**Student’s Individual Study #1**

Full Name: Bekenov Temirlan  
Educational Program: Faculty of IT

**Topic: Project introduction, machines provisioning**

**Project Name: TaskBoard – Simple Task Management System**

**Target**

The main target of this SIS is to define, plan, and prepare the environment for a small-scale web-based project.  
The chosen project, TaskBoard, is a lightweight system for managing and tracking tasks, similar to Trello.  
The purpose is to design the project architecture, provision virtual machines, and determine data and configuration storage locations.

**Tasks**

* Choose and describe the project;
* Create the architecture of the project;
* Prepare machines for the project;
* Define the locations for storing data and configurations.

**1. Project Description**

Project name: TaskBoard  
Main idea: Build a simple task management web application that allows users to create, assign, and track tasks on a visual board divided into “To Do”, “In Progress”, and “Done” sections.

**Key Features:**

* User authentication and registration
* Task creation, editing, and deletion
* Task assignment to users
* Task status tracking

**Legend:**TaskBoard is intended as a simplified internal system for student project teams or small organizations that need a visual way to manage tasks.

**Team:**This project is completed individually. However, the architecture is designed in a scalable way to support multiple developers and environments in the future.

**2. Architecture of the Project**

The TaskBoard system uses a three-layer architecture:

1. Frontend: provides a user interface (React web app).
2. Backend: provides an API for user and task management (Node.js + Express).
3. Database: stores persistent data (PostgreSQL).
4. Proxy server: Nginx handles HTTP requests and static content.

**Architecture Diagram (conceptual):**

[User Browser]

↓

[Nginx Reverse Proxy]

↓

[Backend API - Node.js + Express]

↓

[PostgreSQL Database]

**Components:**

* Frontend server: React build hosted by Nginx
* Backend server: Node.js + Express application
* Database server: PostgreSQL
* Proxy server: Nginx acting as reverse proxy and static server

**3. Machines Preparation**

Environment: VirtualBox Oracle  
Reason: VirtualBox is free, cross-platform, and easy to use for building isolated test environments.

Chosen Method: Create two virtual machines (VMs) manually in VirtualBox.

| Machine | Role | OS | Installed Software |
| --- | --- | --- | --- |
| VM1 | Application Server | Ubuntu Server 22.04 LTS | Node.js, Nginx |
| VM2 | Database Server | Ubuntu Server 22.04 LTS | PostgreSQL |

**Justification of Linux choice:**  
Ubuntu Server 22.04 LTS is stable, widely supported, and compatible with most open-source software.  
It offers long-term security updates and a large package repository, making it ideal for both development and deployment.

Networking:  
Both VMs are connected through an internal VirtualBox network (NAT), ensuring secure and isolated communication.

**4. Data and Configuration Storage**

| Component | Data Storage | Config Files | Temporary Files |
| --- | --- | --- | --- |
| Backend (Node.js) | /var/taskboard/data | /etc/taskboard/config.json | /tmp/taskboard |
| PostgreSQL Database | /var/lib/postgresql/data | /etc/postgresql/postgresql.conf | /tmp/pg\_temp |
| Nginx (Proxy + Frontend) | /var/www/taskboard | /etc/nginx/sites-enabled/taskboard.conf | /var/cache/nginx |

All configuration files are stored in JSON or .conf formats, depending on the software.  
Temporary directories are used for caching and automation scripts (for setup or deployment).

**5. Step-by-Step Report Summary**

1. Selected an existing concept (task management app) suitable for an individual project.
2. Designed a logical system architecture with clearly separated components.
3. Chose VirtualBox for virtual environment creation and Ubuntu Server as OS.
4. Planned two virtual machines with specified roles and configurations.
5. Defined data and configuration storage structure.

**Conclusion**

During this SIS, a project concept was selected, architecture was designed, and a deployment environment was prepared using VirtualBox.  
The project demonstrates understanding of machine provisioning, system design, and environment configuration.  
The chosen technologies are stable and realistic for academic or small production use.