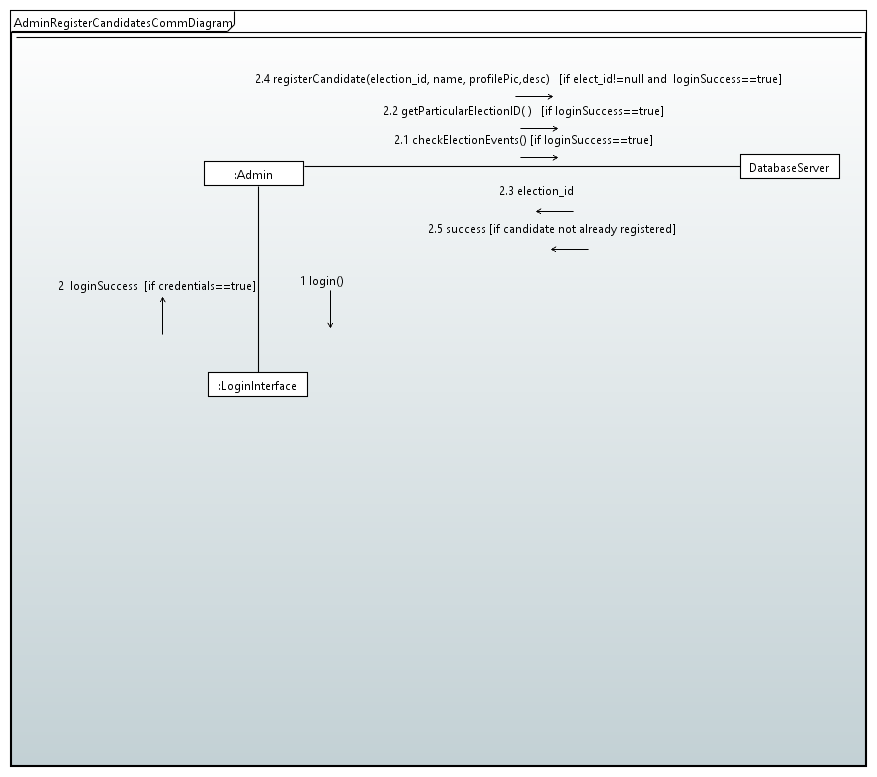
## 5.3 Admin Register Candidates

### **5.3.1** Admin Register Candidates**(Figure 5)**



***Figure 5- Admin Register Candidate Sequence***

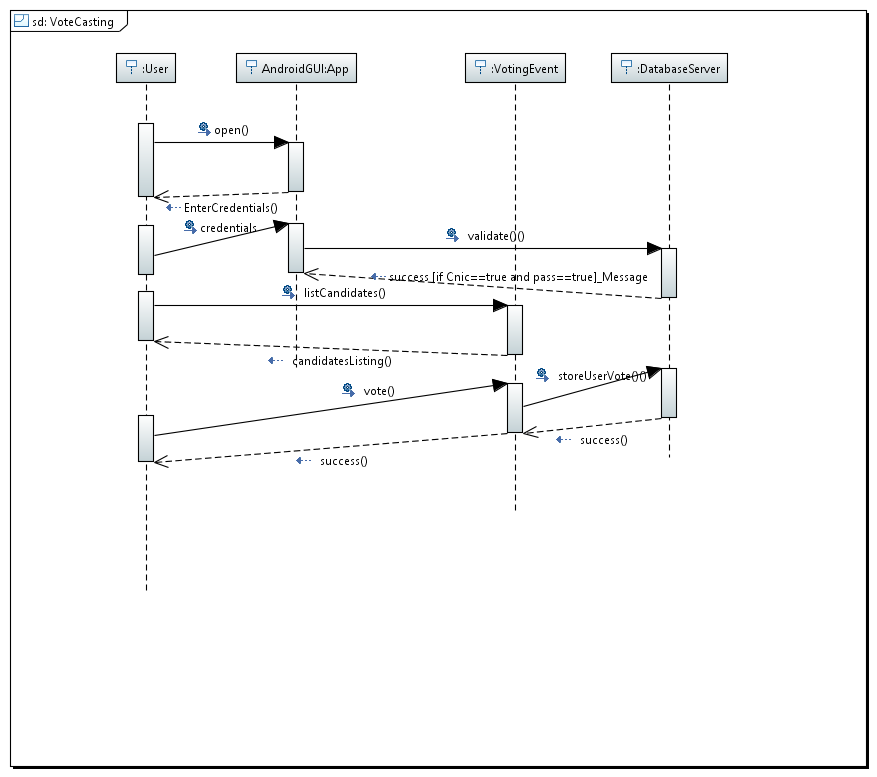
### 5.3.2 Admin Register Candidates Collaboration (Figure 6)



***Figure 6 –Admin Register Candidate Collaboration***

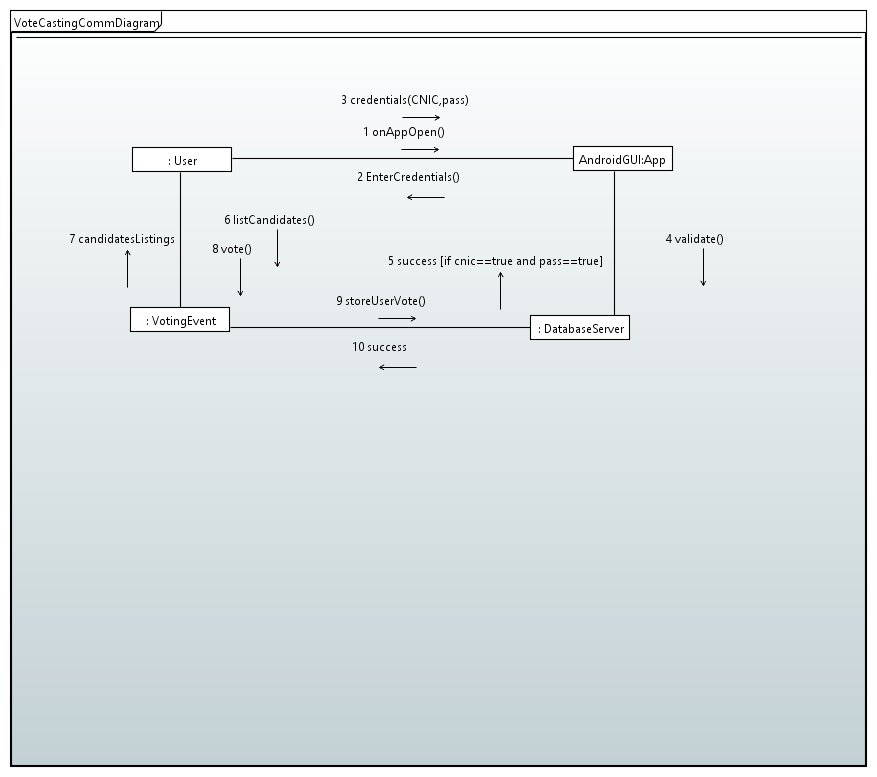
## 5.4 Vote Casting

### 5.4.1 Vote Casting Sequence (Figure 7)



***Figure 7 –Vote Casting Sequence***

### 5.4.2 Vote Casting Collaboration (Figure 8)



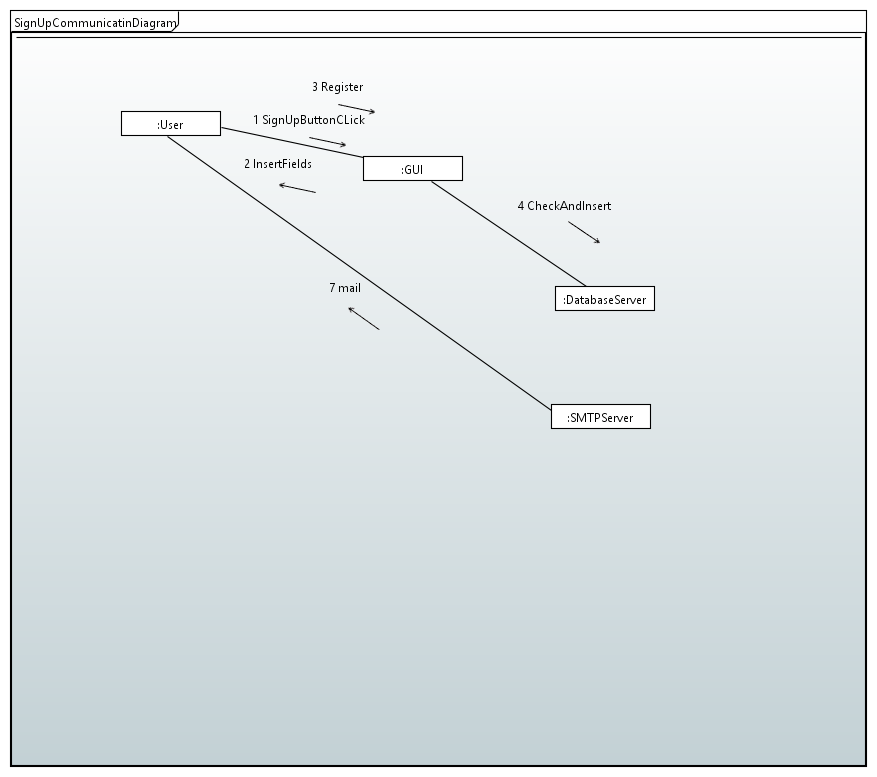
***Figure 8- Vote Casting Collaboration***

## 5.5 Signup

### 5.5.1 Signup Sequence (Figure 9)

***Figure 9 - Signup Sequence***

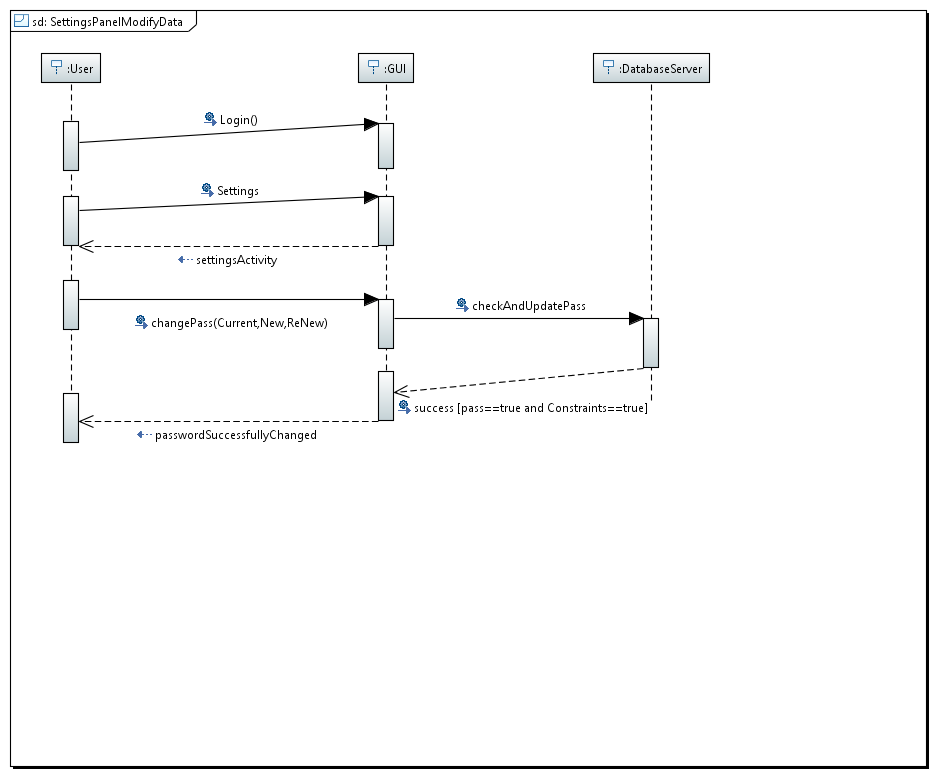
### 5.5.2 Signup Collaboration (Figure 10)



***Figure 10 - Signup Collaboration***

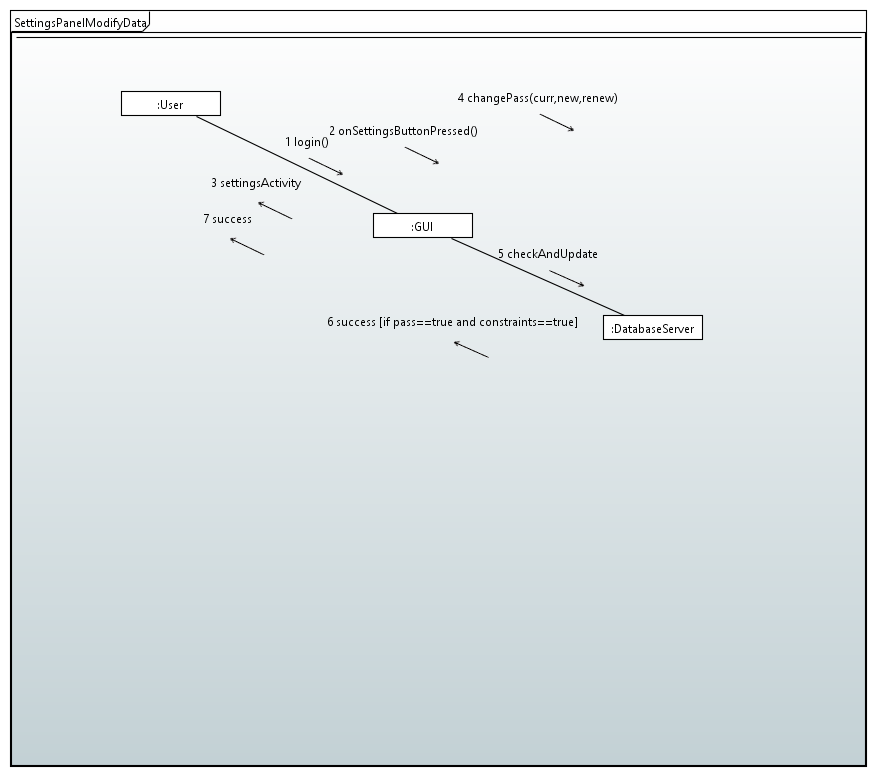
## 5.6 Setting Panel Modify

### 5.6.1 Setting Panel Modify Sequence (Figure 11)



***Figure 11 –Setting Model Modify Sequence***

### 5.6.3 Setting Panel Modify Collaboration (Figure 12)



***Figure 12 Setting Panel Modify Collaboration***

# 6 REFERENCES

## 6.1 References

*Appleton, Brad . A Software Design Specification Template. N.d.*

*<http://www.enteract.com/~bradapp/docs/sdd.html>.*

*Booch, Grady, Ivar Jacobsen, and James Rumbaugh. The Unified Software Development Process(The Addison-Wesley Object Technology Series). 1st. Ed. New York: Addison Wesley, 1999..*

*GCC Home Page - GNU Project . Free Software Foundation. N.d. <http://gcc.gnu.org/>.*

*Sommerville, Ian. Software Engineering. 6th. Ed. New York: Addison Wesley, 2001.*