Contents

[1. INTRODUCTION 1](#_Toc419070983)

[1.1 Purpose 1](#_Toc419070984)

[1.2 Scope 1](#_Toc419070985)

[1.3 Overview 2](#_Toc419070986)

[1.4 Reference Material 3](#_Toc419070987)

[ **Software Requirements Specification (SRS)** 3](#_Toc419070988)

[1.5 Definitions and Acronyms 3](#_Toc419070989)

[2 SYSTEM OVERVIEW 5](#_Toc419070990)

[2.1 Operational Automation 6](#_Toc419070991)

[2.2 Design Methods 6](#_Toc419070992)

[2.3 GUI Design 6](#_Toc419070993)

[3 DESIGN CONSIDERATIONS 8](#_Toc419070994)

[3.1 Assumptions and Dependencies 8](#_Toc419070995)

[3.1.1 Related software, hardware and operating system 8](#_Toc419070996)

[3.1.2 End-user characteristics 8](#_Toc419070997)

[3.2 Software Constraints 8](#_Toc419070998)

[4 SYSTEM ARCHITECTURE 9](#_Toc419070999)

[4.1 Decomposition Description 11](#_Toc419071000)

[5 DATA DESIGN 14](#_Toc419071001)

[5.1 Data Description 14](#_Toc419071002)

[5.2 Data Dictionary 14](#_Toc419071003)

[5.2.1 Forms table 14](#_Toc419071004)

[5.2.2 User table 15](#_Toc419071005)

[5.2.3 Subscriber table 15](#_Toc419071006)

[5.2.4 Organization Table 16](#_Toc419071007)

[5.2.5 Package Table 16](#_Toc419071008)

[5.2.6 FormDataHistory Table 17](#_Toc419071009)

[5.2.7 Request Table 17](#_Toc419071010)

[5.2.8 Signature Table 17](#_Toc419071011)

[6 HUMAN INTERFACE DESIGN 18](#_Toc419071012)

[6.1 Overview of User Interface 18](#_Toc419071013)

[6.2 Screen Images 22](#_Toc419071014)

[7 Appendix 24](#_Toc419071015)

# INTRODUCTION

## Purpose

This software design document has detailed specifications of all aspects of the GoPaperless web application. It gives details about its scope, the components of the system and information pertaining to the implementation of the project with special prominence on its analysis and design elements. The document is written for the purpose of identifying the key tasks which are being handled by the software program which in turn fulfills the purpose of this project. The purpose of this document is to provide a detailed description of the functionalities of the project. The scope of this document involves the proposal description for making a web application. This document will cover each of the system’s intended features, system constraints, interface and interactions with other user applications. This document will give a brief overview of the whole system or project.

This document is intended for both the stakeholders and the developers of the system. For developer it serves as a reference for product design and implementation constraints. *This document uses requirements specified in the Software Recruitment Specification (SRS) document.*

## Scope

This document will be used to provide information on how each part of the system operates during the coding and testing phase. As well as ensuring proper data cohesion for the database design.

The main goal of the GoPaperless software is to provide an easy to use but powerful environment to allow it used to digitize forms and use them as web based forms without any prior experience in programming or database design and management. The goal behind the project is to enable companies to reduce paper usage and doing so becoming “Green” (*environmentally friendly*). Our solution to the problem to have organizations convert paper forms into digital web based forms, However traditionally this would be a difficult, time consuming and expensive process of developing a purpose built software and spending additional money on update and maintain the software.  
To remedy this the GoPaperless software is designed a simple not hassle solutions that is as simple as logging into the website using a easy and intuitive drag and drop UI to build and publish forms and all the complicated database connection and data manipulations is done automatically. A organization can convert all there paper forms into digital ones an a matter of hours not only saving a large about you money which buy not using paper but take a set closer to becoming environmentally conscious.

## Overview

This document is a product of all the team decisions and planning was done on how the proposed software will be implemented. The Document contains sections on software architecture that will be implemented its database and object models as well as the assumptions constraints to the development. This document is the conversion of the use cases and requirements that were documented in the Software Requirement Specification document (SRS).  
The information and designs and not concert and will change when required but the project team is required to ensure that the functionality adheres to the requirements.

This document is specifically for the project development team and the team involved in the future maintenance of the software.

## Reference Material

## **Software Requirements Specification (SRS)**

|  |  |
| --- | --- |
| Version | 1.0 |
| Date | 16/2/2015 |
| Author | Anwer Amin |
| Access information | GitHub Repository //Master/documentation/SRS.doc |
| Publisher | Anwer Amin |

## Definitions and Acronyms

|  |  |
| --- | --- |
| **Table** | **Definitions** |
| User | A person who directly interact with the software |
| Administrator | System administrator for the website |
| Subscriber | User that has the main account linked to the payment |
| Customer | the person(s) who orders the travel packages |
| HCI | Human Computer Interaction |
| Html | Hyper Text Markup Language |
| CSS | Cascading Style Sheets |
| JSP | Java Server Pages |
| Bootstrap | is an open-source JavaScript framework developed by the team at Twitter |
| Web-forms | allows a user to enter data that is sent to a server for processing |
| GUI | Graphical user Interface |
| Front-end users | User site |
| Back-end users | Administrator site |
| Dynamically | web page whose construction is controlled by an application server processing server-side scripts |
| DB | Database |
| PC | Personal Computer |
| JavaScript | Front end scripting language. |
| Paperless | An environment that reduces paper usage or eliminated it all together |

# SYSTEM OVERVIEW

This project is made as a standalone system for a solution that offers functionality to expedite the process of the digitization for organization looking for “Green” paperless office environments. I does this by removing the need for an organization to go through long process of digitization and teaching its employees the new system.  
GoPaperless is essentially a web application that allows its users to simple sign up to the service that handles all back end programming automatically. The user will able to choose from a number of packages that met their requirements.

* This website is responsive, providing appropriate display for any screen size be it PC, smart phones or other gadgets.
* Users will be able to make new forms without programming.
* Users will be able to make forms in advance pending approval.
* Users will be able fill and use forms normally.
* The information will be saving a separate database so speed up data access.
* Admin users will be able to make new accounts for users that can access and work with forms.

This is the context diagram explaining the scope of the system.



Figure 1 Context Diagram

## Operational Automation

The process of automation of code generation is handles by a combinations of Javascript front end and Java back end coding. In practice there can be hundreds of forms that can be created and stored and unlike other website that offer similar functionality the data will not be stored in abstracted form that is meaningless without the GUI. The data in GoPaperless is saved in dynamically generated database tables that are created using the prototype forms that the user constructs.

## Design Methods

To best suit the client requirements and experience of the development team with web technologies we have decided to use HTML5 for the interface, while the main system logic for dynamically displayed pages we will be using JSP and Java programming language.

## GUI Design

As this website’s focus was on marketing we have spent a great amount of effort in flushing out the user interface for this website. The requirement state that the website should be responsive. So do that we have used a combination of HTML5, CSS3, JavaScript (JQuery), Javascript and bootstrap. This allowed us to provide a simple but attractive user interface for the main website and the admin panel. We followed standard HCI practices to ensure usability.

# DESIGN CONSIDERATIONS

## Assumptions and Dependencies

### Related software, hardware and operating system

As the software is being developing Java web technologies it will be hosted on a server configured to use Apache Tomcat servlet container to run the JSP pages and controllers. The server hosting the databases will be required to run MySQL5 databases.

### End-user characteristics

The website user be is the costumer or the admin must use a device with a web browser that supports HTML5. The user will also require an active internet access on the device.

## Software Constraints

The following are required for the webserver to host the website properly:

* Microsoft Windows (NT, 7).
* MySQL
* Apache Tomcat

The device that opens the webpage must support html and JSP file formats as well as running Java Script.

# SYSTEM ARCHITECTURE

System architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.  
The GoPaperless application is designed around MVC model one architecture.

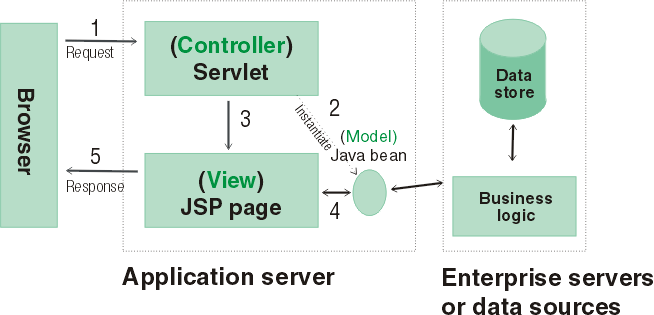


Figure JSP MVC model 1 Architecture

In this three-tier architecture, a servlet and a JSP page are on an application server, and a data store and the business logic are on a data server.

1. The browser sends a request to a servlet.
2. The servlet instantiates a Java bean that is connected to a database.
3. The servlet communicates with a JSP page.
4. The JSP page communicates with the Java bean.
5. The JSP page responds to the browser.

In this using this our proposed implementation of the GoPaperless web application can be representing using an architecture diagram.



Figure GoPaperless Deployment

## Decomposition Description

The project is build using Java web technologies and follows the MVC architecture. This means that your class structure can be divided into 4 major types. The first are the JSP pages while they contain HTML and javascript in reality how JSP works is that is compiled into servlets classes that will handle the GET POST and PUT operations on the pages rendering HTML **view** and send it to the user. As this is done by the Apache Tomcat there need not be a requirement to go into much details into how this is done.

The second type of classes are called **model** classes these are JavaBean that model the database table are essential as these classes are responsible for the communication with the database, They use supplementary Data Access Objects to connect to the database and perform the required operations (insertion, update, selection, deletion).   
The third class is called **controller**  are servlets that contain the core logic and manage both the view and the model, these class act as mediums that handle requests from the view and fetches data thorough the models and send a response with the appropriate information.

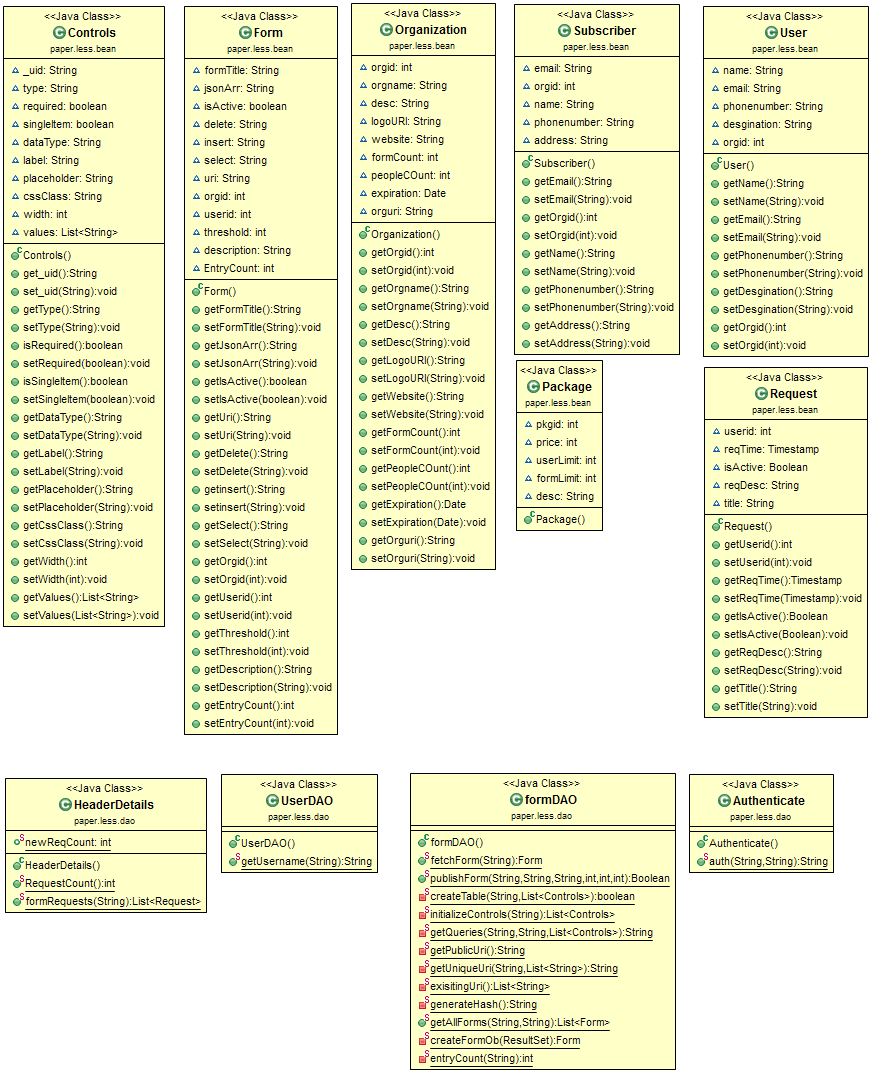
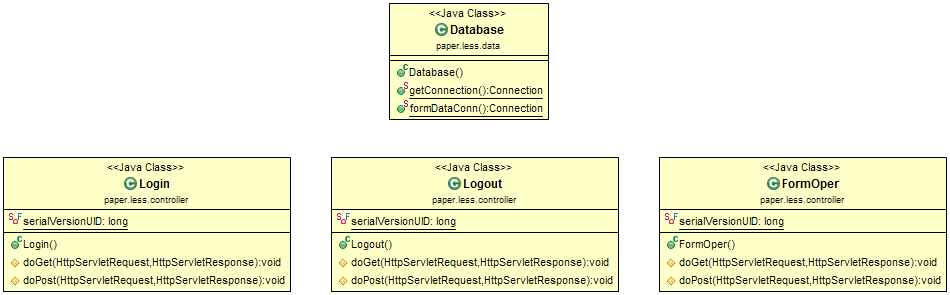


Figure Model (Data Access Objects and Java beans)

These classes are modeled directly on the database, object of a JavaBean can hold one record of a specific table.

Figure 5 contains the servlet classes that handle the core functionalities of the software.  


The Database Class is provides connection with the database(s) to all other classes that require it. It has two connection methods for the 2 different databases in the project.

The Login and Logout servlets handle session details for users.

The FormOper classes handles all the requests that originate from the front end the software. Which include redirections, event handling etc.

# DATA DESIGN

Database design is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a data definition language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

## Data Description

The database Schema and era images are in the Appendix.

## Data Dictionary

### Forms table

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Allow Null | Description |
| formid | int(11) | NO | Primary Key, Form identification number. |
| title | varchar(100) | NO | Title that appears on the form. |
| jsonarr | varchar(10000) | NO | The Json array that defines from structure. |
| insert | varchar(500) | YES | The insertion query |
| delete | varchar(500) | YES | Store deletion query |
| select | varchar(500) | YES | Stores selection query |
| isActive | tinyint(1) | YES | Store state of form. |
| publicuri | varchar(500) | YES |  |
| orgid | int(11) | YES | The organization the forms belongs to. |
| threshold | int(11) | YES | Tracks number of submissions |
|  |  |  |  |

### User table

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Allow Null | Description |
| userid | int(10) unsigned | NO | PRIMARY KEY user to store the user id, autogen |
| email | varchar(100) | NO | Store user email address |
| orgid | int(11) | NO | Organization number the account is linked to. |
| name | varchar(50) | NO | Name of the user |
| address | varchar(100) | YES | User’s address |
| phonenumber | varchar(15) | NO | User’s phone number |
| designation | varchar(50) | NO | User’s designation in the organization |
| signature | int(11) | YES | Auto get signature |
| username | varchar(50) | NO | Username in the system |
| password | varchar(50) | NO | Password in the system |
| usertype | varchar(5) | NO | The role of the user |

### Subscriber table

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Allow Null | Description |
| subid | int(11) | NO | PRIMARY KEY auto gen id |
| userid | int(11) | NO | The user id of the subscriber |
| orgid | int(11) | NO | The organization of the subscriber |

### Organization Table

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Allow Null | Description |
| orgid | int(11) | NO | PRIMARY KEY the organization autogen id |
| orgName | varchar(100) | NO | The name of the organization |
| logouri | varchar(100) | YES | The organizations logo image link |
| website | varchar(100) | YES | The organizations website |
| formsCount | int(11) | NO | Total number of forms being used |
| peopleCount | int(11) | NO | Total number of employees |
| isActive | tinyint(1) | YES | State of the organization |
| expiration | Datetime | NO | Expirations date of the package |
| description | varchar(200) | YES | Organizations description |
| orguri | varchar(20) | NO | Direct link to the organization for public access the PK |
| userid | int(11) | YES | Store subscribers ID |
| formid | int(11) | YES |  |
| approval | tinyint(1) | YES | Payment approval |

### Package Table

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Allow Null | Description |
| pkgid | int(11) | NO | PRIAMARY KEY the package id |
| price | int(11) | NO | Price in PKR |
| userlimit | int(11) | NO | The total number of user accounts allowed |
| formlimit | int(11) | NO | The total number of Forms allowed to organization |
| desc | varchar(200) | NO | The description of the packages |

### FormDataHistory Table

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Allow Null | Description |
| hsid | int(11) | NO | PRIMARY KEY the history id |
| userid | int(11) | YES | The user that inserted the data |
| orgid | int(11) | YES | The organization of the form and users |
| dateTime | date | YES | The data and time when the record was inserted. |

### Request Table

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type** | **Allow Null** | **Description** |
| **userid** | int(11) | NO | The user id (Admin) that generated the request |
| **req\_time** | datetime | NO | The request time and date |
| **isActive** | tinyint(1) | YES | The request status |
| **title** | varchar(100) | YES | The titles of the request |
| **reqdesc** | varchar(200) | YES | The description of the request |
| **seen** | tinyint(1) | YES | Bool for listing. |

### Signature Table

|  |  |  |  |
| --- | --- | --- | --- |
| Field Name | Data Type | Allow Null | Description |
| userid | int(11) | YES | The user id |
| formid | int(11) | YES | The form id |
| approval | tinyint(1) | YES | Check if the form was approved |

There can’t be a database dictionary for the second database as all the tables are generated dynamically by the program.

# HUMAN INTERFACE DESIGN

The interface as mentioned before uses a variety of web technologies and scripting languages to achieve an attractive be functional design.

## Overview of User Interface

The design for the user interface must be able to perform the functions that are shown in the sequence diagrams shown below.

The sequence diagram only show the major interface operations that are related to the users the others can be done and any manner as long as the requirements are met.



Figure Subscriber sequence diagram



Figure Admin sequence diagram



Figure User sequence diagram

## Screen Images

As we have a good idea of the design language we have to use to make the product as defined in the SRS we have chosen some mock up forms are show below.  
We have decided on a theme from studying the SRS and hence the design for the portal and landing page will be responsive maintaining a simple but elegant minimalist design. As we are using a theme we have good idea as to what the finished product will look like.

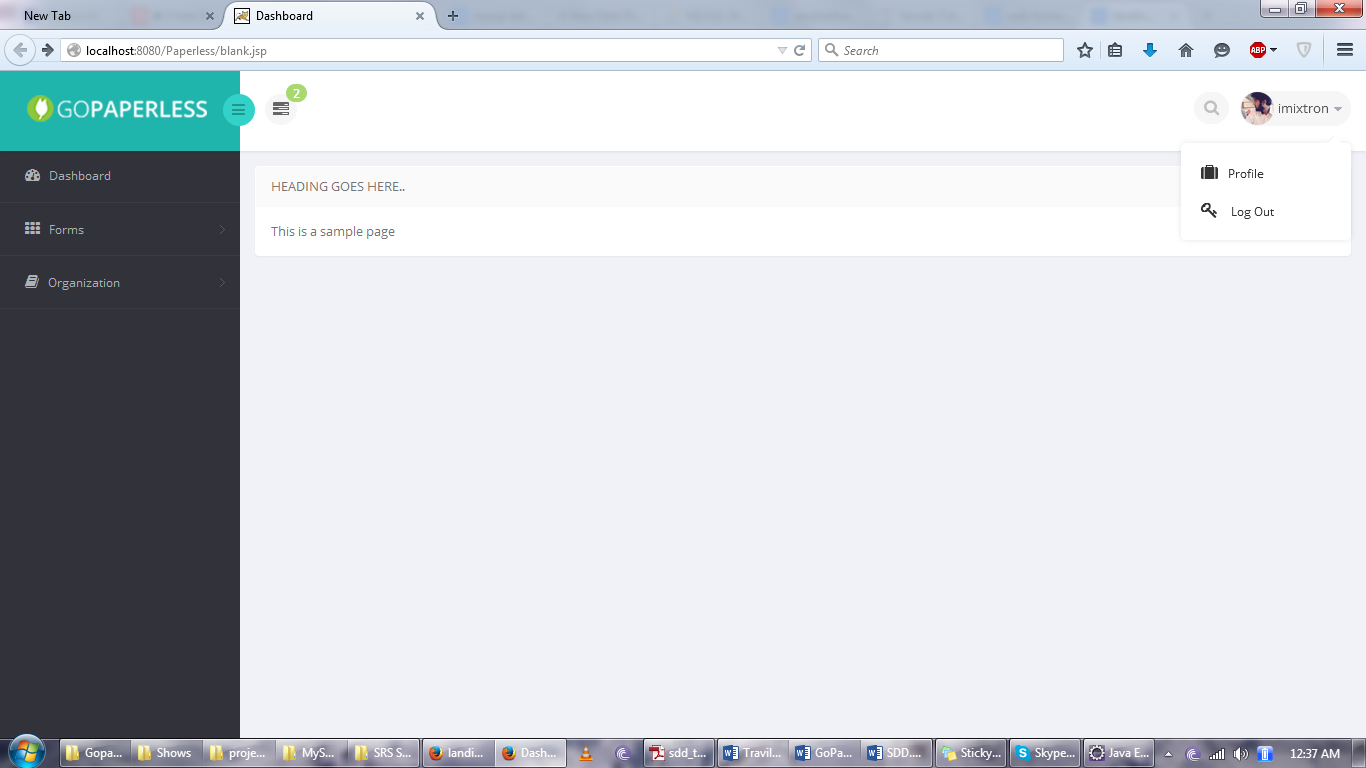


Figure Blank Layout

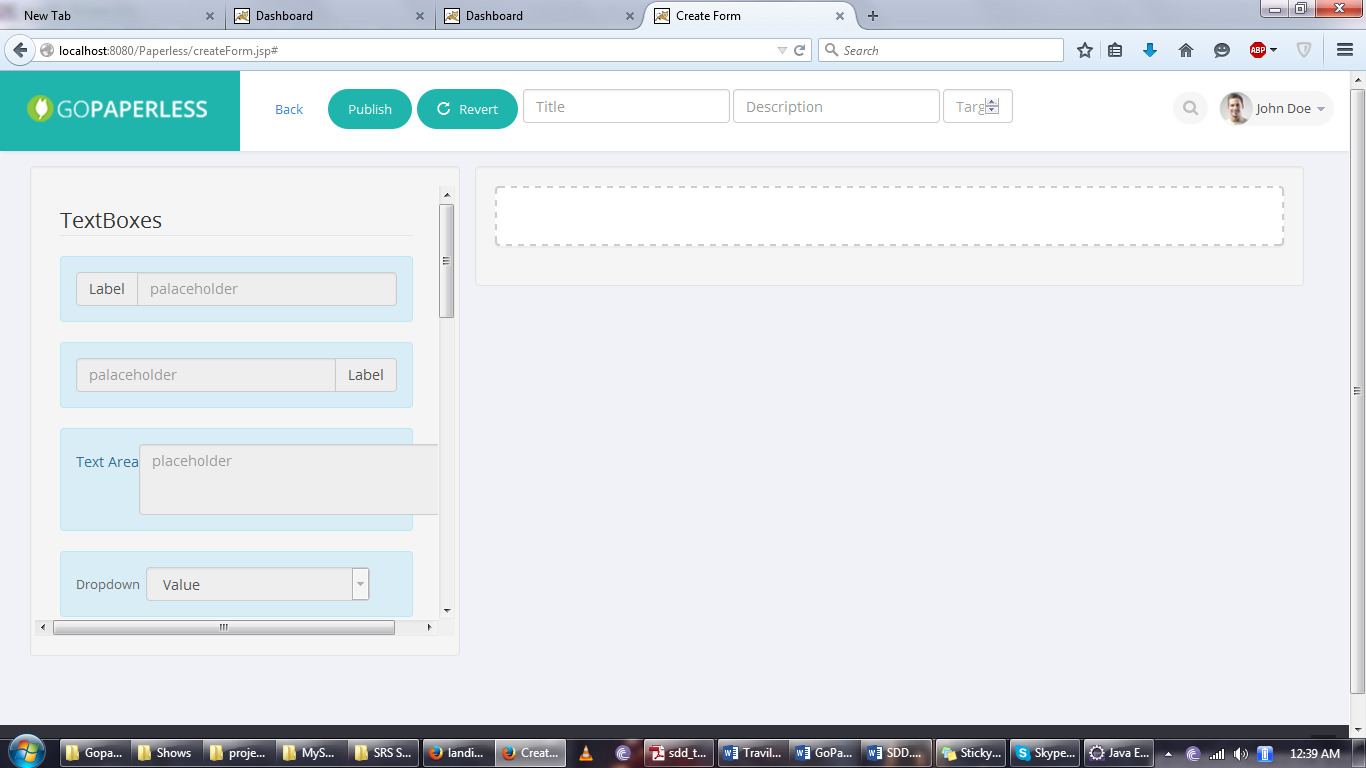


Figure Form Builder layout

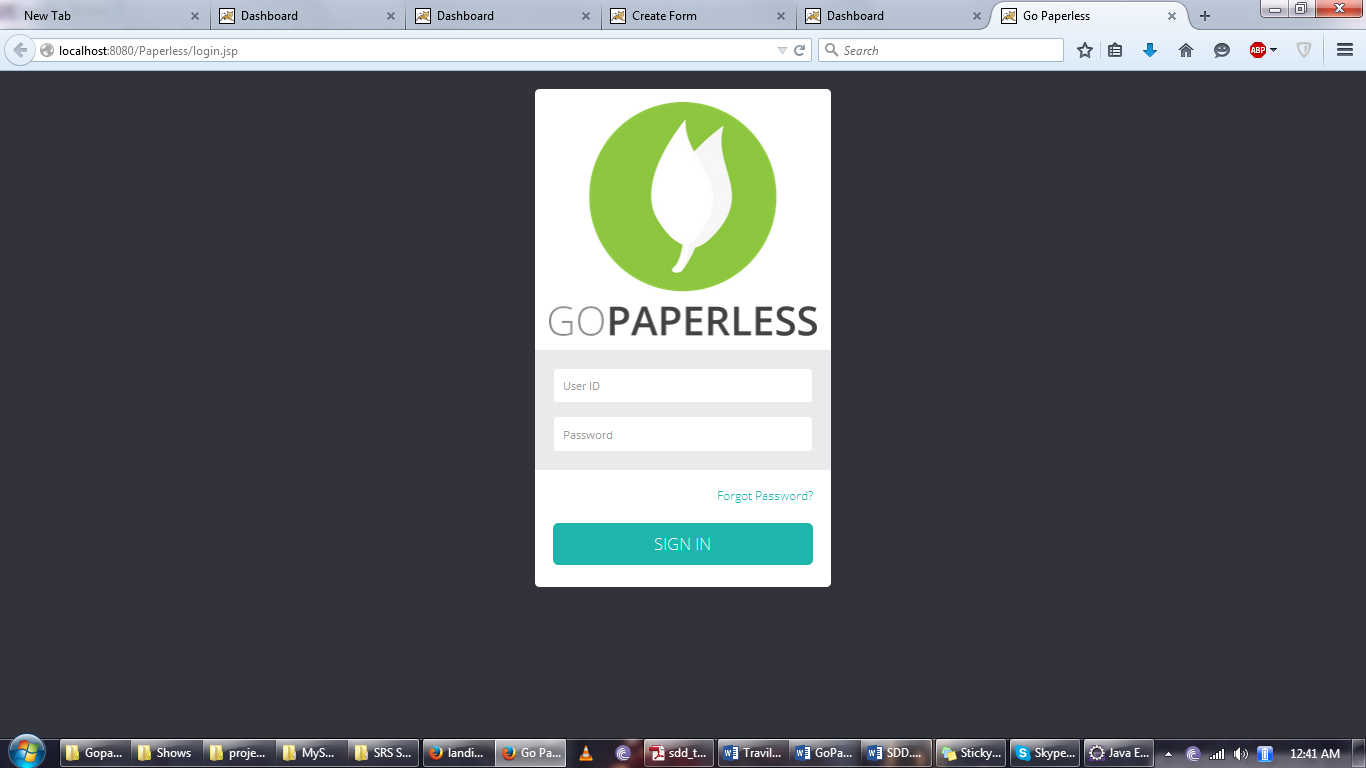


Figure login template

# Appendix



Figure GoPaperless DB ERD

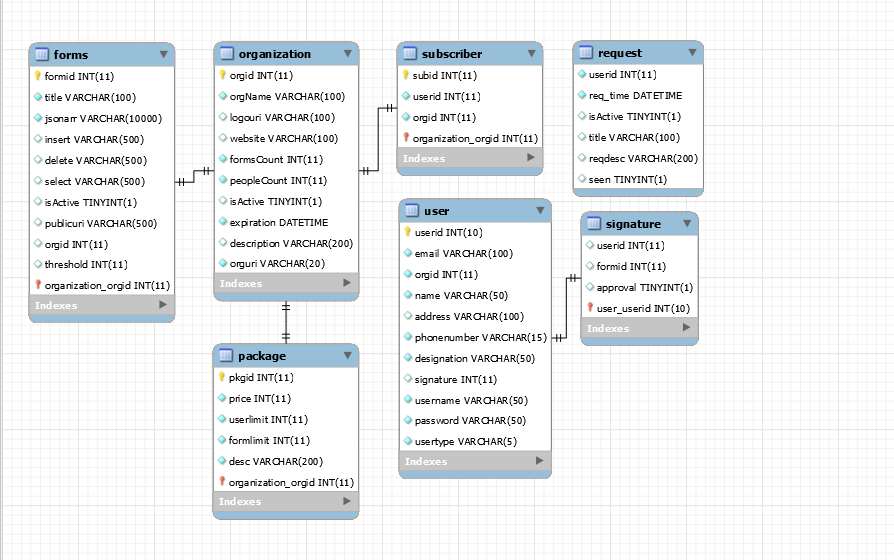


Figure Database Schema

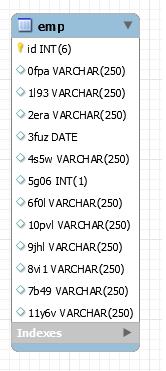


Figure FormData DB prototype table