**Webfactory**

Our Project is based on a small company/start-up named '**Webfactory**' consisting of less than 10 employees that focuses on delivering Web development services.

The process we have undertaken is to '**Build and Deliver a Website**' to our client.

**Workflow:**  
A client approached our company to make a website. We offer him/her a set of templates they can select based on their requirements. Any additional request/change apart from the one mentioned in the template is treated as a chargeable customization like including a world map etc in the website.

The Account Manager handles the request, gathers requirements, assigns a developer to the project. Once the project is built, the Invoice is generated(includes the template charges and additional customization charges) and sent to the client. Once the payment is made, the built website is then hosted on the company’s in-house server (client chooses a hosting plan from a list of packages prior) and the end product website link and access are delivered to the client.

Note: We are not responsible for the maintenance of the website. We are just responsible for developing the website and hosting it onto the server.

So the **Entities** in our System are Client, Work-Order, Website, Hosting, Template\_Type, Invoice, Customization and Employee(indicated in Yellow). Additionally, there are 3 more **Associative Entities** (Indicated in Blue), Website\_Template\_Assosiation, Work\_Emp\_Assosiation and Website\_Customization\_Assosiation.   
  
**Business details and User requirements for our process are as follows**:

1) Whenever a client approaches the company with a requirement, every requirement results in a work order; thus a customer can have one or many work orders but a work order would belong to one and one customer only as work order is unique for a customer to avoid confusion.

2) Each work order can have one or many employees and at the same time each employee can work on zero or many work orders in parallel to increase efficiency and thus to resolve this, we introduced Work\_Emp\_Association, as a bridge connecting Employee and Work\_Order.

3) Each work order can have one or more websites belonging to a single client as one client can order multiple websites in the same work order. But each website belongs to a single work order.

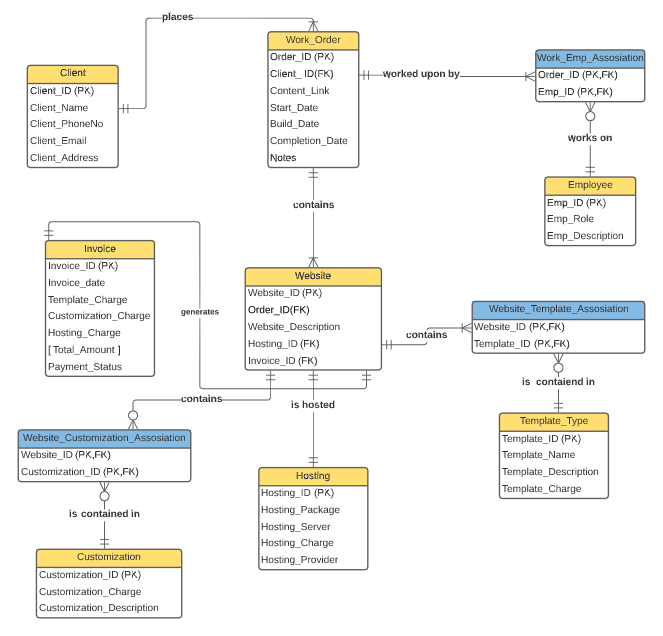
4) Each website will have an invoice associated with it which will track the overall charges incurred- cost of templates chosen, cost of customizations if any and cost of hosting the website. This is a one-to-one relationship as each website will have only one invoice and each invoice will belong to only one website to avoid redundancy.

5) A client can choose 1 or more templates or a combination of them for the website. Hence a website can have 1 or more templates to provide client flexibility to choose multiple templates for the same website and a template can belong to multiple websites. Thus to resolve this many to many relationship, we introduced a Website\_Template\_Association between them.

6) Each website once built is hosted on the server. Hosting plan which will be unique to each website, and one hosting cannot be shared by multiple websites thus it is a one to one relationship as 1 website will only have 1 unique hosting plan.

7) There can be zero or more customization requirements for a website and a particular customization can belong to zero or more websites to give customers a variety of customizations to choose from and thus a many to many relationship. This was resolved using an associative entity: Website\_Customization\_Association.

Note: Here when we say customization, we refer to an additional specific feature that is not present in the template. Eg.Photo Gallery or Map Feature.

[**ERD**](https://lucid.app/lucidchart/abb2fed4-326c-4f3d-9a1a-61015f74bb72/edit?invitationId=inv_f223c4e5-d98b-4ba9-aeeb-e7eb23e80338&page=0_0#) **:** Data types for each field are mentioned in the data dictionary  
  


**Problems faced with the current model:**

-> Too much time elapsed between customer request and end-product delivery.

-> Excel sheets were used to maintain Employee, Template and Customization data. It was too tedious to handle them and to assign Employees to Work Order with necessary requirements.

-> Manual task delegation and follow up.

-> Manual delegation of Developer to Website.

-> In case of an existing client, no prior record of client details or pre-existing projects/preferences. Every customer is treated as a new customer.

-> The client does not get timely updates.

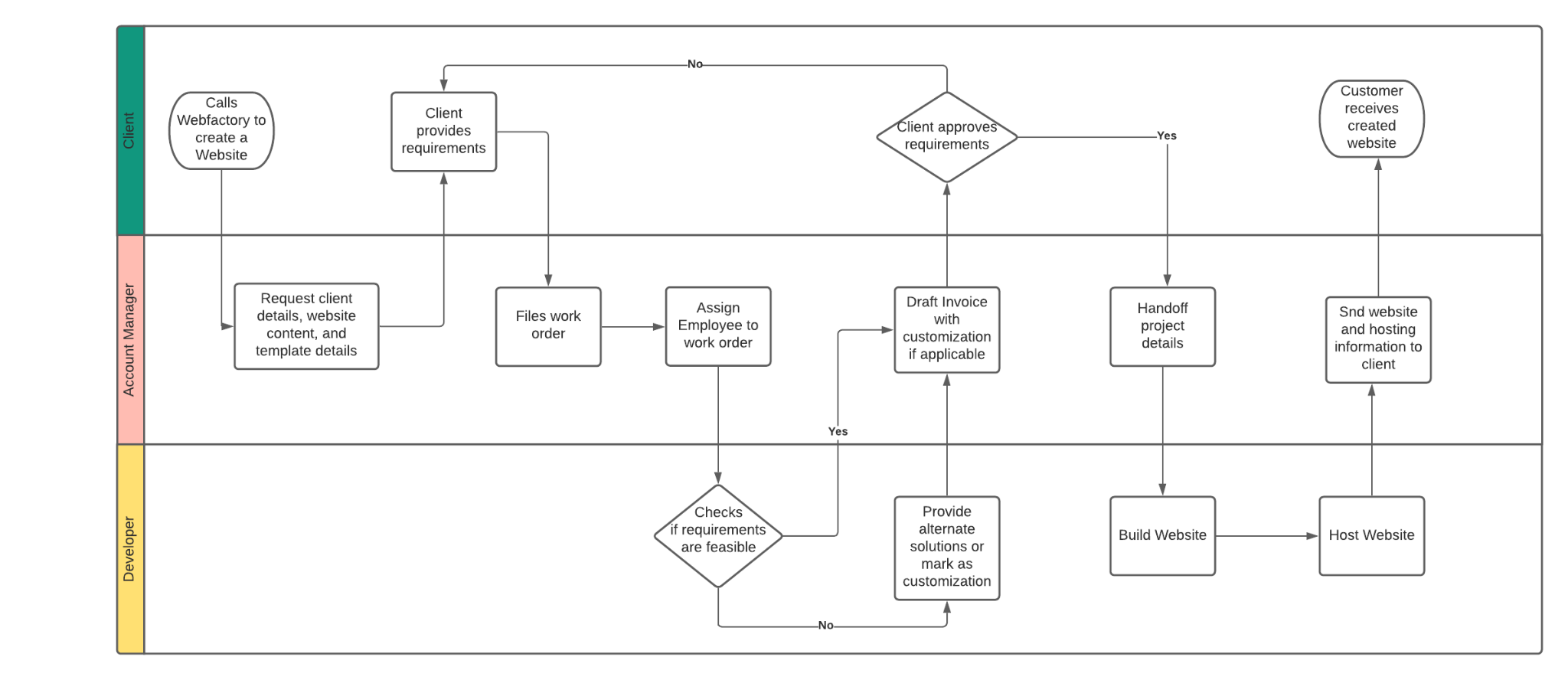
-> Accumulation/Pile-up of projects.

-> Low efficiency of process.

-> Too many handoffs

-> Feasibility of requirements.

We can clearly see these Issues in our Model’s [**As is diagram**](https://lucid.app/lucidchart/4eb31b3e-2504-4fda-85dc-7479b41f38fe/edit?viewport_loc=-62%2C-434%2C2901%2C1334%2C0_0&invitationId=inv_d3ff964b-3670-4efe-b80a-a2666b8351ee) :



**Our solution :**

**Enabler:** Information Systems, Workflow design

->We Introduced a database system that helps optimize workflow and reduce need of human intervention in process  
  
**How has the introduction of Database System helped workflow?**

->Database stores client details. Hence, returning customers and their information is identified.

-> Database stores Employee details, template details and customization details which makes it easier to assign employees to work order with necessary requirements.

->Client directly logs all of their requirements- template, website content, preferences, hosting details, customizations etc into the system through a google sheet. This automatically triggers creation of a Work order.

-> Employees and developers who are available, are automatically assigned to the work order and respective employees and developers are notified as well.

-> Invoice generated by the system, consisting of all agreed upon charges - customization, hosting, template etc.

->Chances of error are reduced as manual work is reduced. Efficiency is increased.

-> Once the website is built, website and hosting details are automatically sent to the client.

**Objectives and Benefits:**

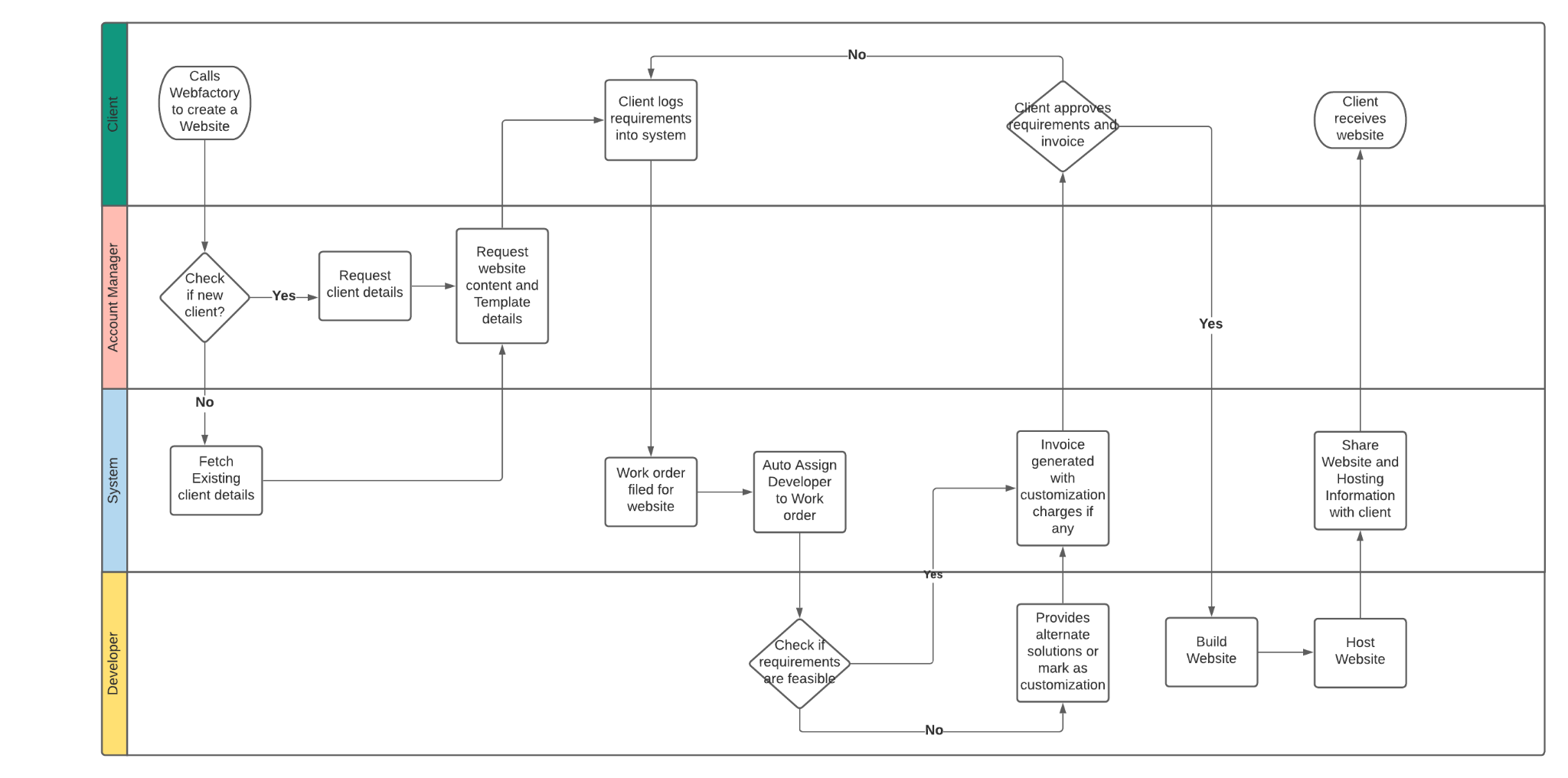
-> Improve process efficiency.

-> Reduce time elapsed between project request and completion.

-> Create a database that tracks every client, project, requirement, and progress.

-> Auto total cost of customizations and generate invoice to customer.

-> Account Manager can log details into database directly as opposed to manual handoff to developer.

How will our diagram look like after the solution :  
  
[**To be diagram :**](https://lucid.app/lucidchart/26f35305-f62d-4cb5-96d5-42dc1658c6d2/edit?viewport_loc=-378%2C-336%2C3330%2C1461%2C0_0&invitationId=inv_5ccd976e-0190-4f5d-858f-c3a8f95ead08)  
  


**Data Dictionary :**[**Data Dictionary**](https://docs.google.com/spreadsheets/d/1T8Slz0NMAjTeEH91d_xy57i2A8RAZ-9hgYDsZMDvMtY/edit#gid=0)

****