AD 715 Business Simulation Migration
Applied Business Analytics
Report

Sree Kumar Valath Bhuan Das
Jan 14, 2024

TABLE OF CONTENTS:

Introduction **3**
Technology Stack **3**
Implementation Strategy **3**
Development process **4**
System architecture and VM infrastructure **5**
Infrastructure Components: 5
Considerations for VM Deployment: 5
Virtual Machine Deployment: 6
Simulation Workflow Approach **7**
Approach: 7
Advantages of the Approach: 8
Key Features of the Simulation System **8**
Improved Performance: 8
Focus on Collaboration: 8
Compatible Solution: 8
Quicker Decision Making and Smooth Operation: 8
Emphasis on Learning Experience: 9
Identified risks, and potential mitigation methods **9**
Dedicated Personnel for Development and Deployment Support: 9
Evolution of Dependent Environments: 9
Limited Knowledge with Newer Technologies: 10
Creating Protocols and Documentation: 10
Protecting the Intellectual Property (IP): 10
Development calendar **10**
Phase 1: [Jan 15 - Feb 16] 10
Phase 2: [Feb 19 - Mar 31] 10

Introduction

The project aims to implement a web-based simulation using the Django framework as part of the migration from existing Excel tool. The website's frontend will be designed using HTML, CSS, and Bootstrap, providing a user-friendly interface. The backend of the simulation will utilize a package-based Python setup, coupled with a PostgreSQL database for efficient data storage and retrieval.

Technology Stack

- ➔ Framework: Django
- ➔ Frontend Technologies: HTML, CSS, Bootstrap
- ➔ Backend Technologies: Python, JavaScript
- ➔ Database: PostgreSQL or MS SQL Server
- ➔ Output Formats: JSON files, Excel files, and PDF documents.

Implementation Strategy

This diversified output strategy ensures that the project addresses various needs, from technical debugging to analysis and documentation which enhances both the user experience and development process.

The chosen technologies and output formats have been carefully selected to ensure an efficient and user-friendly implementation of the simulation project. For in-browser dynamic computations, JavaScript can be incorporated. The use of Django, along with HTML, CSS, and Bootstrap, ensures a robust frontend, while the backend, powered by Python and PostgreSQL, provides a solid foundation for data management. The release of JSON files for technical purposes and Excel/PDF files for user analysis and documentation further enhances the project's versatility.

Development process

We have identified three interactions that users will have with the simulation. For each of these we will have an interface (view) and database (model) and the interactions in the pages will be handled by functions (controller) defined using Python and JavaScript for analytical and in-browser computations respectively:

- Input Functional Areas:
    - Marketing management
    - Financial management
    - Operations management
    - Innovation management
    - Organizational management

- Output Functional Areas:
    - 36 month financial planner
    - Sim report
    - D-Analysis

- Help and Resources:
    - How to play? : A gamified documentation for the user to run through the website and get accustomed to the simulation. This will promote self-learning and quick onboarding.
    - Tutorials:
        - Course: Theoretical tutorials related to AD 715 required from the simulation perspective.
        - Simulation: Tutorial documentation for learning the working of the simulation.
        - FAQs: QnA format resources for helping students.
        - External Links: Tutorials of other links.
    - Zoom Recordings:
        - Video help and resources.

- Load/Settings Areas:
    - Login
    - Project start
    - Sim Navigation
    - Project Home Page

- External integrations:
    - Power BI: for visualizations

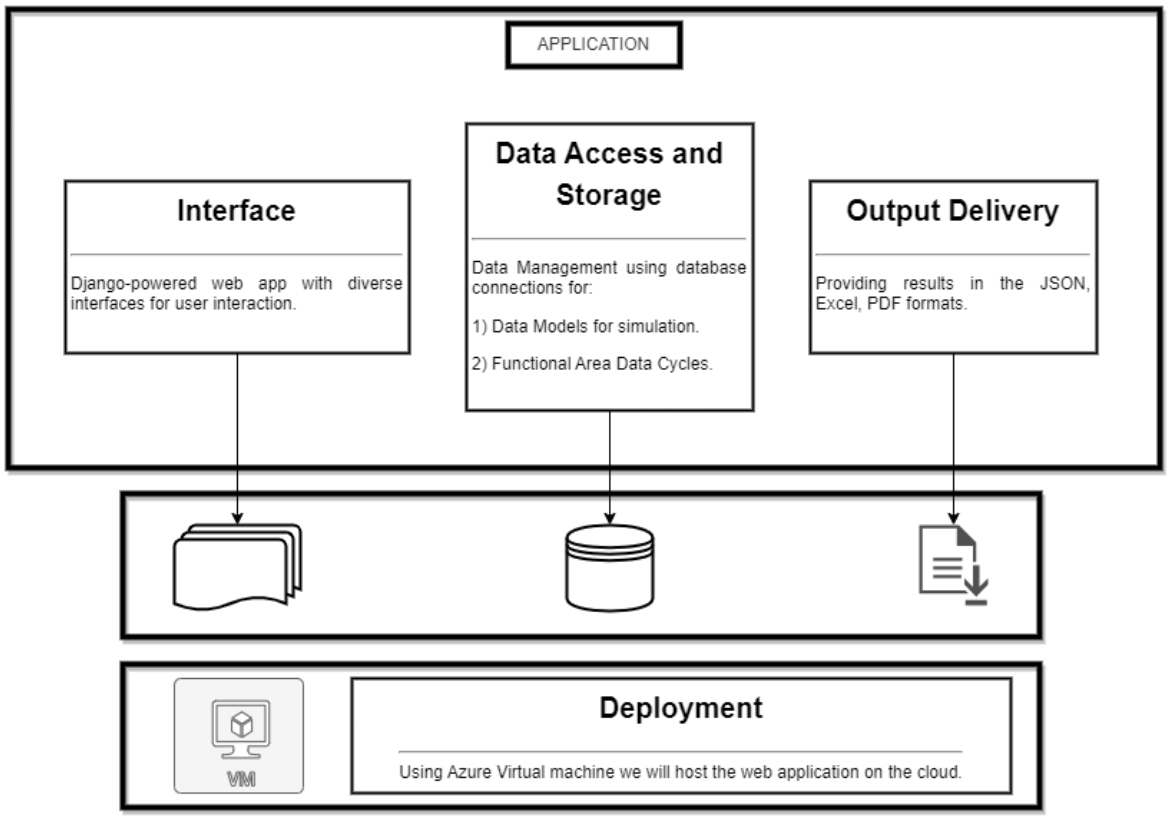
- Bing Chat: generative AI.
- Solver: additional simulations.
- Learning Management system: such as Blackboard, InScribe etc.

System architecture and VM infrastructure

Infrastructure Components:

We have three major components to work on in the application development process: Interface, Data Access, and Output Delivery. For individual versions we shall consider one instance that is locked for a time period and updates can be rolled out whenever required.

Following is a diagram that depicts the proposed architecture.



Considerations for VM Deployment:

- Virtualization Platform: Utilize a reliable virtualization platform based on cloud provider.
- Operating System: Choose an OS that aligns with project requirements.
- Database Server: Set up a PostgreSQL database server for efficient data storage and retrieval.
- Python Environment: Configure the VM with the necessary Python environment for running Django and backend functions.

- Monitoring Tools: Implement monitoring tools for tracking VM performance and resource utilization.

Virtual Machine Deployment:

- Flexibility: Virtual Machines offer flexibility in terms of operating systems and software configurations, allowing seamless integration with project requirements.

- Isolation: VMs provide isolation, ensuring that the simulation environment is independent and does not interfere with other systems.

- Scalability: Virtual Machines can be easily scaled up or down based on resource needs, accommodating future growth and demand.

- Resource Optimization: VMs allow efficient resource utilization by allocating specific resources (CPU, memory, storage) to the simulation.

Simulation Workflow Approach

Approach:

1. User Input:
    a. Users input simulation parameters through designated input functional areas on the interface layer (Django-powered).
    b. Various input fields allow users to configure different aspects of the simulation.

2. Configuration JSON File:
    a. On submission of the input parameters, the system generates a configuration JSON file representing the user's simulation settings.
    b. This JSON file encapsulates the user's configuration (cycle) and serves as a record of the simulation state.

3. Sharing and Collaboration:
    a. Users can download and share the configuration JSON file with team members and course facilitators.
    b. Enables collaboration and discussion based on shared simulation settings.

4. Loading Configuration:
    a. Users can load a previously generated configuration JSON file as an instance, depicting the underlying cycle in the old Excel simulation.
    b. Loading the configuration allows users to revisit and analyze past simulation states.

5. Updating and Iterating:
    a. After loading a configuration, users can make further changes to the input fields based on new parameters or adjustments.
    b. The system updates the simulation based on the modified input, generating updated results.

6. Final Configuration State:
    a. Users can mark a configuration state as "final" when satisfied with the simulation outcomes for a specific functional area.
    b. Marks the completion of a simulation cycle and triggers the generation of document output files.
7. Document Output Files:
    a. For each "final" configuration state, users can generate and download document output files in Excel or PDF formats.
    b. These files provide a comprehensive summary and documentation of the simulation results.

Advantages of the Approach:

- Traceability: Configuration JSON files serve as a traceable record of simulation settings, enhancing transparency and reproducibility.
- Collaboration: Sharing configuration files enables effective collaboration among team members and course facilitators.
- Iterative Analysis: Users can iterate through different simulation states, refining parameters and analyzing outcomes.
- Documentation: Document output files provide a structured summary of final simulation results for reference and reporting.

Key Features of the Simulation System

Improved Performance:

- Optimized Codebase: Emphasis on efficient coding practices and optimization to ensure fast and responsive performance.
- Scalable Architecture: Designed to handle increased user load and data volume, ensuring sustained performance.

Focus on Collaboration:

- Configuration Sharing: Users can easily share configuration JSON files, fostering collaboration among team members and course facilitators.
- Real-time Collaboration: Implement features that enable real-time collaboration, allowing multiple users to work on the simulation simultaneously.

Compatible Solution:

- Cross-Browser Compatibility: Ensures compatibility across various web browsers, providing a seamless experience for users on different platforms.
- Responsive Design: The user interface is designed to be responsive, catering to a range of devices, including desktops, tablets, and smartphones.

Quicker Decision Making and Smooth Operation:

- Fast Iteration: Quick loading and updating of simulation states to facilitate rapid iteration and decision-making.
- Smooth Operation: Minimized downtime through robust error handling and graceful degradation mechanisms for smooth user experience.

Emphasis on Learning Experience:

- User-Friendly Interface: An intuitive and user-friendly interface designed to enhance the learning experience for users.
- Guided User Onboarding: Incorporate guides, tooltips, or tutorials to assist users in understanding the simulation process and functionalities.

Identified risks, and potential mitigation methods

Dedicated Personnel for Development and Deployment Support:

- Risk: Lack of dedicated personnel could lead to inefficiencies in development and deployment.

- Mitigation:
    - Solution: Identify and assign designated roles through a clear "division of workload," segregating responsibilities based on the team members' areas of expertise.
    - Protocol: Establish a structured workflow with defined roles and responsibilities for each team member.

- Division of workload:
    - ➔ Business: Course Builder, Instructors, Teaching Assistants.
    - ➔ Development: B-Sim Developer team consisting students and project supervisor.
    - ➔ Testing: Business + Development + Students of ABA program.
    - ➔ Deployment: Platform support from MET IT for advice on cloud solutions.
    - ➔ Integration: Resource personnel to create connections between the simulation and the following:
        a) Learning Management System.
        b) Third Party tools and solutions.
        c) Analytical Lab platform.

Evolution of Dependent Environments:

- Risk: Changes in third-party environments like Azure may impact the functionality of the simulation.
- Mitigation:
    - Solution: Implement version-based deployment, coupled with a Virtual Machine architecture, to limit changes during deployed editions of the simulation.
    - Protocol: Develop a version control strategy to track changes and ensure compatibility with external dependencies.

Limited Knowledge with Newer Technologies:

- Risk: Limited expertise in emerging technologies like Generative AI, cloud computing solutions, and external tools.
- Mitigation:
    - Solution: Identify and engage mentors or supervisors with expertise in the relevant technologies to contribute to the project.
    - Protocol: Establish regular knowledge-sharing sessions and create documentation to disseminate information within the team.

Creating Protocols and Documentation:

- Risk: Insufficient documentation could lead to confusion and inefficiencies.
- Mitigation:
    - Solution: Prioritize the creation of comprehensive protocols and documentation for development, deployment, and ongoing maintenance.
    - Protocol: Enforce a documentation standard, and ensure that all team members contribute to maintaining up-to-date documentation.

Protecting the Intellectual Property (IP):

- Risk: Inadequate protection of intellectual property could result in unauthorized use or replication.
- Mitigation:
    - Solution: Identify existing protocols at different levels, including application, college (MET IT), university (Central IT), and platform (Microsoft Azure) levels.
    - Protocol: Collaborate with legal and compliance teams to establish and enforce protocols for IP protection.

Development calendar

Phase 1: [Jan 15 - Feb 16]

- Identifying and assigning personnel into teams/groups. New student support members, MET Support for division of workload.

- Estimating migration costs and working with cloud experts (MET IT) for feasibility of migration.

Phase 2: [Feb 19 - Mar 31]

- Continue development of remaining simulation components based on the architecture, with focus on design of remaining functional areas and cycle implementation.

- Deployment ready to be tested by instructors, online facilitators, and teaching assistants; perform integration testing to test the platform and fix bugs.

- Work with ABA program facilitators to build “Help & Resources” for the web application.

- In-class launch March 31, 2024 for all AD715 classes (OL and FF).