Report on

EmailTracking

Prepared by

Sidharth Choudhary (300990285)

Shiva Bhalla (300985590)

December 14th, 2018

INDEX

Contents

INTRODUCTION

Email, once organized well, is the most effective means of communication. More versatile than anything else, therefore, marking Email as the most effective marketing tactic. The biggest challenge for enterprises is to track the sent emails i.e. to read the status of their sent emails, for example - has the email been delivered to the recipient or is it in the processing state, was it read by the user after delivery, and so on.

Tracking the statistics of the sent emails is a very tedious process, which on the other hand involves a lot of cost for sending bulk emails concurrently. For this, SendGrid provides a cloud-based email delivery service that assists businesses with email delivery. It also allows companies to track email opens, unsubscribes, bounces, and spam reports. These services provided by SendGrid are all at low cost, offering some for free.

Therefore, check if your Emails were Read, Spammed, Opted out, etc. Email Tracking for Gmail, Hotmail / Outlook, Google Inbox, & Yahoo Mail is available through our web application. Simply register as a new user to our web application and send emails (via SendGrid) to the contacts added in your account. For the given instance, our web application works for unlimited email Accounts and 25000 emails/month (Free SendGrid version). Our Email tracker does not ever store or transfer your email contents (except the Subject of the email) in order to ensure your privacy. A user *cannot* add a Contact of themselves (i.e. they cannot use their logged in email ID as the email recipient Contact). At a time, user can send email to only one Contact.

.

DOCUMENTATION

DATA MODEL DESIGN FOR EMAIL TRACKING:

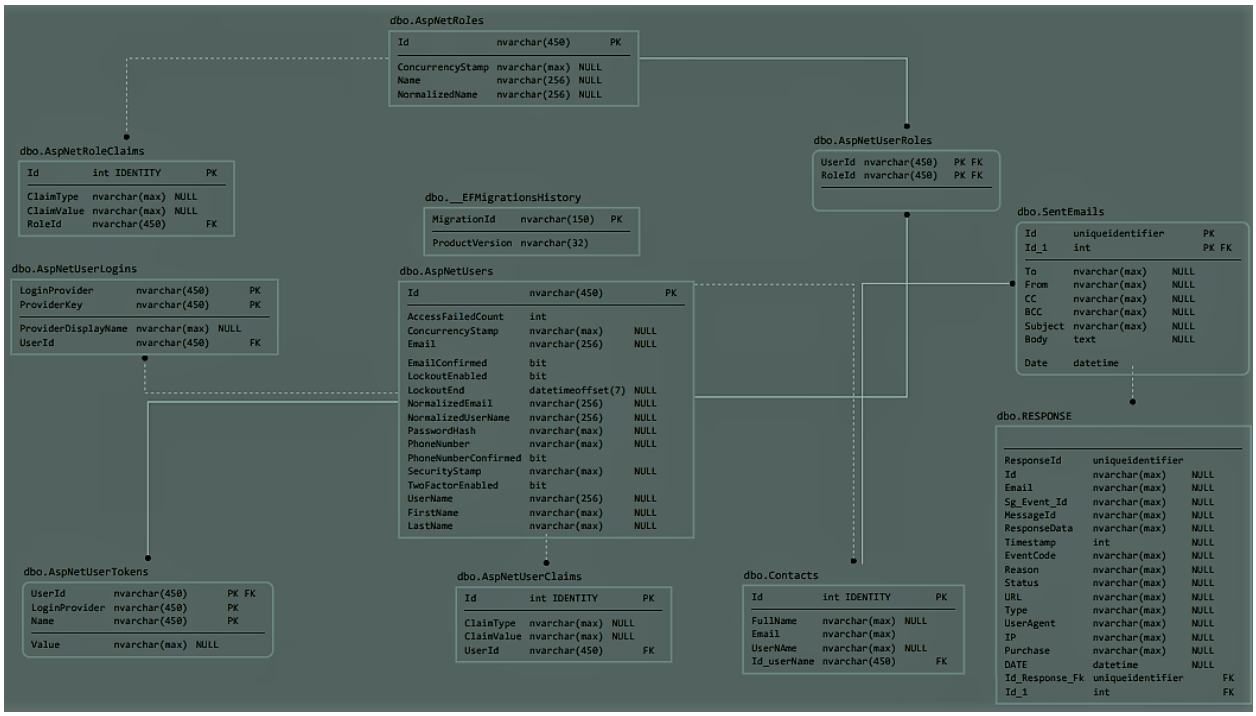

Fig 1.1 Data Model for the Relational Database System

ER-DIAGRAM:

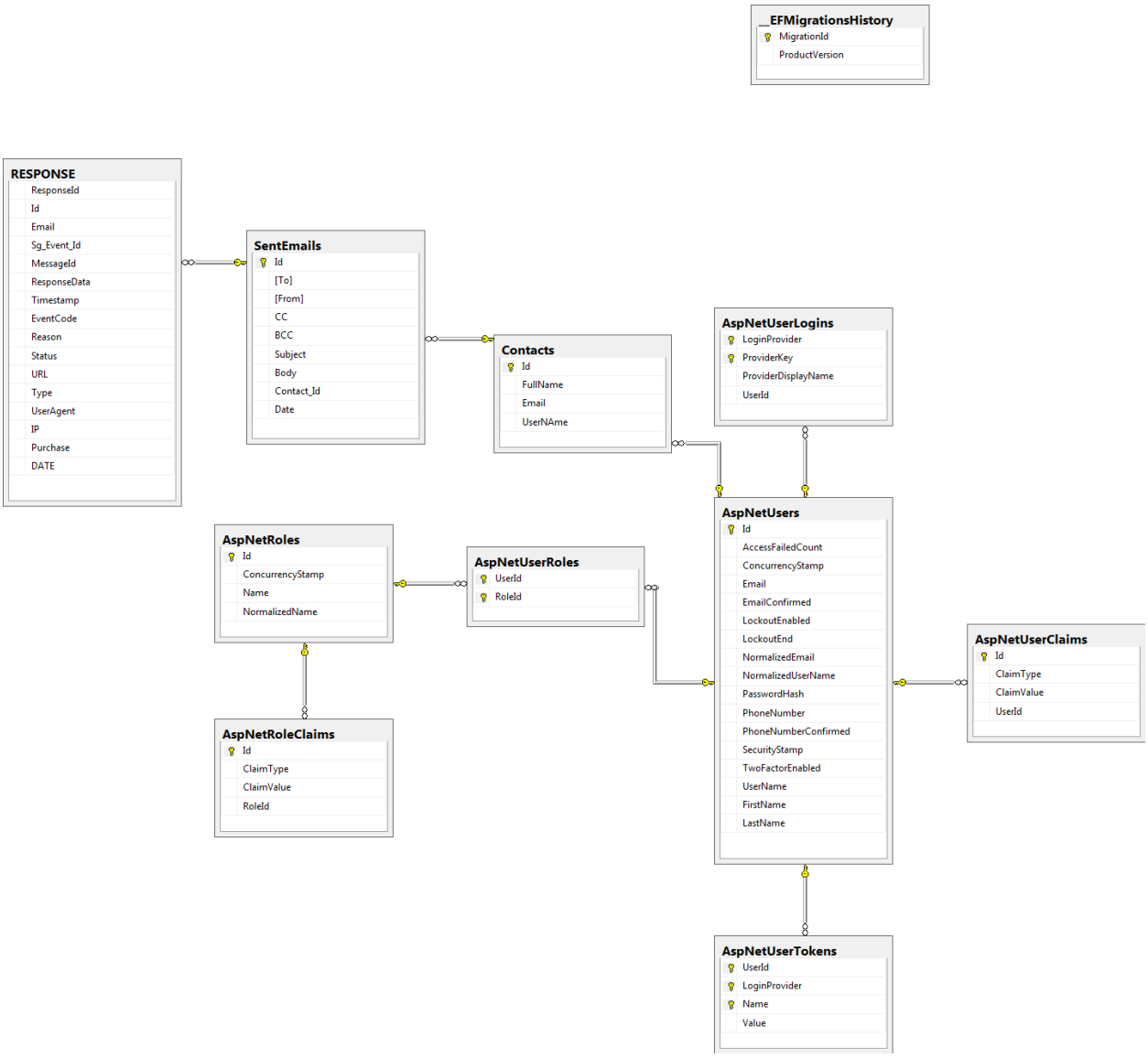

Fig 1.2 ER Diagram for the Relational Database System.

SCRIPT FILE:

```
USE [EmailTracking]
GO
/****** Object:  Table [dbo].[__EFMigrationsHistory]    Script Date: 12/6/2018 3:29:21 PM
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[__EFMigrationsHistory](
	[MigrationId] [nvarchar](150) NOT NULL,
	[ProductVersion] [nvarchar](32) NOT NULL,
 CONSTRAINT [PK___EFMigrationsHistory] PRIMARY KEY CLUSTERED
(
	[MigrationId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]

GO
/****** Object:  Table [dbo].[AspNetRoleClaims]    Script Date: 12/6/2018 3:29:21 PM
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[AspNetRoleClaims](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[ClaimType] [nvarchar](max) NULL,
	[ClaimValue] [nvarchar](max) NULL,
	[RoleId] [nvarchar](450) NOT NULL,
 CONSTRAINT [PK_AspNetRoleClaims] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[AspNetRoles]    Script Date: 12/6/2018 3:29:21 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[AspNetRoles](
	[Id] [nvarchar](450) NOT NULL,
	[ConcurrencyStamp] [nvarchar](max) NULL,
	[Name] [nvarchar](256) NULL,
	[NormalizedName] [nvarchar](256) NULL,
 CONSTRAINT [PK_AspNetRoles] PRIMARY KEY CLUSTERED
(
	[Id] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[AspNetUserClaims]    Script Date: 12/6/2018 3:29:21 PM
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[AspNetUserClaims](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[ClaimType] [nvarchar](max) NULL,
	[ClaimValue] [nvarchar](max) NULL,
	[UserId] [nvarchar](450) NOT NULL,
 CONSTRAINT [PK_AspNetUserClaims] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[AspNetUserLogins]    Script Date: 12/6/2018 3:29:21 PM
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[AspNetUserLogins](
	[LoginProvider] [nvarchar](450) NOT NULL,
	[ProviderKey] [nvarchar](450) NOT NULL,
	[ProviderDisplayName] [nvarchar](max) NULL,
	[UserId] [nvarchar](450) NOT NULL,
 CONSTRAINT [PK_AspNetUserLogins] PRIMARY KEY CLUSTERED
(
	[LoginProvider] ASC,
	[ProviderKey] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[AspNetUserRoles]    Script Date: 12/6/2018 3:29:21 PM
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[AspNetUserRoles](
	[UserId] [nvarchar](450) NOT NULL,
	[RoleId] [nvarchar](450) NOT NULL,
 CONSTRAINT [PK_AspNetUserRoles] PRIMARY KEY CLUSTERED
(
	[UserId] ASC,
	[RoleId] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]

GO
/****** Object:  Table [dbo].[AspNetUsers]    Script Date: 12/6/2018 3:29:21 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[AspNetUsers](
	[Id] [nvarchar](450) NOT NULL,
	[AccessFailedCount] [int] NOT NULL,
	[ConcurrencyStamp] [nvarchar](max) NULL,
	[Email] [nvarchar](256) NULL,
	[EmailConfirmed] [bit] NOT NULL,
	[LockoutEnabled] [bit] NOT NULL,
	[LockoutEnd] [datetimeoffset](7) NULL,
	[NormalizedEmail] [nvarchar](256) NULL,
	[NormalizedUserName] [nvarchar](256) NULL,
	[PasswordHash] [nvarchar](max) NULL,
	[PhoneNumber] [nvarchar](max) NULL,
	[PhoneNumberConfirmed] [bit] NOT NULL,
	[SecurityStamp] [nvarchar](max) NULL,
	[TwoFactorEnabled] [bit] NOT NULL,
	[UserName] [nvarchar](256) NULL,
	[FirstName] [nvarchar](max) NULL,
	[LastName] [nvarchar](max) NULL,
 CONSTRAINT [PK_AspNetUsers] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[AspNetUserTokens]    Script Date: 12/6/2018 3:29:21 PM
******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[AspNetUserTokens](
	[UserId] [nvarchar](450) NOT NULL,
	[LoginProvider] [nvarchar](450) NOT NULL,
	[Name] [nvarchar](450) NOT NULL,
	[Value] [nvarchar](max) NULL,
 CONSTRAINT [PK_AspNetUserTokens] PRIMARY KEY CLUSTERED
(
	[UserId] ASC,
	[LoginProvider] ASC,
	[Name] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[Contacts]    Script Date: 12/6/2018 3:29:21 PM ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Contacts](
	[Id] [int] IDENTITY(1,1) NOT NULL,
	[FullName] [nvarchar](max) NULL,
	[Email] [nvarchar](max) NOT NULL,
	[UserNAme] [nvarchar](450) NOT NULL,
 CONSTRAINT [PK_Contacts] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[Email]    Script Date: 12/6/2018 3:29:21 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Email](
	[Id] [uniqueidentifier] NOT NULL,
	[To] [nvarchar](max) NOT NULL,
	[From] [nvarchar](max) NOT NULL,
	[CC] [nvarchar](max) NULL,
	[BCC] [nvarchar](max) NULL,
	[Subject] [nvarchar](200) NOT NULL,
	[Body] [nvarchar](max) NOT NULL,
	[Date] [datetime2](7) NOT NULL,
 CONSTRAINT [PK_Email] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[RESPONSE]    Script Date: 12/6/2018 3:29:21 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[RESPONSE](
	[ResponseId] [uniqueidentifier] NOT NULL,
	[Id] [nvarchar](max) NULL,
	[Email] [nvarchar](max) NULL,
	[Sg_Event_Id] [nvarchar](max) NULL,
	[MessageId] [uniqueidentifier] NOT NULL,
	[ResponseData] [nvarchar](max) NULL,
	[Timestamp] [int] NULL,
	[EventCode] [nvarchar](max) NULL,
	[Reason] [nvarchar](max) NULL,
	[Status] [nvarchar](max) NULL,
	[URL] [nvarchar](max) NULL,
	[Type] [nvarchar](max) NULL,
	[UserAgent] [nvarchar](max) NULL,
```

```
	[IP] [nvarchar](max) NULL,
	[Purchase] [nvarchar](max) NULL,
	[DATE] [datetime] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
/****** Object:  Table [dbo].[SentEmails]    Script Date: 12/6/2018 3:29:21 PM ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[SentEmails](
	[Id] [uniqueidentifier] NOT NULL,
	[To] [nvarchar](max) NULL,
	[From] [nvarchar](max) NULL,
	[CC] [nvarchar](max) NULL,
	[BCC] [nvarchar](max) NULL,
	[Subject] [nvarchar](max) NULL,
	[Body] [text] NULL,
	[Contact_Id] [int] NOT NULL,
	[Date] [datetime] NOT NULL DEFAULT (getdate()),
PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]

GO
ALTER TABLE [dbo].[RESPONSE] ADD  DEFAULT (getdate()) FOR [DATE]
GO
ALTER TABLE [dbo].[AspNetRoleClaims]  WITH CHECK ADD  CONSTRAINT
[FK_AspNetRoleClaims_AspNetRoles_RoleId] FOREIGN KEY([RoleId])
REFERENCES [dbo].[AspNetRoles] ([Id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[AspNetRoleClaims] CHECK CONSTRAINT
[FK_AspNetRoleClaims_AspNetRoles_RoleId]
GO
ALTER TABLE [dbo].[AspNetUserClaims]  WITH CHECK ADD  CONSTRAINT
[FK_AspNetUserClaims_AspNetUsers_UserId] FOREIGN KEY([UserId])
REFERENCES [dbo].[AspNetUsers] ([Id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[AspNetUserClaims] CHECK CONSTRAINT
[FK_AspNetUserClaims_AspNetUsers_UserId]
GO
ALTER TABLE [dbo].[AspNetUserLogins]  WITH CHECK ADD  CONSTRAINT
[FK_AspNetUserLogins_AspNetUsers_UserId] FOREIGN KEY([UserId])
REFERENCES [dbo].[AspNetUsers] ([Id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[AspNetUserLogins] CHECK CONSTRAINT
[FK_AspNetUserLogins_AspNetUsers_UserId]
GO
ALTER TABLE [dbo].[AspNetUserRoles]  WITH CHECK ADD  CONSTRAINT
[FK_AspNetUserRoles_AspNetRoles_RoleId] FOREIGN KEY([RoleId])
REFERENCES [dbo].[AspNetRoles] ([Id])
```

```
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[AspNetUserRoles] CHECK CONSTRAINT
[FK_AspNetUserRoles_AspNetRoles_RoleId]
GO
ALTER TABLE [dbo].[AspNetUserRoles]  WITH CHECK ADD  CONSTRAINT
[FK_AspNetUserRoles_AspNetUsers_UserId] FOREIGN KEY([UserId])
REFERENCES [dbo].[AspNetUsers] ([Id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[AspNetUserRoles] CHECK CONSTRAINT
[FK_AspNetUserRoles_AspNetUsers_UserId]
GO
ALTER TABLE [dbo].[AspNetUserTokens]  WITH CHECK ADD  CONSTRAINT
[FK_AspNetUserTokens_AspNetUsers_UserId] FOREIGN KEY([UserId])
REFERENCES [dbo].[AspNetUsers] ([Id])
ON DELETE CASCADE
GO
ALTER TABLE [dbo].[AspNetUserTokens] CHECK CONSTRAINT
[FK_AspNetUserTokens_AspNetUsers_UserId]
GO
```

SYSTEM ARCHITECTURE

Architecture or Flow Summary:

Following are the flow summary steps that would be covered by our web application:

a. On landing page of website, user can Sign Up with his email id(unique). This email address would be the primary address to recognize the profile of user.

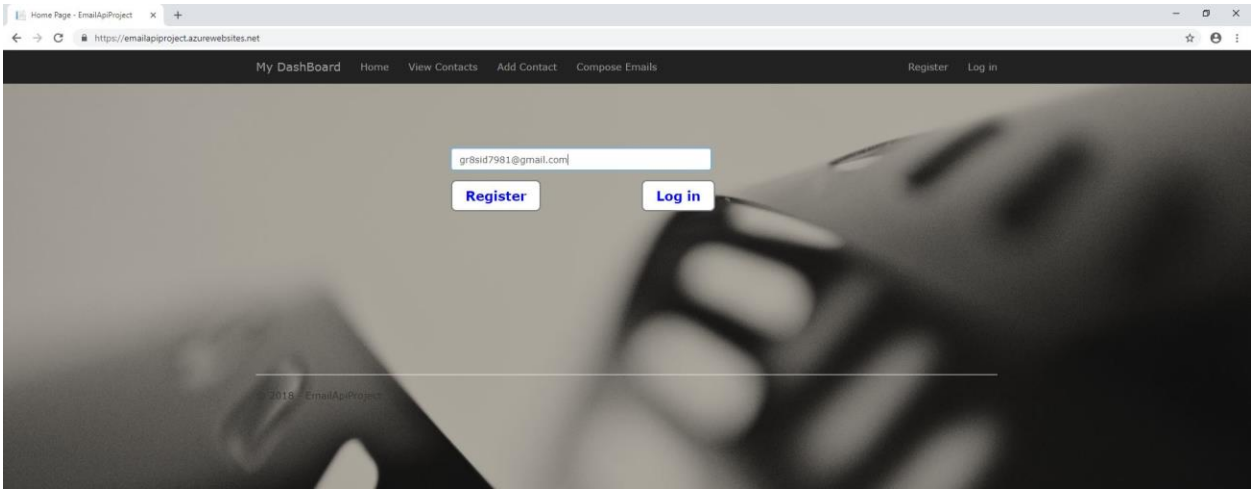

Fig. 2.1 Home Screen

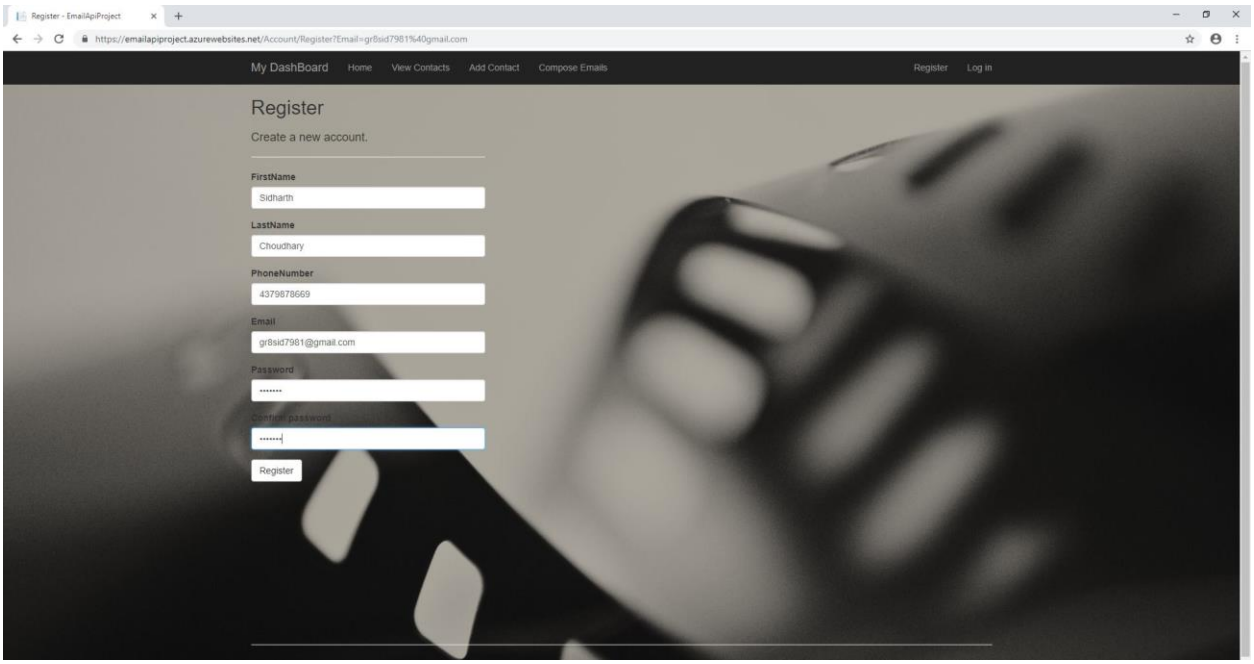

Fig. 2.2 Registration Screen

b. User can login using registered email address and further a menu with different options like Home, Contact List, Compose Emails, Outbox and Logout options.

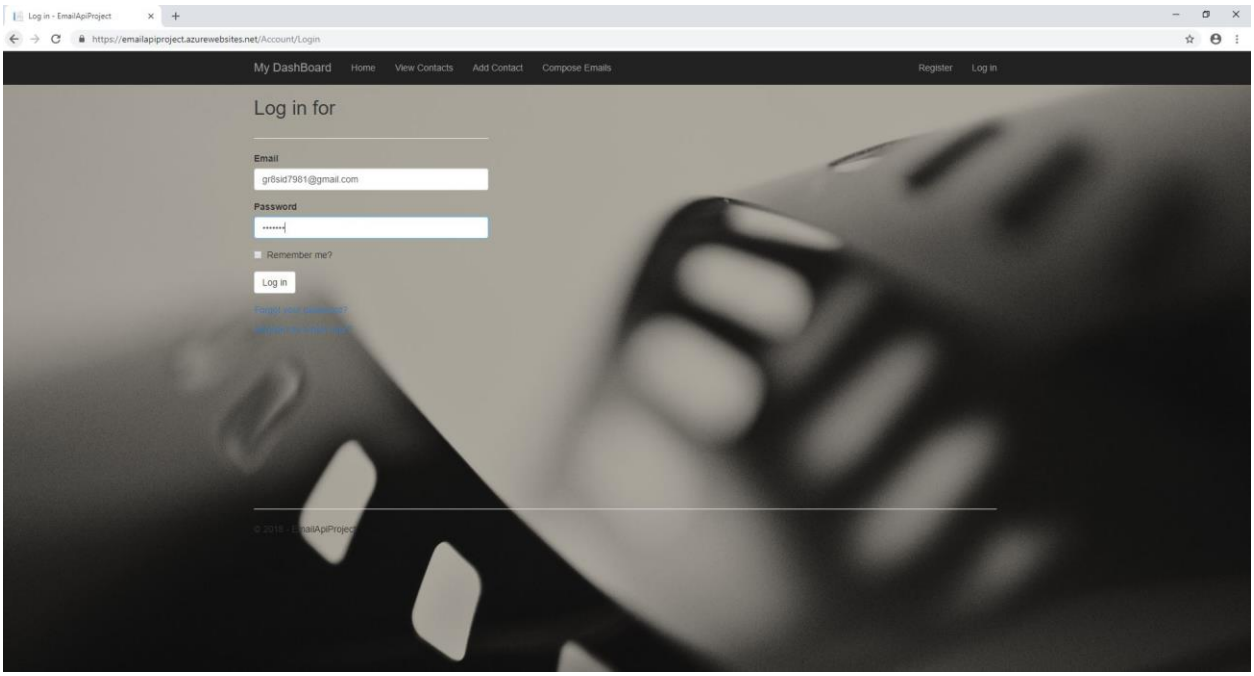

Fig. 2.3 Login Screen

c. A user can add contacts to his/her list and can use these contacts to send them emails.

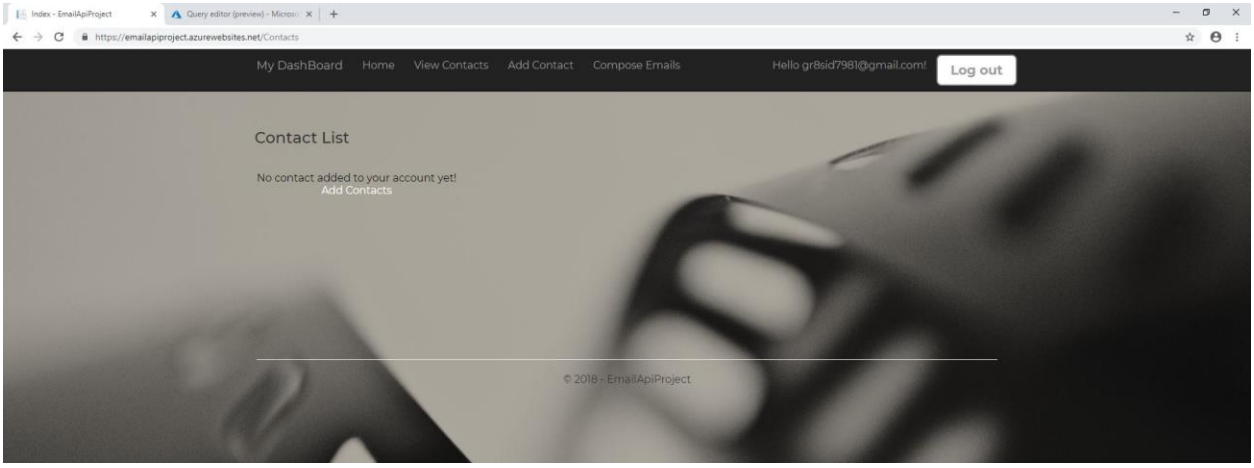

Fig. 2.4 Contact Screen without any contacts added.

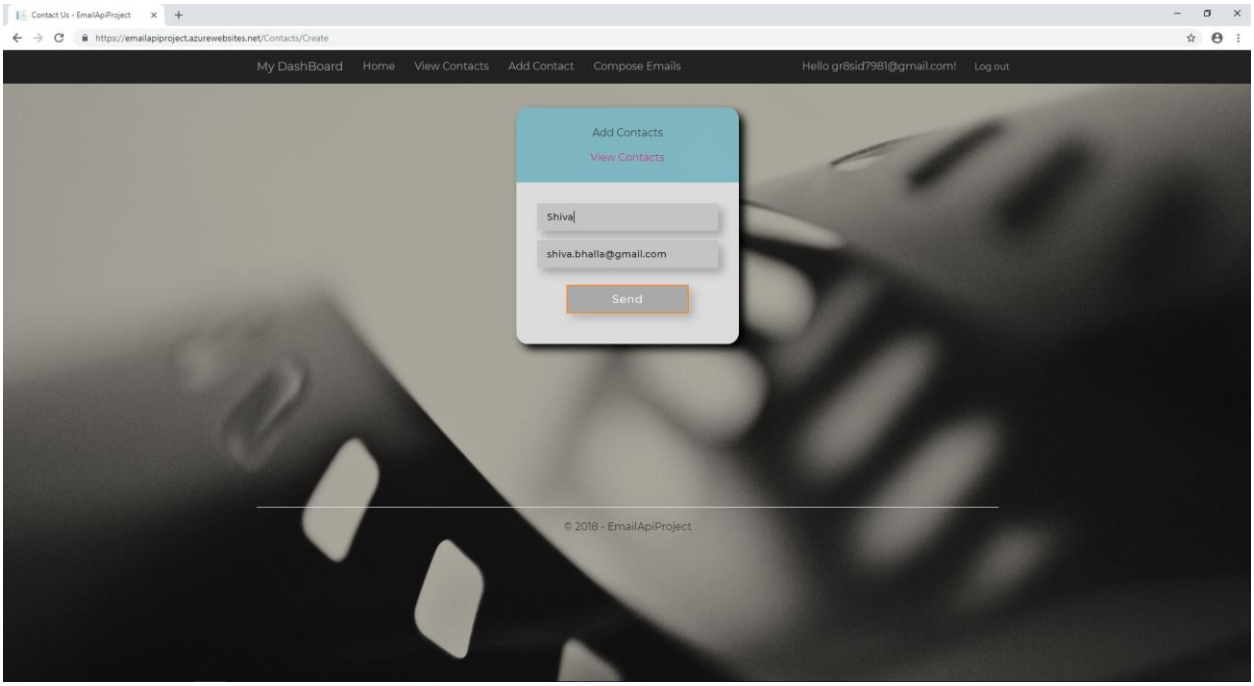

Fig. 2.5 Add Contact Screen

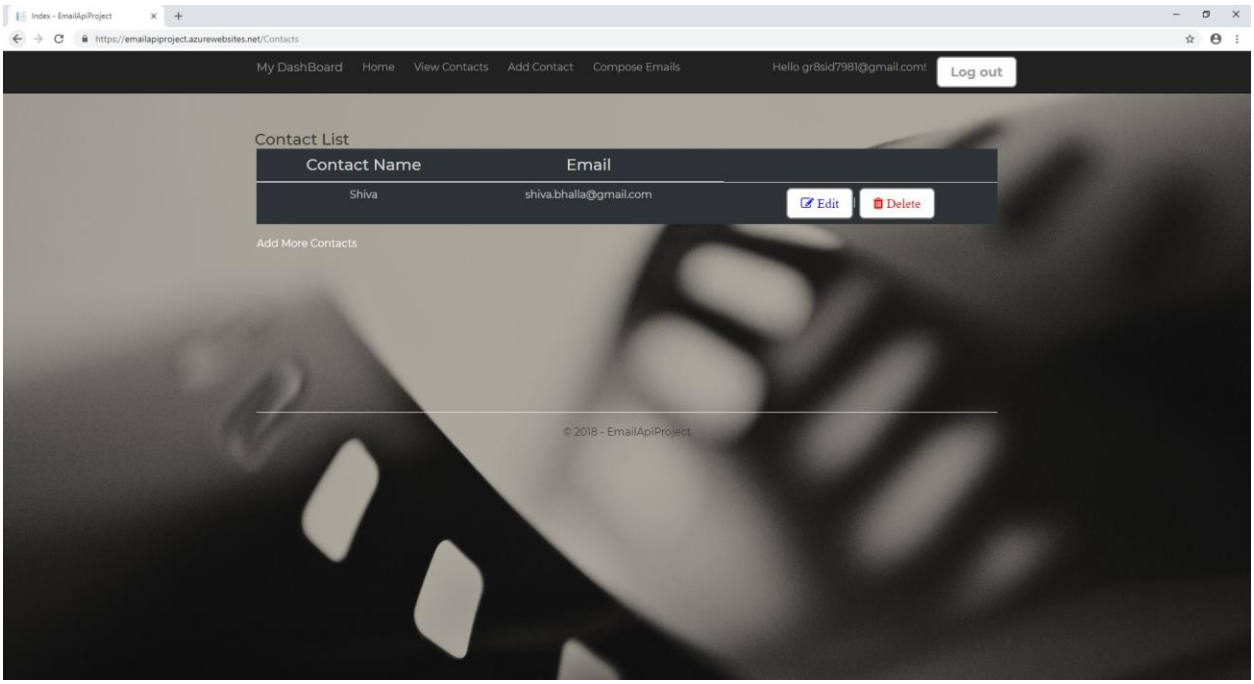

Fig. 2.6 Contact Screen after Adding contact.

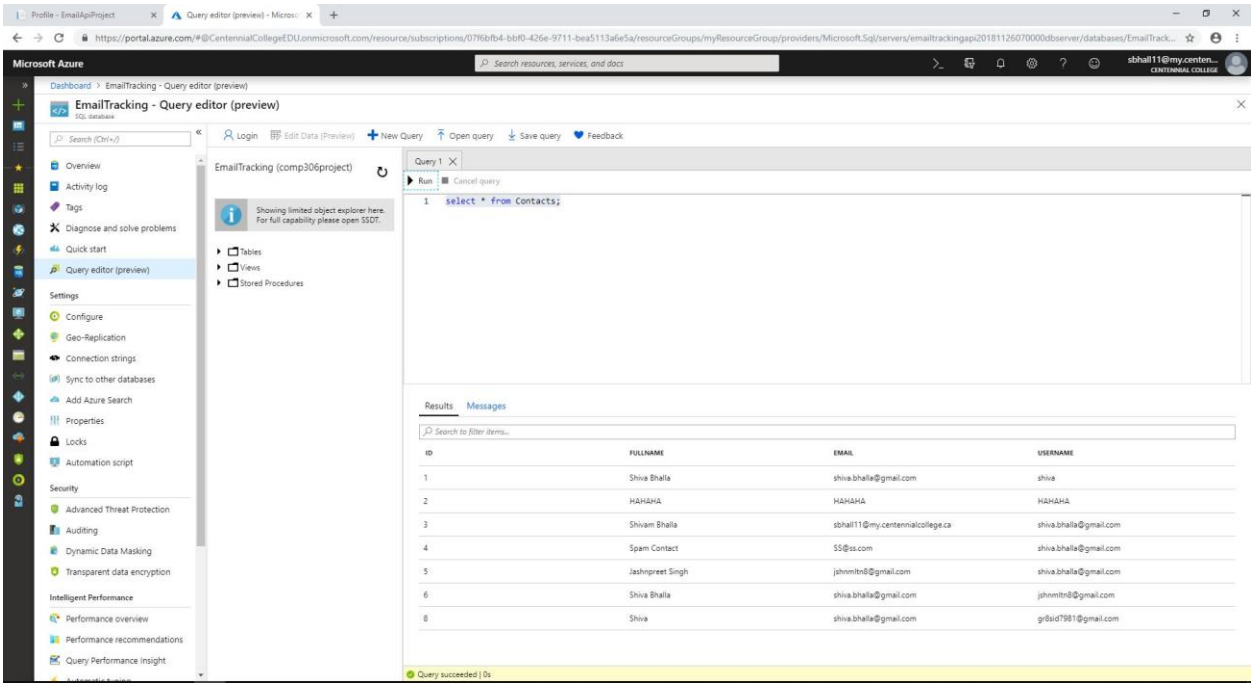

Fig. 2.7 Contacts Database after adding Contact.

d. A user can Edit the Contact List.

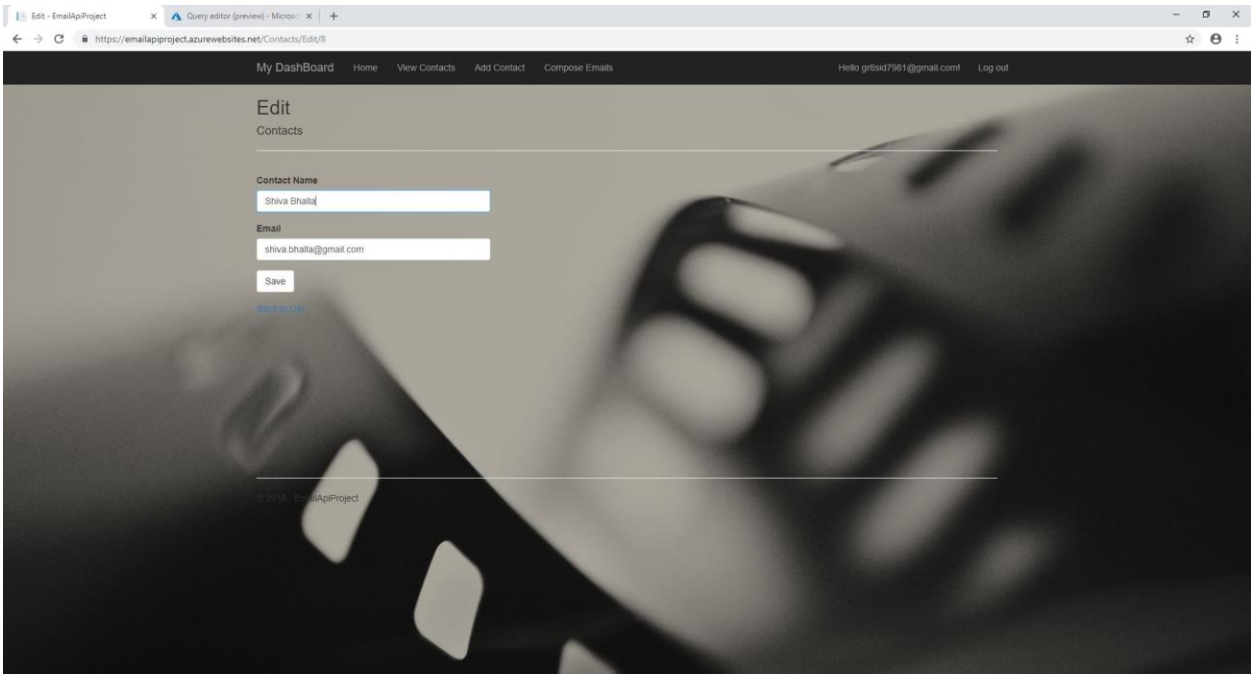

Fig. 2.8 Edit Contact Screen.

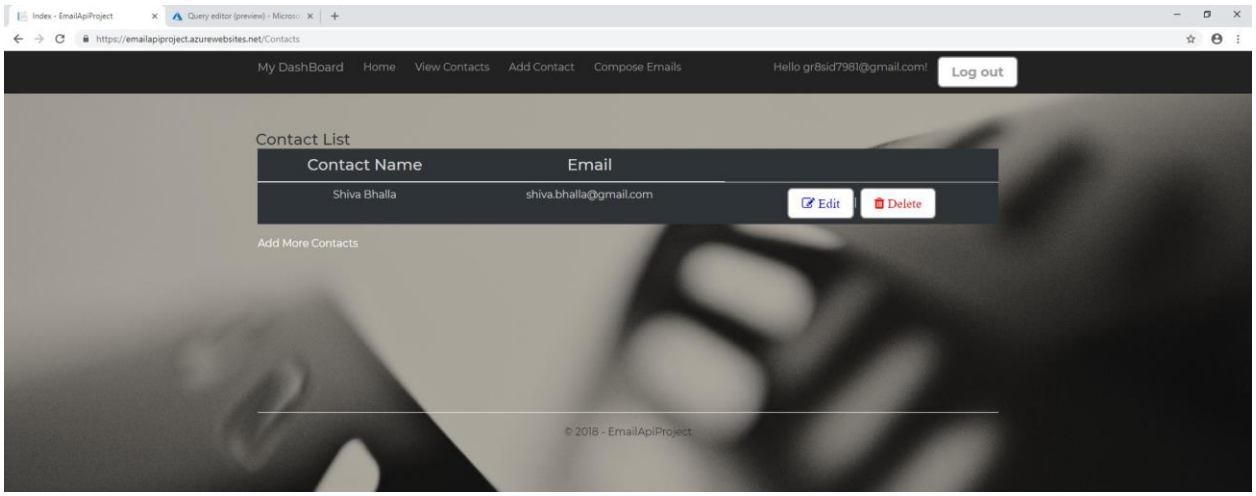

Fig. 2.9 Contact Screen after editing the contact.

e. A user can Delete Contacts from the List.

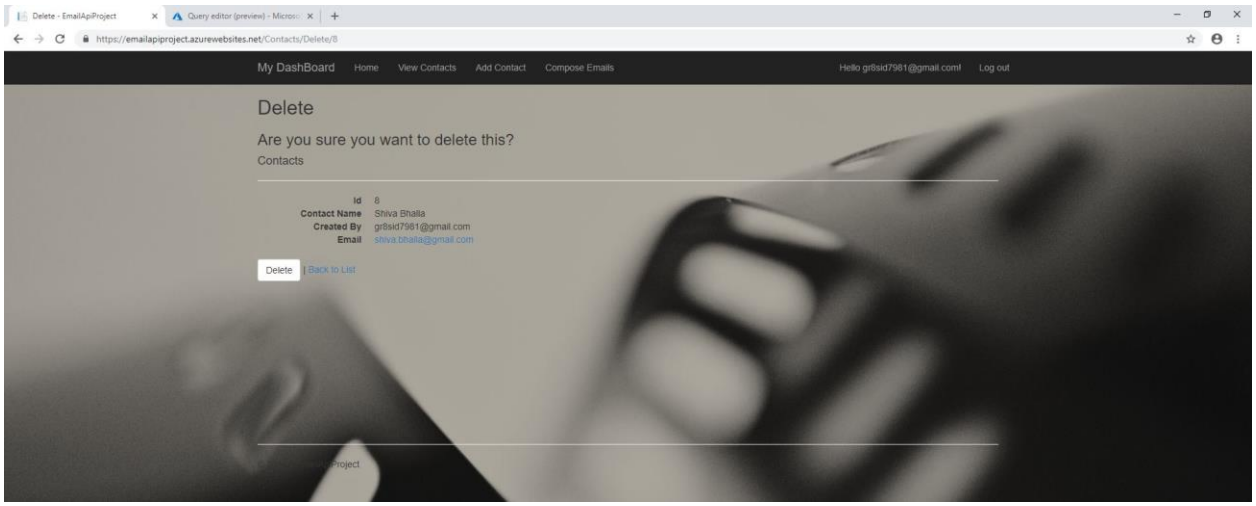

Fig. 2.10 Delete Contact Screen.

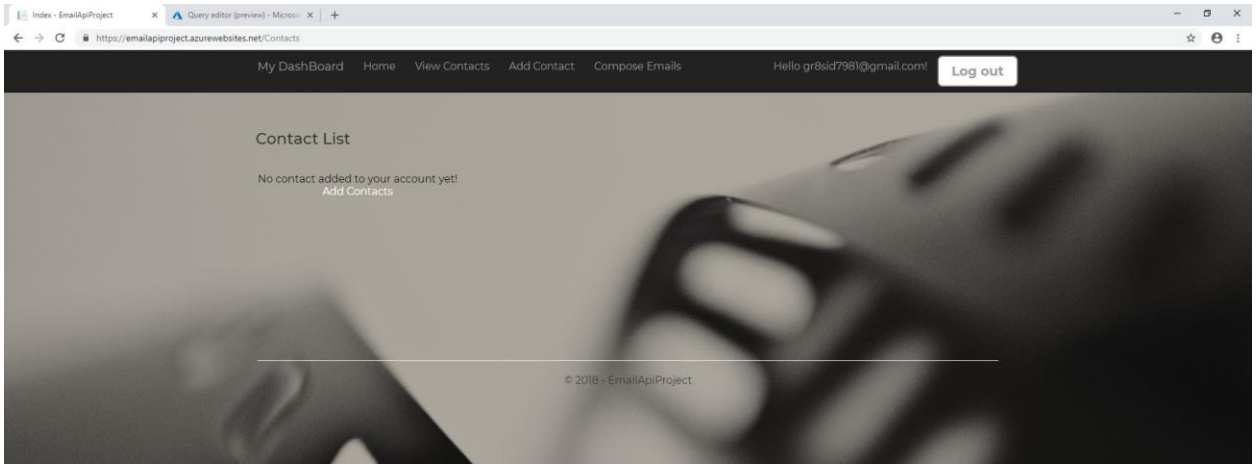

Fig. 2.11 Contact Screen after deleting the contact.

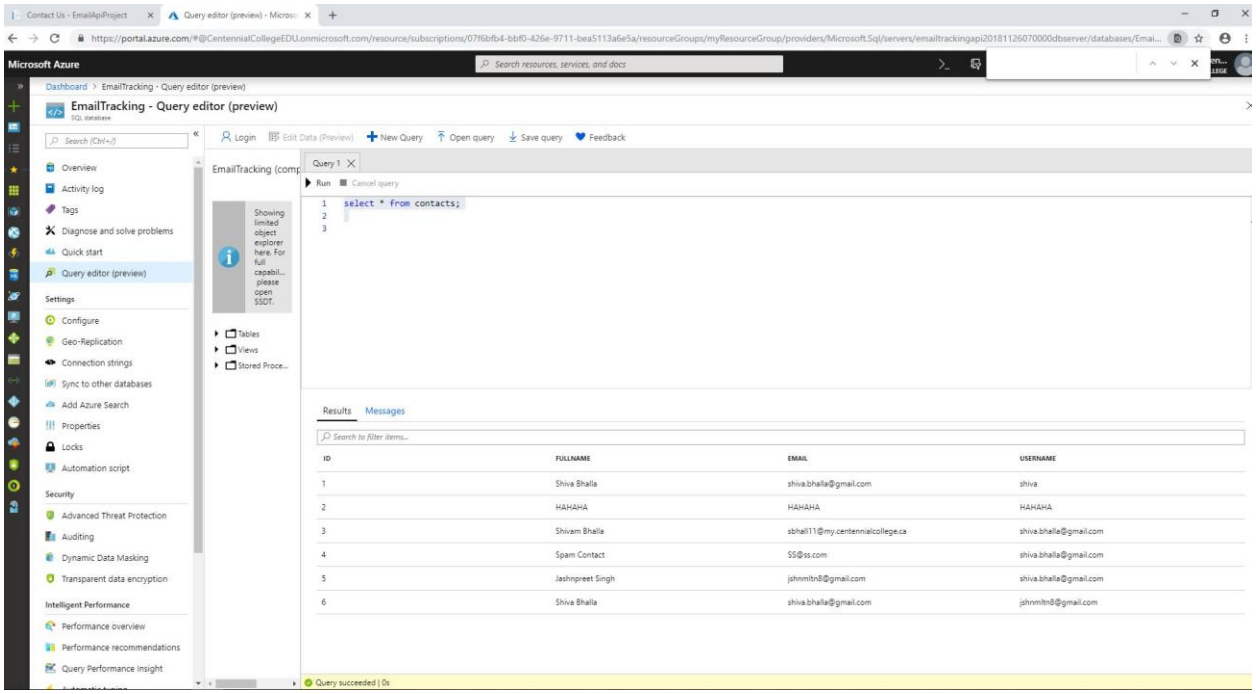

Fig. 2.12 Contacts Database after deleting the contact.

f. User can save his/her complete profile and can add more than one email to be tracked.
g. On compose email, user can compose or write a new email message and there will be a dropdown list of contacts saved in his/her directory. User can send the composed message to one or multiple contacts concurrently (using cc and bcc). All these Emails will be sent via SendGrid. User can check for the sending server by verifying the received email header, where it would be mentioned - sent via SendGrid.

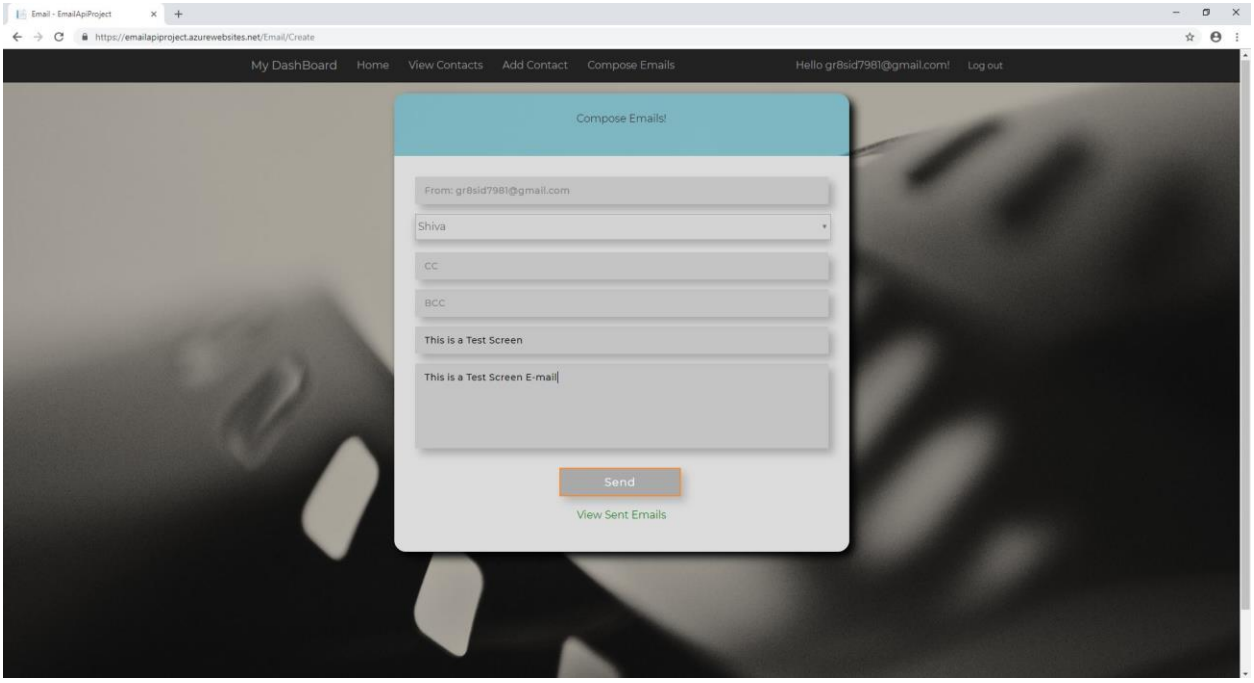

Fig. 2.13 Compose Email Screen.

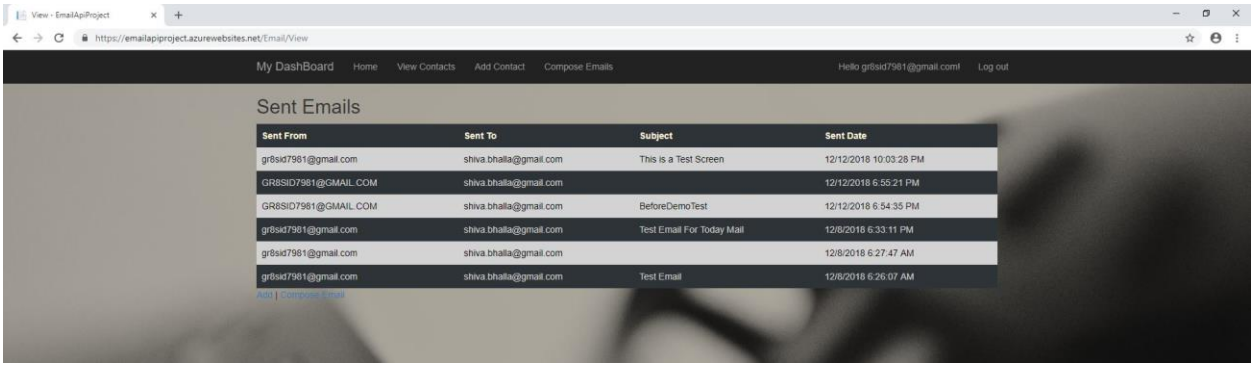

Fig. 2.14 Sent Email Screen.

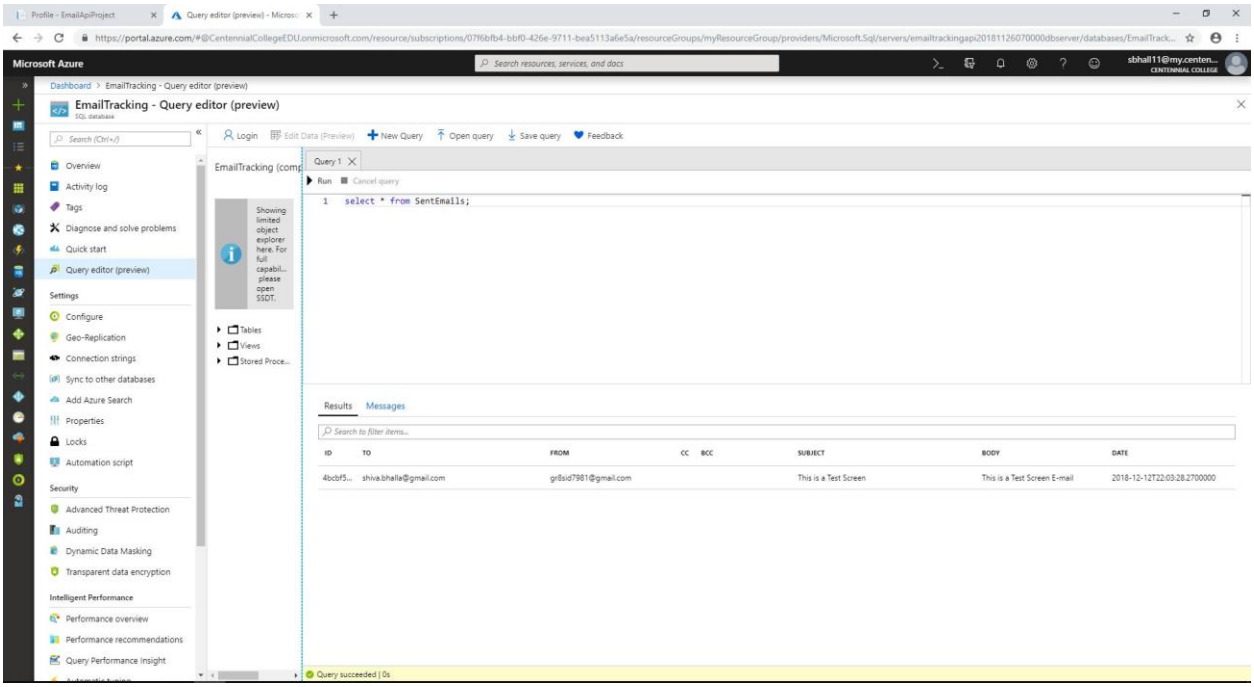

Fig. 2.15 Sent Email Database.

h. All the sent emails can be managed in the Outbox section. Here, user can Check the status of each email whether - read, opened, delivered, opt-out, spam or bounce. 'Yes' represents a True state for a particular email, 'No' representing the False.

i. Dashboard is the main feature of this web application where user can see different emails sent to a particular contact in a formatted list displayed on the dashboard screen.

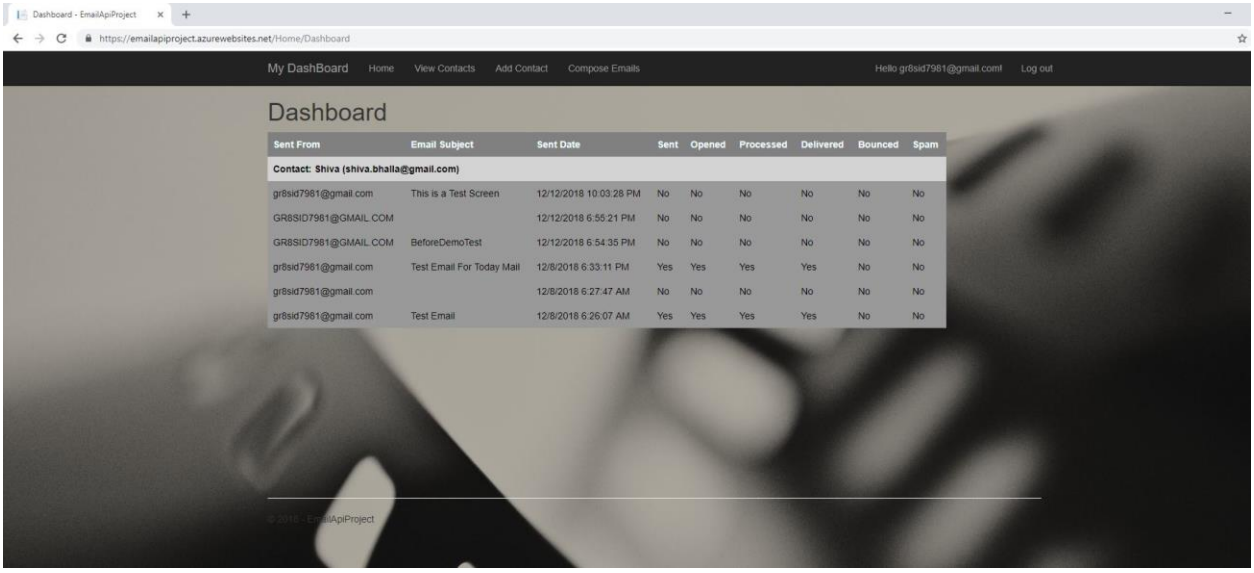

Fig. 2.16 Dashboard Screen after Email Sent.

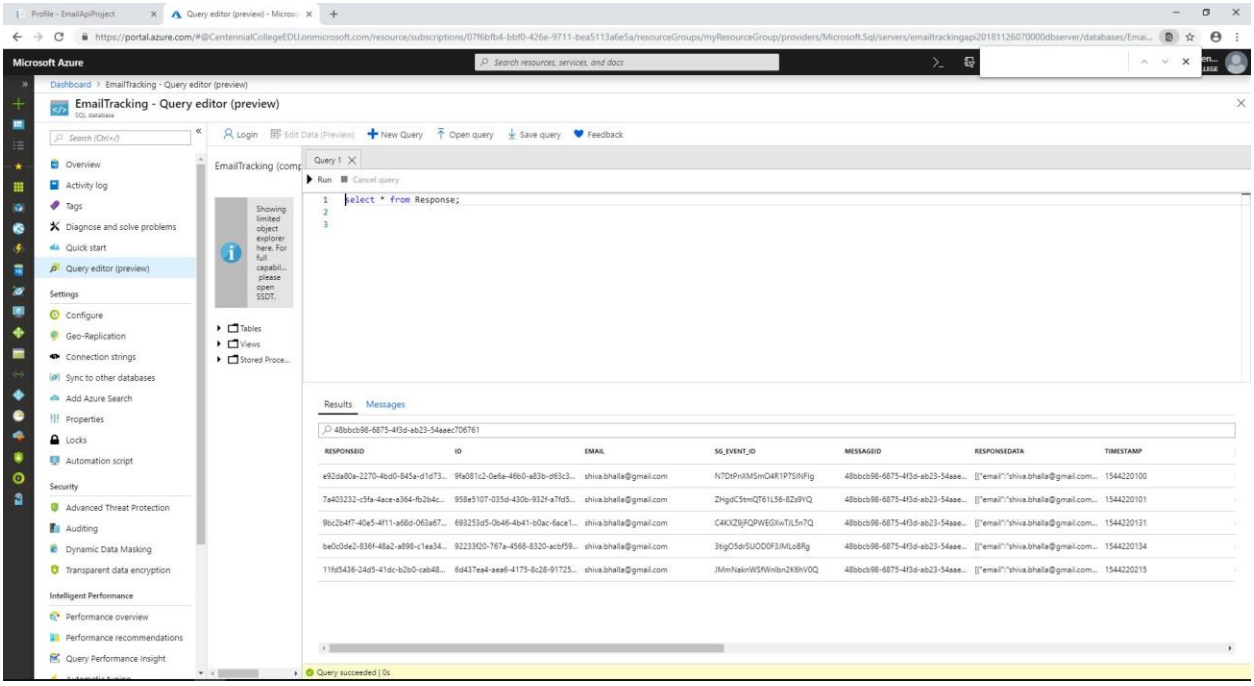

Fig. 2.17 Response Database populating the Dashboard Screen.

The purpose of this web application will be to facilitate customers in tracking their sent Emails. By accessing our application, customers would be able to track various statistics of their mails, for instance, whether the email was read by the recipient, the number of emails opted out by the recipients (of the bulk emails sent), the exact delivery status – filtering the delivered and undelivered emails. Moreover, users can be notified for the bounced emails.

For this web application, we will be using API provided by SendGrid. SendGrid tracks the emails based upon their unique Message ID attribute. Additional benefits of using this API would be to get the exact count of the sent emails. Activities like fetching the number of users who opened and read emails would return the count of users. These statistics are very crucial for any CRM enterprise, for the growth of their business, making a healthy relationship between the client and customers as Clients always know about their valuable and non-valuable customers

Technologies used: ASP .NET, JavaScript, jQuery, Ajax.

Cloud Services: Microsoft Azure (For Deploying the web application).

SQL Database in Microsoft Azure (Cloud)

LESSONS AND EXPERIENCES

1. Understood how the API's get implemented in real world.

2. APIs act as a doorway that people with the right key can get through.
3. APIs let applications (and devices) seamlessly connect and communicate.
4. APIs let you build one app off another app.
5. **Using Apigee** – Using Apigee an API Management tool. It is a full lifecycle API management platform that enables API providers to design, secure, deploy, monitor, and scale APIs.
6. **Using Postman** - Postman is a Google Chrome app for interacting with HTTP APIs. It presents us with a friendly GUI for constructing requests and reading responses.
7. **Using Swagger** - Swagger is the most widely used tooling ecosystem for developing APIs with the Open API Specification (OAS). Swagger consists of both open source as well as professional tools, catering to almost every need and use case.
8. Working on such projects in a team leads to double output of each individual and we get to learn a lot from each other.