

CCS6344 T2510 Assignment 1 Submission

Group Name: Group 12

|  |  |
| --- | --- |
| **Student ID** | **Name** |
| 1211112284 | Nur Dania Iman Binti Desman Desa |
| 1211101800 | Tan Jia Jin |
| 1211101775 | Lam Yuet Xin |

**Deadline**

Submit your deliverables to eBwise on or **before 11:59pm, Monday, 19 th May 2025**. Every member of the group must submit their report individually in the Ebwise

**Calibri with size 12**. All diagrams and figures must be labelled and captioned. Maximum pages allowed is **10**.

Task Distribution

1. Web app
2. Sql database
3. Threat modeling
4. PDPA
5. Security Measures Implementation (choose 3)
6. Access control
7. Encryption
8. Audit
9. backup

Your proposal should cover the following topics:

1. Objectives of the project.

2. Proposed design and implementation of the application.

3. Proposed hardware and software to develop the application.

a. Please explain what programming language and database programme you are

planning to use to develop this application.

b. This includes the type of server OS and webserver that you want to use to

develop this application.

4. Please include the system design and database design for this application.

5. Please explain how you are planning to secure the database using the SQL database

system.

Give a concise explanation of the business strategy and accurately state the project's goal.

Give a precise explanation of the application's design, hardware specifications, and

software. Additionally, make sure to clarify the security precautions needed to safeguard

the application.

# Task 1 Preparation of the proposal

## Objective

The aim of this project is to design, develop, and secure a web-based application using the conventional SQL database system in line with comprehensive security best practices and processes for regulatory compliance with PDPA 2010 regulations. Application is created to effectively demonstrate the implementation of database security measures properly to make it secure against common vulnerabilities and threats.

Threat analysis will be done by applying STRIDE and DREAD.

Validation of protection will be tested through penetration testing. Correct data handling procedures under PDPA 2010 will be implemented. The project will present project documentation, including a technical report and presentation.

* Build a web application with proper authentication mechanism
* Imposing security controls on general threats like SQL injection attacks
* An analysis of threats using the STRIDE and DREAD framework
* Working on data handling processes conforming to the PDPA 2010 regulations
* Validation of security through penetration tests
* Preparation of all professional documentation and presentation

## Design and Implementation of the Application

## This project aims to build a web-based bakery ordering system that allows customers to browse products, place orders, and manage their accounts while providing bakery staff with order management capabilities. It serves the marketing side of the bakery ordering system by charting a quick seamless digital experience that has traditionally present both the bakery business and e-commerce conveniences. The main goal is for the online bakery to step outside the offline walk-in culture, generate more sales through online pre-orders, and lessen offline carrying-width wait times. The system, therefore, pushes through additional income by way of upselling (custom cake designs), repeat sales through customer preference saving, and operational efficiency through inventory management integration. A phased rollout ensures rapid delivery of core features, with revenues later channeled into subscriptions and loyalty programs. Ultimately, the strategy sets the bakery as BenchBeyondiler: a neighborhood favorite endowed with the clever side of tech and artisanal craft juxtaposed with digital convenience and custom-mediated services, competing against both neighborhood bakeries and bulk processes.

??

## hardware and software to develop the application

a. Please explain what programming language and database programme you are planning to use to develop this application.

HTML and CSS are used to design the front-end interface of this application with the sole intent of maintaining a user-friendly and attractive layout. Server-side scripting and database communications are primarily handled by PHP in back-end logic. The database is designed and managed by means of SQL queries which are made available by XAMPP. This allows efficient processes of storing, retrieving, and managing application.

1. Programming Language: PHP for backend, HTML/CSS/JavaScript for frontend.
2. Database: MSSQL for data storage. SSMS for database management.
3. Operating System: Windows 11.
4. Server: XAMPP Local Server

b. This includes the type of server OS and webserver that you want to use to develop this application.

Windows Server 2019 64-bit is being used as the server operating system to run the application under oracle virtual box. This virtualized environment allows the simulation of a server setup for testing and development purposes without the need for any physical hardware. Apache HTTP Server under the XAMPP package serves as the web server. Apache manages requests for the web and performs the execution of PHP scripts, commencing the smooth interaction of the interface with the backend of the application. It is a stable and flexible platform to efficiently develop and test our web-based system.

System design and database design for this application

**SYSTEM DESIGN**

The system is designed using a multi-layered approach:

1. Application Layer: Backend server and business logic, which handles the processing of user requests, application logic, and communication between the presentation layer and the data layer.
2. Presentation Layer: User Interface, which includes the web pages and forms that users interact with.
3. Data Layer: MSSQL database for data storage, which stores all the data required by the application, including user information, product, staff, order and payments.

**DATABASE DESIGN**

The database includes the following tables:

1. Customers: To store the customer’s personal details.
2. Products: To store product’s details.
3. Staff: To store the staff’s personal details.
4. Orders: To store order receipt.
5. Order\_Items: To store order details placed by customers.
6. Payment: To manage payment details.

## Secured database using the SQL database system

To secure the database, the following measures are implemented:

1. Authentication: Users must verify their identity before accessing the database by validating their usernames and passwords against stored credentials.
2. Authorization: After authentication, users are granted access only to the data they are authorized to view or modify, based on their roles and privileges.
3. Data Encryption: Sensitive data stored in the database is encrypted to protect it from unauthorized access.
4. Password Hashing: Passwords are hashed before being stored in the database to ensure that even if the data is compromised, the actual passwords are not exposed.
5. Data Masking: Sensitive information, such as credit card numbers, is masked within the database.

# Task 2 Implementation of the application using SQL Database

Web app = html/css, php

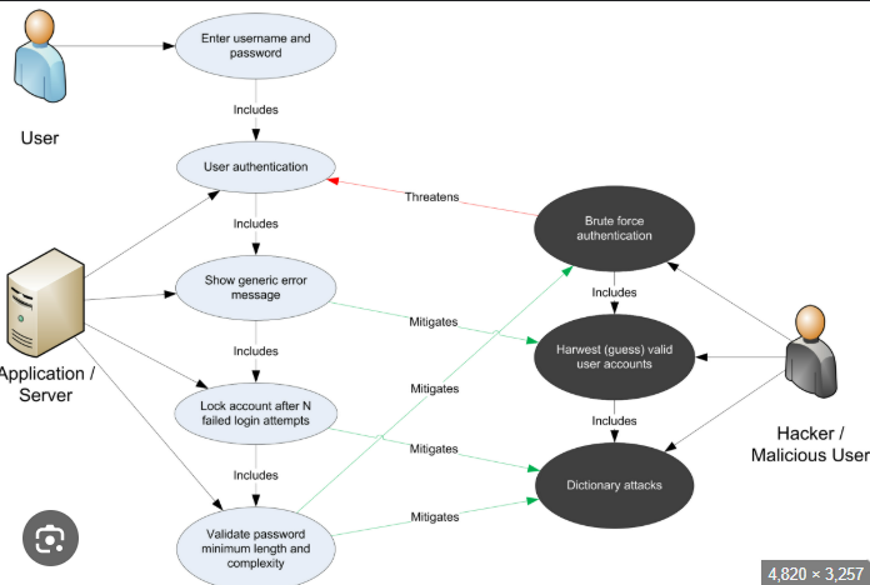
xampp

# Task 3 Threat Modelling

Using the STRIDE and DREAD threat modelling, please explain and provide justification for how your application can pass through these tests.

The application adopts a multi-shield approach to defense against STRIDE with concurrent measures to bring down levels of risk posed by DREAD. To authenticate and control access, we spoof prevention territory (STRIDE) with strong policies on password setting, while the DREAD score for identity risk remains low at 2.4. Least privilege accordingly restricts elevation of privilege based on requirements: administrators have complete permissions, whereas users are granted limited permissions. Data protection-TDE on database files, TLS/SSL on connections, and column-level encryption on sensitive PII-provide a concerted defense against threats of information disclosure (with a DREAD score coming down from 4.4 to lower, manageable levels). Parameterized queries and input sanitization thwart tampering-pray SQL injection; while repudiation gets addressed by audit logging for login attempts and schema changes. Network security comes through firewall rules allowing database access only to an IP of trust and greatly shrinking the attack surface. Business continuity thus gets past denial omnibuses scenarios by allowing our encrypted automated backup with the offsite storage and been tested for disaster recovery. In essence, these are layered security mechanisms that serve a purpose: to provide end-user usability while covering all STRIDE categories and reducing risk to acceptable levels along the lines of Detection, Prevention, and Corrective controls. Exploits will be regularly tested to continually assess these safeguards against an ever-changing threat environment. In essence, these are layered security mechanisms that serve a purpose: to provide end-user usability while covering all STRIDE categories and reducing risk to acceptable levels along the lines of Detection, Prevention, and Corrective controls. Exploits will be regularly tested to continually assess these safeguards against an ever-changing threat environment.

??



# Task 4 PDPA 2010

+ diagram

# Task 5 Security Measures Implementation

Role and SQL User

1. Bakery\_customer
2. Bakery\_staff
3. Bakery\_admin
4. Hr (to create staff account in database) -> audit log hr recruitment and firing
5. dbowner
6. Sysadmin
7. Guest (only can view)

Choose 3

1. **Authentication & Access Control**

* **User Authentication**
  + Implement strong password policies
  + Use multi-factor authentication (if supported)
* **Role-Based Access Control (RBAC)**
  + Assign permissions to roles (e.g., Admin, User, Guest)
  + Restrict access to sensitive tables/operations
* **Least Privilege Principle**
  + Grant only necessary permissions (e.g., SELECT only for reports)

1. **Data Protection**

* **Encryption**
  + **At Rest**: Use TDE (Transparent Data Encryption) for database files
  + **In Transit**: Enable TLS/SSL for connections
  + **Column-Level Encryption** for sensitive data (e.g., passwords, PII)

**3. Threat Prevention**

* **SQL Injection Mitigation**
  + Use **parameterized queries** (prepared statements)
  + Input validation/sanitization
* **Audit Logging**
  + Track login attempts, schema changes, and data access
  + Log failed queries for attack detection
* **Firewall Rules**
  + Restrict database access to trusted IPs

**4. Backup & Recovery**

* **Regular Backups**
  + Automated encrypted backups (full + incremental)
* **Disaster Recovery Plan**
  + Test restoration procedures
  + Store backups offsite