Azure 3-Tier Architecture

Status | Not Started

Document Version: 6.0

Date: January 2026

Audience: All Technical Guys

Executive Summary: The Restaurant Analogy

Think of 3-Tier Architecture as a Fine Dining Restaurant:

Tier	Restaurant Part	Purpose	Access Rules
Web Tier	Dining Area & Host	Customer interface	Public access
App Tier	Kitchen	Food preparation	Staff only
DB Tier	Pantry & Recipe Vault	Ingredients & recipes	Chefs only

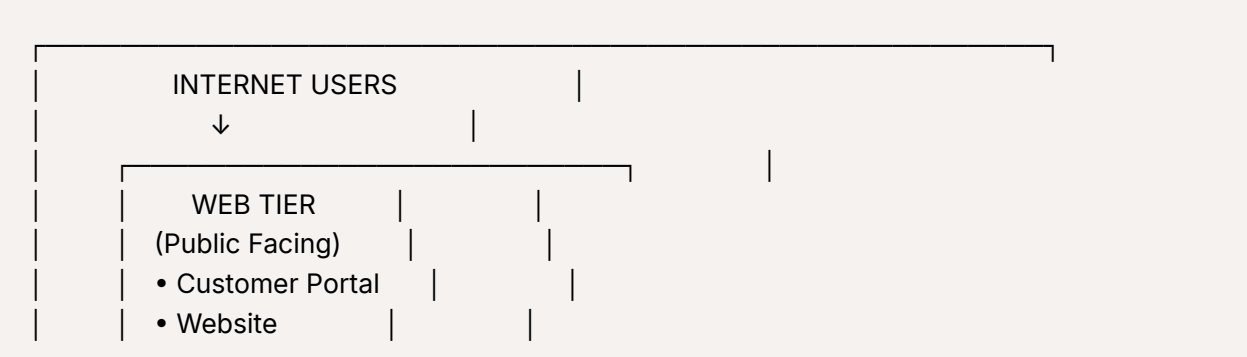
Simple Security Rule: Customers (Internet) can only sit in the dining area. They can't walk into the kitchen or pantry!

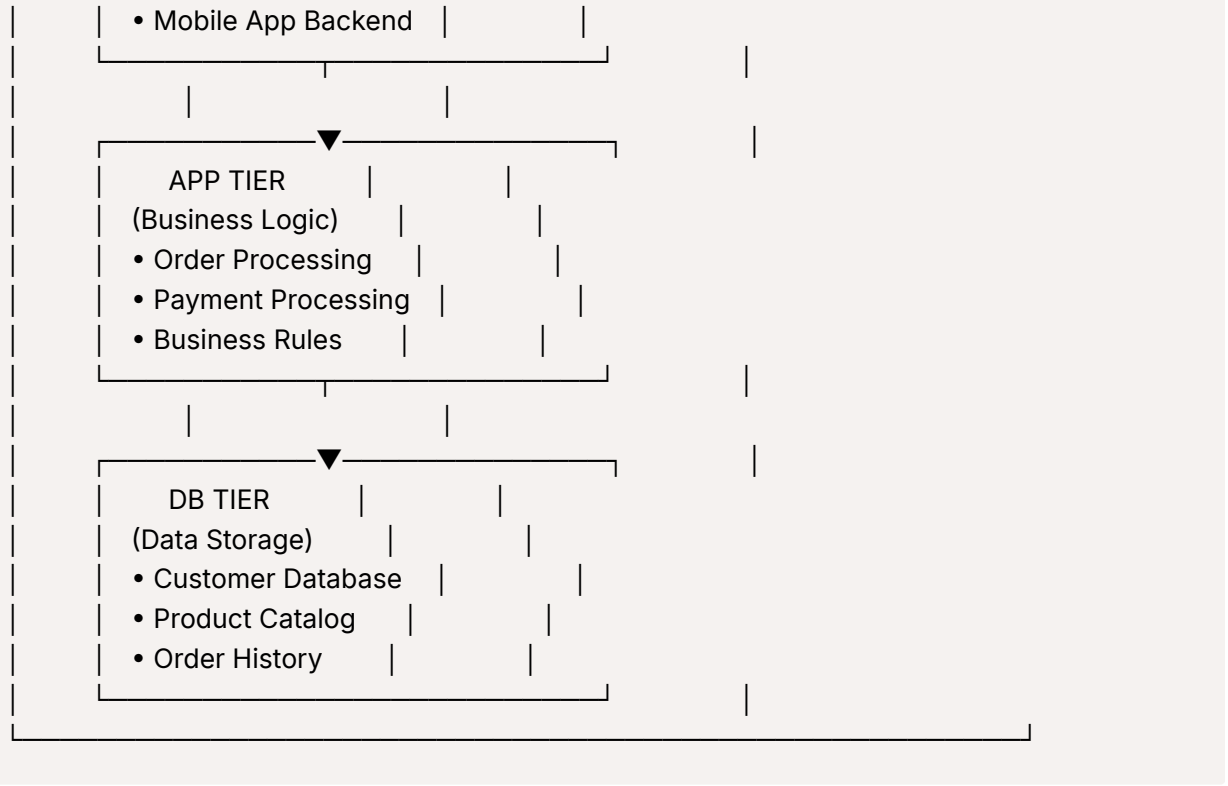
1. Understanding 3-Tier Architecture

1.1 What is 3-Tier Architecture?

Technical Definition: A layered architecture that separates an application into three logical and physical tiers, each with specific responsibilities and security boundaries.

Visual Representation:





1.2 Why Use 3-Tier Architecture?

Benefit	Explanation	Real-world Example
Security	Each tier has its own security rules	Bank: Lobby → Teller → Vault
Scalability	Scale each tier independently	Restaurant: More tables doesn't need more kitchens
Maintainability	Update one tier without affecting others	Car: Fix engine without touching interior
Resilience	Failure in one tier doesn't bring down entire system	Airplane: Entertainment system failure doesn't affect navigation

2. Step-by-Step Implementation with Azure CLI

2.1 Prerequisites Setup

```
#!/bin/bash
# File: 00-prerequisites.sh
# Description: Setup Azure environment for 3-tier architecture

echo "=== STEP 0: PREREQUISITES SETUP ==="

# Login to Azure
az login
```

```
# Set variables
SUBSCRIPTION_ID=$(az account show --query id -o tsv)
TENANT_ID=$(az account show --query tenantId -o tsv)
LOCATION="eastus"
RESOURCE_GROUP="3tier-app-rg"
PROJECT_NAME="myapp"
ENVIRONMENT="prod"

echo "Subscription: $SUBSCRIPTION_ID"
echo "Tenant: $TENANT_ID"
echo "Location: $LOCATION"

# Create Resource Group
echo "Creating Resource Group..."
az group create \
  --name $RESOURCE_GROUP \
  --location $LOCATION \
  --tags "project=$PROJECT_NAME" "environment=$ENVIRONMENT"

echo "✅ Prerequisites setup complete!"
```

2.2 Network Infrastructure Setup

```
#!/bin/bash
# File: 01-network-setup.sh
# Description: Create VNet and subnets for 3-tier architecture

echo "=== STEP 1: NETWORK INFRASTRUCTURE ==="

# Variables
VNET_NAME="vnet-$PROJECT_NAME"
VNET_ADDRESS="10.0.0.0/16"

# Create Virtual Network
echo "Creating Virtual Network..."
az network vnet create \
  --name $VNET_NAME \
  --resource-group $RESOURCE_GROUP \
  --address-prefixes $VNET_ADDRESS \
  --location $LOCATION
```

```
# Create Subnets
echo "Creating Web Tier Subnet..."
az network vnet subnet create \
  --name "snet-web" \
  --resource-group $RESOURCE_GROUP \
  --vnet-name $VNET_NAME \
  --address-prefixes "10.0.1.0/24"

echo "Creating App Tier Subnet..."
az network vnet subnet create \
  --name "snet-app" \
  --resource-group $RESOURCE_GROUP \
  --vnet-name $VNET_NAME \
  --address-prefixes "10.0.2.0/24"

echo "Creating Database Tier Subnet..."
az network vnet subnet create \
  --name "snet-db" \
  --resource-group $RESOURCE_GROUP \
  --vnet-name $VNET_NAME \
  --address-prefixes "10.0.3.0/24"

echo "Creating AzureBastionSubnet..."
az network vnet subnet create \
  --name "AzureBastionSubnet" \
  --resource-group $RESOURCE_GROUP \
  --vnet-name $VNET_NAME \
  --address-prefixes "10.0.100.0/26"

# Create Public IP for Load Balancer
echo "Creating Public IP for Load Balancer..."
az network public-ip create \
  --name "pip-web-lb" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --sku Standard \
  --allocation-method Static \
  --dns-name "$PROJECT_NAME-web"

echo "✅ Network infrastructure created successfully!"
```

3. Tier 1: Web Tier Implementation

3.1 Web Tier Components

What belongs in Web Tier:

- Web servers (IIS, Apache, Nginx)
- Load Balancers
- Web Application Firewall (WAF)
- Content Delivery Network (CDN)

3.2 Create Load Balancer for Web Tier

```
#!/bin/bash
# File: 02-web-tier.sh
# Description: Deploy Web Tier components

echo "=== STEP 2: WEB TIER DEPLOYMENT ==="

# Create Load Balancer
echo "Creating Load Balancer..."
az network lb create \
  --name "lb-web" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --sku Standard \
  --public-ip-address "pip-web-lb" \
  --frontend-ip-name "frontend-web" \
  --backend-pool-name "backend-web"

# Create Health Probe
echo "Creating Health Probe..."
az network lb probe create \
  --resource-group $RESOURCE_GROUP \
  --lb-name "lb-web" \
  --name "probe-http" \
  --protocol tcp \
  --port 80 \
  --interval 5 \
  --threshold 2

# Create Load Balancing Rule
echo "Creating Load Balancing Rule..."
az network lb rule create \
  --resource-group $RESOURCE_GROUP \
```

```
  --lb-name "lb-web" \
  --name "rule-http" \
  --protocol tcp \
  --frontend-port 80 \
  --backend-port 80 \
  --frontend-ip-name "frontend-web" \
  --backend-pool-name "backend-web" \
  --probe-name "probe-http"

# Create NAT Rules for SSH/RDP (for management)
echo "Creating NAT Rules for Management..."
for i in {0..1}; do
  az network lb inbound-nat-rule create \
    --resource-group $RESOURCE_GROUP \
    --lb-name "lb-web" \
    --name "nat-ssh-$i" \
    --protocol tcp \
    --frontend-port $((50000 + $i)) \
    --backend-port 22 \
    --frontend-ip-name "frontend-web"
done

echo "✅ Load Balancer configuration complete!"
```

3.3 Deploy Web Tier Virtual Machines

```
# Create Availability Set for Web VMs
echo "Creating Availability Set for Web VMs..."
az vm availability-set create \
  --name "as-web" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --platform-fault-domain-count 2 \
  --platform-update-domain-count 5

# Create Network Security Group for Web Tier
echo "Creating NSG for Web Tier..."
az network nsg create \
  --name "nsg-web" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION
```

```
# Add NSG Rules for Web Tier
echo "Adding NSG Rules..."
# Allow HTTP from Internet
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-web" \
  --name "AllowHTTP" \
  --priority 100 \
  --access Allow \
  --protocol Tcp \
  --direction Inbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges 80 443

# Allow SSH from specific IP (your IP)
MY_IP=$(curl -s ifconfig.me)
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-web" \
  --name "AllowSSH" \
  --priority 110 \
  --access Allow \
  --protocol Tcp \
  --direction Inbound \
  --source-address-prefixes "$MY_IP/32" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges 22

# Deny all other inbound
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-web" \
  --name "DenyAllInbound" \
  --priority 4096 \
  --access Deny \
  --protocol "*" \
  --direction Inbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
```

```
  --destination-port-ranges "*"

# Associate NSG with Web Subnet
echo "Associating NSG with Web Subnet..."
az network vnet subnet update \
  --resource-group $RESOURCE_GROUP \
  --vnet-name $VNET_NAME \
  --name "snet-web" \
  --network-security-group "nsg-web"

# Deploy Web Tier VMs
echo "Deploying Web Tier Virtual Machines..."
for i in {1..2}; do
  echo "Creating Web VM $i..."

  # Create NIC
  az network nic create \
    --resource-group $RESOURCE_GROUP \
    --name "nic-web-vm$i" \
    --location $LOCATION \
    --subnet "snet-web" \
    --vnet-name $VNET_NAME \
    --network-security-group "nsg-web" \
    --lb-name "lb-web" \
    --lb-address-pools "backend-web" \
    --lb-inbound-nat-rules "nat-ssh-$(($i-1))"

  # Create VM
  az vm create \
    --resource-group $RESOURCE_GROUP \
    --name "web-vm$i" \
    --location $LOCATION \
    --nics "nic-web-vm$i" \
    --image "Ubuntu2204" \
    --size "Standard_B2s" \
    --admin-username "azureuser" \
    --admin-password "Azure123456!!" \
    --availability-set "as-web" \
    --public-ip-address "" \
    --custom-data cloud-init-web.txt

  echo "✅ Web VM $i created successfully!"
done
```

```
echo "✅ Web Tier deployment complete!"
```

3.4 Web Tier Configuration Script

Create `cloud-init-web.txt`:

yaml

```
#cloud-config
package_update: true
package_upgrade: true

packages:
  - nginx
  - curl
  - net-tools

write_files:
  - path: /var/www/html/index.html
    owner: www-data:www-data
    permissions: '0644'
    content: |
      <!DOCTYPE html>
      <html>
      <head>
          <title>Web Tier - 3 Tier Application</title>
          <style>
              body { font-family: Arial, sans-serif; margin: 40px; }
              .container { max-width: 800px; margin: 0 auto; }
              .tier { padding: 20px; margin: 20px 0; border-radius: 5px; }
              .web-tier { background-color: #e3f2fd; border-left: 5px solid #2196f3; }
              .hostname { font-weight: bold; color: #1976d2; }
          </style>
      </head>
      <body>
          <div class="container">
              <h1>Welcome to 3-Tier Application</h1>
              <div class="tier web-tier">
                  <h2>Web Tier - Frontend Service</h2>
                  <p><span class="hostname">Hostname:</span> HOSTNAME_PLACEHOLDER</p>
                  <p><span class="hostname">IP Address:</span> IP_PLACEHOLDER</p>
                  <p>This is the public-facing web tier serving static content.</p>
```

```
            <p>Traffic from Internet → This tier only</p>
          </div>
          <div style="margin: 30px; text-align: center;">↓</div>
          <div style="background-color: #f3e5f5; padding: 10px; border-radius: 5px;">
            <p><em>App Tier (Business Logic) is isolated and secure</em></p>
          </div>
        </div>
      </body>
      </html>

runcmd:
  - systemctl daemon-reload
  - systemctl enable nginx
  - systemctl restart nginx
  - sed -i "s/HOSTNAME_PLACEHOLDER/$(hostname)/" /var/www/html/index.html
  - sed -i "s/IP_PLACEHOLDER/$(hostname -I | awk '{print $1}')/" /var/www/html/index.html
  - echo "Web Tier configured successfully on $(hostname)" > /tmp/web-tier-ready.txt
```

4. Tier 2: App Tier Implementation

4.1 App Tier Components

What belongs in App Tier:

- Application servers (Tomcat, JBoss, Node.js)
- API gateways
- Business logic processing
- Authentication services

4.2 Deploy App Tier Infrastructure

```
#!/bin/bash
# File: 03-app-tier.sh
# Description: Deploy App Tier components

echo "=== STEP 3: APP TIER DEPLOYMENT ==="

# Create Internal Load Balancer for App Tier
echo "Creating Internal Load Balancer for App Tier..."
az network lb create \
  --name "lb-app" \
  --resource-group $RESOURCE_GROUP \
```

```
  --location $LOCATION \
  --sku Standard \
  --vnet-name $VNET_NAME \
  --subnet "snet-app" \
  --frontend-ip-name "frontend-app" \
  --private-ip-address "10.0.2.100" \
  --backend-pool-name "backend-app"

# Create Health Probe for App Tier
echo "Creating Health Probe for App Tier..."
az network lb probe create \
  --resource-group $RESOURCE_GROUP \
  --lb-name "lb-app" \
  --name "probe-app-8080" \
  --protocol tcp \
  --port 8080 \
  --interval 5 \
  --threshold 2

# Create Load Balancing Rule for App Tier
echo "Creating Load Balancing Rule for App Tier..."
az network lb rule create \
  --resource-group $RESOURCE_GROUP \
  --lb-name "lb-app" \
  --name "rule-app-8080" \
  --protocol tcp \
  --frontend-port 8080 \
  --backend-port 8080 \
  --frontend-ip-name "frontend-app" \
  --backend-pool-name "backend-app" \
  --probe-name "probe-app-8080"

# Create Network Security Group for App Tier
echo "Creating NSG for App Tier..."
az network nsg create \
  --name "nsg-app" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION

# Add NSG Rules for App Tier
echo "Adding NSG Rules for App Tier..."

# Allow from Web Tier only (Port 8080)
```

```
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-app" \
  --name "AllowFromWebTier" \
  --priority 100 \
  --access Allow \
  --protocol Tcp \
  --direction Inbound \
  --source-address-prefixes "10.0.1.0/24" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges 8080

# Allow SSH from Bastion Subnet
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-app" \
  --name "AllowSSHFromBastion" \
  --priority 110 \
  --access Allow \
  --protocol Tcp \
  --direction Inbound \
  --source-address-prefixes "10.0.100.0/26" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges 22

# Deny all other inbound (STRICT)
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-app" \
  --name "DenyAllInbound" \
  --priority 4096 \
  --access Deny \
  --protocol "*" \
  --direction Inbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges "*"

# Allow outbound to DB Tier only
az network nsg rule create \
```

```
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-app" \
  --name "AllowOutboundToDB" \
  --priority 100 \
  --access Allow \
  --protocol Tcp \
  --direction Outbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
  --destination-address-prefixes "10.0.3.0/24" \
  --destination-port-ranges 5432 3306

# Allow outbound to Internet for updates
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-app" \
  --name "AllowOutboundToInternet" \
  --priority 110 \
  --access Allow \
  --protocol Tcp \
  --direction Outbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges 443

# Deny all other outbound
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-app" \
  --name "DenyAllOutbound" \
  --priority 4096 \
  --access Deny \
  --protocol "*" \
  --direction Outbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges "*"

# Associate NSG with App Subnet
echo "Associating NSG with App Subnet..."
az network vnet subnet update \
```

```
  --resource-group $RESOURCE_GROUP \
  --vnet-name $VNET_NAME \
  --name "snet-app" \
  --network-security-group "nsg-app"

echo "✅ App Tier NSG configured successfully!"
```

4.3 Deploy App Tier Virtual Machines

```
# Create Availability Set for App VMs
echo "Creating Availability Set for App VMs..."
az vm availability-set create \
  --name "as-app" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --platform-fault-domain-count 2 \
  --platform-update-domain-count 5

# Deploy App Tier VMs
echo "Deploying App Tier Virtual Machines..."
for i in {1..2}; do
  echo "Creating App VM $i..."

  # Create NIC for App VM
  az network nic create \
    --resource-group $RESOURCE_GROUP \
    --name "nic-app-vm$i" \
    --location $LOCATION \
    --subnet "snet-app" \
    --vnet-name $VNET_NAME \
    --network-security-group "nsg-app" \
    --lb-name "lb-app" \
    --lb-address-pools "backend-app"

  # Create App VM
  az vm create \
    --resource-group $RESOURCE_GROUP \
    --name "app-vm$i" \
    --location $LOCATION \
    --nics "nic-app-vm$i" \
    --image "Ubuntu2204" \
    --size "Standard_B2s" \
```

```
    --admin-username "azureuser" \
    --admin-password "Azure123456!!" \
    --availability-set "as-app" \
    --public-ip-address "" \
    --custom-data cloud-init-app.txt

  echo "✅ App VM $i created successfully!"
done

echo "✅ App Tier deployment complete!"
```

4.4 App Tier Configuration Script

Create `cloud-init-app.txt`:

yaml

```
#cloud-config
package_update: true
package_upgrade: true

packages:
  - openjdk-11-jdk
  - tomcat9
  - tomcat9-admin
  - net-tools
  - postgresql-client

write_files:
  - path: /var/lib/tomcat9/webapps/ROOT/index.jsp
    owner: tomcat:tomcat
    permissions: '0644'
    content: |
      <%@ page contentType="text/html;charset=UTF-8" language="java" %>
      <!DOCTYPE html>
      <html>
      <head>
          <title>App Tier - 3 Tier Application</title>
      </head>
      <body>
          <h1>Application Tier Service</h1>
```

```
        <p><b>Status:</b> Running</p>
        <p><b>Hostname:</b> HOSTNAME_PLACEHOLDER</p>
        <p><b>IP Address:</b> IP_PLACEHOLDER</p>
    </body>
    </html>

runcmd:
  - systemctl daemon-reload
  - systemctl enable tomcat9
  - systemctl restart tomcat9
  - ufw allow 8080
  - sed -i "s/HOSTNAME_PLACEHOLDER/$(hostname)/" /var/lib/tomcat9/webapps/ROOT/inde
x.jsp
  - sed -i "s/IP_PLACEHOLDER/$(hostname -I | awk '{print $1}')/" /var/lib/tomcat9/webapps/RO
OT/index.jsp
  - echo "App Tier configured successfully on $(hostname)" > /tmp/app-tier-ready.txt
```

5. Tier 3: Database Tier Implementation

5.1 Database Tier Components

What belongs in DB Tier:

- Database servers (Azure SQL, PostgreSQL, MySQL)
- Database replication
- Backup servers
- Data encryption services

5.2 Deploy Azure Database for PostgreSQL

```
#!/bin/bash
# File: 04-database-tier.sh
# Description: Deploy Database Tier components

echo "=== STEP 4: DATABASE TIER DEPLOYMENT ==="

# Create Network Security Group for DB Tier
echo "Creating NSG for Database Tier..."
az network nsg create \
  --name "nsg-db" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION
```

```
# Add NSG Rules for DB Tier (MOST RESTRICTIVE)
echo "Adding NSG Rules for Database Tier..."

# Allow ONLY from App Tier (Port 5432 - PostgreSQL)
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-db" \
  --name "AllowFromAppTier" \
  --priority 100 \
  --access Allow \
  --protocol Tcp \
  --direction Inbound \
  --source-address-prefixes "10.0.2.0/24" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges 5432

# Allow SSH from Bastion Subnet ONLY
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-db" \
  --name "AllowSSHFromBastion" \
  --priority 110 \
  --access Allow \
  --protocol Tcp \
  --direction Inbound \
  --source-address-prefixes "10.0.100.0/26" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges 22

# DENY ALL OTHER INBOUND (STRICTEST RULE)
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-db" \
  --name "DenyAllInbound" \
  --priority 4096 \
  --access Deny \
  --protocol "*" \
  --direction Inbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
```

```
  --destination-address-prefixes "*" \
  --destination-port-ranges "*"

# Allow outbound for updates
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-db" \
  --name "AllowOutboundUpdates" \
  --priority 100 \
  --access Allow \
  --protocol Tcp \
  --direction Outbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges 443

# DENY ALL OTHER OUTBOUND
az network nsg rule create \
  --resource-group $RESOURCE_GROUP \
  --nsg-name "nsg-db" \
  --name "DenyAllOutbound" \
  --priority 4096 \
  --access Deny \
  --protocol "*" \
  --direction Outbound \
  --source-address-prefixes "*" \
  --source-port-ranges "*" \
  --destination-address-prefixes "*" \
  --destination-port-ranges "*"

# Associate NSG with DB Subnet
echo "Associating NSG with DB Subnet..."
az network vnet subnet update \
  --resource-group $RESOURCE_GROUP \
  --vnet-name $VNET_NAME \
  --name "snet-db" \
  --network-security-group "nsg-db"

echo "✅ Database Tier NSG configured successfully!"
```

5.3 Option A: Deploy Azure Database for PostgreSQL (PaaS)

```
# Create PostgreSQL Server
echo "Creating Azure Database for PostgreSQL..."
az postgres server create \
  --name "psql-$PROJECT_NAME" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --admin-user "dbadmin" \
  --admin-password "Azure123456!!" \
  --sku-name "GP_Gen5_2" \
  --version "11" \
  --storage-size "5120" \
  --backup-retention "7"

# Configure VNet integration
echo "Configuring VNet Service Endpoint..."
az network vnet subnet update \
  --resource-group $RESOURCE_GROUP \
  --vnet-name $VNET_NAME \
  --name "snet-db" \
  --service-endpoints "Microsoft.Sql"

echo "Creating VNet Rule for PostgreSQL..."
az postgres server vnet-rule create \
  --resource-group $RESOURCE_GROUP \
  --server-name "psql-$PROJECT_NAME" \
  --name "AllowAppSubnet" \
  --subnet "snet-db" \
  --vnet-name $VNET_NAME

# Configure Firewall Rules
echo "Configuring Firewall Rules..."
# Allow from App Subnet
az postgres server firewall-rule create \
  --resource-group $RESOURCE_GROUP \
  --server-name "psql-$PROJECT_NAME" \
  --name "AllowAppTier" \
  --start-ip-address "10.0.2.0" \
  --end-ip-address "10.0.2.255"

# Deny all other IPs (already default)

# Create Database
```

```
echo "Creating Application Database..."
az postgres db create \
  --resource-group $RESOURCE_GROUP \
  --server-name "psql-$PROJECT_NAME" \
  --name "appdb"

echo "✅ Azure Database for PostgreSQL deployed successfully!"
```

5.4 Option B: Deploy PostgreSQL on VM (IaaS)

```
# Create DB VM
echo "Creating Database VM..."
az vm create \
  --resource-group $RESOURCE_GROUP \
  --name "db-vm1" \
  --location $LOCATION \
  --image "Ubuntu2204" \
  --size "Standard_B2s" \
  --admin-username "azureuser" \
  --admin-password "Azure123456!!" \
  --subnet "snet-db" \
  --vnet-name $VNET_NAME \
  --public-ip-address "" \
  --nsg "nsg-db" \
  --custom-data cloud-init-db.txt

echo "✅ Database VM created successfully!"
```

5.5 Database Configuration Script

Create `cloud-init-db.txt`:

yaml

```
#cloud-config
package_upgrade: true
packages:
  - postgresql
  - postgresql-contrib
  - net-tools
  - fail2ban
write_files:
  - path: /etc/postgresql/14/main/postgresql.conf
```

```
  content: |
    data_directory = '/var/lib/postgresql/14/main'
    hba_file = '/etc/postgresql/14/main/pg_hba.conf'
    ident_file = '/etc/postgresql/14/main/pg_ident.conf'
    listen_addresses = '10.0.3.4,localhost'
    port = 5432
    max_connections = 100
    ssl = on
    ssl_cert_file = '/etc/ssl/certs/ssl-cert-snakeoil.pem'
    ssl_key_file = '/etc/ssl/private/ssl-cert-snakeoil.key'
- path: /etc/postgresql/14/main/pg_hba.conf
  content: |
    # Database administrative login by Unix domain socket
    local   all             postgres                                peer
    # TYPE  DATABASE        USER            ADDRESS                 METHOD
    # Allow from App Tier only
    host    all             all             10.0.2.0/24             md5
    # Reject all other connections
    host    all             all             0.0.0.0/0               reject
- path: /tmp/db-info.html
  content: |
    <!DOCTYPE html>
    <html>
    <head>
        <title>DB Tier - 3 Tier Application</title>
        <style>
            body { font-family: Arial, sans-serif; margin: 40px; }
            .container { max-width: 800px; margin: 0 auto; }
            .tier { padding: 20px; margin: 20px 0; border-radius: 5px; }
            .db-tier { background-color: #e8f5e8; border-left: 5px solid #4caf50; }
            .hostname { font-weight: bold; color: #388e3c; }
            .security { background-color: #fff3e0; padding: 15px; border-radius: 5px; }
        </style>
    </head>
    <body>
        <div class="container">
            <h1>Database Tier Service</h1>
            <div class="tier db-tier">
                <h2>Database Tier - Data Storage</h2>
                <p><span class="hostname">Hostname:</span> $(hostname)</p>
                <p><span class="hostname">IP Address:</span> $(hostname -I)</p>
                <p><span class="hostname">Database:</span> PostgreSQL 14</p>
                <div class="security">
```

```
                <h3>🔒 Security Configuration (MOST RESTRICTIVE):</h3>
                <ul>
                  <li>✅ Accepts connections from App Tier ONLY (10.0.2.0/24)</li>
                  <li>❌ No connections from Web Tier (Blocked)</li>
                  <li>❌ No connections from Internet (Blocked)</li>
                  <li>✅ SSH only from Bastion Host (10.0.100.0/26)</li>
                  <li>✅ Database encrypted at rest</li>
                  <li>✅ Connection encryption enabled</li>
                </ul>
              </div>
            </div>
          </div>
      </body>
      </html>
runcmd:
  - systemctl enable postgresql
  - systemctl start postgresql
  - sudo -u postgres psql -c "CREATE DATABASE appdb;"
  - sudo -u postgres psql -c "CREATE USER appuser WITH ENCRYPTED PASSWORD 'AppPass
word123!';"
  - sudo -u postgres psql -c "GRANT ALL PRIVILEGES ON DATABASE appdb TO appuser;"
  - sudo -u postgres psql -d appdb -c "CREATE TABLE IF NOT EXISTS visitors (id SERIAL PRIM
ARY KEY, visit_time TIMESTAMP, source_ip TEXT);"
  - echo "Database Tier configured successfully on $(hostname)" > /tmp/db-tier-ready.txt
  - ufw allow 5432
  - systemctl enable fail2ban
  - systemctl start fail2ban
```

6. Security & Connectivity Setup

6.1 Deploy Bastion Host for Secure Management

```
#!/bin/bash
# File: 05-security-setup.sh
# Description: Deploy security components

echo "=== STEP 5: SECURITY COMPONENTS ==="

# Create Public IP for Bastion
echo "Creating Public IP for Bastion..."
az network public-ip create \
  --name "pip-bastion" \
```

```
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --sku Standard \
  --allocation-method Static

# Deploy Azure Bastion
echo "Deploying Azure Bastion..."
az network bastion create \
  --name "bastion-$PROJECT_NAME" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --vnet-name $VNET_NAME \
  --public-ip-address "pip-bastion"

echo "✅ Azure Bastion deployed successfully!"
```

6.2 Configure Application Gateway (Optional - for advanced scenarios)

```
# Create Application Gateway with WAF
echo "Creating Application Gateway..."
az network application-gateway create \
  --name "agw-$PROJECT_NAME" \
  --resource-group $RESOURCE_GROUP \
  --location $LOCATION \
  --capacity 2 \
  --sku WAF_v2 \
  --public-ip-address "pip-web-lb" \
  --vnet-name $VNET_NAME \
  --subnet "snet-web" \
  --servers "10.0.1.4" "10.0.1.5" \
  --private-ip-address "10.0.1.100"

echo "✅ Application Gateway deployed successfully!"
```

6.3 Test Connectivity Between Tiers

```
#!/bin/bash
# File: 06-test-connectivity.sh
# Description: Test connectivity between tiers

echo "=== STEP 6: CONNECTIVITY TESTING ==="
```

```
# Get Web Tier Private IPs
echo "Getting Web Tier VM IPs..."
WEB_VM1_IP=$(az vm show \
  --resource-group $RESOURCE_GROUP \
  --name web-vm1 \
  --show-details \
  --query 'privateIps' -o tsv)

WEB_VM2_IP=$(az vm show \
  --resource-group $RESOURCE_GROUP \
  --name web-vm2 \
  --show-details \
  --query 'privateIps' -o tsv)

# Get App Tier Private IPs
echo "Getting App Tier VM IPs..."
APP_VM1_IP=$(az vm show \
  --resource-group $RESOURCE_GROUP \
  --name app-vm1 \
  --show-details \
  --query 'privateIps' -o tsv)

APP_VM2_IP=$(az vm show \
  --resource-group $RESOURCE_GROUP \
  --name app-vm2 \
  --show-details \
  --query 'privateIps' -o tsv)

# Get DB IP
echo "Getting Database IP..."
if az postgres server show --name "psql-$PROJECT_NAME" --resource-group $RESOURCE_G
ROUP &>/dev/null; then
  DB_IP=$(az postgres server show \
    --name "psql-$PROJECT_NAME" \
    --resource-group $RESOURCE_GROUP \
    --query 'fullyQualifiedDomainName' -o tsv)
  DB_TYPE="Azure PostgreSQL"
else
  DB_IP=$(az vm show \
    --resource-group $RESOURCE_GROUP \
    --name db-vm1 \
    --show-details \
    --query 'privateIps' -o tsv)
```

```
  DB_TYPE="PostgreSQL VM"
fi

# Get Load Balancer Public IP
echo "Getting Load Balancer Public IP..."
LB_PUBLIC_IP=$(az network public-ip show \
  --name "pip-web-lb" \
  --resource-group $RESOURCE_GROUP \
  --query 'ipAddress' -o tsv)

echo ""
echo "================================================================
======"
echo "               3-TIER ARCHITECTURE SUMMARY"
echo "================================================================
======"
echo ""
echo "📊 TIER DETAILS:"
echo "   Web Tier (Public):"
echo "     • VM1: $WEB_VM1_IP"
echo "     • VM2: $WEB_VM2_IP"
echo "     • Public Access: http://$LB_PUBLIC_IP"
echo ""
echo "   App Tier (Private):"
echo "     • VM1: $APP_VM1_IP:8080"
echo "     • VM2: $APP_VM2_IP:8080"
echo ""
echo "   Database Tier (Restricted):"
echo "     • $DB_TYPE: $DB_IP:5432"
echo ""
echo "🔐 SECURITY RULES:"
echo "   ✅ Internet → Web Tier (Port 80/443): ALLOWED"
echo "   ✅ Web Tier → App Tier (Port 8080): ALLOWED"
echo "   ✅ App Tier → DB Tier (Port 5432): ALLOWED"
echo "   ❌ Internet → App Tier: BLOCKED"
echo "   ❌ Internet → DB Tier: BLOCKED"
echo "   ❌ Web Tier → DB Tier: BLOCKED"
echo ""
echo "🛡️  SECURITY COMPONENTS:"
echo "   • Azure Bastion: Secure SSH/RDP access"
echo "   • NSG Rules: Tier isolation"
echo "   • Private IPs: No public IPs on App/DB tiers"
echo ""
```

```
echo "============================================================
======"
echo ""
echo "🔍 TESTING CONNECTIVITY:"
echo "1. Test Web Tier (from Internet):"
echo "   curl http://$LB_PUBLIC_IP"
echo ""
echo "2. Test App Tier (from Web Tier):"
echo "   ssh azureuser@$WEB_VM1_IP 'curl http://$APP_VM1_IP:8080'"
echo ""
echo "3. Test DB Tier (from App Tier):"
echo "   ssh azureuser@$APP_VM1_IP 'psql -h $DB_IP -U appuser -d appdb -c \"SELECT versi
on();\"'"
echo ""
echo "⚠️  IMPORTANT SECURITY NOTES:"
echo "   • App/DB tiers have NO public IP addresses"
echo "   • Use Azure Bastion for management access"
echo "   • All traffic between tiers is encrypted"
echo "   • Regular security audits recommended"
echo ""
echo "✅ 3-Tier Architecture deployment complete!"
```

7. Complete Deployment Script

7.1 Master Deployment Script

```
#!/bin/bash
# File: deploy-3tier.sh
# Description: Complete 3-Tier Architecture Deployment

echo "================================================"
echo "     AZURE 3-TIER ARCHITECTURE DEPLOYMENT      "
echo "================================================"

# Step 0: Prerequisites
echo ""
echo "🔧 STEP 0: Checking Prerequisites..."
chmod +x 00-prerequisites.sh
./00-prerequisites.sh

# Step 1: Network Setup
echo ""
```

```
echo "🌐 STEP 1: Setting up Network Infrastructure..."
chmod +x 01-network-setup.sh
./01-network-setup.sh

# Step 2: Web Tier
echo ""
echo "🖥️  STEP 2: Deploying Web Tier..."
chmod +x 02-web-tier.sh
./02-web-tier.sh

# Step 3: App Tier
echo ""
echo "⚙️  STEP 3: Deploying App Tier..."
chmod +x 03-app-tier.sh
./03-app-tier.sh

# Step 4: Database Tier
echo ""
echo "🗄️  STEP 4: Deploying Database Tier..."
chmod +x 04-database-tier.sh
./04-database-tier.sh

# Step 5: Security
echo ""
echo "🔐 STEP 5: Deploying Security Components..."
chmod +x 05-security-setup.sh
./05-security-setup.sh

# Step 6: Testing
echo ""
echo "🧪 STEP 6: Testing Connectivity..."
chmod +x 06-test-connectivity.sh
./06-test-connectivity.sh

echo ""
echo "================================================"
echo "          DEPLOYMENT COMPLETE!                  "
echo "================================================"
echo ""
echo "🎉 Your 3-Tier Architecture is now deployed!"
echo ""
echo "📚 NEXT STEPS:"
echo "   1. Access your application: http://$(az network public-ip show \
```

```
  --name "pip-web-lb" \
  --resource-group $RESOURCE_GROUP \
  --query 'ipAddress' -o tsv)"
echo ""
echo "   2. Manage VMs via Azure Bastion:"
echo "      - Go to Azure Portal"
echo "      - Navigate to Bastion resource"
echo "      - Connect to any VM securely"
echo ""
echo "   3. Monitor security:"
echo "      - Check NSG flow logs"
echo "      - Review security center recommendations"
echo "      - Set up alerts for suspicious activity"
echo ""
echo "   4. Scale resources:"
echo "      - Add more VMs to any tier"
echo "      - Configure auto-scaling"
echo "      - Set up database read replicas"
echo ""
echo "🔧 MAINTENANCE:"
echo "   • Update NSG rules as needed"
echo "   • Rotate passwords regularly"
echo "   • Backup database daily"
echo "   • Update VMs monthly"
echo ""
echo "💡 TIPS:"
echo "   • Use Terraform/ARM for production"
echo "   • Implement CI/CD pipeline"
echo "   • Regular security audits"
echo "   • Monitor costs with Azure Cost Management"
echo ""
echo "================================================="
```

7.2 Cleanup Script

```
#!/bin/bash
# File: cleanup-3tier.sh
# Description: Clean up all resources

echo "⚠️  WARNING: This will delete ALL resources in resource group!"
read -p "Are you sure? Type 'yes' to continue: " confirmation
```

```
if [ "$confirmation" != "yes" ]; then
    echo "Cleanup cancelled."
    exit 0
fi

RESOURCE_GROUP="3tier-app-rg"

echo "Deleting resource group: $RESOURCE_GROUP"
az group delete --name $RESOURCE_GROUP --yes --no-wait

echo "Cleanup initiated. Check Azure Portal for progress."
```

---

8. Security Best Practices

8.1 Network Security Configuration

```
#!/bin/bash
# File: security-hardening.sh
# Description: Additional security hardening

echo "🔒 ADDITIONAL SECURITY HARDENING"

# Enable NSG Flow Logs
echo "Enabling NSG Flow Logs..."
az network watcher flow-log create \
  --resource-group $RESOURCE_GROUP \
  --nsg "nsg-web" \
  --storage-account $(az storage account list \
    --resource-group $RESOURCE_GROUP \
    --query '[0].name' -o tsv) \
  --enabled true \
  --format json \
  --interval 10

# Enable Azure DDoS Protection
echo "Enabling DDoS Protection..."
az network ddos-protection create \
  --resource-group $RESOURCE_GROUP \
  --name "ddos-$PROJECT_NAME" \
  --location $LOCATION \
  --vnets $VNET_NAME
```

```
# Configure Azure Security Center
echo "Configuring Azure Security Center..."
az security auto-provisioning-setting update \
  --name "default" \
  --auto-provision "On"

# Enable SQL Threat Detection (if using Azure SQL)
if az postgres server show --name "psql-$PROJECT_NAME" --resource-group $RESOURCE_G
ROUP &>/dev/null; then
  echo "Enabling PostgreSQL Threat Detection..."
  az monitor diagnostic-settings create \
    --resource $(az postgres server show \
      --name "psql-$PROJECT_NAME" \
      --resource-group $RESOURCE_GROUP \
      --query id -o tsv) \
    --name "PostgresSecurityLogs" \
    --workspace $(az monitor log-analytics workspace list \
      --resource-group $RESOURCE_GROUP \
      --query '[0].id' -o tsv) \
    --logs '[{"category": "PostgreSQLLogs", "enabled": true}]'
fi

echo "✅ Security hardening complete!"
```

8.2 Monitoring Setup

```
#!/bin/bash
# File: monitoring-setup.sh
# Description: Set up monitoring and alerts

echo "📊 SETTING UP MONITORING"

# Create Log Analytics Workspace
echo "Creating Log Analytics Workspace..."
az monitor log-analytics workspace create \
  --resource-group $RESOURCE_GROUP \
  --workspace-name "logs-$PROJECT_NAME" \
  --location $LOCATION

# Create Action Group for Alerts
echo "Creating Action Group..."
az monitor action-group create \
```

```
  --resource-group $RESOURCE_GROUP \
  --name "CriticalAlerts" \
  --action email admin "admin@mycompany.com" \
  --short-name "CritAlert"

# Create Alert Rules
echo "Creating Alert Rules..."

# Alert for High CPU on Web Tier
az monitor metrics alert create \
  --name "HighCPU-WebTier" \
  --resource-group $RESOURCE_GROUP \
  --scopes $(az vm show \
    --resource-group $RESOURCE_GROUP \
    --name web-vm1 \
    --query id -o tsv) \
  --condition "avg Percentage CPU > 80" \
  --window-size "5m" \
  --evaluation-frequency "1m" \
  --action-group "CriticalAlerts"

# Alert for Failed Health Probes
az monitor metrics alert create \
  --name "FailedHealthProbes" \
  --resource-group $RESOURCE_GROUP \
  --scopes $(az network lb show \
    --resource-group $RESOURCE_GROUP \
    --name lb-web \
    --query id -o tsv) \
  --condition "count DipAvailability > 0" \
  --window-size "5m" \
  --evaluation-frequency "1m" \
  --action-group "CriticalAlerts"

echo "✅ Monitoring setup complete!"
```

9. Troubleshooting Guide

Common Issues and Solutions

Issue 1: Cannot access web tier from internet

```
# Check Load Balancer health
az network lb show \
  --name lb-web \
  --resource-group $RESOURCE_GROUP \
  --query '{provisioningState:provisioningState, frontendIPs:frontendIpConfigurations}'

# Check NSG rules
az network nsg rule list \
  --nsg-name nsg-web \
  --resource-group $RESOURCE_GROUP \
  --output table

# Check VM network
az vm show \
  --name web-vm1 \
  --resource-group $RESOURCE_GROUP \
  --show-details \
  --query '{privateIp:privateIps, publicIp:publicIps}'
```

Issue 2: App tier cannot connect to database

```
# Test connectivity from app VM
az vm run-command invoke \
  --command-id RunShellScript \
  --name app-vm1 \
  --resource-group $RESOURCE_GROUP \
  --scripts "nc -zv $DB_IP 5432"

# Check NSG rules on DB tier
az network nsg rule list \
  --nsg-name nsg-db \
  --resource-group $RESOURCE_GROUP \
  --output table

# Check database firewall rules (if Azure PostgreSQL)
az postgres server firewall-rule list \
  --server-name "psql-$PROJECT_NAME" \
  --resource-group $RESOURCE_GROUP \
  --output table
```

Issue 3: Bastion connection fails

```
# Check Bastion status
az network bastion show \
  --name "bastion-$PROJECT_NAME" \
  --resource-group $RESOURCE_GROUP \
  --query provisioningState

# Check NSG allows SSH from Bastion subnet
az network nsg rule list \
  --nsg-name nsg-app \
  --resource-group $RESOURCE_GROUP \
  --query "[?contains(name, 'Bastion')]" \
  --output table
```

---

10. Architecture Diagrams & Documentation

10.1 Complete Architecture Diagram

10.2 Security Flow Diagram

```
┌─────────────────────────────────────────────────────────────────────┐
│                    SECURITY FLOW                                    │
├─────────────────────────────────────────────────────────────────────┤
│                                                                     │
│  INTERNET TRAFFIC:                                                  │
│  ✅ Allowed → Web Tier (Port 80/443)                                │
│  ❌ Blocked → App Tier                                              │
│  ❌ Blocked → DB Tier                                               │
│                                                                     │
│  WEB → APP:                                                         │
│  ✅ Allowed → Port 8080 only                                        │
│  ❌ Blocked → All other ports                                       │
│                                                                     │
│  APP → DB:                                                          │
│  ✅ Allowed → Port 5432 only                                        │
│  ❌ Blocked → All other ports                                       │
│                                                                     │
│  MANAGEMENT:                                                        │
│  ✅ Bastion → All tiers (Port 22)                                   │
│  ❌ Direct SSH from Internet → Blocked                              │
└─────────────────────────────────────────────────────────────────────┘
```

10.3 IP Address Planning Table

Tier	Subnet	IP Range	Gateway	Usable IPs	Purpose
Web	snet-web	10.0.1.0/24	10.0.1.1	10.0.1.4-254	Public-facing web servers
App	snet-app	10.0.2.0/24	10.0.2.1	10.0.2.4-254	Business logic servers
DB	snet-db	10.0.3.0/24	10.0.3.1	10.0.3.4-254	Database servers
Bastion	AzureBastionSubnet	10.0.100.0/26	10.0.100.1	10.0.100.4-62	Secure management

11. Cost Optimization

11.1 Cost Estimation

```
#!/bin/bash
# File: cost-estimation.sh
# Description: Estimate monthly costs

echo "💰 COST ESTIMATION (Monthly - East US)"

echo ""
echo "COMPUTE:"
echo "  • Web VMs (2x B2s):       ~$40/month"
echo "  • App VMs (2x B2s):       ~$40/month"
echo "  • DB VM (1x B2s):         ~$20/month"
echo "  • Total Compute:          ~$100/month"

echo ""
echo "NETWORKING:"
echo "  • Load Balancer:          ~$18/month"
echo "  • Application Gateway:    ~$150/month (optional)"
echo "  • Azure Bastion:          ~$130/month"
echo "  • Bandwidth (first 5GB):  ~$0.50/GB"
echo "  • Total Networking:       ~$150-$300/month"

echo ""
echo "DATABASE (if using PaaS):"
echo "  • Azure PostgreSQL Basic: ~$25/month"
echo "  • Azure SQL Basic:        ~$30/month"

echo ""
echo "STORAGE:"
```

```
echo "  • Managed Disks:          ~$5/VM/month"
echo "  • Total Storage:          ~$15/month"

echo ""
echo "📊 TOTAL ESTIMATED COST:"
echo "  • IaaS Solution:          ~$265-$415/month"
echo "  • PaaS Solution:          ~$290-$440/month"

echo ""
echo "💡 COST OPTIMIZATION TIPS:"
echo "  1. Use Reserved Instances (save up to 72%)"
echo "  2. Shut down dev environments when not in use"
echo "  3. Use Azure Hybrid Benefit for Windows/Linux"
echo "  4. Implement auto-scaling"
echo "  5. Monitor with Azure Cost Management"
```

12. Summary & Best Practices

12.1 Key Takeaways

1. **Security First:**
   - Each tier has its own security boundary
   - Use NSGs to enforce tier isolation
   - No public IPs on private tiers
2. **Scalability:**
   - Scale each tier independently
   - Use load balancers for distribution
   - Implement auto-scaling rules
3. **Resilience:**
   - Deploy VMs in availability sets
   - Use managed disks for durability
   - Implement backup strategies
4. **Maintainability:**
   - Use Infrastructure as Code (IaC)
   - Implement CI/CD pipelines
   - Regular security updates

12.2 Production Checklist

- Enable Azure Backup for all VMs
- Configure Azure Site Recovery
- Implement Azure Monitor alerts
- Set up Azure Cost Management budgets
- Enable Azure Security Center
- Configure Azure Active Directory integration
- Implement Azure Policy for compliance
- Regular penetration testing
- Disaster recovery drills