**Group 4 Project Proposal & Draft Object Model**Lazaro Del Pino, Osahon Obayagbona

Elena Ozyman, Jose Ramos, Corey Slayton

ISM6106 RG7 – Dr. Arijit Sengupta

Master of Science in Information Systems

Chapman Graduate School of Business



**Project XXX**

**DRAFT Project NAME UML Object Model:**

Diagram

Description automatically generatedDiagram

Description automatically generated

**Business Problem:**

Team YYY has been chartered to design an application for Project XXX that allows customers to build their own website using a subscription-based platform like wix.

**Value Proposition:**

You can be able to use ready templates and predefined billing system, perhaps some security modules etc.

**Roles:**

Project Title –

Team Name –

Roles –

1. Team Lead –
2. Design Lead –
3. Developers –
4. Testers –

**Requirement Specification:**

**Division of Responsibilities:**

1. Business Problem:
2. Value Proposition:
3. Specification Statement:
4. Class Diagram:
5. State Diagrams:
6. Scenarios:
7. Interactive Diagrams:
8. GitHub Repository:
9. Division of Work:
10. Percentage of Work Done:
11. Final Analysis/Lessons Learned:

**System Development Outline:**

* + 1. System Conception:
       1. Idea for an application (elaborating concept)
          1. Who is it for?
          2. What problems will it solve?
          3. Where will it be used?
          4. When is it needed?
          5. Why is it needed?
          6. How will it work?
       2. Prepare Business Case
          1. Business Problem Statement

Requirements

Design

Implementation

* + - * 1. Value Proposition
    1. Analysis:
       1. Scrutinize requirements by developing models
          1. Describe Objects
          2. Create Classes

Assign Attributes & Operations/Methods

* + - * 1. Describe Link and Association Concepts
        2. Specify:

Multiplicity

Association end names

Ordered set of objects

Bags or sequences

Associated classes

Qualified/Not Qualified associations

Relationships between super/sub classes

Plan inheritance and overriding features

* + - * 1. Develop Class Model
        2. Prepare a Data Dictionary
      1. Consider the available resources of information
      2. Resolve Ambiguities
         1. Domain Analysis – focuses on real-world things whose semantics the application captures
         2. Application Analysis – addresses the computer aspects of the application that are visible to the users
    1. System Design:
       1. Formulate an architecture
          1. Update Class Model

Substantiate

Enumerations

Multiplicity

Scope

Visibility

Association Ends

N-nary Associations

Aggregation

Abstract Classes

Multiple Inheritance

Multiple Classification

Metadata

Reification

Constraints

Objects

Generalization Sets

Links

Derived Data

Packages

* + - 1. Create State Model
         1. Events

Signal Events

Change Events

Time Events

* + - * 1. States
        2. Transitions and Conditions
        3. Guard condition
        4. State Diagrams

Behavior:

Activity Effects

Do-Activities

Entry & Exit Activities

Completion Transition

Sending Signals

* + - * 1. Advanced:

Nested State Diagrams

Nested States

Signal Generalization

Concurrency

Aggregation

Within an object

Synchronization of concurrent activities

* + - 1. Interaction Modeling
         1. Use Case Models

Actors

Use Cases

Use Case Diagrams

* + - * 1. Sequence Models

Scenarios

Sequence Diagrams

* + - * 1. Activity Models

Activities

Branches

Initiation and Termination

Concurrent Activities

Executable Activity Diagrams

Advanced:

Use Case Relationships

Include Relationship

Extend Relationship

Generalization

Combinations of Use Case Relationships

Procedural Sequence Models

Sequence Diagrams with Passive Objects

Sequence Diagrams with Transient Objects

Special Constructs for Activity Models

Sending and Receiving Signals

Swimlanes

Object flows

* + - 1. Choose global strategies and policies to guide the subsequent, more detailed portion of design
    1. Class Design:
       1. Expand and optimize analysis models
    2. Implementation:
       1. Map design elements to programming language and database code
    3. Testing:
       1. Revisit the original business requirements
       2. Verify that the system delivers the proper functionality
       3. Uncover accidental errors (bugs)
       4. Tested on all operating system platforms it is intended to work for
       5. Execute Unit Tests
       6. Execute System Tests
       7. Performance
          1. Create indexes for commonly run queries, joins, and views
       8. Authorization
          1. Ensure all roles have authorization only to appropriate data
       9. Security
          1. Ethical Hacking
          2. Insider threat
          3. Malware prevention
          4. Firewall
          5. Hash values for passwords and sensitive information
       10. Rules
           1. Ensure data is consistent with structural and business rules
       11. Recovery
           1. Database systems must recover from failures and restore the database to a consistent state without loss of data
           2. Cloud or backup plan
       12. Transaction Management
           1. Ensure transactions are processed completely or not at all
           2. Prevent conflicts between concurrent transactions
           3. Ensure transaction results are never lost
    4. Training:
       1. Prepare user documentation
       2. Check the software against the user documentation to ensure the software meets its original goals
       3. Train users so they can fully benefit from the application
    5. Deployment:
       1. Tune the system under various loads
       2. Write scripts and install procedures
    6. Maintenance: