

Scope Document

Bulk Bill Payment

GrameenPhone

Version 1.1

Table of Contents

[Table of Contents 3](#_Toc141341278)

[1 Introduction 5](#_Toc141341279)

[1.1 Purpose 5](#_Toc141341280)

[1.2 Scope 5](#_Toc141341281)

[1.3 Audience 5](#_Toc141341282)

[2 Bulk BillPayment Portal 6](#_Toc141341283)

[2.1 Login Flow: 6](#_Toc141341284)

[2.2 BillPayment Flow: 8](#_Toc141341285)

[2.3 Password Management Flow: 9](#_Toc141341286)

[2.4 Existing Flow Diagrams: 10](#_Toc141341287)

[3 Technical Solution Requirement 12](#_Toc141341288)

[3.1 Solution Overview 12](#_Toc141341289)

[3.1.1 Tech stack to be used to deploy and run existing code 12](#_Toc141341290)

[3.2 Deployment Specifications of existing module 12](#_Toc141341291)

[3.2.1 DB Readiness 12](#_Toc141341292)

[3.2.2 Application readiness 13](#_Toc141341293)

[3.3 Project Specific Scope 13](#_Toc141341294)

[3.3.1 Current Implemenation 13](#_Toc141341295)

[3.3.2 Graphical User Interface 13](#_Toc141341296)

[3.3.3 Training 14](#_Toc141341297)

[3.3.4 Documentation 14](#_Toc141341298)

[3.3.5 Postman Collection 14](#_Toc141341299)

[3.4 Testing and Acceptance 14](#_Toc141341300)

[3.4.1 System Integration Testing 15](#_Toc141341301)

[3.4.2 User Acceptance Procedure 15](#_Toc141341302)

[3.5 Risks, Impact and Mitigation 15](#_Toc141341303)

[3.6 Responsibility Matrix 15](#_Toc141341304)

[3.7 Out of Scope 15](#_Toc141341305)

[3.8 Code management 15](#_Toc141341306)

[3.9 Project handover 16](#_Toc141341307)

[3.10 Project Timelines 16](#_Toc141341308)

[3.11 Support and maintenance 16](#_Toc141341309)

[4 Document Change History 17](#_Toc141341310)

# Introduction

This document describes scope of work for Bulk BillPayment UI requirement for GrameenPhone

## Purpose

It is the scope of work document for the changes needed in current Bulk BillPayment module of Grameenphone to integrate with upgraded mobiquity PayX Solution. This will give flexibility to admin users to fetch bill details of account holders in bulk and then pay all pending bills in bulk.

## Scope

Grameenphone has already shared their code of existing Bulk BillPayment UI from old version and want same to be implemented in PayX solution provided by comviva in Phase B.

On existing feature, below are changes that need to be done:

* While fetching pending bill details of all bill account selected on UI : Change in existing integration logic and update to Mobiquity Integration API
* For Bill payment : Change in existing integration logic and update to Mobiquity Integration API
* Ambiguous payment handling scenarios(for enquiry and not reversal or refund) request will be routed to Mobiquity , for any dispute management
* Email and SMS handling should be there as per current setup : integration with Email server and SMSGW
* Login page to be created on WEB to enable user login with credentials. In old system, login page was from IDP. But in latest deployment Login feature should be created in Bulk UI itself.
* Password management (change and forgot password) feature has to be implemented on Bulk UI : Integration with Mobiquity for backend logic

Note : There are 2 UI for Bulk Bill Payment and Reporting. Current scope is just to give feature of Bulk Billpayment UI . Reporting UI requirements will be fulfilled from Pentaho which is part of Mobiquity solution.



## Audience

Comviva technical teams, partner teams

-

# Bulk BillPayment Portal

Admin user would be able to access Bulk BillPayment portal. Each of them will have their specific userId and password

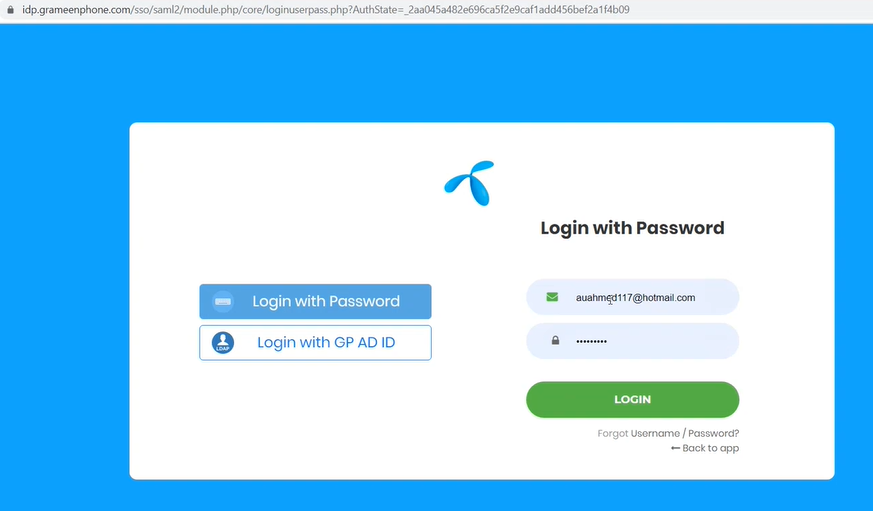
## Login Flow:

**Current Flow:**

In old architecture, User authentication while Login is maintained by IDP. So whenever user is logging into the system, application routes user to login page of IDP. Where userid and password is asked from user.

IDP validated credentials, on success it calls Billpay Dashboard page and give access to admin user.

**Login Page of IDP :**



**BillPay Dashboard snapshot is below:**

A person holding a phone

Description automatically generated

**New Flow:**

Partner need to create this Login page into Bulk UI code itself. And w.r.t. authentication, hit is send to Mobiquity for user validation. Since there is no password storage in Bulk Bill mflex DB. Only userId and email id of users are stored. If user is verified by Mobiquity, then dashboard will be shown.

In order to support this approach, we must ensure that admin users who are present in mobiquity systems can only access bulk billpay dashboard. And they must need mobiquity login id and credentials only to login after registration in Bulk Billpay application. When any mobiquity user want access on Bulk Pay dashboard, then they have to get registered in BulkPay UI also with same login ID.

So login ID of user in mobiquity and BulkUI DB should match.

BulkUI application will send request to generate system token at time of login, once system token is fetched, login request will be sent with system token and login credentials and in return if user is valid in mobiquity system then userToken is sent in response to Login API along with refresh Token.

On receiving successful login, Bulk UI dashboard must be made visible to user.

**Expections from Partner:**

* Create Login page
* Integrate with mobiquity to fetch system token and Login validation
* Basis mobiquity Login API response, route request to already existing Bulk dashboard in system.

## BillPayment Flow:

**Current Flow:**

* On UI (named as CBP / web-portal-application) : to fetch bill details of accounts and to pay pending bills in bulk, request is initiated from UI and it is stored in CBP DB for further processing. At this point request is just queued, no fetching or processing has take place.
* Second component scheduler will then pick these request from DB on timely basis and communicates with communicator component which will make hit to backend financial system(mobiquity) for Payment and third party system(Biller platform) for actual processing

**New Flow:**

**Process flow will be similar to existing approach , but only change in integration.**

* On UI (named as CBP / web-portal-application) : to fetch bill details of accounts and to pay pending bills in bulk, request is initiated from UI and it is stored in CBP DB for further processing. At this point request is just queued, no fetching or processing has take place.
* Second component scheduler will then pick these request from DB on timely basis and communicates with communicator component which will make hit to backend financial system(mobiquity) to fetch bill details or Payment request instead of Biller Platform.
* In fetch bill details it is simple API hit to mobiquity to fetch bill details instead of biller platform.
* For Payment also, communicator will send payment hit to just mobiquity. Not to biller. Mobiquity will in turn send hit to biller platform for any kind of third party integration. Communicator will not interact with biller platform.
* While sending hit to mobiquity, CBP has to ensure a unique reference ID is created, stored in mflex DB corresponding to each bill request and is sent to mobiquity to manage for any disputes in future.
* Communicator will just wait for Mobiquity response of Payment API. Response handling is mentioned below:
  + If it is Failed, then payment is also marked failed in mflex DB.
  + If success/pending or fulfilment initiated : then it is assumed that request is properly posted in mobiquity and now to check status of payment etc. mobiquity DB/ reports has to be inspected. All mobiquity reports will show corresponding unique ID created by CBP so as to track the transaction in mobiquity.
  + If no response from mobiquity for payment request, then all such transactions will be enquired in mobiquity with unique reference ID created at time of payment.
  + If success/fail response of enquiry then status is cleared in mFlex DB.
  + But if no response of enquiry, for a predefined configurable retries, then those are left uncleared in mflex DB, and all such transactions can be checked in Mobiquity reports.

**Note :**

Scheduler will pick only those records for enquiry for which they didn’t get response back from mobiquity platform. No retry feature will be available in this scenario. If scheduler gets response back for payment API, then also any reference to the payments and third party response should be from mobiquity reports. It should not be checked from Bulk Billpay UI.

**Expectations from partner :**

* Communicator code need to be updated so it can now integrate with new mobiquity platform with updated API specifications to fetch bill details and for payment. These will be routed to mobiquity instead of direct biller integration.
* Logic of unique ID creation to be sent in mobiquity payment APIs
* Mobiquity response handling and Ambiguous payments settlement logic (enquiry , **not refund**) need to be updated accordingly in BillPay code.
* Email and SMS handling should be there as per current setup : integration with Email server and SMSGW

## Password Management Flow:

When any user forgets his password then user should have flexibility to get these reset via forget password feature on login page. Also once he login, then user can change his password on his own.

**Current Flow:**

In current platform, all password management is done at IDP page.

**New Flow and Expectations from Partners:**

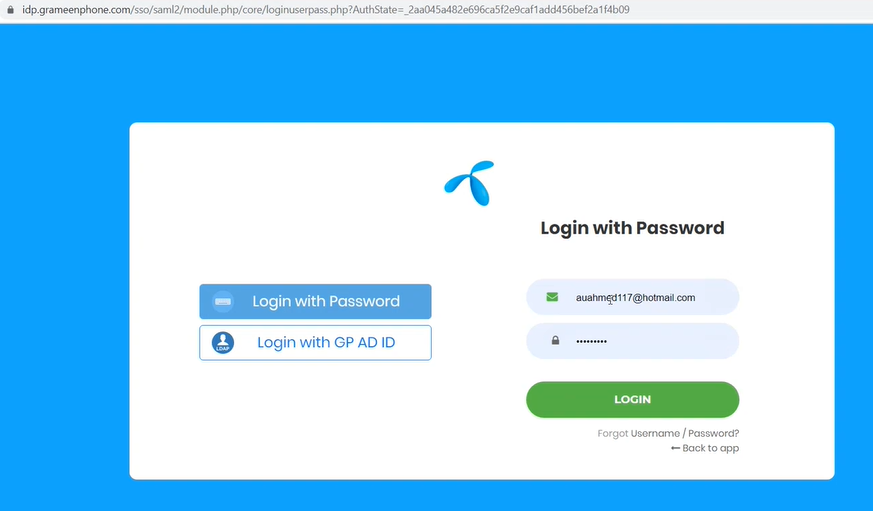
Password management to be incorporated on Bulk UI portal in 2 ways:

* 1. Give forgot password option on login page.
  2. User himself once logged in, can be given with Change password option

All 2 APIs need to have direct integration with Mobiquity for forgot password and change password.

Mobiquity will share APIs to achieve same. Partner can design pages.

Forgot password option may come like below:



Change Own password or reset password pages are new.

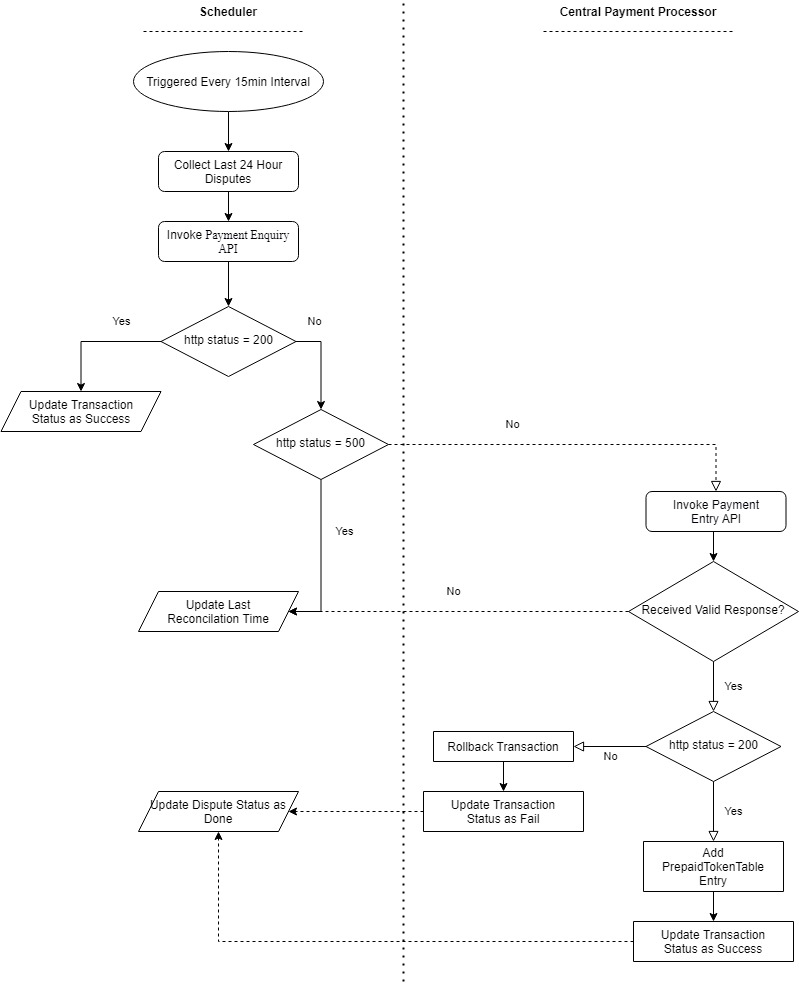
## Existing Flow Diagrams:

**Flow diagram of Billpayment Request**

A diagram of a flowchart

Description automatically generated

**Call flow of scheduler : to pick bulk request(fetch bill and payment) initiated from UI**



Note:

* All technical training session happened with Wipro will be shared with partner that will give detailed view of current implemented request flow .
* Postman collection of existing system APIs is also shared.
* Git code link of existing system will be shared by Comviva to partner

# Technical Solution Requirement

## Solution Overview

1. the solution must be created using open-source technologies widely accepted and used by the industry.

### Tech stack to be used to deploy and run existing code

1. Comviva suggest to use below listed technology stack for the development of the application:

|  |
| --- |
| Angular 9.1.1 |
| npm 6.13.4 |
| Node 12.16.1 |
| tomcat 9 or above |
| Jdk 14 |
| MySql |

## Deployment Specifications of existing module

1. The solution must be deployed with below steps in order.

### DB Readiness

* Install MySql in your local system.
* Install MySql Workbench in your local system.
* execute Below all the files in MySQL Workbench for setup database in your local system.

z

### Application readiness

* There are Four Modules which need to be deployed in same order
* 1. web application
* 2. async job
* 3. mfs-communication
* 4. gpay-C

Steps to deploy:

* ( Web Application – master)

- Go in this folder

1. cd src/ng

2. open command prompt or git bash

3. npm init -y

4. npm install

5. npm run dev

- Now, open the project and do some changes in .properties (DB configuration, etc…)

- run main class which have @SpringBootAplication annotations.

- open any Browser and hit : <http://localhost:8080/>

## Project Specific Scope

### Current Implemenation

1. All technical hands-on recording will be shared with Parner

### Graphical User Interface

1. GUI should be as below:
2. A person holding a phone

   Description automatically generated

Rest can be referred from Recordings:

### Training

1. Once the application is delivered on site, the partner must conduct technical training to the Comviva development team and provide necessary information and code walk through for changes made in existing Code.

### Documentation

Flow diagram has been attached above to explain process flow of Request

### Postman Collection

Attached is postman collection is existing system APIs to integrate within system and with biller platform



Note: Comviva will share it’s API (Fetch bill, payment, ambiguous settlement)

## Testing and Acceptance

### System Integration Testing

Application must pass all the test cases defined by partner’s internal QA. Comviva QA will audit the report and also may execute their own designed cases as per the functionalities if required, Partner must fix all issues raised by Comviva QA.

Note: Any bug identified during QA testing which is present in existing/new code must be fixed by Partner.

### User Acceptance Procedure

Once the application in cleared by Comviva QA, it must be offered to customer for their testing and acceptance. User acceptance test cases must be shared by partner and passed in order to get user acceptance certificate. All UAT defects must be fixed by Partner.

## Risks, Impact and Mitigation

1. Partner must identify and define all the risks and share its impact as well as mitigation plan.

## Responsibility Matrix

1. Partner must prepare and share responsibility matrix for various tasks required to be executed during the lifecycle of this project.

## Out of Scope

1. Third party (Biller )APIs integration would be out of scope for partner, though partner must define their own list of out of scope items and share with us.
2. All third party integrations will be under mobiquity scope. Partner has to just integrate with Mobiquity Payment APIs.
3. **Reporting UI shown in demo is out of scope of partner . Mobiquity will support all report related need through existing reports present in product.**

## Code management

1. Partner must commit and update all the codes, designs, and documents into Comviva version control systems (GIT) and provide necessary access information for Comviva development team. Partner should adhere to all the Comviva software development guidelines and follow GDPR processes.

## Project handover

1. Once the project is completed, partner must handover updated software to Comviva and conduct various workshops (technical, functional, non-functional etc.) for teams and provide them hands on experience and KT to enable them to take care of future change requests as well as bug fixes. This will be for changes done by Partner on top of existing code.

## Project Timelines

1. This project must be completed within the defines and agreed timelines. The project is expected to be completed within 30 days.

## Support and maintenance

* Partner will be responsible for UAT as well as production defects
* Partner must provide technical support and maintain the project upto 1 year after delivery.
* Partner must assign a dedicated team to develop and support this project
* Production defects must be fixed with in defined SLA.

# Document Change History

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Version** | **Change Type**  **(A/M/D)** | **Change Description** | **Prepared By** | **Reviewer** | **Approved by** | **Date** |
| 1.0 | A | Initial version | Anshika Aggarwal | Pranay Agrawal | Kamal Kishor Arya | 27-July-2023 |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

Disclaimer

Copyright © 2021:  Comviva Technologies Limited 5th, 7th & 8th floor, Capital CyberScape, Village Ullahwas, Sector 59, Golf Course Extn. Road, Gurugram – 122102, Haryana, India.

All rights about this document are reserved and  shall not be , in whole or in part, copied, photocopied, reproduced, translated, or reduced to any manner including but not limited to electronic, mechanical, machine readable ,photographic, optic recording or otherwise without prior consent, in writing, of Comviva Technologies Ltd  (the Company).

1. The information in this document is subject to changes without notice. This describes only the product defined in the introduction of this documentation. This document is intended for the use of prospective customers of the Company  Products Solutions and or Services for the sole purpose of the transaction for which the document is submitted. No part of it may be reproduced or transmitted in any form or manner whatsoever without the prior written permission of the company. The Customer, who/which assumes full responsibility for using the document appropriately. The Company welcomes customer comments as part of the process of continuous development and improvement.
2. The Company, has made all reasonable efforts to ensure that the information contained in the document are adequate, sufficient and free of material errors and omissions. The Company will, if necessary, explain issues, which may not be covered by the document. However, the Company does not assume any liability of whatsoever nature , for any errors in the document except the responsibility to provide correct information when any such error is brought to company’s knowledge. The Company will not be responsible, in any event, for errors in this document or for any damages, incidental or consequential, including monetary losses that might arise from the use of this document or of the information contained in it.
3. This document and the Products, Solutions and Services it describes are intellectual property of the Company and/or of the respective owners thereof, whether such IPR is registered, registrable, pending for registration, applied for registration or not.

The only warranties for the Company Products, Solutions and Services are set forth in the express warranty statements accompanying its products and services. Nothing herein should be construed as constituting an additional warranty. The Company shall not be liable for technical or editorial errors or omissions contained herein.

1. The Company logo is a trademark of the Company. Other products, names, logos  mentioned in this document , if any , may be trademarks of their respective owners.
2. Copyright © 2021:  Comviva Technologies Limited. All rights reserved.
3. **Thank You**

Visit us at [*www.comviva.com*](http://www.mahindracomviva.com)