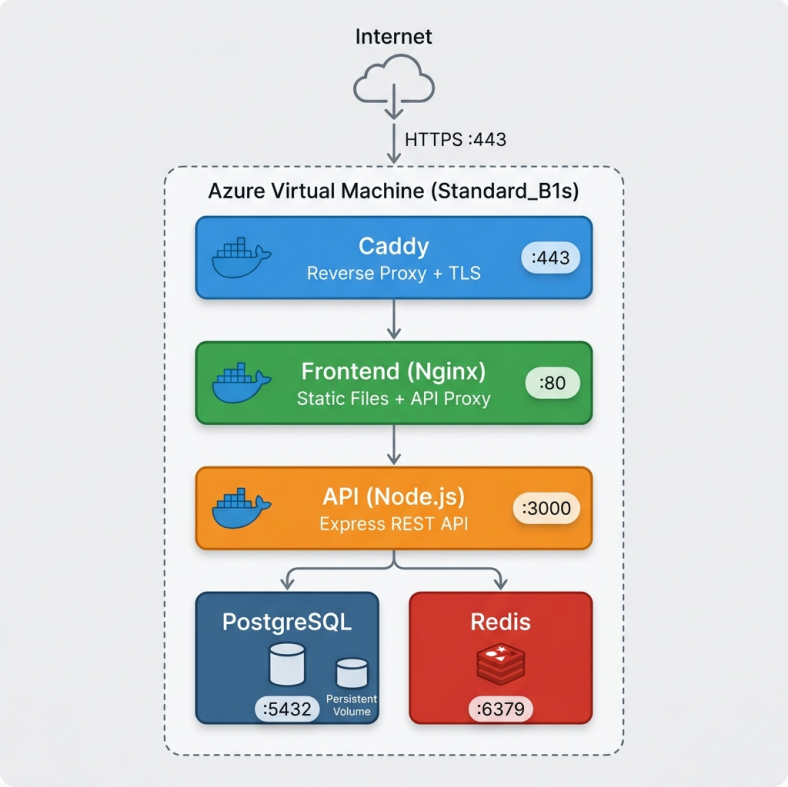
Task Manager

Cloud Deployment Documentation

Course:	Cloud Technologies / Web Application Deployment
Student:	Gopi Suvanam
Login:	gs699he
Date:	May 19, 2026
Cloud:	Microsoft Azure (Azure for Students)
Repository:	https://git.kemt.fei.tuke.sk/gs699he/zkt25



Technical University of Košice – KEMT FEI

Contents

Application Overview

The **Task Manager** is a full-stack web application for creating, reading, updating, and deleting (CRUD) tasks. It is deployed to **Microsoft Azure** using Docker Compose on an Azure Virtual Machine.

Key Features

- RESTful API with full CRUD operations
- Apple-inspired premium UI with glassmorphism
- Redis caching (30s TTL) for API optimization
- PostgreSQL database with persistent volume and backup scripts
- Automatic HTTPS via Caddy + Let's Encrypt
- Fully automated deployment via shell scripts
- Auto-restart on container failure

Architecture

The application uses a **5-container microservices architecture** on an Azure VM, orchestrated by Docker Compose.

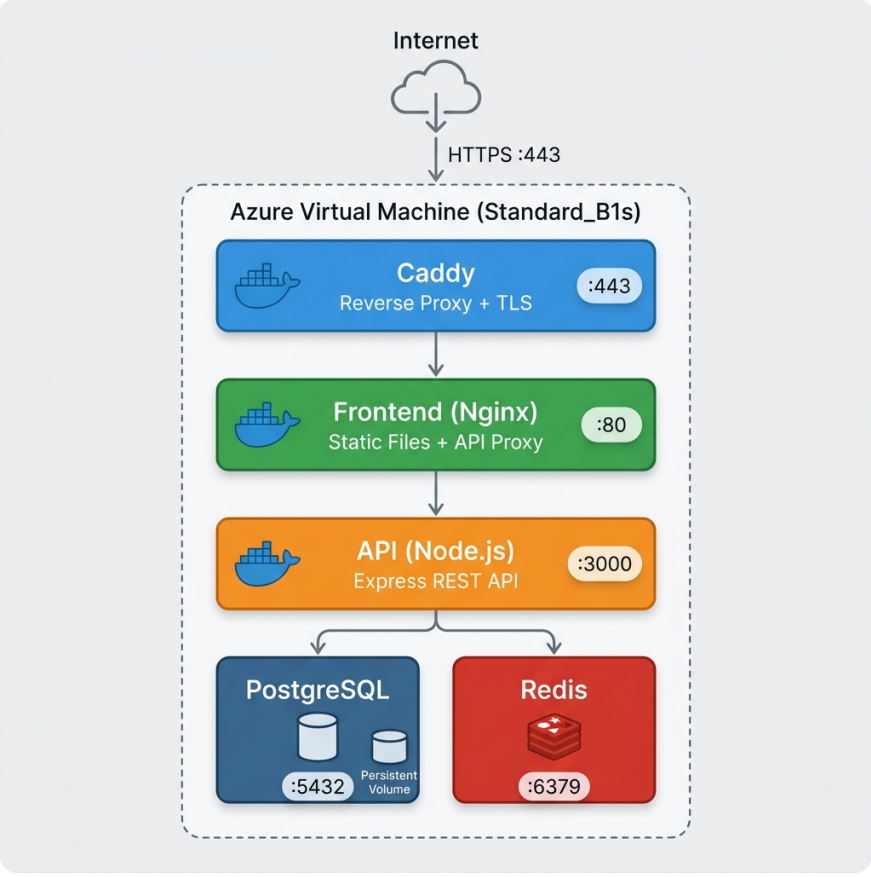


Figure 1: Architecture – 5 Docker containers on Azure VM

Container Descriptions

Container	Image	Purpose
Caddy	caddy:2-alpine	Reverse proxy with automatic HTTPS/TLS via Let's Encrypt. Ports 80, 443.
Frontend	nginx:alpine	Serves static HTML/CSS/JS, proxies `/api/` to backend.
API	node:20-alpine	Express.js REST API with CRUD endpoints.
PostgreSQL	postgres:16-alpine	Persistent database with Docker named volume.
Redis	redis:7-alpine	In-memory cache with AOF persistence.

Network Isolation

- `frontend-net` – Caddy, Frontend, API
- `backend-net` – API, PostgreSQL, Redis

Database and cache are **not accessible** from the internet. Only the API bridges both networks.

Azure Services Used

Service	Usage
Azure VM (Standard_B1s)	Ubuntu 24.04, 1 vCPU, 1 GB RAM. Runs Docker.
Public IP + DNS	`*.westeurope.cloudapp.azure.com` for HTTPS.
Azure for Students	$100 free credit, no credit card required.
Let's Encrypt	Free TLS certificates via Caddy (auto-renewal).
cloud-init	Installs Docker on first VM boot automatically.

File Descriptions

File	Description
`docker-compose.yaml`	5 services, 2 networks, persistent volumes.
`prepare-app.sh`	Creates Azure VM, installs Docker, deploys app.
`remove-app.sh`	Deletes Azure resource group (all resources).

File	Description
`backup-db.sh`	SSH + pg_dump for database backups.
`view-logs.sh`	Retrieves container logs via SSH.
`cloud-init.yaml`	Auto-installs Docker on VM first boot.
`env.example`	Template for secrets (copied to .env).
`caddy/Caddyfile`	HTTPS reverse proxy configuration.
`frontend/`	Nginx Dockerfile, config, static files.
`api/`	Node.js Dockerfile, Express server, DB pool.
`db/init.sql`	Database schema and sample data.
`docs/`	This documentation + architecture diagram.

Configuration

All settings via environment variables in `.env` (gitignored).

Variable	Description	Default
`POSTGRES_PASSWORD`	Database password	*(required)*
`AZURE_RESOURCE_GROUP`	Resource group name	taskmanager-rg
`AZURE_LOCATION`	Azure region	westeurope
`AZURE_VM_SIZE`	VM size	Standard_B1s
`AZURE_DNS_LABEL`	DNS subdomain	taskmanager-gs699he

Security

- `.env` in `.gitignore` – secrets never in Git
- Databases on isolated Docker network
- HTTPS enforced by Caddy (HTTP → HTTPS redirect)
- Security headers in Nginx
- SSH key authentication (no password login)

Deployment Instructions

Prerequisites

1. Azure CLI: `curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash`
2. Azure for Students subscription activated
3. `az login` completed

Deploy

```
# Clone and configure
git clone git@git.kemt.fei.tuke.sk:gs699he/zkt25.git
cd zkt25/sk1
cp env.example .env
nano .env  # Set POSTGRES_PASSWORD

# Deploy to Azure
./prepare-app.sh
# Creates: Resource Group -> VM -> Docker -> Containers

# Access:
# https://taskmanager-gs699he.westeurope.cloudapp.azure.com
```

Remove (after exam)

```
./remove-app.sh
# Deletes entire resource group (VM, IP, disk, NSG)
```

Data Persistence and Backups

Volumes

- `taskapp-pgdata` – PostgreSQL data
- `taskapp-redisdata` – Redis AOF
- `taskapp-caddy-data` – TLS certificates

Backup

```
./backup-db.sh
# Output: backups/taskmanager_backup_20260519_020000.sql
```

Auto-Restart

All containers: `restart: always`. Docker starts on boot via systemd.

Access Logs

```
./view-logs.sh              # All logs
./view-logs.sh --caddy      # HTTPS proxy logs
./view-logs.sh --api -f     # Follow API logs real-time
```

Log sources:

- **Caddy**: JSON – client IP, status, TLS version

- **API**: `METHOD /path STATUS duration`
- **Nginx**: Combined log format

Cost Analysis

Azure for Students – Monthly Costs

Resource	Configuration	Monthly
Azure VM	Standard_B1s (1 vCPU, 1 GB)	$7.59
Public IP	Static IPv4	$3.00
OS Disk	30 GB Premium SSD	$1.20
TLS Certificate	Let's Encrypt (free)	$0.00
Total		**$11.79**
Annual		**$141.48**

Azure for Students provides $100 free credit – enough for approximately 8 months of operation.

Comparison

Provider	Plan	Monthly
Azure (Students)	B1s + $100 credit	$11.79
AWS	t3.micro (free 12mo)	$8.50
DigitalOcean	Basic Droplet 2 GB	$12.00
Hostinger	KVM 1 (4 GB)	$5.99

AI Usage Declaration

- **Google Antigravity (Gemini)**: Generated scripts, Docker Compose, Nginx/-Caddy config, and this documentation.
- **Architecture diagram**: AI-generated from technical specification.

All content reviewed, tested, and adapted. Student understands all components.

Exam Defense Preparation

Q: How does HTTPS work?
A: Caddy uses the ACME protocol to automatically get a free Let's Encrypt TLS certificate for the Azure DNS hostname. It renews every 90 days.

Q: What if a container crashes?
A: `restart: always` in docker-compose.yaml. Docker auto-restarts failed containers. Docker starts on boot via systemd.

Q: How are secrets managed?
A: `.env` file (gitignored). Docker Compose injects vars into containers. `env.example` has template without values.

Q: How do you back up the database?
A: `backup-db.sh` runs `pg_dump` via SSH inside the PostgreSQL container. SQL dump saved locally.

Q: Can deployment be reproduced?
A: Yes. `prepare-app.sh` creates all Azure resources from scratch via CLI. `remove-app.sh` deletes everything. No web UI needed.

Q: Why 5 containers?
A: Separation of concerns. Caddy handles HTTPS, Nginx serves static files, Node.js handles API logic, PostgreSQL stores data, Redis caches queries. Each can be scaled independently.

Q: How does caching work?
A: Redis caches `GET /api/tasks` for 30 seconds. Write operations invalidate the cache.