**Chapter 1**

**Introduction**

**1.1Overview**

The Most annoying thing if you are a publisher or a website or simply want information and access to a tender from the internet you first have to download the file to your hard-drive and then upload to the file hosting company like Google. This thing takes up too much time plus if you are on limited bandwidth plan then it will make you grab your hair one by one and pull them out because of the excessive data usage.

Wouldn't that be awesome if you somehow transfer the file directly from the internet to the file host and then share it with all the friends or colleagues? The answer is yes and it is called Remote Url or file upload. Many Cloud hosting sites offer such option but they delete the file after some time because of inactivity. Then the real or original file is lost and you have to re download and then re upload and do the whole procedure again and again for all or the inactive files. But most companies like Google don't have any issue with you in activity and they give you maximum space. But they don't support direct file upload or URL upload. So what can you do? Do the procedure of downloading and uploading and then sharing or find and alternative? The Alternative is here.

Simply put the URL of the file you want to upload in the File Url and then click on Upload to Google Drive. Then on the next page you will be asked to allow the application to upload or something like that and then you are done. Your account will be linked to the uploaders and it is totally safe so don't worry about anything.

It reduces time to load file in Cloud storage. It also reduce user’s bandwidth usage if in limited data plan, because we don’t need to first download the file from source and then upload it on cloud. It save file to cloud storage for future use as some time file get moved from URL. It can also Share and download file without login cloud storage site.

**1.2 Motivation**

We all using UC browser for web surfing and downloading. It is the most easy and efficient browser for mobile use.UC browser provided one feature for downloading that is cloud download. In that feature we can upload any remote file to the cloud storage. The benefit of this feature is we save our time and data pack.

The downloading is takes place on server to server, therefore the speed of downloading is very fast. But as the time goes, Government banned that feature of UC browser, so the users are very upset even our friends also want that feature again. So the thought was comes in our mind that, why don’t we develop such featured application? So now that thought is became reality, and we develop a web application who can upload remote files.

**1.3 Problem Statement**

The Most annoying thing if you are a publisher of a blog or a website or simply want to share a file from the internet.

You first have to download the file to your hard-drive and then upload to the file hosting company like Google.

This thing takes up too much time plus if you are on limited bandwidth plan then it will make you grab your hair one by one and pull them out because of the excessive data usage.

Wouldn't that be awesome if you somehow transfer the file directly from the internet to the file host and then share it with all the friends or colleagues? The answer is yes and it is called Remote Url or file upload.

Many Cloud hosting sites offer such option but they delete the file after some time because of inactivity.

Then the real or original file is lost and you have to re download and then re upload and do the whole procedure again and again for all or the inactive files.

But most companies like Google don't have any issue with you in activity and they give you maximum space. But they don't support direct file upload or URL upload.

So what can you do? Do the procedure of downloading and uploading and then sharing or find and alternative.

The Alternative is here: Simply put the URL of the file you want to upload in the File Url and then click on Upload to Google Drive.

**1.4 Objectives**

To provide an interface for upload remote files on cloud storage and that interface will be user familiar and easy to use.

**1.5 Organization of Report**

This report has nine chapters beginning with a brief introduction and motivation in Chapter 1. Chapter 2 describes the modules of our process and the methodology we used and provides detailed information on the architecture of the Backend IT process. Chapter 3 presents the brief information on the web extractor. Chapter 4gives a description of transferring the data from regional to live server using Local2Live app. Chapter 5 presents the report on CMS Entry Panel implementation to finish and polish the unfinished data as well as manual entry of tenders. Chapter 6 explores the Archive Applications to dump the expired data in a reliable and manageable storage. Chapter 7 is all about the detailed information of displaying the uploaded data on the website using Level-Up. Chapter8 is the structural description of the Database design used to store and transfer data. Chapter 9 gives the conclusion to the process and vision of the future work which can be implemented.

**Chapter 2**

**Proposed Architecture**

**2.1 Introduction**

For system developers, they need system architecture diagrams to understand, clarify, and communicate ideas about the system structure and the user requirements that the system must support. It's a basic framework can be used at the system planning phase helping partners understand the architecture, discuss changes, and communicate intentions clearly.

In this Backend process we are using the Client-Server Architecture where multiple web-extractors upload and insert data to their respective regional servers. These regional servers then push the data to the website’s server which then eventually reflects on the website.

**2.2 Architecture Model Overview**

An architectural model (in software) is a rich and rigorous diagram, created using available standards, in which the primary concern is to illustrate a specific set of tradeoffs inherent in the structure and design of a system or ecosystem modeling in general can be done at different levels of abstraction. It is useful to model the architecture of a specific application, complete with components and interactions.

**2.3 Architecture of Backend IT Process Diagram**

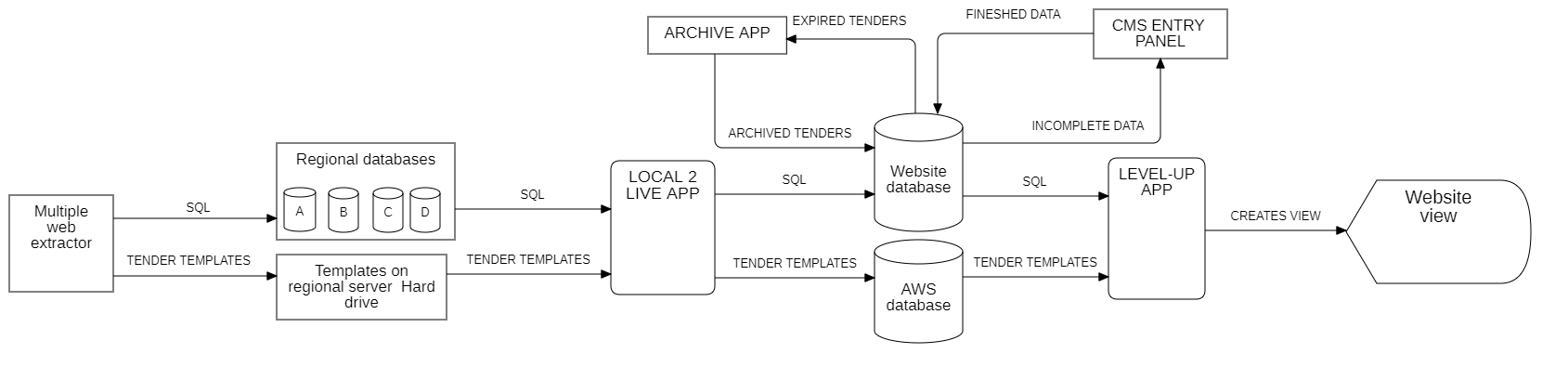


Fig 2.1 Architecture of Backend IT Process Goodtenders

In this backend IT structure there are multiple applications and processes which play an important role to maintain smooth flow of data. In this backend process there is a hunt made for an official tender offering website, then a specific web extractor is build to scrape the tender related data, this data is filtered using different attributes and translated and then it is given a easy to read format and then this data is inserted and stored into a regional server which consists of databases and network mapped hardware. Using Local2Live App which is build to upload data on the live servers the tender data is then transferred to the website database and the html easy to read templates is uploaded on the AWS database. Gradually LevelUp App creates a view for this newly uploaded data on the website and a link to the respective html template is attached to the view and makes it available for the users on the website. As there are even more apps to this backend process which includes Archive App and CMS Entry panel, respectively apps dump the expired tenders to the archive table year wise and manual entry of the unfinished data is done.

**2.4 Modules**

a. Web Extractor

b. Local2Live App

c. Level-Up App

d. Archive App

e. CMS Entry Panel

f. Database Design

## 2.5 Use case Diagram

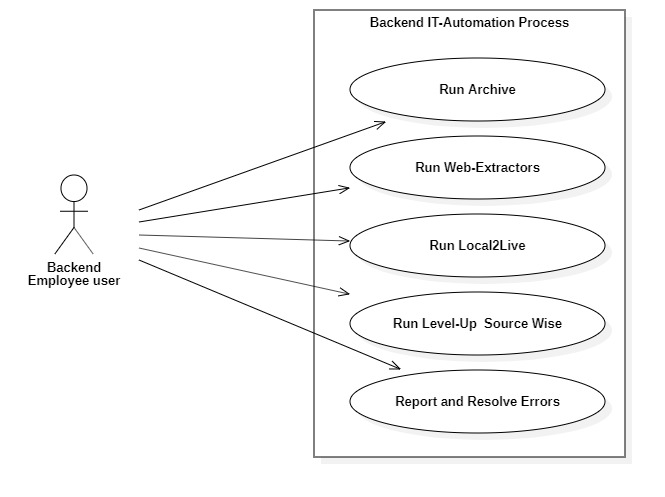


Fig 2.2 Use case Diagram

## 2.6 Entity-Relationship Diagramp1eg6n3tptc2p4h71pc0ht51ba24-0.jpg

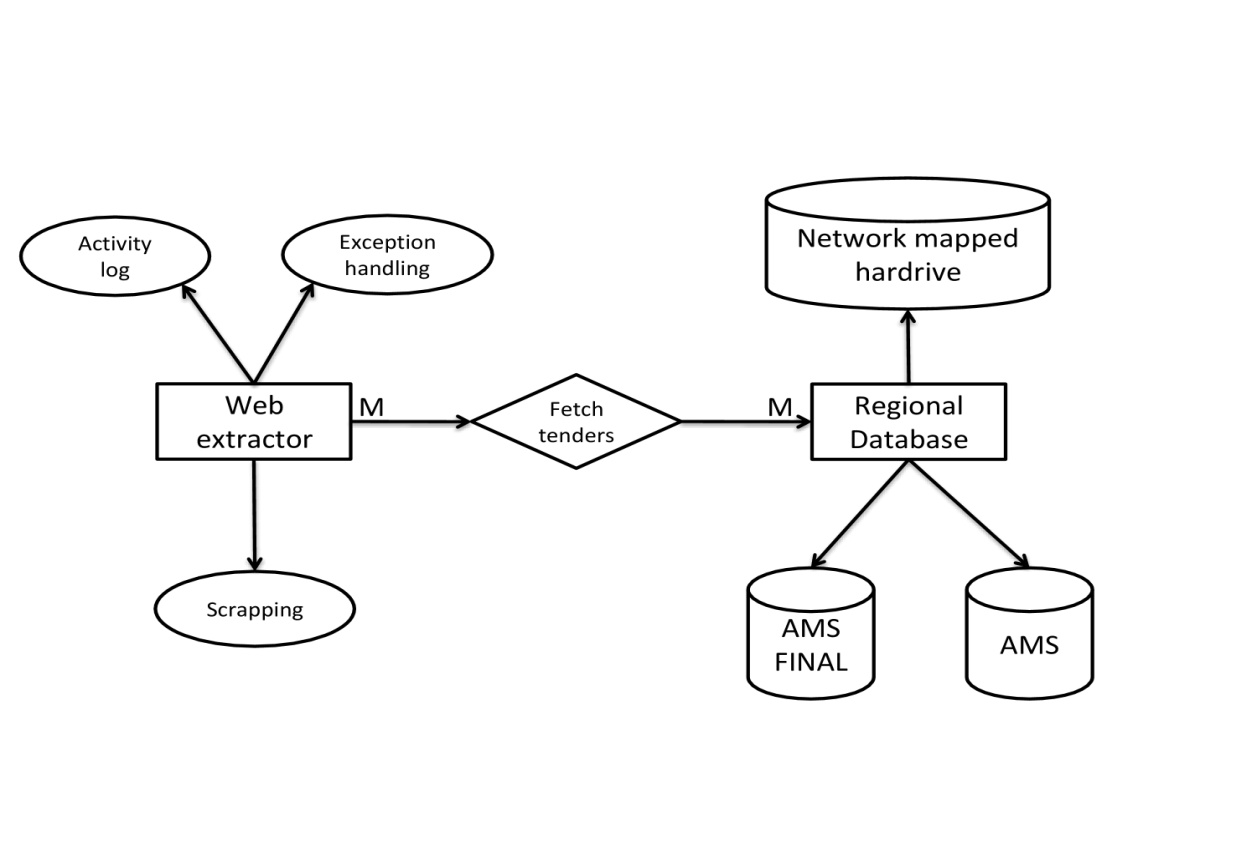
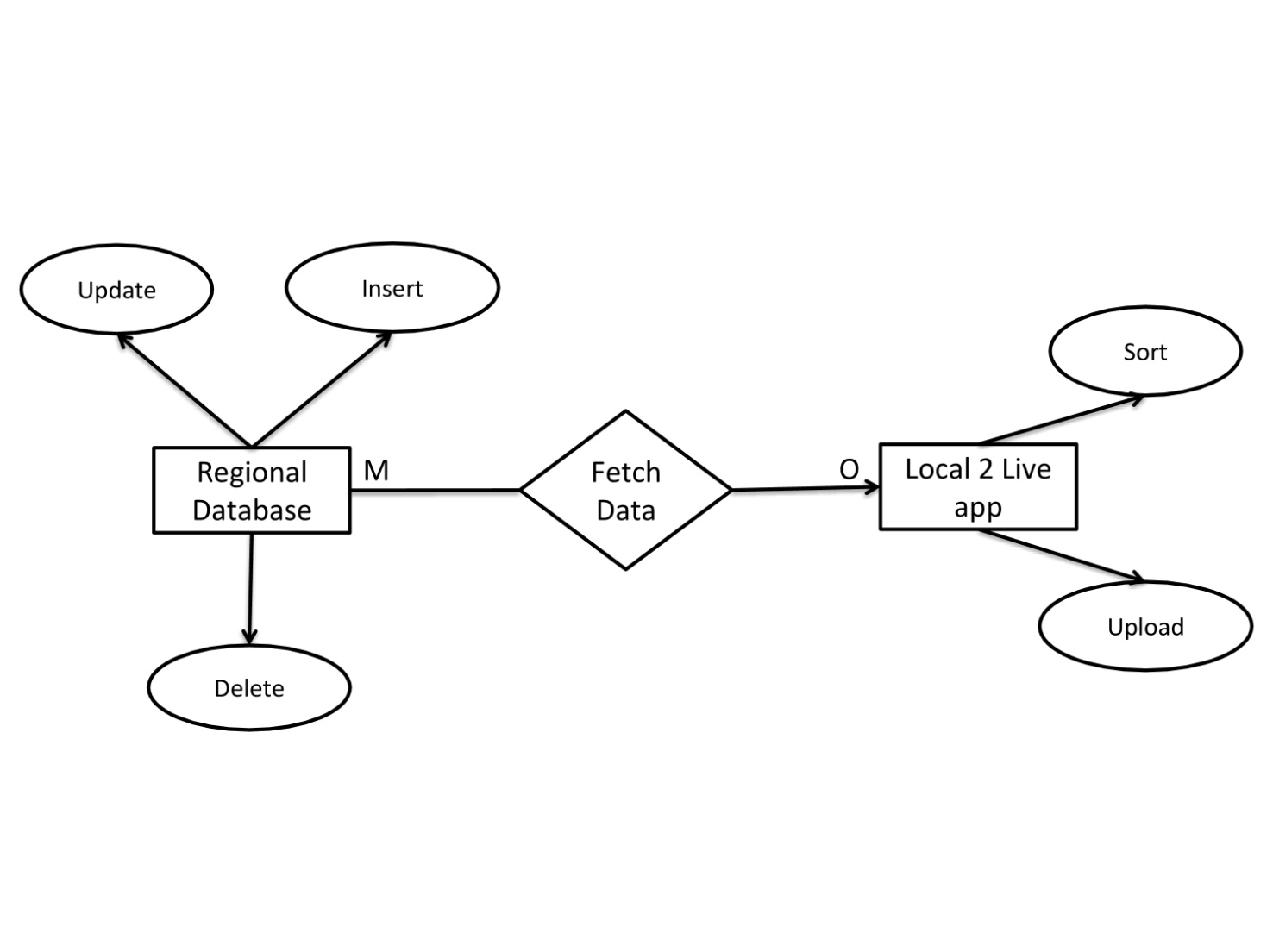
Fig 2.3.1Tender Website-Web Extractor E-R Diagram 

Fig 2.3.2 Web Extractor-Regional Database E-R Diagram

 Fig 2.3.3 Regional Database-Local2Live App E-R Diagram

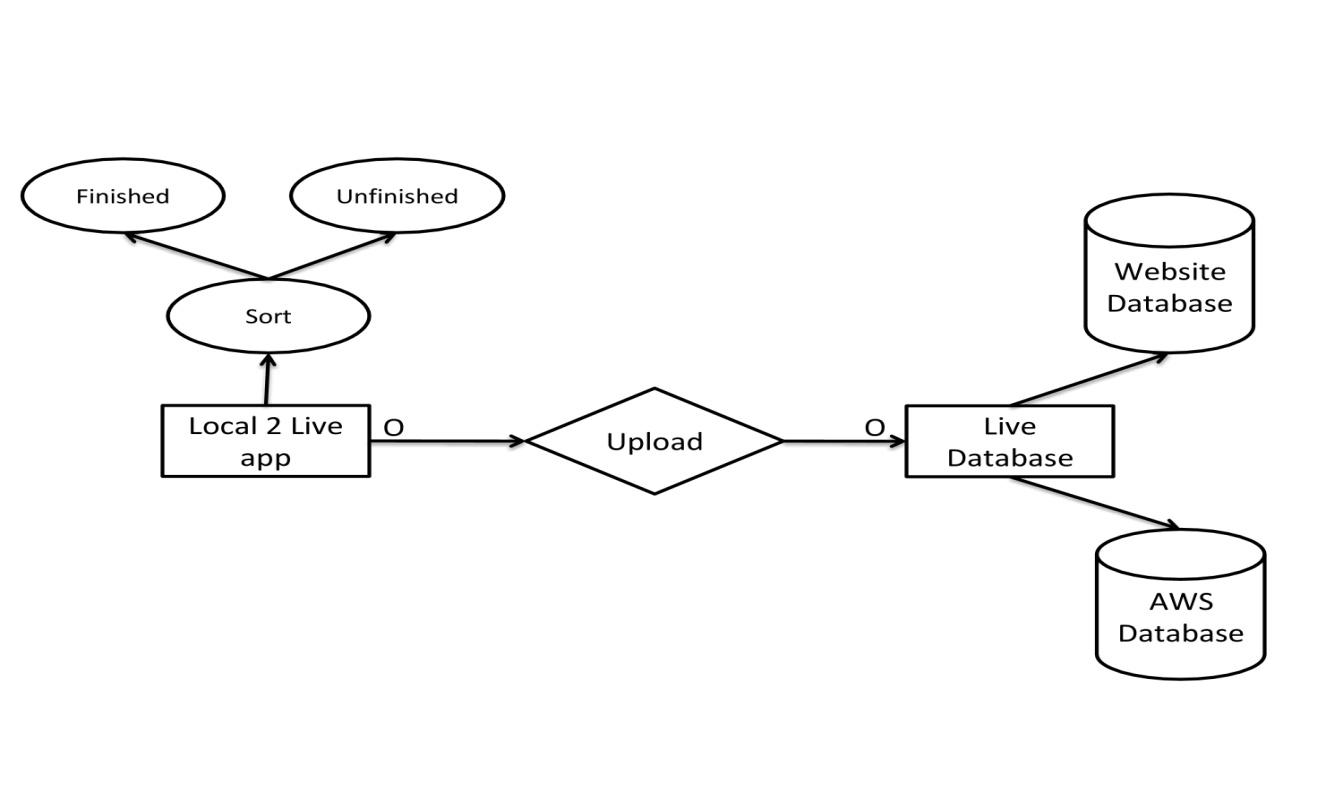


Fig 2.3.4 Local2Live App-Live Database E-R Diagram

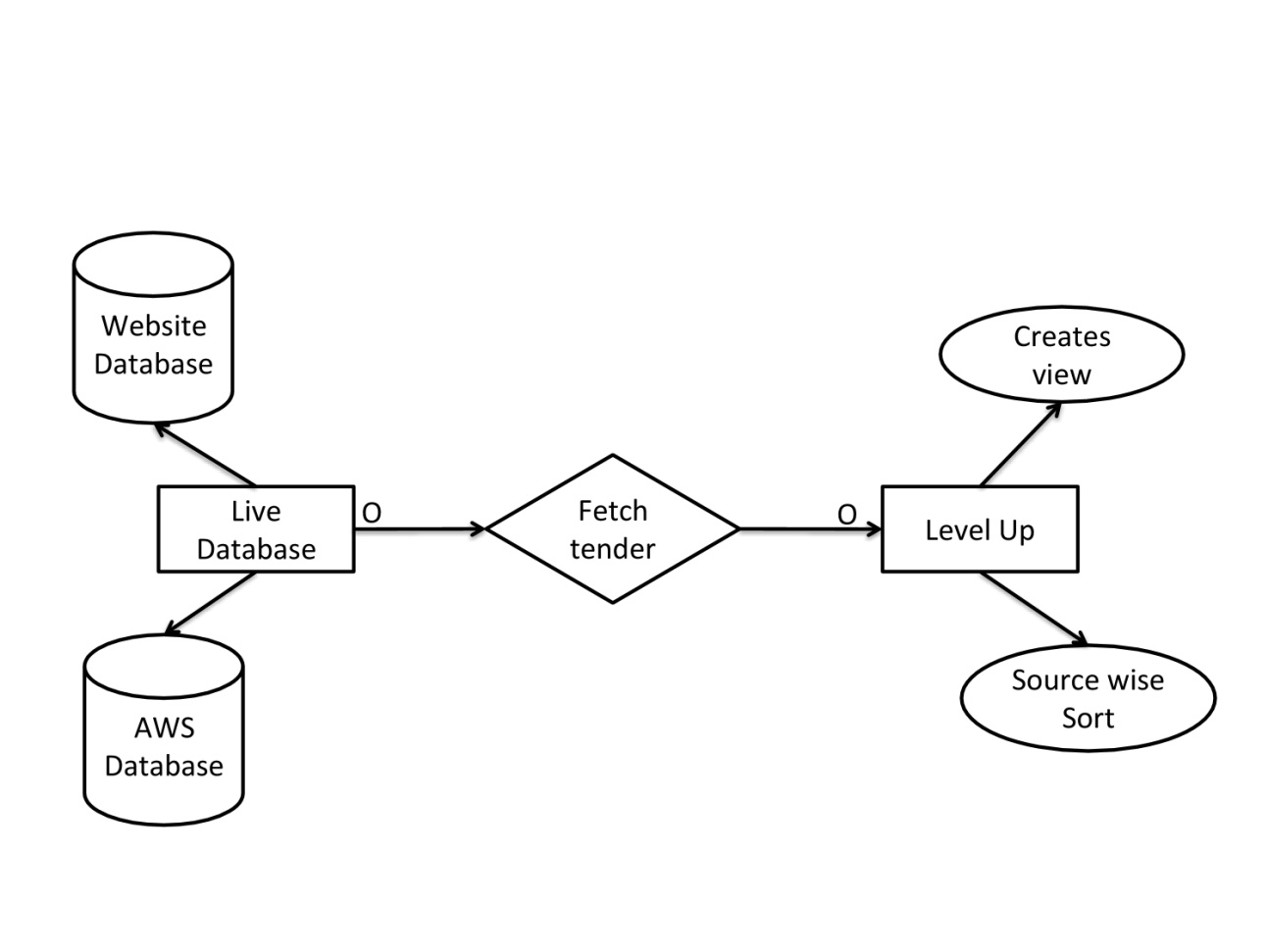


Fig 2.3.5 Live Database-Level up App E-R Diagram

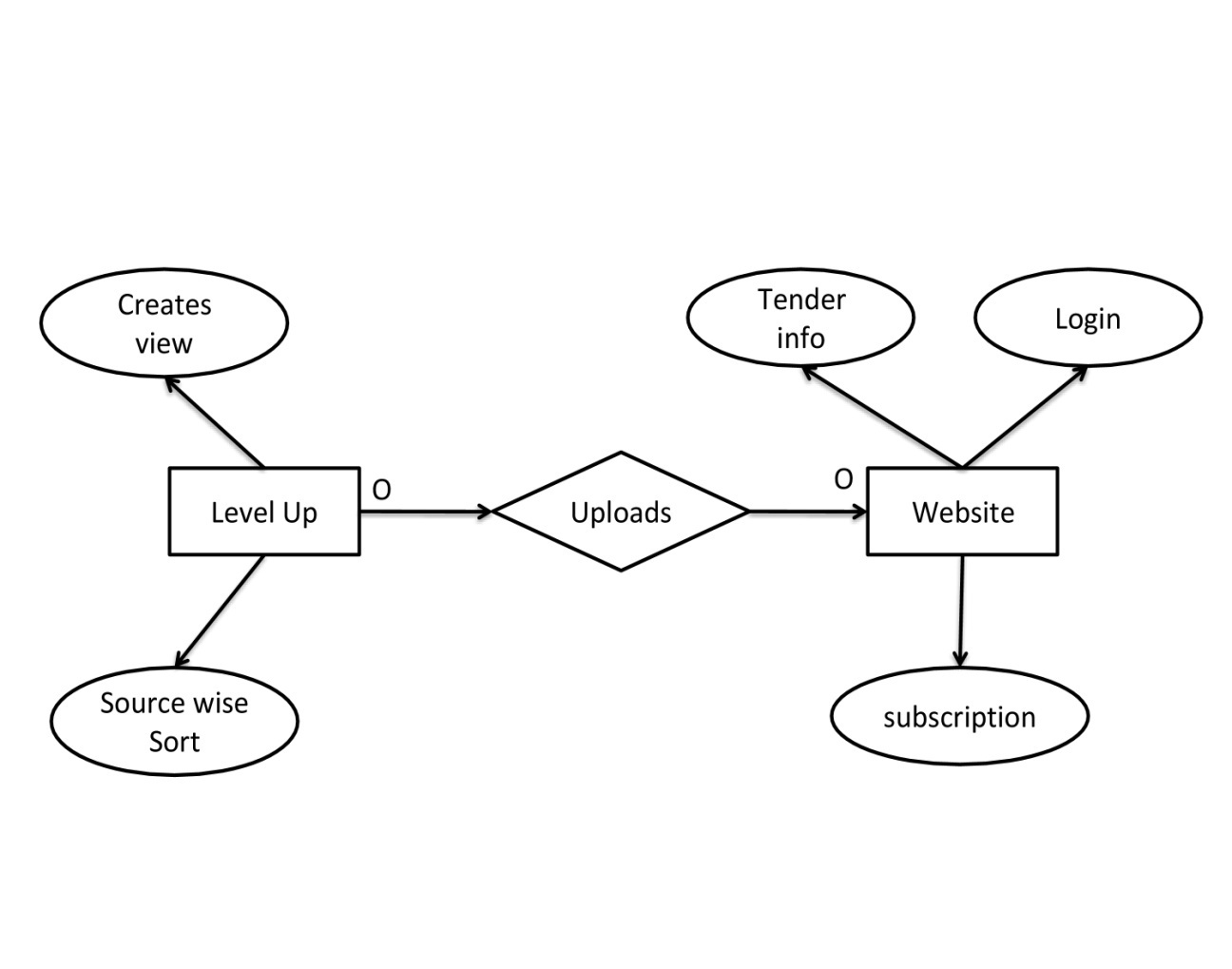


Fig 2.3.6 Level up App-Website E-R Diagram

**Chapter 3**

**Web Extractor**

**3.1 Overview**

Web Extractor is a software application which is made on a asp.net framework which uses C# as a backend programming language. These Web Extractor applications are built using Microsoft Visual Studio 2017 desktop express version. This Visual Studio provides an environment in which windows form applications are created which is the basic structure of this Web Extractor application. The Web Extractor consists of a web browser which is embedded into the windows form built structure and it consists of various information providing labels and textboxes and progress bar etc.

**3.2 Work Flow**

In this backend process Web Extractors are built with a specific task and for a particular website. So before creating a new Web Extractor application, a hunt for official tender offering website is conducted, once the focus is set on a particular website, the web- flow of the website to fetch the tender is analyzed. After this analysis the flow of web extractor is decided. Most commonly the web extractor follows specific footsteps. Firstly the Web Extractor navigates to the link of the website. Then by code creating an automation to apply filters on the website to fetch tenders. After applying the filters it checks weather the tenders are available or not, if not then it is instructed to quit. If yes it proceeds further to collect the links of the available tenders, in the similar way as above it checks weather next page exists or not, if yes it iterates similarly till the last page where the tenders are available.

Once all the tender links are collected, this pool of links is compared with the list of links of tenders which are already inserted into the database to discard the duplicate links from the list. The links in the newly created list without redundancy are then navigated one by one.

Eventually the link navigates the Extractor to a webpage with tender details. These specific details are scraped using string operations in C#. If the details provided needs translation it is translated using google or bing translation.

The tender extracted goes through the check-duplicate method which checks duplicates with parameters such as tender number, tender title, deadline, etc. As the method name already gives much information about the purpose of the method it eventually discards the duplicate tenders who are already in the database but also discard the tenders which are expired using deadline as a parameter.

The new tenders are then processed through the insertion method where a connection is made to the regional database and local network mapped hard drive. The tender data is then structured in a tabular form and stored in the AMS and AMS Final database which is created in MS Sql and simultaneously the same tender html template is prepared. Then bootstrap is added only to this html template which gives this data a refined and filtered look then after this html template is stored on the local network mapped drive and if no error are displayed then the data is inserted successfully which reflects on the labels added to the windows form structure. This whole cycle continues until all the collected links are navigated and then lastly it displays a message box providing information about the total numbers of tenders inserted and skipped.

**3.3 Advantages**

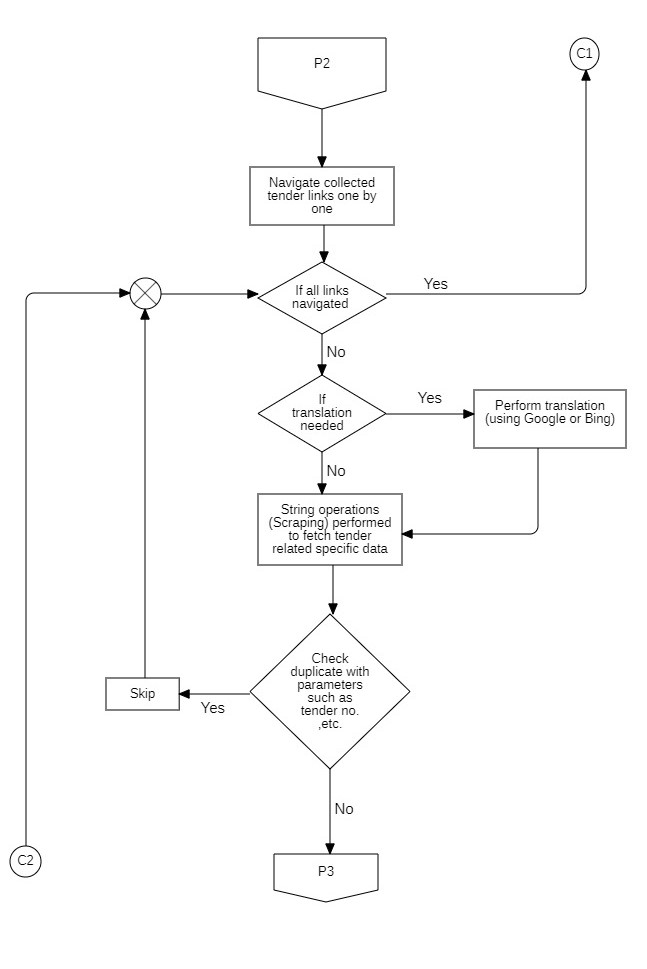
* These web extractor applications are a good example of web automation which saves a lot of time and manual work.
* This application provides automation, filtration, and insertion with timestamp which makes it to manage and locate the data.
* Moreover it helps a lot in removing redundancy and duplication.

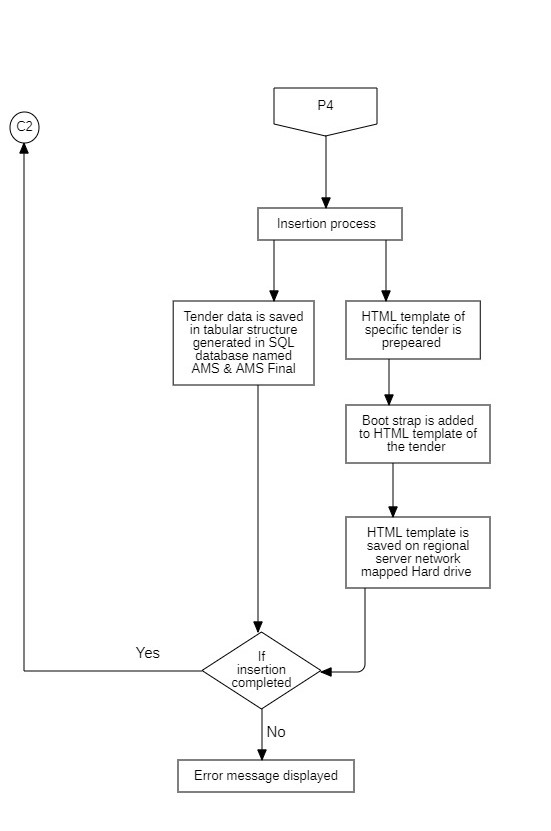
**3.4 Disadvantages**

* The major drawback of the web extractor is that, it needs periodic maintenance.
* To maintain the accurate and smooth extraction of data many changes from the origin websites should be handled dynamically in the program.
* In some web extractors the automation of the process is limited or can be achieved up to a limited extent as the origin website tend to have session or captcha to avoid network traffic.

## FlowchartDiagram1.jpg3.5 Flow Chart

Fig. 3.1 Web Extractor Flow Chart



****

## 3.6 Sequence Diagram

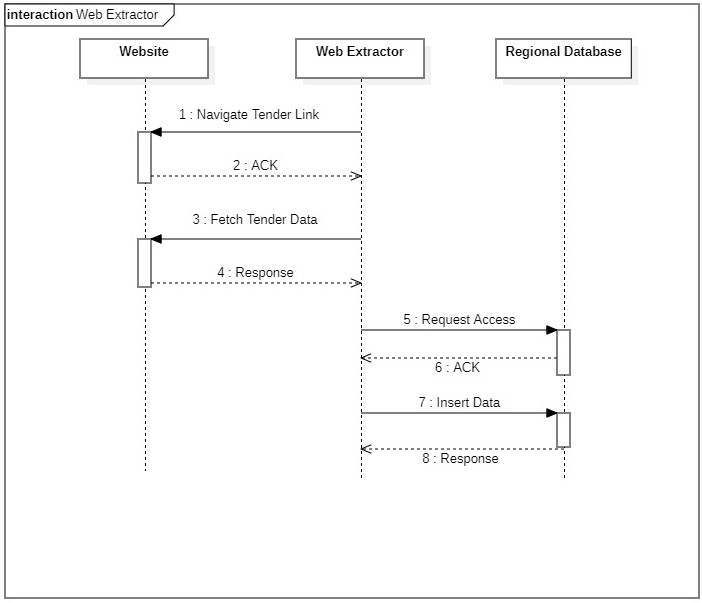
****

Fig. 3.2 Web Extractor Sequence Diagram

## 3.7 Activity Diagram

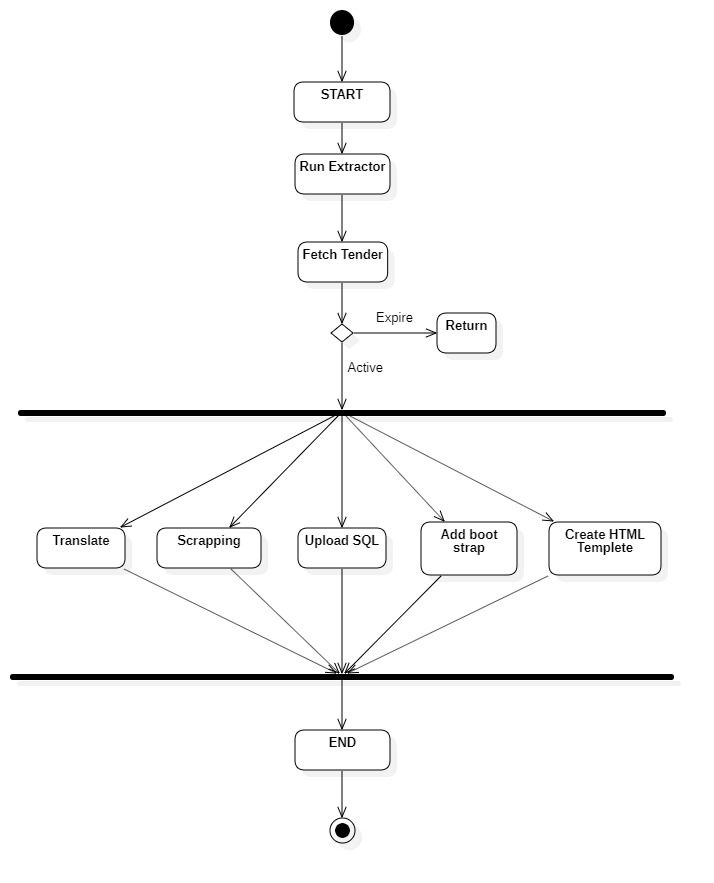


Fig. 3.3 Web Extractor Activity Diagram

## 3.8 Screenshots

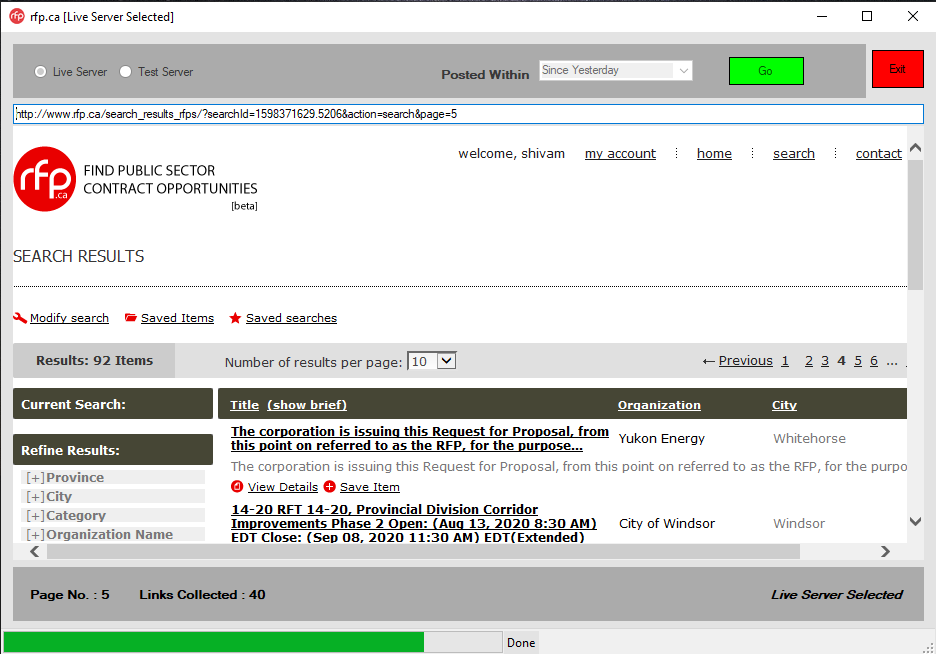


Fig. 3.4 Web Extractor link collection

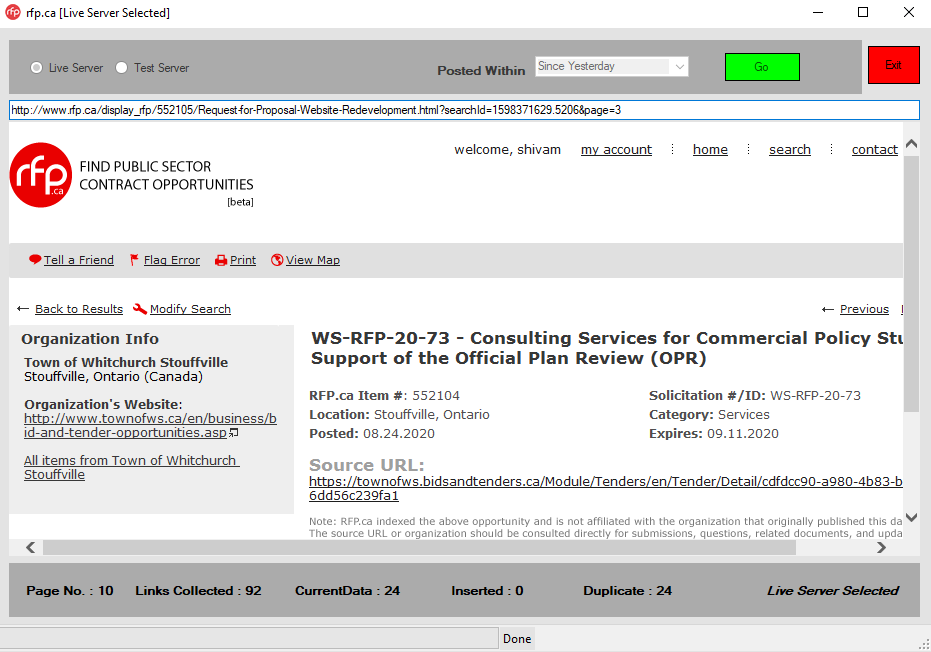


Fig. 3.5 Web Extractor tender page navigation

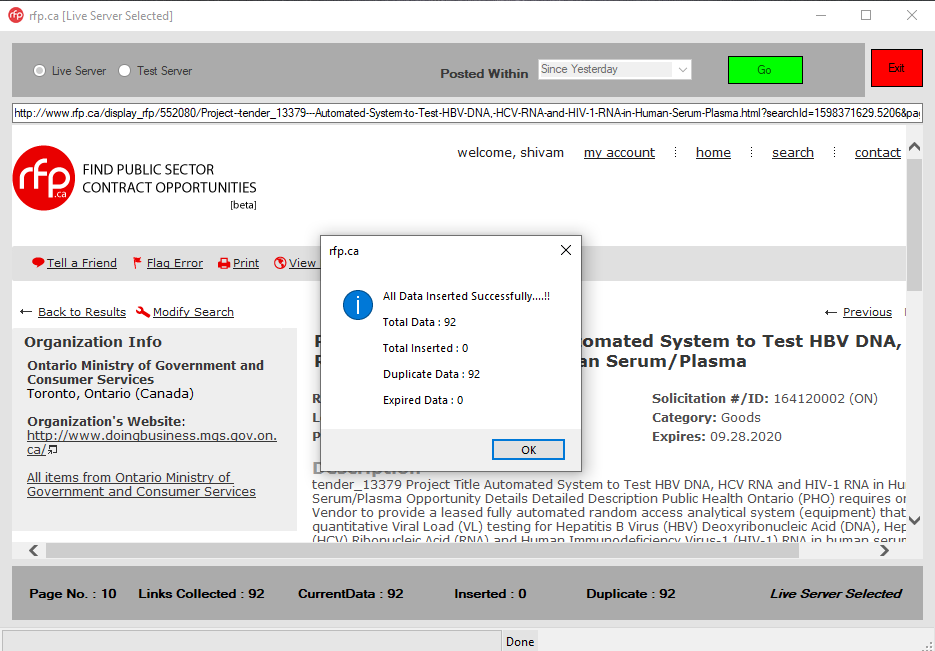


Fig. 3.6 Web Extractor process completion

**Chapter 4**

**Local2Live App**

**4.1 Introduction**

Local2Live app is a software application which is made on a asp.net framework which uses C# as a backend programming language. These Local2Live app applications are built using Microsoft Visual Studio 2017 desktop express version. This Visual Studio provides an environment in which windows form applications are created which is the basic structure of this Local2Live app application.

**4.2 Objective**

Local2Live app transfers Structured (Sql) data from regional database servers to the live database server as well as transfers the HTML tender templates to the AWS database server.

**4.3 Work Flow**

Local2Live app is an interface between the live and local (regional) databases .When the application is initialized a connection made with both the local and website database. It also plays a role of a platform for the data as both the databases run on different platforms such as MS Sql and MySQL.Local2Live uploads tender data from AMS\_Final database to website database. Firstly it checks weather if data is available in the AMS\_Final database and then fetches the tender data and simultaneously using the GT\_ID (goodtenders\_ID) it fetches the same tender html template which is stored in the local network mapped hard drive.

Local2Live app sorts the tender data into finished and unfinished data on the basis of country code, deadline, and tender title etc. Then after a MySQL query is built to upload the unfinished Sql data. The unfinished data is then uploaded to the website database but in the semi entry table then the unfinished data is then managed by the manual entry process

Then after a MySQL query is built again to upload the finished Sql data which is retrieved from the MS Sql local database and after this the data is uploaded to the website database in the current tenders table.

Using the AWSSDK a connection is made with Amazon Web Services database and the tender html template is uploaded to the AWS database. All this is done in a batch of 200 tender at a time and the tenders are identified on the basis of GT\_ID .Once these multiple tenders are uploaded to the live databases then these are deleted from the local databases except from AMS database which used to check duplicates on the local level while running the Web Extractors. This cycle of process continues in a batch of 200 tender until there are no tenders left to upload in the AMS\_Final database .The application has a built in timer which makes sure that the cycle runs at an interval of every 15 minutes.

**4.4 Advantages**

* This application is built to perform crucial tasks which includes uploading and deleting.
* The built in timer which runs the program at intervals saves manual work and human interaction to initialize the process again.
* The application provides an informative label which shows the progress of the cycle.

**4.5 Disadvantages**

* Local2Live is an interface between two platforms where it transfers data from one different platform to another which makes it error prone as some formats are supported in MS Sql but not in MySQL. e.g.: Special Character, Special HTML characters etc.
* As it transfers HTML Templates it requires a more bandwidth.

**4.6 Flow Chart**

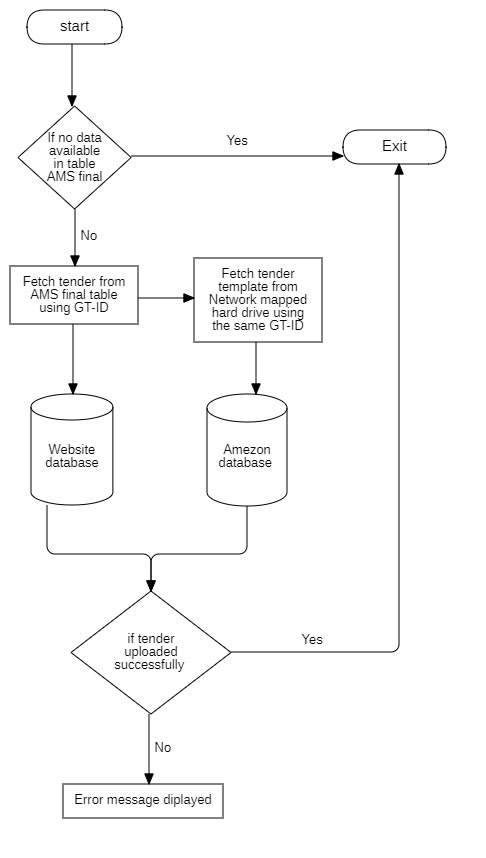


Fig. 4.1 Local2Live Flow Chart

**4.7 Sequence Diagram**

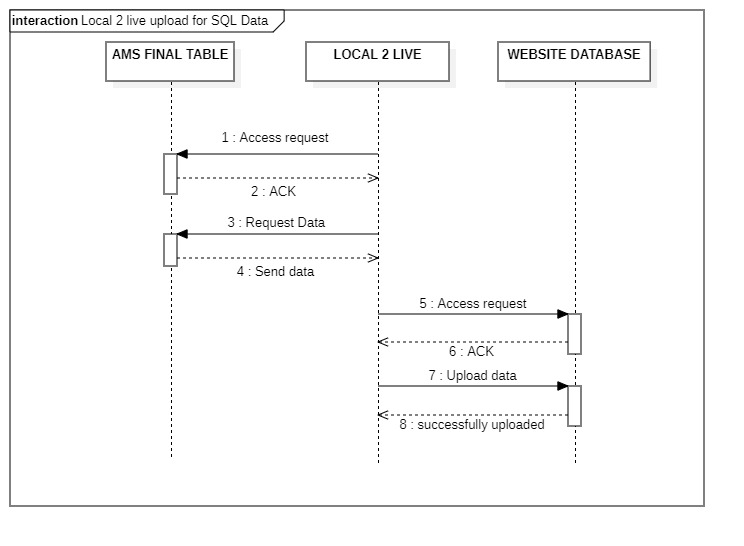


Fig. 4.2.1 Local2live SQL Upload Sequence Diagram

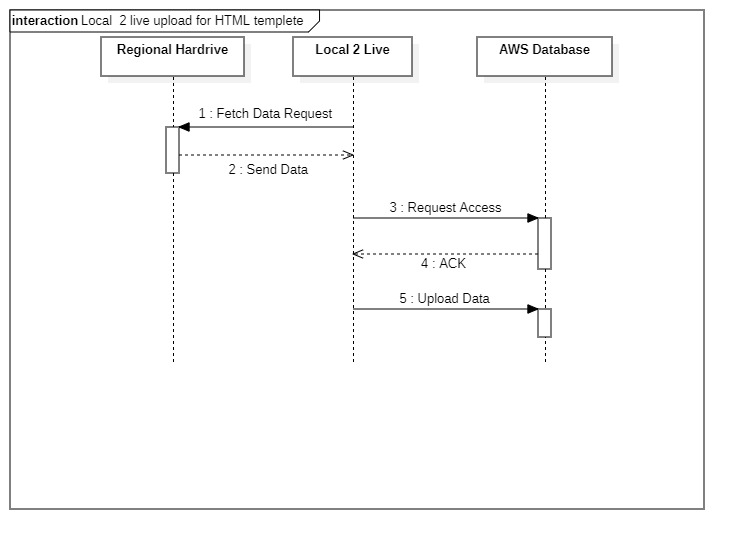


Fig. 4.2.2 Local2live HTML Upload Sequence Diagram

**4.8 Activity Diagram**

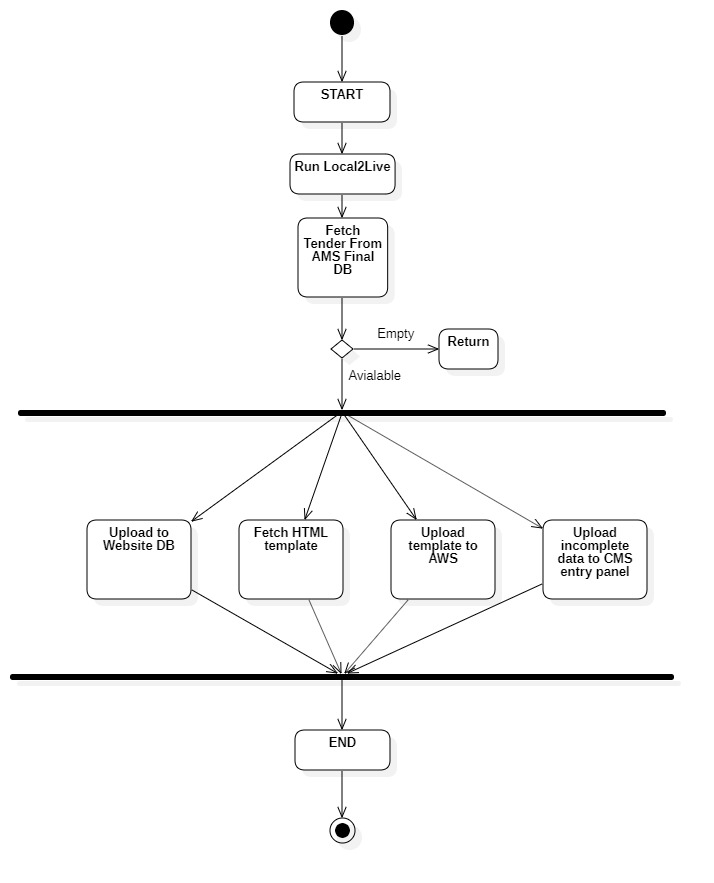


Fig. 4.3 Local2live Activity Diagram

**4.8 Screenshot**

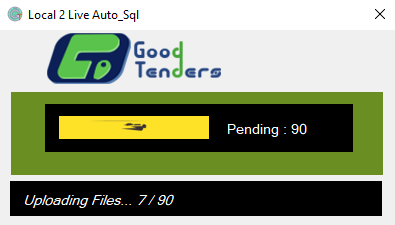
****

Fig. 4.4 Local2Live data uploading

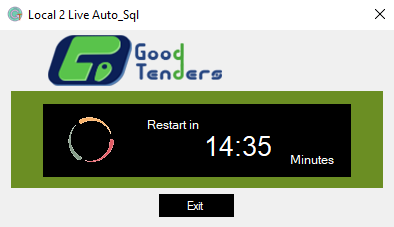
****

Fig. 4.5 Local2Live interval

**Chapter 5**

**CMS-Entry Panel**

**5.1 Introduction**

CMS-Entry Panel is a software application which enables the user and makes it easier for the user to make manual entries of the tenders. It is also built on the same platform as Web Extractor and other applications but with a more interactive UI as it was not built with the purpose of automation but for manual entries. The UI is made in such a way that the screen area is distributed in two half’s where one half is embedded with a web browser and the other half of the form with textbox and labels to take inputs from the end user.

**5.2 Work Flow**

As the CMS-Entry Panel application initializes it builds a connection with website database which works on MySQL platform as well as using AWSSDK it also builds a connection to the AWS database.

This gives the applications the access to multiple tables in the website database and access to the tender html templates in the AWS database.

The Entry panel firstly checks weather tenders are available in the semi entry table, if yes then it fetches the tenders one by one according to the GT\_ID allocated to the tender using the same GT\_ID the it fetch its tender html from the AWS database and renders it on the web browser which is embedded in the application. The web browser acts as html previewer which helps the end user in manual entry of the tender.

The application already fills the rows of tender detail which are complete using automation and the columns which are unfinished are completed by the user.

It provides skip functionality to the user to skip the current displayed tender on the screen and move to the next.

The entry form also uses validation at the backend; the tender won’t be inserted if mandatory details aren’t filled.

Validations are used for the deadline, emails etc formats which avoids glitches and verifies that only filtered and finished data is inserted.

Once the data is inserted by the user, in the backend of the application the tender is then sorted by the deadline which compared by current date.

If the tender is expired it is moved to the archive table year wise and if the tender is an active tender then it is moved to the current tenders table. The tenders without deadline are an exception and they are moved to the archive 1900 year table.

This cycle of rendering unfinished tenders continues until tenders are available in the semi-entry table.

**5.3 Advantages**

* The merged view in the CMS-Entry Panel makes it easier for the end user to deal with the unfinished data which gradually increases the performance and speed of the number of entries made in a day.
* The validation and verification features of the entry panel help to maintain a filtered and precise smooth flow of data.

**5.4 Flow Chart**

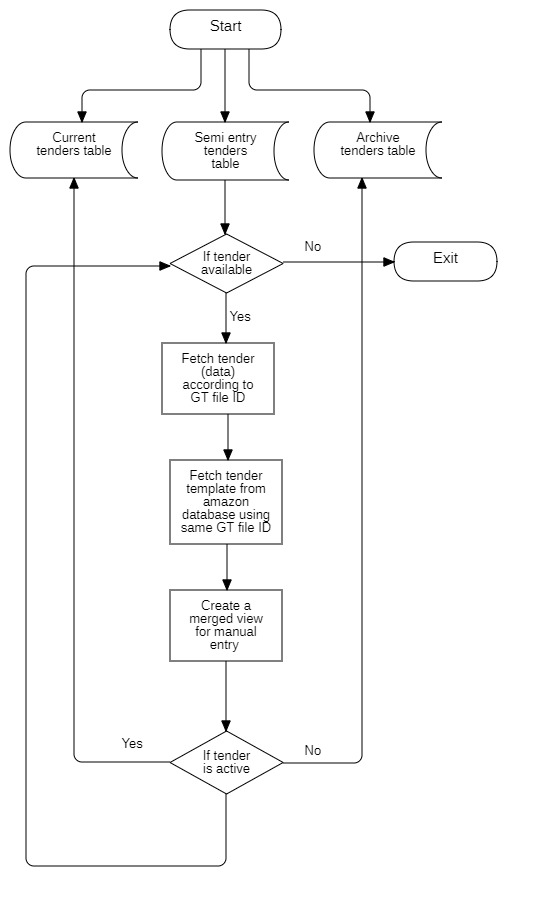


Fig. 5.1 CMS Entry Panel Flow Chart

**5.5 Sequence Diagram**

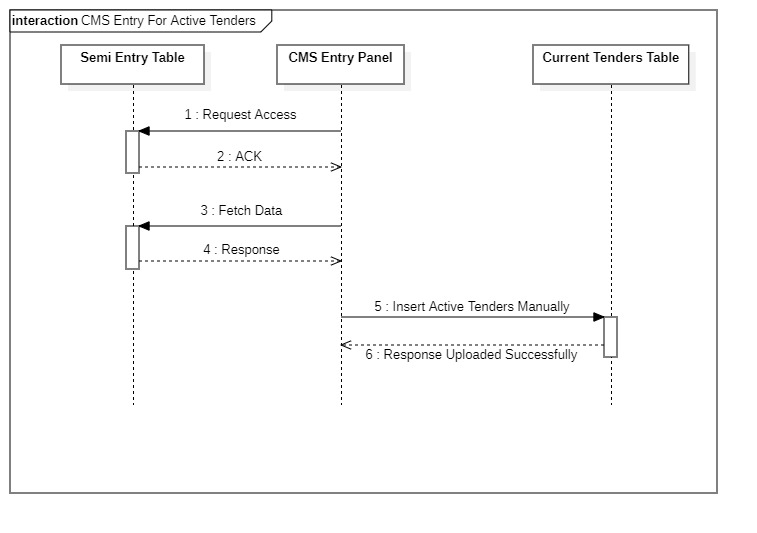


Fig. 5.2.1 CMS Entry Panel For Active Tenders Sequence Diagram

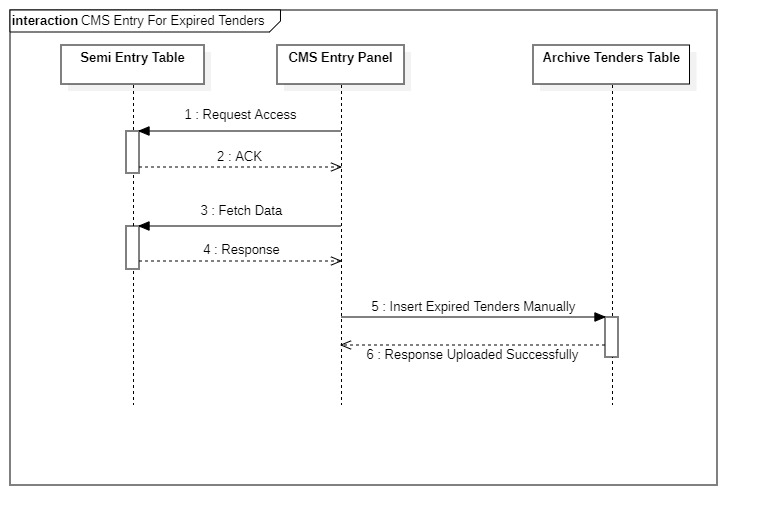


Fig. 5.2.2 CMS Entry Panel Expired Tenders Sequence Diagram

**5.6 Activity Diagram**

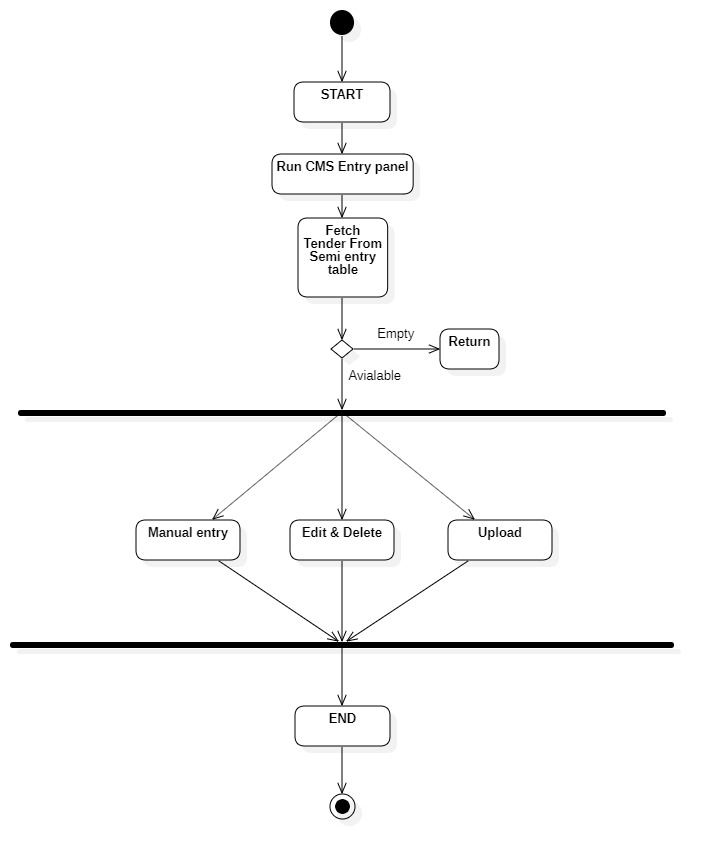
****

Fig. 5.3 CMS Entry Panel Activity Diagram

**Chapter 6**

**Archive Application**

**6.1 Introduction**

Archive app is a software application which is made on an asp.net framework which uses C# as a backend programming language. Its UI structure is added with informative label and interactive elements like drop-down box which enables user to manually choose the items, and buttons, progress bars etc, to perform their appropriate and assigned tasks.

**6.2 Objective**

The main purpose of this app can be easily determined by its name, where it moves the expired tenders which are live on website to the archive tables.

**6.3 Work Flow**

The archive app moves the expired tenders from the current table to the archive table. This process is to be conducted daily at the start of backend process and at the end of the process which verifies that no expired tenders are floating on the website but only the active ones.

As the application initializes it builds connection with the current tenders table in the website database. A tender is determined expired or active using its deadline, so deadline is a very prominent factor in this process. Once the connection is built multiple items are displayed in drop down box displaying years from 2014 until the last year from the current year (E.g.:2014 to 2019 as the current year is 2020) and archiveb4\_2014 (which represents year tables before 2014 as one).There is one more item in the drop down box displaying 1900.This “1900” represents a archive table where all the tenders without deadlines are inserted. The end user selects any one year at a time, once any item is selected at the backend it detects the select event fired and then it fetches the tenders with deadline of the year which is selected by the user from the current tenders table in the website database, the application program sorts and groups the tenders source wise and then on the UI using another drop-down box the multiple sources of expired are displayed of the selected year. Then the user has to select any source (one at a time), then click the insert button, once the button click event is detected at the backend; the program inserts all the tenders of the selected source to the selected year archive table and then all these tenders are deleted from the current tenders table. Similarly the whole process continues until all the expired tenders of the year mentioned in the drop-down box are cleared and inserted to their respective year archive table and eventually the process ends.

There is another version of archive app which deals with the expired tenders of the current year. This is version very simple at its UI and as it has very minor functionality almost everything is done by backend program, the user just has to click the insert button. As this version of archive app which is named “date wise archive app” and as the name says it all , the expired tenders of the current year is fetched date wise until the date before current date.

The counts of the expired tenders which are fetched is displayed on the UI, then once the user clicks the insert button these tenders are inserted into its respective current year’s archive table.

**6.4 Advantages**

* This archive app helps the users to sort and move the expired tenders to the archive tables in minimum time and limited actions.

**6.5 Flow Chart**

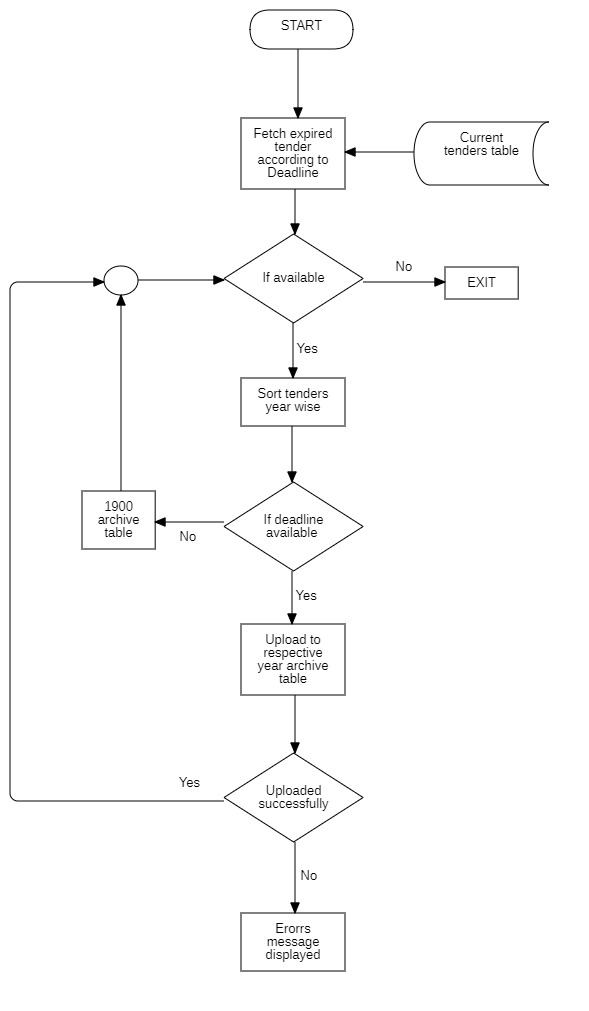
****

Fig. 6.1 Archive App Flow Chart

**6.6 Sequence Diagram**

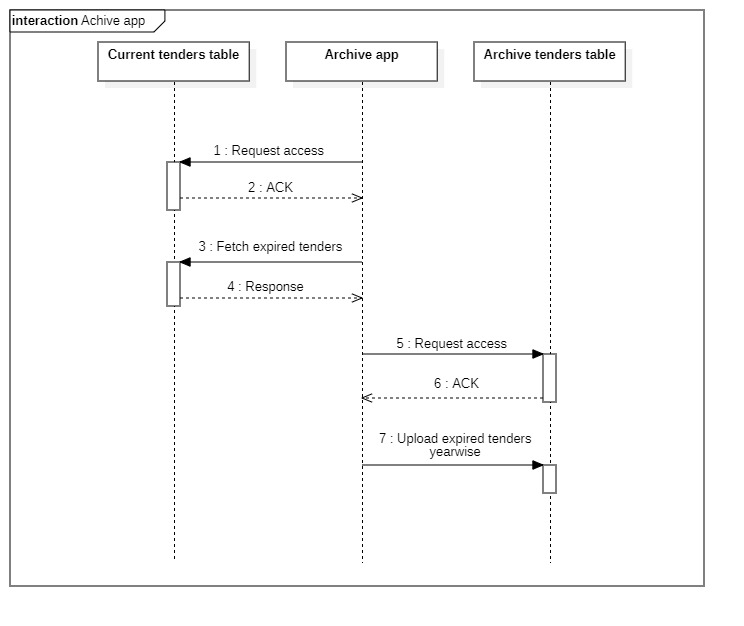


Fig. 6.2 Archive App Sequence Diagram

**6.7 Activity Diagram**

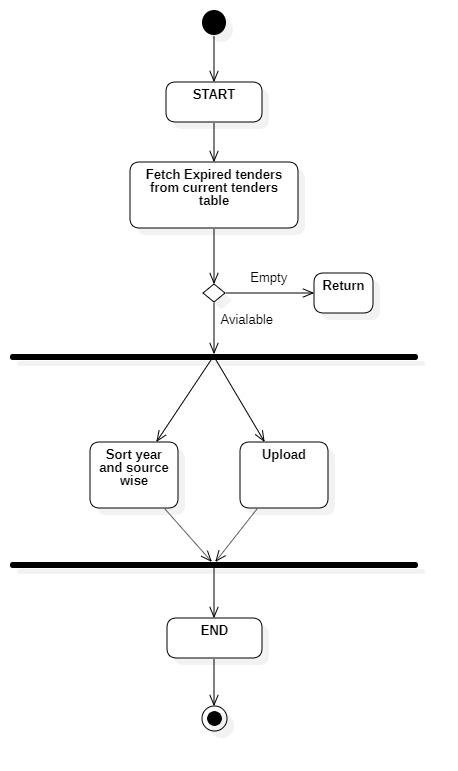
****

Fig. 6.3 Archive App Activity Diagram

**6.8 Screenshots**

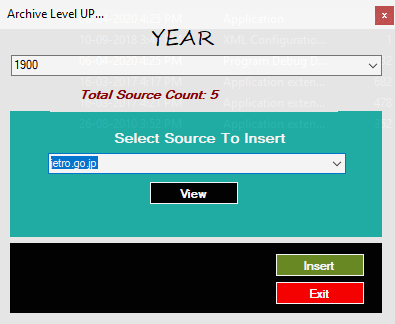
****

Fig. 6.4.1 Archive App Ver1

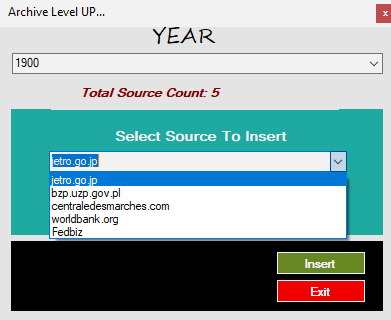
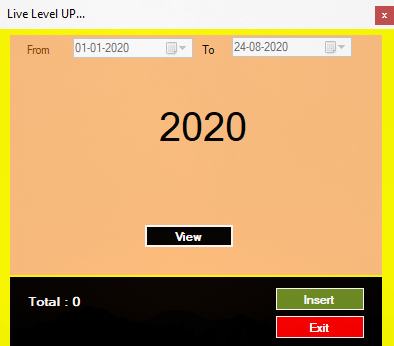
****

Fig. 6.4.2 Archive App Ver1 Select Source

Fig. 6.4.3 Archive App Ver1 Select Year

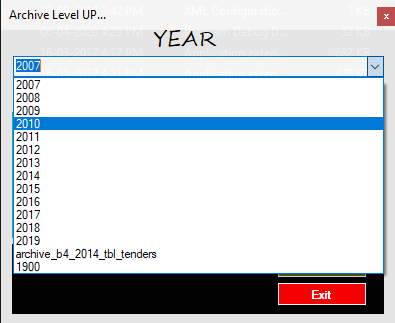
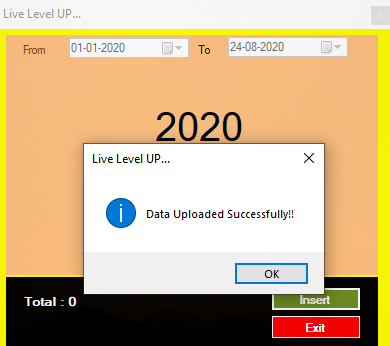
****Fig. 6.4.4 Archive App Ver2

Fig. 6.4.5 Archive App Ver2 Process Completion

****

**Chapter 7**

**Level-Up Application**

**7.1 Introduction**

Level-Up app is a software application which is made on an asp.net framework which uses C# as a backend programming language. Its UI structure is added with informative labels and interactive elements like drop-down box which enables user to manually choose the items, and buttons, progress bars etc, to perform their appropriate and assigned tasks.

**7.2 Objective**

This application is made with the aim to perform specific task of creating the view of the newly inserted tenders and make them float on the website.

**7.3 Work Flow**

The Level-Up makes the newly added tenders to float on the website. As the application is initialized is builds connection with the current tenders table. It fetches the newly added tenders in the website database. It then sorts and groups the tenders by their source name.

These multiple source name of the tenders is displayed in the drop down box as items to the end user. The end user selects any one source name at a time and click on the insert button, as the button click event is detected at the backend, the application program creates view of the multiple tenders of the particular source selected one by one. In each view created of the newly added tender its AWS html tender template link is bind to a (More Detail) button in the view .This is link is obtained and prepared by the File-ID and GT-ID information stored in the tender table for each tender .The progress of the task is displayed on using the progress bar and informative labels. This process continues until all the sources displayed in the drop down box are inserted by the end users and the drop down clears out.

**7.4 Advantages**

* Level-Up app makes it easier for the user to upload tenders as it groups the tenders by its source for e.g. if hundred tenders of FedBiz source is inserted it uploads and creates view of all the hundred tenders in one go.

**7.5 Disadvantages**

* Level-Up app sorts and groups the tenders by its source name which is an advantage to the process but as the number of source have increased over the time which is reached a range of 400-450 which makes it complex for the end user to upload one source at a time makes it a time consuming and hectic process.

**7.6 Flow Chart**

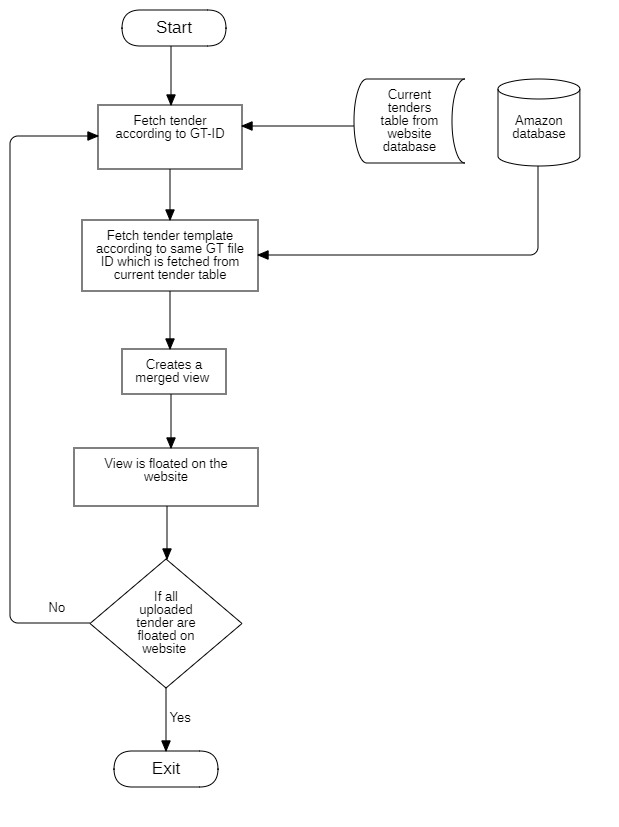
****

Fig. 7.1 Level up Flow Chart

**7.7 Sequence Diagram**

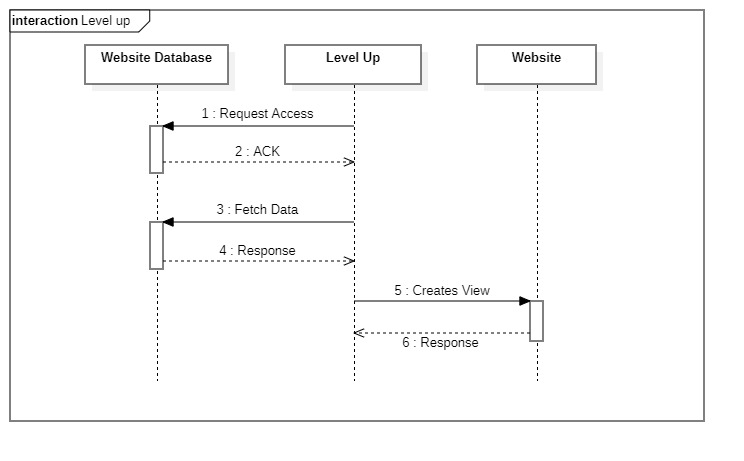


Fig. 7.2.1 Level up Sequence Diagram

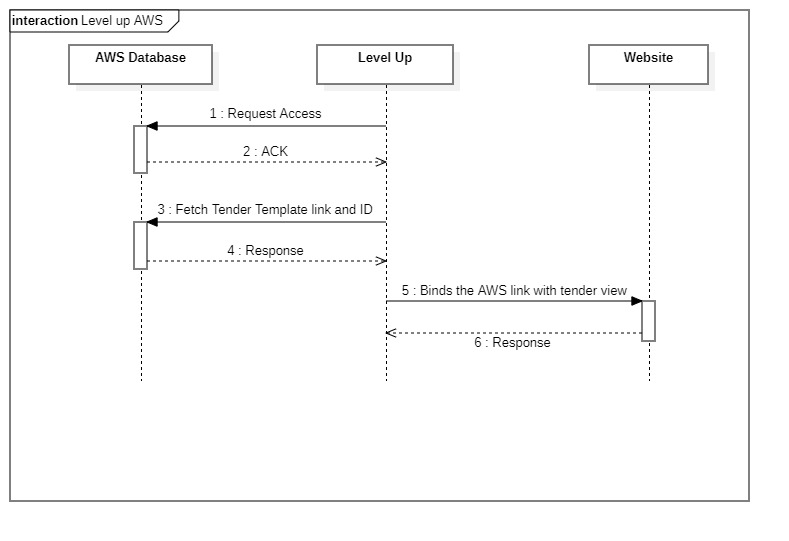
****

Fig. 7.2.2 Level up AWS Sequence Diagram

**7.8 Activity Diagram**

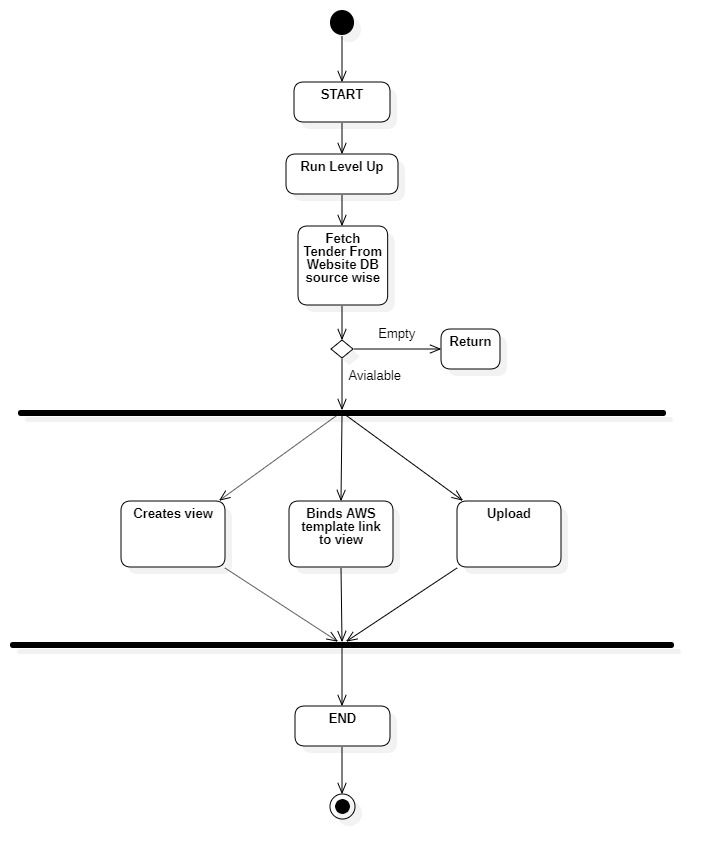
****

Fig. 7.3 Level up Activity Diagram

**Chapter 8**

**Database Design**

The Database infrastructure of “Good tenders backend IT process” consists of three major databases.

* Regional database.
* Live database (Website db).
* Amazon Web Services (AWS) database.

**8.1 Regional Database**

The regional databases are structured and set-upped in the working location. These multiple databases are created on multiple servers in the room. These databases are created on a MS Sql platform and are distributed region wise as the tenders are collected various from regions and types. For the purpose of managing and retrieving data efficiently the tender data is collected in this distributed architecture. This distribution helps in handling the data more precisely making it easier to find specific data.

**8.2 Live (Website) Database**

The Live Database of the website is hosted online on a server. This database works on the MySQL Platform. This online server of the website consists of multiple tables used to manage data. The structured data from the regional servers are uploaded to this live database server.

**8.2 AWS Database**

An Amazon S3 bucket is a public cloud storage resource available in Amazon Web Services' (AWS) Simple Storage Service (S3), an object storage offering. Amazon S3 buckets, which are similar to file folders, store objects, which consist of data and its descriptive metadata. Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. This means customers of all sizes and industries can use it to store and protect any amount of data for a range of use cases, such as websites, mobile applications, backup and restore, archive, enterprise applications, IoT devices, and big data analytics. Amazon S3 provides easy-to-use management features so you can organize your data and configure finely-tuned access controls to meet your specific business, organizational and compliance requirements.

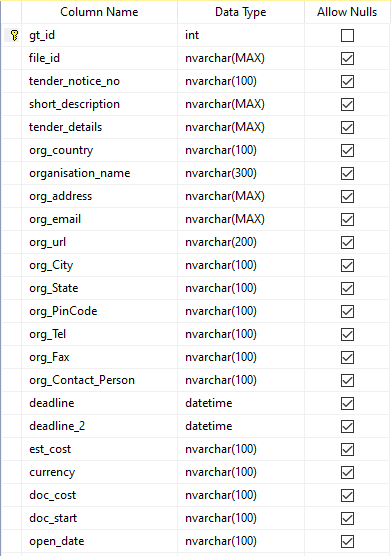
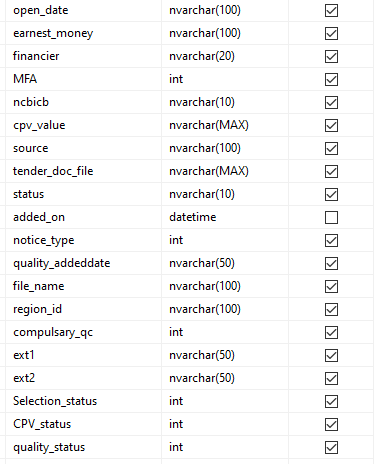


Fig. 8.1 Table structure design

Fig. 8.1 Table structure design



**8.1 Database Design Diagram**

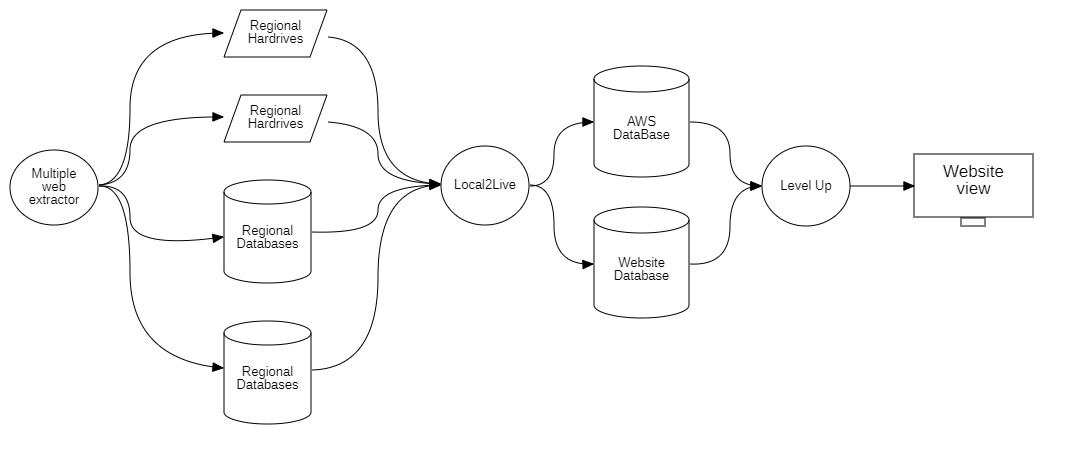
****

Fig. 8.2 Database Design Diagram

**Chapter 9**

**Conclusion**

**9.1Conclusion**

The Backend IT process of Goodtenders actually filters and uploads the tenders which are resided on the remote link. We used .NET technology for developing the applications, also CSS and JavaScript for designing and scripting the website. The process stores the filtered and translated tenders on the website database. It also categorizes the uploaded tenders by their various attributes type.

**9.2Future Scope**

**Appendix I**

**A. Technology Used**

1. **MS Visual Studio 2017.**

****

Visual Studio is an **Integrated Development Environment (IDE)** developed by Microsoft to develop GUI (Graphical User Interface), console, Web applications, web apps, mobile apps, cloud, and web services, etc. With the help of this IDE, you can create managed code as well as native code. It uses the various platforms of Microsoft software development software like Windows store, Microsoft Silver light, and Windows API, etc. It is not a language-specific IDE as you can use this to write code in C#, C++, VB (Visual Basic), Python, JavaScript, and many more languages. It provides support for 36 different programming languages. It is available for Windows as well as for macOS.

1. **JavaScript.**

****

JavaScript is the programming language of HTML and the Web. Programming makes computers do what you want them to do.

1. **CSS3.**

****

CSS stands for Cascading Style Sheets

CSS describes how HTML elements are to be displayed on screen, paper, or in other media

CSS saves a lot of work. It can control the layout of multiple web pages all at once

External style sheets are stored in CSS files.

1. **JQuery.**

****

JQuery is a lightweight, "write less, do more", JavaScript library.

The purpose of jQuery is to make it much easier to use JavaScript on your website.

JQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.

JQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.

The jQuery library contains the following features:

* HTML/DOM manipulation
* CSS manipulation
* HTML event methods
* Effects and animations

1. **MS SQL Server (MS SQL Server 2017)**

****

SQL Server is Microsoft's relational database management system (RDBMS). It is a full-featured database primarily designed to compete against competitors Oracle Database (DB) and MYSQL.

Like all major RBDMS, SQL Server supports ANSI SQL, the standard SQL language. However, SQL Server also contains T-SQL, its own SQL implementation. SQL Server Management Studio (SSMS) (previously known as Enterprise Manager) is SQL Server's main interface tool, and it supports 32-bit and 64-bit environments. SQL Server is sometimes referred to as MSSQL and Microsoft SQL Server.

1. **MySQL**

****

MySQL is a database management system that allows you to manage relational databases. It is open source software backed by Oracle. It means you can use MySQL without paying a dime. Also, if you want, you can change its source code to suit your needs.

* Even though MySQL is open source software, you can buy a commercial license version from Oracle to get premium support services.
* MySQL is pretty easy to master in comparison with other database software like Oracle Database, or Microsoft SQL Server.

1. **PHP**

****

PHP is a server side scripting language. that is used to develop Static websites or Dynamic websites or Web applications. PHP stands for Hypertext Pre-processor, that earlier stood for Personal Home Pages.

* PHP is an acronym for "PHP: Hypertext Preprocessor".
* PHP is a widely-used, open source scripting language.
* PHP scripts are executed on the server.
* PHP is free to download and use.

1. **AWS S3 SERVER**

****

**Amazon S3** (Simple Storage Service) is a scalable, high-speed, low-cost web-based service designed for online backup and archiving of data and application programs. It allows uploading, store, and downloading any type of files up to 5 TB in size. This service allows the subscribers to access the same systems that Amazon uses to run its own web sites. The subscriber has control over the accessibility of data, i.e. privately/publicly accessible.