Step-by-Step Technical Guide to Configure a Debian Bookworm VM on Google Cloud for a TypeScript, Vite, and React-Based Landing Page

- Create a Debian Bookworm VM with 2 CPU cores, 8GB RAM, and 10GB SSD on Google Cloud.
- Install Node.js, npm, Git, and configure Antigravity VS Code for development.
- Set up Vite with React and TypeScript, clone from GitHub, and automate builds with Git hooks.
- Configure Nginx for static file delivery with SSL, caching, and compression.
- Install tmux for persistent npm sessions and terminal tools for monitoring.
- Automate the entire setup with a comprehensive bash script including error handling and logging.

Prerequisites

- A Google Cloud account with billing enabled.
- Familiarity with Google Cloud Console and `gcloud` CLI.
- PuTTY or equivalent SSH client for terminal access.
- A GitHub repository URL for the landing page project.

Step-by-Step Manual Setup

1. Create the VM Instance

Using Google Cloud Console or **`gcloud`** CLI, create a VM instance with the following specifications:

- **Machine Type**: Custom (2 vCPUs, 8GB RAM)
- **Boot Disk**: Debian 12 Bookworm, 10GB SSD
- **Zone**: Choose an appropriate zone (e.g., `us-central1-a`)
- **Firewall**: Allow HTTP (80), HTTPS (443), SSH (22), and FTP (21) ports.

Command Example:

```
gcloud compute instances create landing-page-vm \
  --zone=us-central1-a \
  --machine-type=custom-2-8192 \
  --create-disk=auto-delete=yes,boot=yes,device-name=landing-page-disk,image=projects/debian-clouc
  --tags=http-server,https-server,ftp-server \
  --metadata=startup-script-url=gs://your-bucket/startup-script.sh
```

References:
- Creating a VM with custom machine type
- Debian 12 Bookworm image

2. Install Node.js, npm, and Git

Once the VM is running, connect via SSH using PuTTY or `gcloud compute ssh`:

```
gcloud compute ssh landing-page-vm --zone=us-central1-a
```

Install Node.js, npm, and Git:

```
sudo apt update
sudo apt install -y nodejs npm git
```

Verify installations:

```
node -v
npm -v
git --version
```

References:
- Installing Node.js and npm on Debian
- Installing Git on Debian

3. Install and Configure Antigravity VS Code

Google Antigravity is a fork of VS Code with AI-powered features. Install it via Google's official APT repository:

```
sudo apt update
sudo apt install -y curl gpg
curl -fsSL https://us-central1-apt.pkg.dev/doc/repo-signing-key.gpg | sudo gpg --dearmor -o /usr/s
echo "Types: deb\nURIs: https://us-central1-apt.pkg.dev/projects/antigravity-auto-updater-dev/\nSu
sudo apt update
sudo apt install -y antigravity
```

Launch Antigravity:

```
antigravity
```

References:
- Google Antigravity Installation Guide

4. Set Up Version Control with GitHub

Clone the GitHub repository containing the landing page:

```
git clone https://github.com/your-username/landing-page.git /var/www/landing-page
cd /var/www/landing-page
```

Set up a Git hook to automatically rebuild the landing page on `git pull`:

Create a `post-receive` hook in the Git repository:

```
cd /var/www/landing-page/.git/hooks
touch post-receive
chmod +x post-receive
```

Edit `post-receive`:

```
#!/bin/bash
cd /var/www/landing-page
npm install
npm run build
echo "Build completed at $(date)" >> /var/log/landing-page-build.log
```

References:
- GitHub Clone Instructions
- Git Hooks Guide

5. Install and Configure Vite, React, and TypeScript

Initialize a Vite project with React and TypeScript:

```
cd /var/www/landing-page
npm create vite@latest . --template react-ts
npm install
```

Configure Vite to use TypeScript by ensuring `vite.config.ts` includes the TypeScript plugin:

```
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react'
import typescript from '@rollup/plugin-typescript'

export default defineConfig({
  plugins: [react(), typescript()]
})
```

Install TypeScript and React type definitions:

```
npm install --save-dev typescript @types/react
```

References:
- Vite Getting Started
- Vite TypeScript Support

6. Configure Nginx for Static File Delivery

Install Nginx:

```
sudo apt install -y nginx
```

Configure Nginx to serve the landing page from `/var/www/landing-page/dist`:

```
sudo vim /etc/nginx/sites-available/default
```

Add the following configuration:

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    server_name your-domain.com;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    location /static/ {
        alias /var/www/landing-page/dist/;
        expires 30d;
        add_header Cache-Control "public, no-transform";
        gzip on;
        brotli on;
    }
}
```

Enable the site and restart Nginx:

```
sudo ln -s /etc/nginx/sites-available/default /etc/nginx/sites-enabled/
sudo systemctl restart nginx
```

References:
- Nginx Configuration Guide
- Nginx Caching and Compression

7. Set Up FTP Access with VSFTPD

Install VSFTPD:

```
sudo apt install -y vsftpd
```

Configure VSFTPD:

```
sudo vim /etc/vsftpd.conf
```

Ensure the following settings:

```
listen=YES
anonymous_enable=NO
local_enable=YES
write_enable=YES
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
chroot_local_user=YES
secure_chroot_dir=/var/run/vsftpd/empty
pam_service_name=vsftpd
rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
ssl_enable=YES
allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=NO
require_ssl_reuse=NO
ssl_ciphers=HIGH
```

Create an FTP user and set permissions:

```
sudo useradd -m ftpuser
sudo passwd ftpuser
sudo mkdir -p /var/www/landing-page/ftp
sudo chown ftpuser:ftpuser /var/www/landing-page/ftp
```

Restart VSFTPD:

```
sudo systemctl restart vsftpd
```

References:
- VSFTPD Configuration
- FTP on Google Cloud

8. Configure tmux and Terminal Tools

Install tmux:

```
sudo apt install -y tmux
```

Create a tmux session for development:

```
tmux new-session -s landing-page-dev
```

Install monitoring tools:

```
sudo apt install -y htop net-tools ncdu
```

References:
- tmux Installation and Usage
- Google Cloud Monitoring Agent

9. Automated Bash Script

Below is a self-contained bash script that automates the entire setup process, including error handling and logging:

```
#!/bin/bash

# User-defined variables
GITHUB_REPO_URL="https://github.com/your-username/landing-page.git"
DOMAIN_NAME="your-domain.com"
FTP_USERNAME="ftpuser"
FTP_PASSWORD="securepassword"
LOG_FILE="/var/log/landing-page-setup.log"

# Function to log messages
log() {
    echo "[$(date)] $1" | tee -a $LOG_FILE
}

# Function to handle errors
error_exit() {
    log "ERROR: $1"
    exit 1
}
```

```
# Install dependencies
install_dependencies() {
    log "Installing dependencies..."
    sudo apt update || error_exit "Failed to update package list"
    sudo apt install -y nodejs npm git nginx vsftpd tmux htop net-tools ncdu || error_exit "Failed
}

# Configure Nginx
configure_nginx() {
    log "Configuring Nginx..."
    sudo cp /etc/nginx/sites-available/default /etc/nginx/sites-available/default.bak
    sudo bash -c "cat > /etc/nginx/sites-available/default" << 'EOF'
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    server_name $DOMAIN_NAME;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    location /static/ {
        alias /var/www/landing-page/dist/;
        expires 30d;
        add_header Cache-Control "public, no-transform";
        gzip on;
        brotli on;
    }
}
EOF
    sudo ln -s /etc/nginx/sites-available/default /etc/nginx/sites-enabled/
    sudo systemctl restart nginx || error_exit "Failed to restart Nginx"
}

# Configure FTP
configure_ftp() {
    log "Configuring FTP..."
    sudo cp /etc/vsftpd.conf /etc/vsftpd.conf.bak
    sudo bash -c "cat > /etc/vsftpd.conf" << 'EOF'
listen=YES
anonymous_enable=NO
local_enable=YES
write_enable=YES
dirmessage_enable=YES
use_localtime=YES
xferlog_enable=YES
connect_from_port_20=YES
```

```
chroot_local_user=YES
secure_chroot_dir=/var/run/vsftpd/empty
pam_service_name=vsftpd
rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
ssl_enable=YES
allow_anon_ssl=NO
force_local_data_ssl=YES
force_local_logins_ssl=YES
ssl_tlsv1=YES
ssl_sslv2=NO
ssl_sslv3=NO
require_ssl_reuse=NO
ssl_ciphers=HIGH
EOF
    sudo useradd -m $FTP_USERNAME
    echo "$FTP_USERNAME:$FTP_PASSWORD" | sudo chpasswd
    sudo mkdir -p /var/www/landing-page/ftp
    sudo chown $FTP_USERNAME:$FTP_USERNAME /var/www/landing-page/ftp
    sudo systemctl restart vsftpd || error_exit "Failed to restart VSFTPD"
}

# Clone GitHub repository and set up Git hook
setup_git() {
    log "Setting up Git repository..."
    sudo git clone $GITHUB_REPO_URL /var/www/landing-page || error_exit "Failed to clone repositor
    cd /var/www/landing-page
    sudo chown -R $FTP_USERNAME:$FTP_USERNAME /var/www/landing-page
    sudo mkdir -p .git/hooks
    sudo bash -c "cat > .git/hooks/post-receive" << 'EOF'
#!/bin/bash
cd /var/www/landing-page
npm install
npm run build
echo "Build completed at $(date)" >> /var/log/landing-page-build.log
EOF
    sudo chmod +x .git/hooks/post-receive
}

# Install and configure tmux
setup_tmux() {
    log "Setting up tmux..."
    sudo apt install -y tmux || error_exit "Failed to install tmux"
    tmux new-session -d -s landing-page-dev
}

# Main execution
main() {
    install_dependencies
    configure_nginx
    configure_ftp
    setup_git
    setup_tmux
    log "Setup completed successfully!"
```

```
}

main
```

Verification Steps

- **Test Nginx**: Open a browser and navigate to `http://your-domain.com` or the VM's external IP.
- **Test FTP**: Use an FTP client to connect to the VM using the FTP credentials.
- **Test tmux**: Reattach to the tmux session (`tmux attach-session -t landing-page-dev`) and verify npm processes are running.
- **Test Git Hook**: Make a change to the repository, commit, and push to trigger the `post-receive` hook.

Troubleshooting

Issue	Solution
Nginx fails to start	Check Nginx logs: `sudo journalctl -xe` or `sudo tail -f /var/log/nginx/error.log`
FTP connection refused	Verify VSFTPD is running: `sudo systemctl status vsftpd` and check firewall rules
Git hook not executing	Ensure the hook is executable: `chmod +x .git/hooks/post-receive` and check logs
tmux session not persisting	Verify tmux is installed and running: `tmux ls` and `tmux attach-session -t landing-page-dev`

Optimization Notes

- **CDN Integration**: Use Google Cloud CDN to cache static assets globally for faster delivery.
- **Vite Build Flags**: Optimize Vite build with flags like `--minify` and `--sourcemap` for production.
- **Monitoring**: Set up Google Cloud Monitoring for performance metrics and alerts.
- **Security**: Regularly update packages and restrict SSH/FTP access to trusted IPs.

This guide provides a comprehensive, step-by-step approach to configuring a production-ready Debian Bookworm VM on Google Cloud for hosting a TypeScript, Vite, and React-based landing page, with automation and observability tools integrated. The accompanying bash script automates the entire process, ensuring a consistent and efficient setup.