**Online Survey Assistant**

**High-Level Design Document**

v0.00

Robosoft Technologies Pvt. Ltd.

Registered & Corporate Office

217, N.H.66, Santhekatte

New Udupi – 576 105

Karnataka, India

Tel: +91 820 2593930

Fax: +91 820 2593900

**Document Revision History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Date** | **Version** | **Authors** | **Reviewed By** | **Approved By** | **Brief Description** |
| <dd-Mmm-yy> | 0.01 | <Author> | <Reviewer>  (One person/ Group Review - PQAL) | - | Initial draft |
| <dd-Mmm-yy> | 0.02 | <Author> | Verified by | - | List the changes incorporated as per review |
| <dd-Mmm-yy> | 0.03 | <Author> | <Customer> | <Customer> | List the changes incorporated as per Customer feedback and sent to Customer for approval |
| <dd-Mmm-yy> | 1.00 |  |  |  | Approved by Customer and baselined |

**Contents**

1. High-Level Design 4
   1. Introduction 4
      1. Purpose 4
      2. Scope 4
      3. Audience 4
      4. References 4
      5. Terms and Abbreviations 4
   2. Design Considerations 4
      1. Assumptions and dependencies 4

**Figures**

# High-Level Design

## Introduction

The High Level Design Document (HLDD) provides an overview of the Online Survey Assistant project. It provides a detailed description to create survey, add questions and publish to the participants. It also contains information on the design considerations and the different components of the project, its architecture, responsibilities and the control flow.

### Purpose

The purpose of this High Level Design Document (HLDD) is to provide architecture, functionality and methodologies and road map for the developers to implement the solution for Online Survey Assistant.

### Scope

This document provides the system and design details for the Online Survey Assistant software to be developed.

### Audience

This document is intended for the customer, project manager, development team and the quality assurance team.

### References

Requirement docs provided by customer on 13-02-2021

### Terms and Abbreviations

OnlineSurvey– codename used for this project**.**

## Design Considerations

### Assumptions and dependencies

* Super admin would be able to create admins. Admins would be able to create surveys, add/edit questions and publish the surveys to participants.
* Only super admins would be able to create the admins and manage user management.
* Once the admins are created in the backend password set link should be sent to their emailIds and they have to set the password. Then they can login to the system.
* After login admins can create survey and add question then can save the survey as draft.
* They can then publish the survey to participants using email ids.
* The invited users will get invitation link to the survey in their mails and after they can login or as a guest user they can view the survey questions and answer them. Only after filling all the required questions they can submit the survey.

Functionalities should be implemented in the client side:

* Showing tabs based on user permissions
* Change the order of the questions using drag and drop
* Prevent users submitting survey without answering required questions
* Storing answers when the browser closes, until the user submits the survey.

**API LIST**:

1. Login

Description

API for the admins to login to the app

HTTP Method

POST

Input Parameters

username : EmailId (String)

password: String (Encrypted)

Output Parameters

On Success:

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

accessToken: String

user: {userId: Integer, userName: String, userEmailID: String}

2. Create Admin

Description

API for the super admins to create admin

HTTP Method

POST

Input Parameters

accessToken: String (Encrypted)

User: {employeeName: String, employeeCode: String, mailId: String, designation: string, createSurvey: boolean, allSurvey: boolean, deletedSurvey: boolean, userManagement: boolean }

Output Parameters

On Success: In the backend send set password link to the user created to set the password.

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

3. Edit Admin

Description

API for the super admins to edit the admin

HTTP Method

PATCH

Input Parameters

accessToken: String (Encrypted)

User: {employeeName: String, employeeCode: String, mailId: String, designation: string, createSurvey: boolean, allSurvey: boolean, deletedSurvey: boolean, userManagement: boolean }

Output Parameters

On Success:

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

4. Delete Admin

Description

API for the super admins to delete the admin

HTTP Method

DELETE

Input Parameters

accessToken: String (Encrypted)

userId: integer

Output Parameters

On Success:

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

5. Create Survey

Description

API for the admins to create Survey

HTTP Method

POST

Input Parameters

accessToken: String (Encrypted)

survey: {title: String, banner\_image: Image}

Output Parameters

On Success:

surveyId: integer

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

6. Add Question

Description

API for the admins to add Question

HTTP Method

POST

Input Parameters

accessToken: String (Encrypted)

question: {surveyId: integer, pageNo: integer, title: string, answerType: string, options: Array of options}

Output Parameters

On Success:

surveyId: integer

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure

7. Edit Question

Description

API for the admins to add Question

HTTP Method

PATCH

Input Parameters

accessToken: String (Encrypted)

question: {surveyId: integer, pageNo: integer, title: string, answerType: string, options: Array of options}

Output Parameters

On Success:

surveyId: integer

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

8. DeleteSurvey

Description

API for the admins to delete the survey

HTTP Method

DELETE

Input Parameters

accessToken: String (Encrypted)

surveyId: integer

Output Parameters

On Success:

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

9. Publish Survey

Description

API for the admins to publish the survey to participants

HTTP Method

POST

Input Parameters

accessToken: String (Encrypted)

survey: {surveyId: integer, emailds: Array of emailIds}

Output Parameters

On Success: In the backend invitation link for the given survey has to be sent to each email id.

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

10. List Survey

Description

API for the admins to list the surveys

HTTP Method

GET

Input Parameters

accessToken: String (Encrypted)

surveyType: all/deleted/draft

Output Parameters

On Success:

Surveys: Array of survey([{title: string, date: datetime, noOfPeopleResponded: integer, positiveResponse: integer, negativeResponse: integer}..])

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

On Failure:

errrorStatusCode: Integer

errorMessage: String

11. Get Survey detail

Description

API for the participants to see the survey details

HTTP Method

GET

Input Parameters

accessToken: String (Encrypted)

surveyID: integer

Output Parameters

On Success:

Survey: {title: String, bannerImage: image\_path, total\_pages: integer, pages:[{page\_no: integer, questions:[{questionId: integer, title: string, answerType: string, options: []}]}..]}

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

On Failure:

errrorStatusCode: Integer

errorMessage: String

12. Submit Survey

Description

API for the admins to add Question

HTTP Method

POST

Input Parameters

accessToken: String (Encrypted)

survey: {userId: integer, surveyId: integer, questions:[{questionId: integer, optionSelected: String}..]}

Output Parameters

On Success:

surveyId: integer

statusCode : String (success: “1” , failure: ”0”) (Encrypted)

message : String (Success/ Error message on success/failure)

On Failure:

errrorStatusCode: Integer

errorMessage: String

13. Forgot Password API

Description

Forgot Password

HTTP Method

POST

Input Parameters

email: String (Encrypted)

Output Parameters

On Success:

statusCode : Integer (success: 1 , failure: 0)

message : String

On Failure:

errrorStatusCode: Integer

errorMessage: String

14. Set Password API

Description

Forgot Password

HTTP Method

POST

Input Parameters

accessToken: String (Encrypted)

newPassword: String (Encrypted)

Output Parameters

On Success:

statusCode : Integer (success: 1 , failure: 0)

message : String

On Failure:

errrorStatusCode: Integer

errorMessage: String

15. Logout API

Description

Forgot Password

HTTP Method

DELETE

Input Parameters

accessToken: String (Encrypted)

email: String

Output Parameters

On Success:

statusCode : Integer (success: 1 , failure: 0)

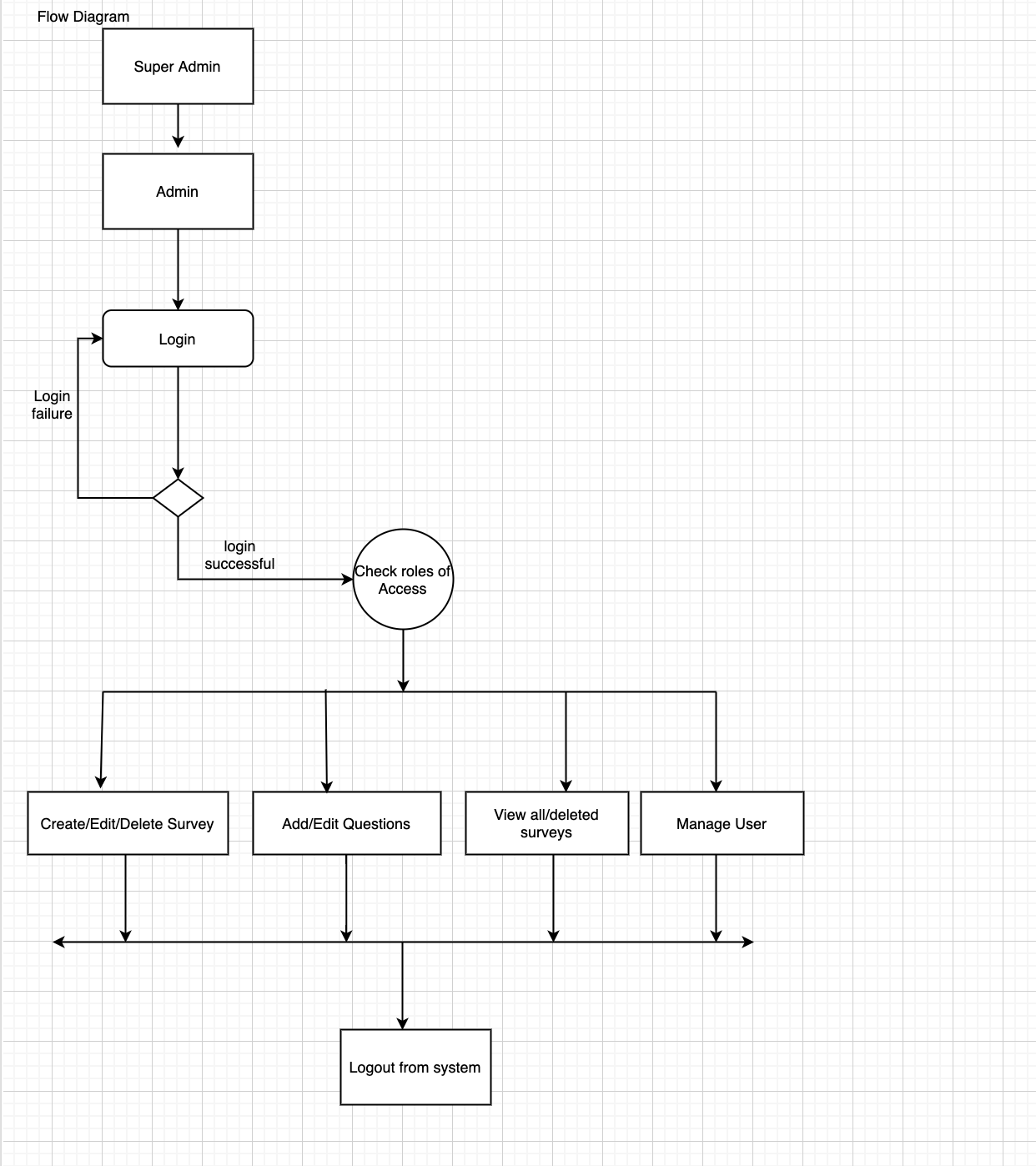
message : String

On Failure:

errrorStatusCode: Integer

errorMessage: String

Data flow diagram:



Database structure: