WSL2 + Docker Setup - Quick Start Checklist

Print this page and check off each step as you complete it.

Pre-Installation Checklist

- [ ] Windows 10/11 Build 19041 or higher
- [ ] At least 12GB RAM
- [ ] Administrator access to Windows
- [ ] 50GB+ free disk space
- [ ] Stable internet connection
- [ ] GitHub account created

Phase 1: WSL2 Configuration (30 minutes)

Windows Side

- [ ] Open PowerShell as Administrator
- [ ] Run: `wsl --version` (verify WSL is installed)
- [ ] Create `C:\Users\<YourUsername>\.wslconfig`
- [ ] Copy provided `.wslconfig` content
- [ ] Save file
- [ ] Run: `wsl --shutdown`
- [ ] Wait 10 seconds
- [ ] Run: `wsl` (restart WSL)

WSL Ubuntu Side

- [ ] Open WSL Ubuntu terminal
- [ ] Run: `sudo nano /etc/wsl.conf`
- [ ] Paste provided `wsl.conf` content
- [ ] Save: `Ctrl+X`, then `Y`, then `Enter`
- [ ] Exit WSL: `exit`
- [ ] From PowerShell: `wsl --shutdown`
- [ ] Restart: `wsl`
- [ ] Verify systemd: `systemctl --version`

✅ **Checkpoint:** Systemd version should appear

Phase 2: Docker Installation (30 minutes)

Install Docker in WSL

- [ ] Update packages: `sudo apt-get update`
- [ ] Install prerequisites (ca-certificates, curl, gnupg, lsb-release)
- [ ] Add Docker GPG key
- [ ] Set up Docker repository
- [ ] Install Docker Engine: `sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin`
- [ ] Add user to docker group: `sudo usermod -aG docker $USER`
- [ ] Apply group: `newgrp docker`
- [ ] Test: `docker run hello-world`

✅ **Checkpoint:** "Hello from Docker!" message appears

Install Docker Desktop (Windows)

- [ ] Download Docker Desktop from docker.com
- [ ] Run installer
- [ ] Restart computer if prompted
- [ ] Open Docker Desktop
- [ ] Skip tutorial
- [ ] Go to Settings → General:
- [ ] Check "Use WSL 2 based engine"
- [ ] Uncheck "Start Docker Desktop when you log in"
- [ ] Go to Settings → Resources → WSL Integration:
- [ ] Enable "Ubuntu" (or your distro)
- [ ] Go to Settings → Resources → Advanced:
- [ ] Memory: 4 GB
- [ ] CPUs: 2
- [ ] Click "Apply & Restart"

✅ **Checkpoint:** Docker Desktop shows "Running"

Phase 3: Project Structure (15 minutes)

Create Directories

```bash
# Copy and paste this entire block
mkdir -p ~/projects/{web-apps,microservices,ml-models,databases,mcp-services,_templates,_shared,_docs}
mkdir -p ~/projects/_shared/{docker-networks,configs,scripts,volumes}
```

- [ ] Run above command
- [ ] Verify: `tree ~/projects -L 2`

Create Shared Networks

- [ ] `cd ~/projects/_shared/docker-networks`
- [ ] Create `docker-compose.yml` with provided content
- [ ] Run: `docker compose up -d`
- [ ] Verify: `docker network ls`

✅ **Checkpoint:** See networks: app-network, db-network, mcp-network

---

Phase 4: VS Code Setup (20 minutes)

Install Extensions

- [ ] Open VS Code
- [ ] Install: Remote - WSL
- [ ] Install: Docker
- [ ] Install: Python
- [ ] Install: Dev Containers
- [ ] Install: GitLens (optional)

Connect to WSL

- [ ] In WSL terminal: `code ~/projects`
- [ ] VS Code opens in WSL mode
- [ ] Bottom-left corner shows "WSL: Ubuntu"

✅ **Checkpoint:** VS Code connected to WSL

Phase 5: Git Configuration (10 minutes)

- [ ] In WSL: `git config --global user.name "Your Name"`
- [ ] `git config --global user.email "your@email.com"`
- [ ] `git config --global core.autocrlf input`
- [ ] `git config --global core.filemode false`
- [ ] `git config --global pull.rebase false`
- [ ] Verify: `git config --list`

✅ **Checkpoint:** Name and email appear in config

Phase 6: Create First Project (15 minutes)

From Template

- [ ] `cd ~/projects/_templates`
- [ ] Create `python-microservice/` directory
- [ ] Copy all template files from artifacts:
- [ ] `.gitignore`
- [ ] `.dockerignore`
- [ ] `Dockerfile`
- [ ] `docker-compose.yml`
- [ ] `.env.example`
- [ ] `.devcontainer/devcontainer.json`
- [ ] `.vscode/settings.json`

Test Project

- [ ] `cp -r ~/projects/_templates/python-microservice ~/projects/microservices/test-app`
- [ ] `cd ~/projects/microservices/test-app`
- [ ] `cp .env.example .env`
- [ ] `nano .env` (edit if needed)
- [ ] `docker compose up -d`
- [ ] `docker compose ps` (verify services are running)
- [ ] `docker compose logs -f` (check logs)
- [ ] `docker compose down` (stop containers)

✅ **Checkpoint:** Containers start and stop successfully

Phase 7: GitHub Integration (20 minutes)

Setup SSH Key

- [ ] `ssh-keygen -t ed25519 -C "your@email.com"`
- [ ] Press Enter (accept default location)
- [ ] Enter passphrase (optional)
- [ ] `cat ~/.ssh/id_ed25519.pub` (copy output)
- [ ] Go to GitHub → Settings → SSH Keys → New SSH key
- [ ] Paste key, give it a name (e.g., "WSL2-Laptop")
- [ ] Save
- [ ] Test: `ssh -T git@github.com`

✅ **Checkpoint:** "Hi <username>! You've successfully authenticated"

Create Test Repository

- [ ] On GitHub, create new repository "test-wsl-setup"
- [ ] Copy SSH clone URL
- [ ] In WSL: `cd ~/projects/microservices/test-app`
- [ ] `git init`
- [ ] `git add .`
- [ ] `git commit -m "Initial commit"`
- [ ] `git branch -M main`
- [ ] `git remote add origin git@github.com:username/test-wsl-setup.git`
- [ ] `git push -u origin main`

✅ **Checkpoint:** Code appears on GitHub

Phase 8: Final Verification (10 minutes)

System Check

- [ ] WSL version: `wsl --version`
- [ ] Docker version: `docker --version`
- [ ] Docker Compose: `docker compose version`
- [ ] Python: `python3 --version`
- [ ] Git: `git --version`
- [ ] VS Code: `code --version`

Resource Check

- [ ] In PowerShell: Task Manager → Performance
- [ ] Check "Vmmem" process (should be ~1-2GB idle)
- [ ] In WSL: `docker stats` (when containers running)
- [ ] Verify limits are respected

Functionality Check

- [ ] Can create project from template: ✓
- [ ] Can start/stop containers: ✓
- [ ] Can commit to Git: ✓
- [ ] Can push to GitHub: ✓
- [ ] VS Code connects to WSL: ✓
- [ ] Dev containers work: ✓

✅ **Checkpoint:** All checks pass

Common First-Time Issues

Issue: "WSL not found"

Fix: Install WSL from Microsoft Store, then restart

Issue: "Docker daemon not running"

Fix: `sudo systemctl start docker`

Issue: "Permission denied (docker)"

Fix: `sudo usermod -aG docker $USER`, then `newgrp docker`

Issue: "Cannot find module 'xyz'"

Fix: Inside container: `pip install xyz`

Issue: "Port already in use"

Fix: Change port in `docker-compose.yml` or kill process

Post-Setup Tasks

Optional but Recommended

- [ ] Install GitHub CLI: `sudo apt install gh`
- [ ] Configure git aliases (provided in artifacts)
- [ ] Set up Docker Desktop notifications
- [ ] Bookmark this checklist
- [ ] Join course Discord/Slack for help

Create Helper Scripts

- [ ] Copy `create-project.sh` to `~/projects/_shared/scripts/`
- [ ] Copy `pre-switch.sh` to `~/projects/_shared/scripts/`
- [ ] Copy `toggle-mcp.sh` to `~/projects/_shared/scripts/`
- [ ] Make executable: `chmod +x ~/projects/_shared/scripts/*.sh`

Success Criteria

You're ready to start developing when:

✅ **Performance:**

- [ ] WSL2 uses ≤6GB RAM
- [ ] Docker uses ≤4GB RAM
- [ ] System remains responsive with containers running

✅ **Functionality:**

- [ ] Can create new projects from template
- [ ] Can start/stop containers independently
- [ ] Can develop inside VS Code dev containers
- [ ] Git commits work without permission errors

✅ **Multi-Machine:**

- [ ] Can push code to GitHub
- [ ] Can clone on another machine
- [ ] Containers work identically on both machines

Estimated Total Time

- **Experienced users:** 90 minutes
- **First-time users:** 2-3 hours
- **With troubleshooting:** 3-4 hours

Don't rush! Take breaks between phases.

Getting Help

If stuck:

1. Check the Troubleshooting section in COMPLETE_SETUP_GUIDE.md
2. Review the specific phase instructions
3. Search error message on Google/Stack Overflow
4. Ask in course forum/Discord
5. Review Docker/WSL documentation

Common documentation links:

- WSL Docs: https://learn.microsoft.com/en-us/windows/wsl/
- Docker Docs: https://docs.docker.com/
- VS Code Remote: https://code.visualstudio.com/docs/remote/wsl

---

Final Notes

📝 **Keep this checklist** for setting up additional machines

🎯 **Goal:** Complete all checkboxes before starting development

⏰ **Budget:** 2-4 hours for first-time setup

💾 **Backup:** Commit your configuration files to a "dotfiles" repo

🎓 **Learn:** Understand each step, don't just copy-paste

🎉 **Congratulations on completing the setup!**

Now you're ready to build production-grade applications with professional developer tools.

Quick Commands Reference Card

Cut this out and keep it visible:

```
# Start work
cd ~/projects/microservices/my-app
git pull origin main
docker compose up -d
code .

# End work
git add .
git commit -m "feat: description"
git push origin main
docker compose down

# Check resources
docker stats
docker compose ps

# Clean up
docker system prune -a
```