IT PROJECT 700

NAME	:	TRECHEL
SURNAME	:	MABUNDA
STUDENT NUMBER	:	401911580

PAINTILES CONSTRUCTION PROJECT PROPOSAL PHASE 2

Table of Content

1) Introduction

The System design phase is a crucial part of this project, where ideas are transformed into a tangible product or service. It is a phase where the project's objectives, requirements, and constraints are defined, and the project's solution is designed. The design phase plays a vital role in ensuring that the project's outcome meets the needs of the stakeholders and delivers the intended value. In this article, I will define the design phase in project management and evaluate its role in the project management process. This design phase is a systematic process that involves several activities, including planning, creating, and organizing the project's solution. The primary objective of the design phase is to develop a detailed plan that outlines how the project will be implemented. This plan includes defining the project's architecture, developing components, and creating design specifications. The design phase will also involves making critical decisions that will impact the project's success, such as selecting the appropriate technology, determining the project's security requirements, and defining the project's user interface. The design phase plays a vital role in ensuring that the project's outcome meets the needs of the stakeholders and delivers the intended value. The following are some of the key reasons why the design phase is essential:

It will ensures that the project's solution is feasible and meets the objectives: The design phase will help to ensure that the project's solution is feasible and meets the objectives by defining the project's requirements, constraints, and objectives. This helps to avoid costly rework and ensures that the project's outcome is aligned with the stakeholders' needs. It helps to reduce risks and uncertainties: The design phase helps to identify potential risks and uncertainties associated with the project. This allows the project team to develop strategies to mitigate these risks and ensure that the project's outcome is delivered on time and within budget. It helps to ensure that the project's solution is efficient and effective: The design phase helps to ensure that the project's solution is efficient and effective by defining the project's architecture, components, and design specifications. This helps to ensure that the project's solution is scalable, reliable, and meets the needs of the stakeholders.It helps to ensure that the project's solution is user-friendly: The design phase helps to ensure that the project's solution is user-friendly by defining the project's user interface and defining the project's security requirements. This helps to ensure that the project's outcome is easy to use and meets the needs of the stakeholders. Final answer: In conclusion, the design phase is a crucial part of the project management process, where ideas are transformed into a tangible product or service. The design phase plays a vital role in ensuring that the project's outcome meets the needs of the stakeholders and delivers the intended value. The design phase helps to ensure that the project's solution is feasible, efficient, effective, and user-friendly. Therefore, it is essential to invest time and resources in the design phase to ensure that the project's outcome is successful.

2) System Design (Description of Proposed System)

a) Requirements Gathering: The web-based Application is expected to provide clients with quotations and answer frequently asked question about the business
b) Requirement Analysis: According to our analysis, the requirement analysis of the web-based application shows that there will be no conflict or ambiguities.
c) Feasibility study: A feasibility study was conducted to assess the technical, economic, and operational feasibility of the proposed software system which is the Web Based Application. This has helped in determining whether the project should proceed to the next phase or not.
d) System modeling: Various modeling techniques, such as use case diagrams, activity diagrams, and data flow diagrams, were used to represent the system requirements and its behavior. These models help in visualizing and understanding the system.
e) Risk assessment: Potential risks and challenges associated with the software development project are identified and analyzed. This helps in developing strategies to mitigate or manage these risks.
f) Cost estimation: The cost of developing the software system is estimated based on the requirements and the complexity of the project. This helps in budgeting and resource allocation.
g) Stakeholder communication: Throughout the analysis phase, effective communication with the stakeholders is crucial. Regular meetings and discussions are held to ensure that the requirements are well-understood and any changes or clarifications are addressed.

3) Architectural Design (Software Architectural Design, Hardware Architectural Design, Network Architectural Design, class diagram)

Software Architectural Design

Client-Server Architecture: The system is divided into client and server components, with the client making requests to the server for data or services. Client-server architectures is a type of network architecture where a central server provides services or resources to multiple client devices. The clients request and receive data or services from the server, which manages and coordinates the resources. This architecture is commonly used in distributed computing systems and allows for efficient sharing of resources and centralized control.

Required attributes for client-server architectures:

- Central server: A central server will provides services or resources to clients.
- Clients: Multiple client devices that request and receive data or services from the server.
- Network: A network infrastructure that connects the server and clients, allowing communication between them.
- Request-response model: Clients send requests to the server, which responds with the requested data or services.
- Scalability: The architecture should be able to handle a large number of clients and scale as the number of clients increases.
- Security: Measures should be in place to ensure the security and privacy of data transmitted between the server and clients.

Hardware Architectural Design

The hardware architectural design typically involves several key components to ensure scalability, reliability, and performance.

Here are the main elements:

- Web Server: This handles HTTP requests from clients. Common choices include Apache, Nginx, and Microsoft IIS.
- Application Server: This processes the business logic of the application. Examples include Node.js, Django, and Ruby on Rails.
- Database Server: This stores and manages the application's data. Options include MySQL, PostgreSQL, and MongoDB.
- Load Balancer: This distributes incoming network traffic across multiple servers to ensure no single server becomes overwhelmed. Examples are HAProxy and AWS Elastic Load Balancing.
- Cache: This stores frequently accessed data to reduce load on the database and improve response times. Common caching solutions are Redis and Memcached.
- Content Delivery Network (CDN): This distributes static content like images, CSS, and JavaScript files to servers closer to the user, reducing latency. Examples include Cloudflare and AWS CloudFront.

- Firewall and Security: This protects the application from malicious attacks. Solutions include hardware firewalls and services like AWS WAF.
- Monitoring and Logging: This tracks the performance and health of the application. Tools include Prometheus, Grafana, and ELK Stack (Elasticsearch, Logstash, Kibana).

A typical request flow might look like this:

- A user makes a request to the web server.
- The web server forwards the request to the load balancer.
- The load balancer distributes the request to one of the application servers.
- The application server processes the request, possibly querying the database server or cache.
- The response is sent back through the load balancer and web server to the user.

In terms of hardware:

- Web Server: 2 CPUs, 4 GB RAM
- Application Server: 4 CPUs, 8 GB RAM
- Database Server: 8 CPUs, 16 GB RAM, SSD storage
- Cache Server: 2 CPUs, 4 GB RAM

Network Architectural Design

To design a network architecture for a web application, you need to consider several key components and their interactions. Here’s a concise outline:

- Client-Side: This includes web browsers or mobile apps that interact with the web application. They send requests to the server and display the responses to the user.
- Load Balancer: Distributes incoming traffic across multiple servers to ensure no single server becomes a bottleneck. This improves reliability and performance.

- Web Servers: Handle incoming HTTP requests, execute application logic, and return responses. Common web servers include Apache, Nginx, and IIS.
- Application Servers: Execute the core business logic of the web application. They process requests from the web servers and interact with the database servers.
- Database Servers: Store and manage data. They handle queries from the application servers. Common databases include MySQL and MongoDB.
- Caching Layer: Improves performance by storing frequently accessed data in memory. Common caching solutions include Redis and Memcached.
- Content Delivery Network (CDN): Distributes static content (like images, CSS, and JavaScript files) across multiple geographical locations to reduce latency.
- Firewall: Protects the network by filtering incoming and outgoing traffic based on security rules.
- Monitoring and Logging: Tools to monitor the health and performance of the application and log events for troubleshooting and analysis.
- Backup and Recovery: Ensures data is regularly backed up and can be restored in case of failure.

Here is an example to a network architecture for connecting to a database

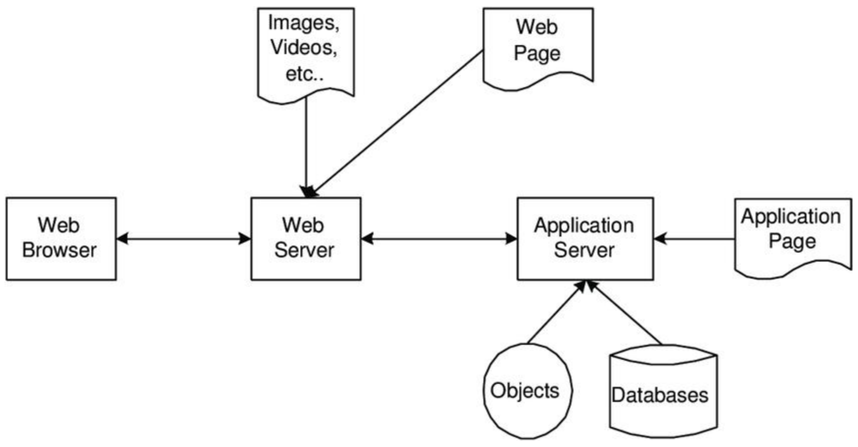
```
<?php
$servername = "localhost";
$username = "your_username";
$password = "your_password";
$database = "your_database";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $database);

// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";

// Perform database operations here

// Close connection
mysqli_close($conn);
?>
```

Class Diagram

Clent Class Diagram

```
–––––––––––––––––––––
|   Client          |
–––––––––––––––––––––
| – id: int         |
| – name: string    |
| – age: int        |
| – address: string |
| – phone: string   |
–––––––––––––––––––––
```

4) Physical Design for my web application



5) Database Design

Creating a Database for my Web Application

Connect to MySQL

```
mysql -u root -p
```

```
------------------------------Create a new database---------------

CREATE DATABASE webapp_db;

-- Use the database---

USE webapp_db;
```

Create Tables
Creating a simple user management system with users and roles tables.

sql

```
------Create the roles table------

CREATE TABLE roles (
   id INT AUTO_INCREMENT PRIMARY KEY,
   role_name VARCHAR(255) NOT NULL
);

-- Create the users table
CREATE TABLE users (
   id INT AUTO_INCREMENT PRIMARY KEY,
   username VARCHAR(255) NOT NULL,
   password VARCHAR(255) NOT NULL,
   email VARCHAR(255) NOT NULL,
   role_id INT,
   FOREIGN KEY (role_id) REFERENCES roles(id)
);
```

Insert Initial Data
sql

```
-- Insert roles--

INSERT INTO roles (role_name) VALUES ('Admin'), ('User');

-- Insert users
INSERT INTO users (username, password, email, role_id) VALUES
('admin', 'admin_password', 'admin@example.com', 1),
('user1', 'user1_password', 'user1@example.com', 2);
```

Step 5: Connect to the Database in Your Web Application
Here is an example using Node.js with the mysql2 package.

Install mysql2 package:

bash

```
npm install mysql2
```

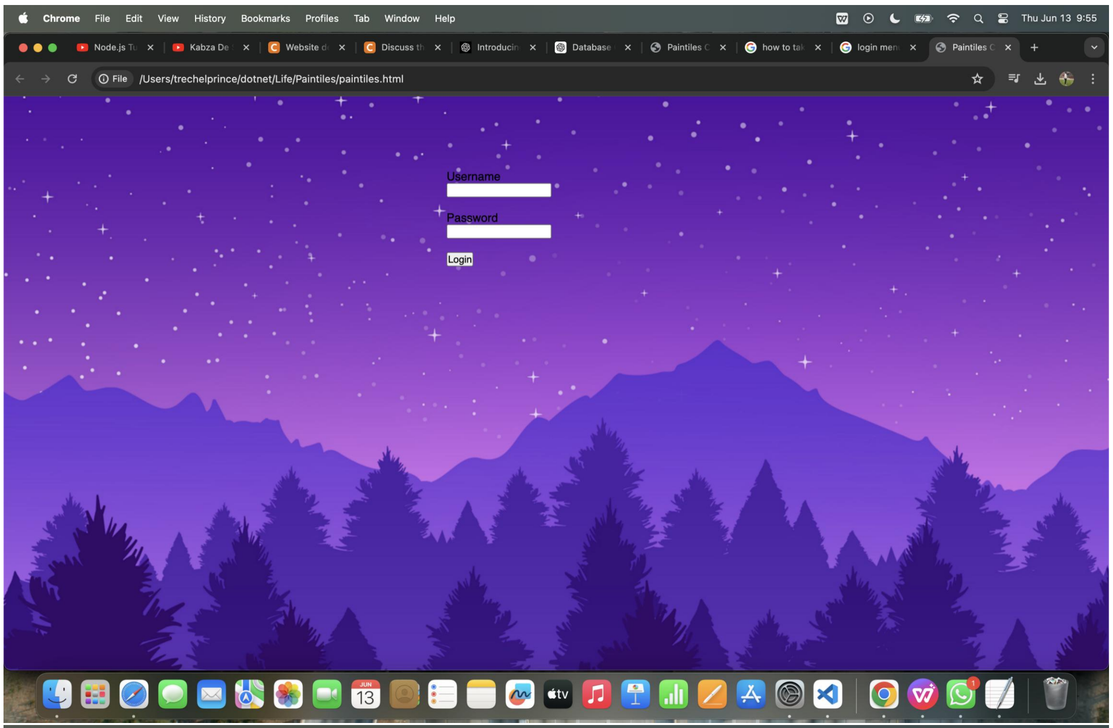
Create a database connection:

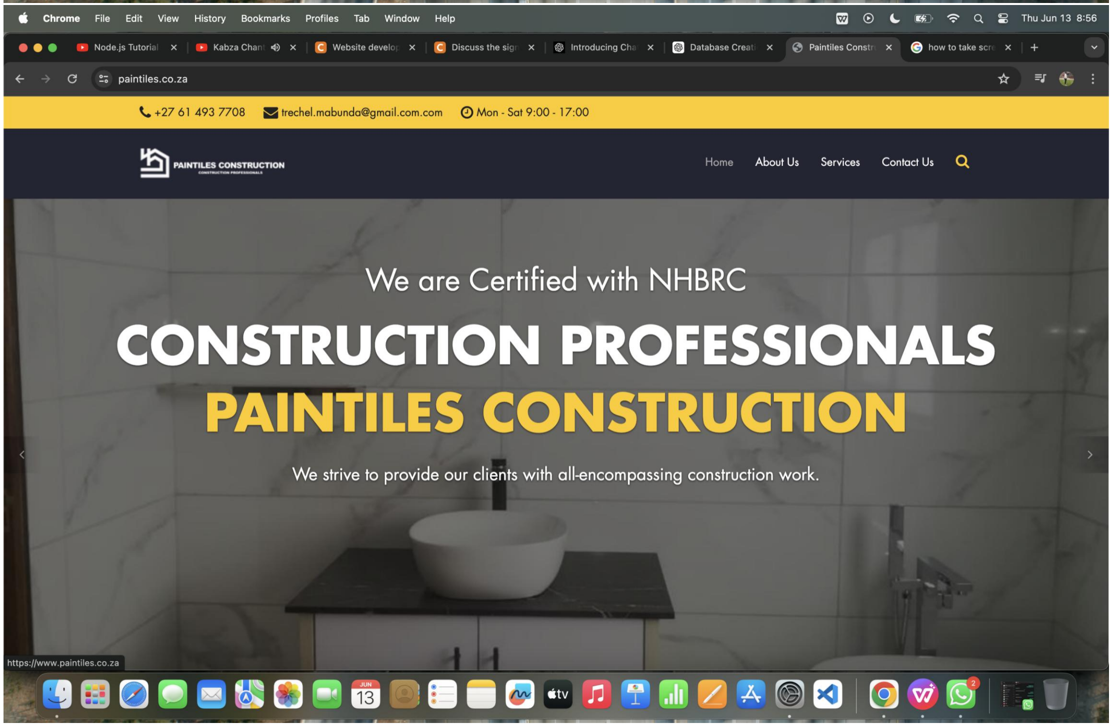
```
javascript

const mysql = require('mysql2');

// Create a connection to the database
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: 'your_password',
  database: 'webapp_db'
});

// Connect to the database
connection.connect(error => {
  if (error) throw error;
  console.log('Successfully connected to the database.');
});

// Example query
connection.query('SELECT * FROM users', (error, results) => {
  if (error) throw error;
  console.log(results);
});
```

6) Program design (Pseudo Code)

```
// Define necessary data structures and variables

// User Authentication
function login(username, password):
    // Validate username and password
    // Authenticate user
    // Generate session token
    // Return session token or error message

// Add a client
function addclient(ClientData):
    // Validate Client data
    // Generate unique Client ID
    // Save Client data in the database
    // Return success message or error

function getClient(ClientID):
    // Retrieve Client data from the database using Client ID
    // Return Client data or error

function updateClient(ClientID, newData):
    // Validate and update Client data in the database using Client ID
    // Return success message or error

function deleteClient(ClientID):
    // Delete Clinet data from the database using Client ID
    // Return success message or error
// Quotation Management
function scheduleQuotation(ClientID, EmplyeeID, appointmentDateTime):
    // Check if the appointment slot is available
    // Create a new appointment record with Client ID, Employee ID, and appointment date/time
    // Return success message or error

function cancelAppointment(appointmentID):
    // Retrieve appointment data using appointment ID
    // Delete appointment record from the database
    // Return success message or error

// Billing Management
function generateInvoice(ClientID, services):
    // Calculate the total cost of services
    // Generate an invoice with Client details, services, and total cost
    // Return the invoice or error

 exit the program
```

7) Interface Design (Menu InterfaceDesign, Input Design, Output Design)




paintiles.co.za
+27 61 493 7708
trechel.mabunda@gmail.com.com
Mon - Sat 9:00 - 17:00
PAINTILES CONSTRUCTION
Home
About Us
Services
Contact Us
We are Certified with NHBRC
CONSTRUCTION PROFESSIONALS
PAINTILES CONSTRUCTION
We strive to provide our clients with all-encompassing construction work.

8) Security back up design (Software concern)

Strategy to the backup website and its database

1. Manual Backup

In the manual backup, I will mostly do backup in our local machine only. I will just save all files of our website as well as its database in our local machine. It is a very low-cost method to backup a website. The main disadvantage of this method is that it takes a lot of time to download all files and database of a website as well as we have to make our own logics to save it in our machine so that we can easily access and manage them. Another issue is that it also download all updates as well when our website or database is updated.

2. By Using CPanel backups

We will also look into Cpanel backup option also to backup our website as well as the database. There is a backup icon present in Cpanel from where we can simply click on the generate backup button and enter our email id. We will receive a mail when our backup is ready and then we can download it and keep in multiple hard drives or any storage media. The main disadvantage of this method is that we have to download all updates on our website when our website or database is updated every time.

3. Backup in Cloud

This is the most efficient way of backup od any website as well as its database. Many Cloud storage providers such as AWS(Amazon Web Services). Google Cloud Platform and Microsoft Azure are there which provide website backup facilities on their cloud at a reasonable cost and guarantee us for protection of our website as well as its data from any virus or malware attack. This method of website backup should always be used. The proble with this cost is that it is costly.

4. Using Rsync

This method of website and its database backup is also very efficient. Rsync is simply a software which copies all files from one server to another server. It copies all files of the website to another server where we want a backup. It also backs up the database of our website. It uses the MYSQL database for backup of database. It simply runs a script known as mysqldump which creates a backup for the database of our website. Rsync is more effective when the second server is Linux because just by typing a simple command it automatically creates all backup of the websites along with its database as MYSQL is by default installed in Linux. My last option is expensive.