Presentation Day Checklist - Thursday

Morning Setup (1 Hour Before)

Environment Check

- [ ] Laptop fully charged + power adapter ready
- [ ] Projector/screen tested with laptop
- [ ] HDMI/DisplayPort adapter if needed
- [ ] Backup adapter in bag
- [ ] Mouse connected (easier than trackpad for live coding)
- [ ] Water bottle filled

Git Repository Status

```
# Verify all branches exist and are up to date
git fetch origin
git branch -a | grep demo/

# Should see:
# demo/stage-0-blank
# demo/stage-1-naive-csharp
# demo/stage-2-first-du
# demo/stage-3-pattern-matching
```

Test Branch Checkouts (Critical!)

```
# Start clean
cd c:\git\NovuckFSharpPresentation
git status  # Should be clean

# Test each stage
git checkout demo/stage-1-naive-csharp
dotnet build
dotnet run --project CustomerLiveDemo

git checkout demo/stage-2-first-du
dotnet build
dotnet run --project CustomerLiveDemo

git checkout demo/stage-3-pattern-matching
dotnet build
dotnet run --project CustomerLiveDemo

# Start on Stage 1 for presentation
git checkout demo/stage-1-naive-csharp
```

Visual Studio / VS Code Setup

- [ ] Open solution in VS Code: `src/Live/CustomerLive.sln`
- [ ] Font size increased for projector: 18pt minimum
  - Settings → Text Editor → Font Size → 18
- [ ] Terminal font size increased: 16pt minimum
- [ ] Color theme: Light theme for better projector visibility
- [ ] Close all unnecessary panels (maximize editor space)
- [ ] Pin frequently used files: `Library.fs`, `Program.cs`
- [ ] Zoom level: 150% or higher

Screen Layout

- [ ] Editor: Left 60% of screen
- [ ] Terminal: Right 40% of screen (or bottom panel)
- [ ] Hide file explorer during live coding (toggle back if needed)
- [ ] Close email, Slack, notifications (DO NOT DISTURB MODE!)

Build Test

```
# From src/Live/
dotnet clean
dotnet build
dotnet run --project CustomerLiveDemo

# Should output:
# John (£100): £90.0
# Grinch (£100): £100
```

Timer Setup

- [ ] Phone timer app ready (or laptop timer)
- [ ] Set for 30 minutes (your total presentation time)
- [ ] 5-minute warning alarm set

Presentation Materials

Print and Bring

- [ ] `stage1-cheatsheet.md` (printed)
- [ ] `stage2-cheatsheet.md` (printed)
- [ ] `stage3-cheatsheet.md` (printed)
- [ ] `recovery-playbook.md` (printed, highlighted)
- [ ] Backup USB drive with repository clone

Have Open on Laptop (Tabs)

- [ ] GitHub repo in browser (backup reference)
- [ ] F# documentation: https://fsharp.org/
- [ ] This checklist
- [ ] Timer

---

15 Minutes Before Presentation

Final Checks

- [ ] Laptop connected to projector (test display)
- [ ] Audio works (if needed for demos)
- [ ] VS Code open with Stage 1 code visible
- [ ] Terminal open and ready
- [ ] Cheatsheets on podium/table
- [ ] Water within reach
- [ ] Phone on DO NOT DISTURB

Git Status Verification

```
# Verify you're on Stage 1
git branch --show-current
# Should output: demo/stage-1-naive-csharp

# Verify no uncommitted changes
git status
# Should output: nothing to commit, working tree clean
```

Quick Mental Rehearsal

- [ ] Remember opening line
- [ ] Know transition points between stages
- [ ] Know recovery commands by heart: `git reset --hard`, `git checkout demo/stage-X`
- [ ] Know key talking points: "illegal states", "compiler-enforced", "cleaner C#"

During Presentation

Stage 1 (Minutes 5-15)

- [ ] Show Stage 1 code
- [ ] Demonstrate the bug (Grinch gets discount)
- [ ] Explain illegal state problem
- [ ] Ask: "Has anyone forgotten a check like this?"

Stage 2 (Minutes 15-20)

- [ ] Live code: Create RegisteredCustomer, UnregisteredCustomer

- [ ] Live code: Create Customer DU
- [ ] Live code: Update C# with factory methods
- [ ] Build and run
- [ ] Show prevention: Try to create illegal state
- [ ] **Fallback:** If stuck, `git reset --hard; git checkout demo/stage-2-first-du`

Stage 3 (Minutes 20-25)

- [ ] Live code: Remove IsEligible boolean
- [ ] Live code: Add Standard tier to DU
- [ ] Live code: Update C# to NewStandard, NewRegistered, NewGuest
- [ ] Live code: Simplify CalculateTotal
- [ ] Build and run
- [ ] **Fallback:** If stuck, `git reset --hard; git checkout demo/stage-3-pattern-matching`

VIP Extension (Minutes 25-30)

- [ ] Live code: Add VIP tier to DU
- [ ] Live code: Add alice as VIP customer
- [ ] Live code: Update CalculateTotal with VIP case
- [ ] Build and run (show Alice gets 15% discount)
- [ ] **Fallback:** Skip if running out of time

Emergency Procedures

Build Fails

1. Read error message out loud
2. Fix obvious typo
3. If not obvious: `git checkout demo/stage-X` (current stage fallback)
4. Continue from working code

Complete Disaster

1. Stay calm: "Let me jump to the working version"
2. `git reset --hard`
3. `git checkout demo/stage-3-pattern-matching`
4. Continue with VIP extension demo

Running Behind Schedule

- Skip detailed explanation of Stage 2
- Jump straight to Stage 3 (`git checkout demo/stage-3-pattern-matching`)
- Do VIP extension only

Running Ahead of Schedule

- Do VIP extension slowly with more explanation
- Show pattern matching alternative

- Take more audience questions
- Show F# pattern matching syntax

After Presentation

Immediate

- [ ] Thank audience
- [ ] Answer questions
- [ ] Share repository link: `github.com/billkuck/NovuckFSharpPresentation`
- [ ] Mention contact info / LinkedIn

Later (Same Day)

- [ ] Tag the presentation commit

```
git tag presentation-2024-12-19
git push --tags
```

- [ ] Commit any interesting variations from live demo
- [ ] Update notes with what worked / didn't work
- [ ] Email slides/repo link to organizer

Recovery Commands (Memorize These!)

```
# Discard all changes, return to clean state
git reset --hard

# Jump to Stage 1 fallback
git checkout demo/stage-1-naive-csharp

# Jump to Stage 2 fallback
git checkout demo/stage-2-first-du

# Jump to Stage 3 fallback
git checkout demo/stage-3-pattern-matching

# Check current branch
git branch --show-current

# Check git status
git status
```

Key Messages to Emphasize

- ✨ **"Make illegal states unrepresentable"**
- ✨ **"The more we break down F# types, the cleaner the C# gets"**
- ✨ **"Compiler-enforced correctness"**
- ✨ **"Domain language lives in the type system"**
- ✨ **"F# and C# work together seamlessly"**

Confidence Reminders

- [x] You've built this three times - you know it works
- [x] All branches are tested and pushed to origin
- [x] You have fallback branches if live coding fails
- [x] The cheatsheets have all the code you need
- [x] The story is clear: Problem → Solution → Better Solution
- [x] You've practiced the refactorings
- [x] The audience wants you to succeed!

Good Luck! ☘

You've got this. The code is solid, the story is clear, and you're prepared.

If something goes wrong: Stay calm, use the fallback branches, keep presenting.

Most important: Have fun and show your passion for F#!