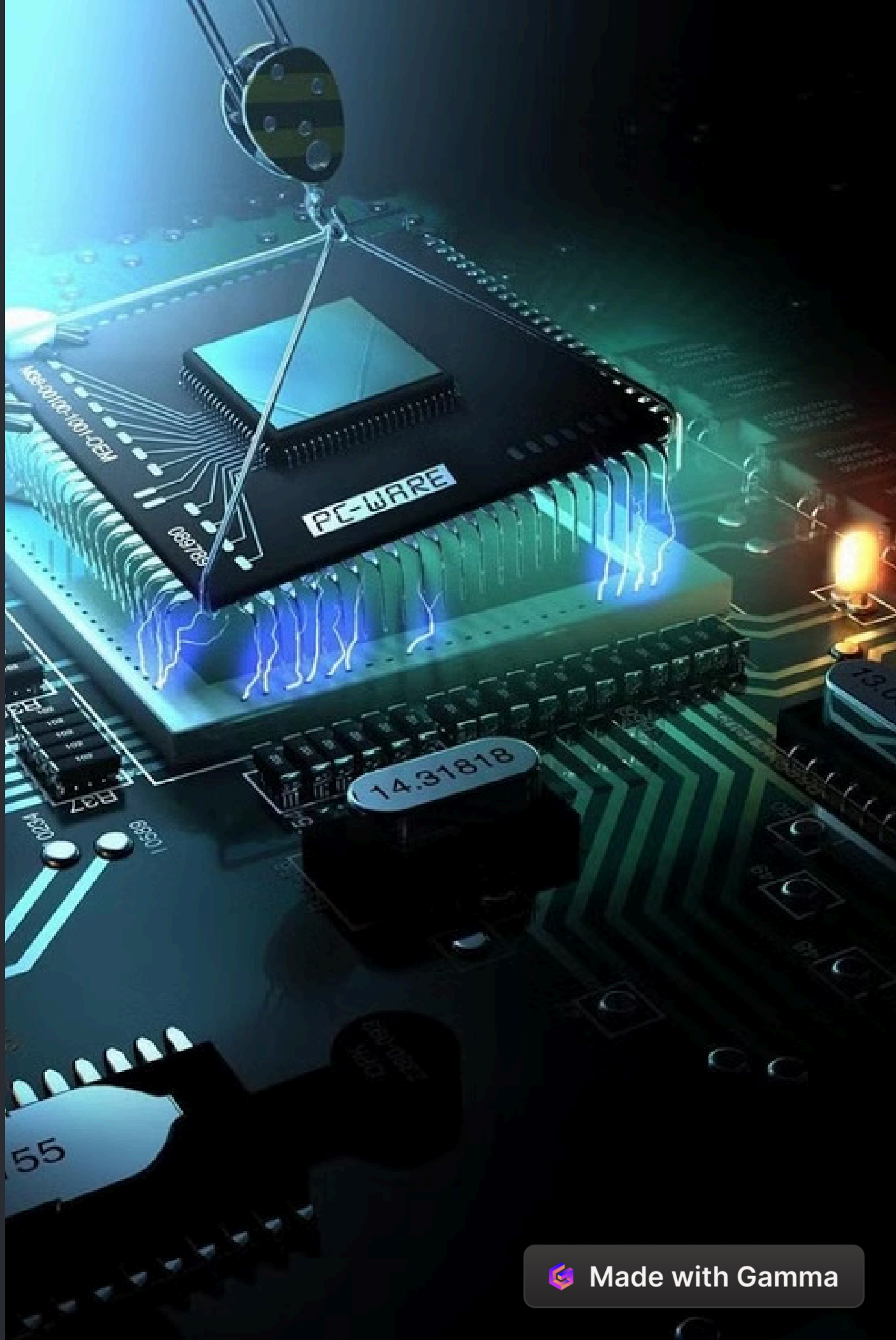
Course Management System (CMS)

This presentation provides a brief overview of the Course Management System (CMS) developed by a team of students at Adventist University of Central Africa (AUCA) for PL/SQL Database course. The project aimed to create a user-friendly database that streamlines course management tasks for lecturers and enhances student engagement.

by MUGISHA JULIEN



Project Team and Tools

Team Cardinals

- IRADUKUNDA Delphine (Leader)
- MUGISHA Julien
- SHEJA N M Yves
- ISHIMWE Mireille

Tools used

- Oracle Enterprise Manager
- SQL
- CMD
- Figma

Project Phases

1. **Introduction**

   Define project scope and objectives, highlighting the need for a centralized course management system.

2. **Business Process Modeling**

   Model the system's core functionalities, focusing on attendance, assignments, and grading.

3. **Logical Model Design**

   Define entities and relationships, ensuring a robust and structured data model.

4. **Database Creation**

   Create the database using Oracle, configuring tablespaces and managing user access.

5. **Table Creation & Data Insertion**

   Create tables for departments, lecturers, students, courses, assignments, submissions, grades, and attendance.

6. **Database Interactions & Transactions**

   Develop views and JOINS for data retrieval and create transactions for data integrity.

7. **Advanced Database Programming & Auditing**

   Implement triggers, packages, cursors, and auditing mechanisms to enhance system functionality and security.

Phase 1: Introduction

1 Challenge

Managing multiple courses and large classes can be overwhelming for university lecturers without a centralized system.

2 Solution

The CMS provides a user-friendly database for streamlined course management.

3 Integration

Integrate attendance tracking, assignment organization, and grade monitoring into one platform.

4 Benefits

Reduce administrative burdens, enhance efficiency, and improve the overall learning experience for students.

Phase 2: Business Process Modeling

Scope

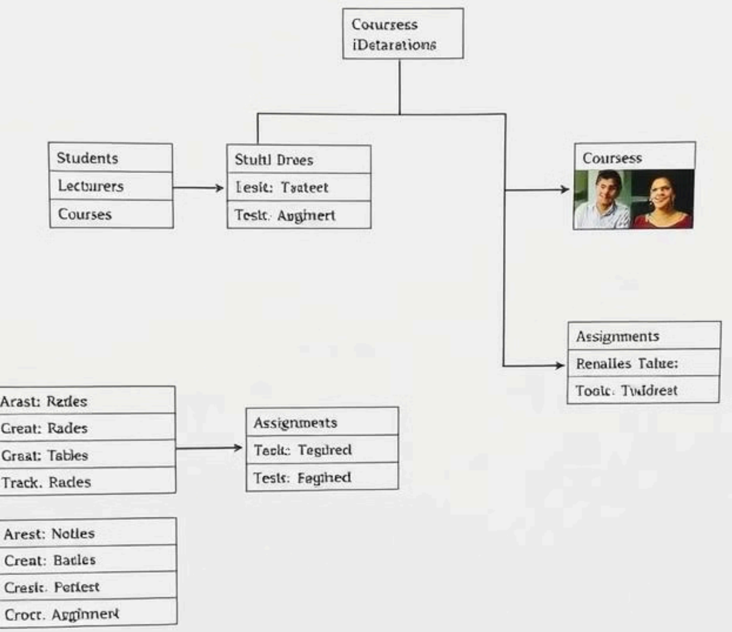
Centralize course management tasks like attendance, assignments, and grading.

Objectives

Automate attendance tracking, simplify grading, and offer real-time performance data.

Significance

Provide insights into student performance to support decision-making.



Phase 3: Logical Model Design

Entities

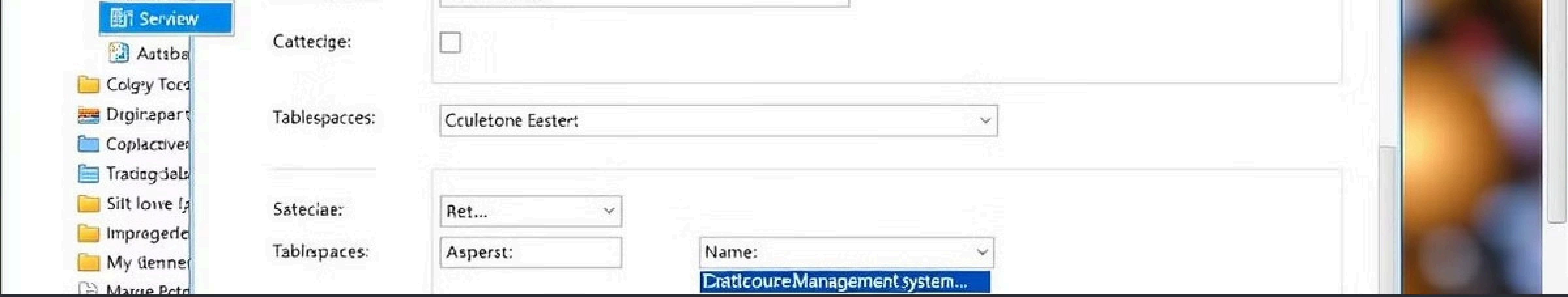
Department, Lecturer, Course, Student, Assignment, Submission, Grade, and Attendance.

Relationships

Defined using primary and foreign keys, connecting students, lecturers, and courses.

Goals

Organize data structure for accurate and accessible data, aiding in tracking progress and workload.



Phase 4: Database Creation

1 Pluggable Database

Created using Oracle with the name TUE_CARDINALS_COURSEMANAGEMENTSYSTEM.

2 Database Creation Code

CREATE PLUGGABLE DATABASE tue_cardinals_CourseManagementSystem ADMIN USER tue_cardinals IDENTIFIED BY cardinals ROLES = (DBA) FILE_NAME_CONVERT = ('C:\app\CIOOL\product\21c\oradata\XE\pdbseed', 'C:\app\CIOOL\product\21c\oradata\XE\tue_cardinals_CourseManagementSystem/');

3 Oracle Enterprise Manager

Used to open the database using the ALTER PLUGGABLE DATABASE command.

4 Tablespace Configuration

Configured the database with tablespaces like SYSAUX, SYSTEM, TEMP, UNDOTBS1 for different data storage needs.

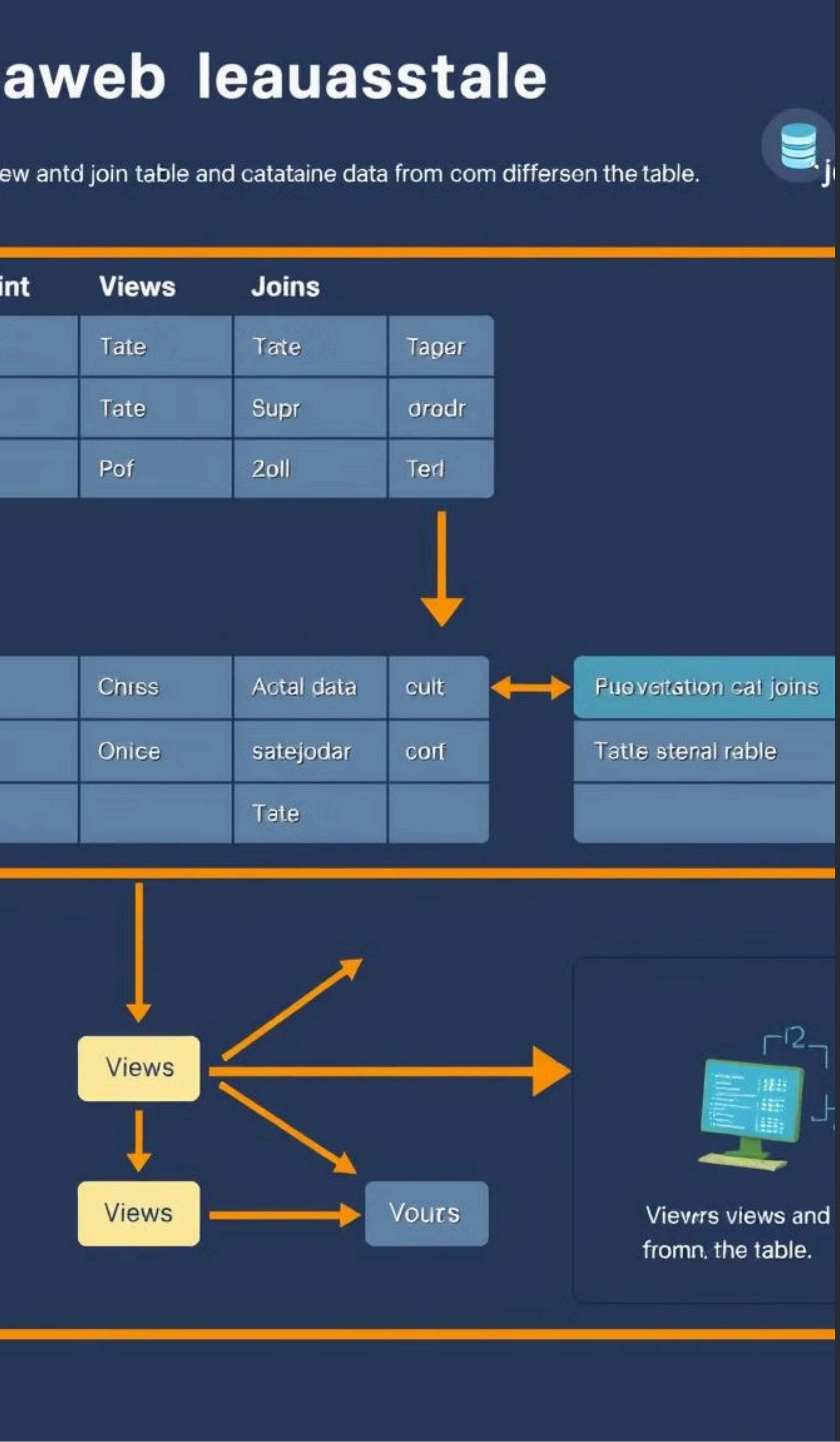
Phase 5: Table Creation & Data Insertion

Department Table

```
CREATE TABLE DEPARTMENT (
    Department_ID INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Contact_Details VARCHAR(255) NOT NULL
);
```

Lecturer Table

```
CREATE TABLE LECTURER (
    Lecturer_ID INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Contact_Details VARCHAR(255) NOT NULL,
    Department_ID INT NOT NULL,
    FOREIGN KEY (Department_ID) REFERENCES
DEPARTMENT(Department_ID)
);
```



Phase 6: Database Interactions & Transactions

1 View Creation

Created views to simplify data retrieval, such as the StudentMarks view to display student information, grades, and assignment details.

2 Transaction Management

Implemented transactions to ensure data integrity and consistency, such as updating attendance and course capacity.

3 Transaction Code

BEGIN TRANSACTION; INSERT INTO ATTENDANCE (Attendance_ID, Course_ID, Student_ID, Status) VALUES (26, 1, 40, 'Present'); UPDATE COURSE SET Seats_Available = Seats_Available - 1 WHERE Course_ID = 1; COMMIT;

4 Database Interactions

Utilized JOINS to combine data from different tables to retrieve comprehensive information about students, courses, and grades.

Phase 7: Advanced Database Programming & Auditing

Triggers

Enforce data integrity and automate workflows, such as the before_attendance_insert trigger to validate attendance status.

Packages

Group related procedures for better organization and reusability, such as the cms_package for logging audits and updating course capacity.

Cursors

Enable efficient row-by-row data processing, such as calculating the average grade for each student using explicit cursors.

Auditing

Improve security and accountability by logging changes to sensitive data.

Made with Gamma