

# Gilt Testing Checklist

Tester: \_\_\_\_\_ Date: \_\_\_\_\_ Version: \_\_\_\_\_ Commit: \_\_\_\_\_

## Prerequisites

1. Run `./tests/setup-test-repo.sh` to create the test repo
2. Optionally: `GITHUB_REMOTE=git@github.com:parenworks/gilt-test-repo.git ./tests/setup-test-repo.sh`
3. `cd gilt-test-repo && gilt`
4. Run `./tests/run-tests.sh` for automated smoke tests first

**Legend:** [P] Pass | [F] Fail | [S] Skip | [N/A] Not applicable | **Auto** column: [Y] = covered by `run-tests.sh`

## 1. Startup & Display

#	Test	Key	Expected	Result	Auto	Notes
1.1	Launch gilt	<code>gilt</code>	TUI renders with 5 left panels + main + cmdlog	[ ]	[Y]	Binary validated
1.2	Status panel shows repo name	—	"[1] Status" with repo name and branch	[ ]		
1.3	Files panel shows files	—	Modified, staged, untracked files visible	[ ]	[Y]	File state verified
1.4	Branches panel shows branches	—	master (green *), feature branches listed	[ ]	[Y]	Branches verified
1.5	Commits panel shows history	—	Commits with yellow hash, initials, message	[ ]	[Y]	Commit count verified
1.6	Stash panel shows stashes	—	2 stashes from setup script	[ ]	[Y]	Stash count verified
1.7	Help bar visible	—	Context-sensitive hints at bottom	[ ]		
1.8	Main panel shows diff	—	Diff for selected file in right panel	[ ]	[Y]	Diff output verified
1.9	Command log visible	—	Bottom-right shows recent commands	[ ]		
1.10	Recent repo saved	—	Check <code>~/.config/gilt/recent-repos.txt</code> exists	[ ]	[Y]	Config file verified

## 2. Navigation

#	Test	Key	Expected	Result	Auto	Notes
2.1	Move down	j or ↓	Selection moves down	[ ]		
2.2	Move up	k or ↑	Selection moves up	[ ]		
2.3	Next panel	Tab or ⏎	Focus moves to next panel	[ ]		
2.4	Previous panel	h	Focus moves to previous panel	[ ]		
2.5	Jump to panel 1	1	Status panel focused	[ ]		
2.6	Jump to panel 2	2	Files panel focused	[ ]		
2.7	Jump to panel 3	3	Branches panel focused	[ ]		
2.8	Jump to panel 4	4	Commits panel focused	[ ]		
2.9	Jump to panel 5	5	Stash panel focused	[ ]		
2.10	Focus main panel	0	Main panel gets focus, j/k scrolls it	[ ]		
2.11	Unfocus main panel	0 again	Left panel re-focused	[ ]		
2.12	Page down	PgDn	Selection jumps ~10 items down	[ ]		
2.13	Page up	PgUp	Selection jumps ~10 items up	[ ]		
2.14	Refresh	r	All panels refresh data	[ ]		
2.15	Help screen	?	Full help screen displayed	[ ]		
2.16	Close help	? or Esc	Help screen closes	[ ]		
2.17	Quit	q	Gilt exits cleanly, terminal restored	[ ]		

## 3. Files Panel — Staging

#	Test	Key	Expected	Result	Auto	Notes
3.1	Stage a file	Space on unstaged	File moves to staged, indicator changes	[ ]	[Y]	git add verified
3.2	Unstage a file	Space on staged	File moves to unstaged	[ ]	[Y]	git reset verified
3.3	Stage all	a	All files staged, cmdlog shows “git add -A”	[ ]		
3.4	Unstage all	a again	All files unstaged, cmdlog shows “git reset HEAD”	[ ]		

#	Test	Key	Expected	Result	Auto	Notes
3.5	Diff updates on select	j/k	Main panel diff changes per selected file	[ ]	[Y]	Diff output verified

#### 4. Files Panel — Commit

#	Test	Key	Expected	Result	Auto	Notes
4.1	Open commit dialog	c	Dialog with text input appears	[ ]		
4.2	Type message	type text	Text appears in input field	[ ]		
4.3	Confirm commit	Tab → Enter	Commit created, panels refresh	[ ]		
4.4	Cancel commit	Escape	Dialog closes, no commit	[ ]		
4.5	Commit with editor	C	TUI suspends, \$EDITOR opens, commit on save	[ ]		
4.6	Commit no-hook	w	Commit bypasses pre-commit hook	[ ]		

#### 5. Files Panel — File Operations

#	Test	Key	Expected	Result	Auto	Notes
5.1	Discard changes	d	Confirmation dialog, file reverted on confirm	[ ]		
5.2	Edit file	e on conflict	\$EDITOR opens with file	[ ]		
5.3	Hunk staging	e on modified	Main panel shows hunks, Space stages hunk	[ ]		
5.4	Blame view	b	Main panel shows blame: hash, author, line	[ ]	[Y]	git blame verified
5.5	Blame commit detail	Enter in blame	Dialog shows commit details for line	[ ]		
5.6	Exit blame	Escape	Returns to normal view	[ ]		
5.7	File tree toggle	T	Files grouped by directory with tree indent	[ ]	[Y]	Dir structure verified
5.8	Tree toggle back	T again	Flat file list restored	[ ]		

#	Test	Key	Expected	Result	Auto	Notes
5.9	Range select start	v	Cmdlog shows "Range select started at N"	[ ]		
5.10	Range select end	v again	Range of files staged/unstaged, cmdlog shows count	[ ]		
5.11	Ignore file	I	Dialog asks to add to .gitignore, confirm adds it	[ ]	[Y]	.gitignore verified
5.12	Verify .gitignore	—	Check .gitignore contains the file	[ ]	[Y]	Pattern verified
5.13	Copy file path	y	Cmdlog shows "Copied file path: ..."	[ ]	[Y]	Clipboard verified
5.14	Verify clipboard	paste	Pasted text matches file path	[ ]	[Y]	Round-trip verified
5.15	External diff tool	x	TUI suspends, difftool opens, TUI restores	[ ]	[Y]	Difftool config verified

## 6. Files Panel — View Cycling

#	Test	Key	Expected	Result	Auto	Notes
6.1	Cycle to Worktrees	w	Panel title changes to "Worktrees"	[ ]		
6.2	Cycle to Stashes	w	Panel title changes to "Stashes"	[ ]		
6.3	Cycle back to Files	w	Panel title changes back to "Files"	[ ]		
6.4	Stash apply	Enter on stash	Stash applied, files appear	[ ]		
6.5	Stash pop	P on stash	Stash applied and removed from list	[ ]	[Y]	Stash pop verified
6.6	Stash drop	D on stash	Confirmation, stash removed	[ ]		
6.7	Stash rename	R on stash	Input dialog, stash renamed	[ ]		
6.8	Branch from stash	B on stash	Input dialog, new branch created	[ ]		
6.9	Stash diff	select stash	Main panel shows stash diff	[ ]		

## 7. Branches Panel — Local

#	Test	Key	Expected	Result	Auto	Notes
7.1	Checkout branch	Enter	Branch checked out, status updates	[ ]	[Y]	Branch checkout verified Branch create verified
7.2	New branch	n	Input dialog, branch created and checked out	[ ]	[Y]	
7.3	Rename branch	N	Input dialog, branch renamed	[ ]		
7.4	Delete branch	D	Confirmation, branch deleted	[ ]	[Y]	Branch delete verified
7.5	Merge branch	M	Dialog with Merge/Squash options	[ ]		
7.6	Squash merge	M → Squash	Squash merge performed	[ ]		
7.7	Rebase onto	R	Confirmation, rebase performed	[ ]		
7.8	Fast-forward	F	Branch fast-forwarded to upstream	[ ]		
7.9	Set upstream	u	Upstream set, cmdlog confirms	[ ]		
7.10	Unset upstream	u again	Upstream removed	[ ]		
7.11	Sort by name	s	Branches sorted alphabetically	[ ]		
7.12	Sort by date	s	Branches sorted by date	[ ]		
7.13	Sort by recent	s	Branches sorted by recent activity	[ ]		
7.14	Copy branch name	y	Cmdlog shows “Copied branch name: ...”	[ ]	[Y]	Clipboard verified
7.15	Open in browser	o	Browser opens branch URL	[ ]	[Y]	Browser tool verified
7.16	Create PR	0	Browser opens compare URL for PR	[ ]	[Y]	Remote URL verified
7.17	Cherry-pick from branch	C	Main panel shows branch commits, Enter picks	[ ]		

## 8. Branches Panel — Remotes

#	Test	Key	Expected	Result	Auto	Notes
8.1	Cycle to Remotes	w	Panel title changes to “Remotes”	[ ]		
8.2	Track remote branch	Enter	Local tracking branch created	[ ]		

#	Test	Key	Expected	Result	Auto	Notes
8.3	Add remote	A	Two-step dialog: name then URL	[ ]		
8.4	Rename remote	R	Input dialog, remote renamed	[ ]		
8.5	Delete remote branch	D	Confirmation, remote branch deleted	[ ]		
8.6	Fetch	f	Fetch dialog, fetches from selected remote	[ ]	[Y]	Remote URL verified

## 9. Branches Panel — Tags

#	Test	Key	Expected	Result	Auto	Notes
9.1	Cycle to Tags	w	Panel title changes to “Tags”	[ ]	[Y]	Tags exist verified
9.2	Create lightweight tag	t	Input dialog, tag created	[ ]		
9.3	Create annotated tag	t	Select “Annotated”, message input	[ ]		
9.4	Checkout tag	Space	Detached HEAD at tag	[ ]		
9.5	Push tag	T	Tag pushed to remote	[ ]		
9.6	Delete tag	D	Confirmation, tag deleted	[ ]		

## 10. Branches Panel — Submodules

#	Test	Key	Expected	Result	Auto	Notes
10.1	Cycle to Submodules	w	Panel title changes to “Submodules”	[ ]	[Y]	Submodule configured
10.2	Enter submodule	Enter	Repo context changes to submodule	[ ]		
10.3	Verify submodule	—	Status panel shows submodule name	[ ]		
10.4	Update submodule	U	Submodule updated	[ ]		

## 11. Commits Panel

#	Test	Key	Expected	Result	Auto	Notes
11.1	View commit detail	select commit	Main panel shows hash, author, date, message	[ ]	[Y]	git log verified
11.2	Search commits	/	Input dialog, results filtered	[ ]		
11.3	Exit search	Escape	Full commit list restored	[ ]		
11.4	Create tag on commit	t	Tag created on selected commit	[ ]		
11.5	Amend HEAD	A on first commit	Dialog: Amend / Amend with new message	[ ]		
11.6	Reset soft	X → Soft	Soft reset, changes preserved staged	[ ]		
11.7	Reset mixed	X → Mixed	Mixed reset, changes preserved unstaged	[ ]		
11.8	Reset hard	X → Hard	Hard reset, changes discarded	[ ]		
11.9	Fixup commit	F	Fixup commit created	[ ]		
11.10	Squash commits	S	Select range, input message, commits squashed	[ ]		
11.11	Cherry-pick	C	Confirmation, commit cherry-picked	[ ]		
11.12	Revert	R	Confirmation, revert commit created	[ ]		
11.13	Copy commit hash	y	Cmdlog shows "Copied commit hash: ..."	[ ]	[Y]	Clipboard verified
11.14	Open in browser	o	Browser opens commit URL	[ ]	[Y]	Browser tool verified

## 12. Interactive Rebase

#	Test	Key	Expected	Result	Auto	Notes
12.1	Enter rebase mode	i on 3rd+ commit	Panel title changes, actions shown	[ ]		
12.2	Pick action	p	Commit marked green "pick"	[ ]		
12.3	Reword action	r	Commit marked cyan "reword"	[ ]		
12.4	Squash action	s	Commit marked magenta "squash" (not first)	[ ]		

#	Test	Key	Expected	Result	Auto	Notes
12.5	Fixup action	f	Commit marked magenta “fixup” (not first)	[ ]		
12.6	Drop action	d	Commit marked red “drop”	[ ]		
12.7	Reorder down	J	Commit moves down in list	[ ]		
12.8	Reorder up	K	Commit moves up in list	[ ]		
12.9	Execute rebase	Enter	Rebase performed, commits updated	[ ]		
12.10	Cancel rebase	q	Rebase cancelled, original state	[ ]		

### 13. Bisect

#	Test	Key	Expected	Result	Auto	Notes
13.1	Start bisect	b on commit	Dialog: “Start bisect with X as bad?”	[ ]		
13.2	Confirm start	Start (bad)	Bisect mode active, title changes	[ ]		
13.3	Mark good	g	Bisect advances	[ ]		
13.4	Mark bad	b	Bisect advances	[ ]		
13.5	Reset bisect	Q	Bisect ended, normal mode	[ ]		

### 14. Stash Panel

#	Test	Key	Expected	Result	Auto	Notes
14.1	Create stash	s	Dialog, stash created	[ ]	[Y]	Stash push verified
14.2	Pop stash	g	Top stash applied and removed	[ ]	[Y]	Stash pop verified

### 15. Search / Filter

#	Test	Key	Expected	Result	Auto	Notes
15.1	Filter files	/ on files panel	Input dialog, files filtered	[ ]		
15.2	Filter shows in title	—	Panel title shows “(filter: query)”	[ ]		



#	Test	Key	Expected	Result	Auto	Notes
15.3	Clear filter	/ → Clear	All files shown again	[ ]		
15.4	Filter branches	/ on branches panel	Branches filtered	[ ]		
15.5	Filter stashes	/ on stash panel	Stashes filtered	[ ]		
15.6	Search commits	/ on commits panel	Commits searched by message/author	[ ]		

## 16. Diff Options

#	Test	Key	Expected	Result	Auto	Notes
16.1	Increase context	}	Diff shows more context lines	[ ]		
16.2	Decrease context	{	Diff shows fewer context lines	[ ]		
16.3	Toggle whitespace	w	Whitespace changes hidden/shown	[ ]		

## 17. Push / Pull / Fetch

#	Test	Key	Expected	Result	Auto	Notes
17.1	Push	P	Push to remote, cmdlog shows result	[ ]	[Y]	Remote verified
17.2	Pull	p	Pull from remote, cmdlog shows result	[ ]	[Y]	Remote verified
17.3	Fetch	f on branches	Fetch dialog, fetches from remote	[ ]	[Y]	Remote verified

## 18. Undo / Redo

#	Test	Key	Expected	Result	Auto	Notes
18.1	Undo	z	Last git command undone via reflog	[ ]		
18.2	Redo	Z	Undone command re-applied	[ ]		

## 19. Merge Conflict Resolution

**Setup:** git merge conflict-branch from master (or use Gilt's M key)

#	Test	Key	Expected	Result	Auto	Notes
19.1	Conflict detected	—	Status shows MERGING, files show !	[ ]	[Y]	Conflict branch verified
19.2	Keep ours	o on conflict	File resolved with our version	[ ]		
19.3	Keep theirs	t on conflict	File resolved with their version	[ ]		
19.4	Edit conflict	e on conflict	\$EDITOR opens with conflict markers	[ ]		
19.5	Abort merge	X	Merge aborted, clean state	[ ]		

## 20. Config Viewer

#	Test	Key	Expected	Result	Auto	Notes
20.1	Open config	G	Main panel shows git config entries	[ ]		
20.2	Cycle scope	w	Filters: All → Local → Global → System	[ ]		
20.3	Navigate entries	j/k	Selection moves through config	[ ]		
20.4	Close config	G or Esc	Normal view restored	[ ]		

## 21. Clipboard & Browser

#	Test	Key	Expected	Result	Auto	Notes
21.1	Copy file path	y on files	Clipboard has file path	[ ]	[Y]	Clipboard verified
21.2	Copy branch name	y on branches	Clipboard has branch name	[ ]	[Y]	Clipboard verified
21.3	Copy commit hash	y on commits	Clipboard has short hash	[ ]	[Y]	Clipboard verified
21.4	Open commit URL	o on commits	Browser opens commit page	[ ]	[Y]	Browser tool verified
21.5	Open branch URL	o on branches	Browser opens branch page	[ ]	[Y]	Browser tool verified

## 22. Shell Command

#	Test	Key	Expected	Result	Auto	Notes
22.1	Open shell dialog	:	Input dialog appears	[ ]		
22.2	Run command	type ls → Enter	Output shown in cmdlog	[ ]		
22.3	Run git command	type git log -- oneline -3	Output shown	[ ]		

## 23. Screen Mode

#	Test	Key	Expected	Result	Auto	Notes
23.1	Cycle to half	+	Left panels take ~50% width	[ ]		
23.2	Cycle to full	+	Left panels take nearly full width	[ ]		
23.3	Cycle to normal	+	Left panels back to ~40% width	[ ]		

## 24. Recent Repos

#	Test	Key	Expected	Result	Auto	Notes
24.1	Show recent repos	L	Main panel shows recent repo list	[ ]		
24.2	Config file exists	—	~/.config/gilt/recent-repos.txt has entries	[Y]		Config file verified

## 25. Git-Flow

**Prerequisite:** git flow must be installed (apt install git-flow)

#	Test	Key	Expected	Result	Auto	Notes
25.1	Open git-flow menu	E	Dialog with flow options	[ ]	[Y]	git-flow available
25.2	Init	Init button	git flow initialized	[ ]		

#	Test	Key	Expected	Result	Auto	Notes
25.3	Feature start	Feature Start	Input dialog, feature branch created	[ ]		
25.4	Feature finish	Feature Finish	Feature branch merged and deleted	[ ]		

## 26. Create Pull Request

**Prerequisite:** Remote must be configured (GitHub/GitLab)

#	Test	Key	Expected	Result	Auto	Notes
26.1	Create PR	0 on branch	Browser opens compare URL	[ ]	[Y]	Remote + browser verified
26.2	URL format correct	—	URL is repo/compare/branch?expand=1	[ ]		

## 27. Custom Commands

**Setup:** Edit ~/.config/gilt/commands.conf, add: 9=git log --oneline -5

#	Test	Key	Expected	Result	Auto	Notes
27.1	Custom command runs	9	Cmdlog shows \$ git log --oneline -5 + output	[ ]	[Y]	Config file verified
27.2	Unbound key no-op	unused key	Nothing happens	[ ]		

## 28. Ignore File

#	Test	Key	Expected	Result	Auto	Notes
28.1	Ignore file	I on file	Confirmation dialog	[ ]		
28.2	Confirm ignore	Ignore button	File added to .gitignore	[ ]	[Y]	.gitignore verified
28.3	File disappears	r	Ignored file no longer in files list	[ ]		

## 29. External Diff Tool

**Prerequisite:** git difftool configured (e.g. git config diff.tool vimdiff)

#	Test	Key	Expected	Result	Auto	Notes
29.1	Launch difftool	x on modified file	TUI suspends, difftool opens	[ ]	[Y]	Difftool configured
29.2	Return to gilt	exit difftool	TUI restores cleanly	[ ]		

## 30. Enter Submodule

#	Test	Key	Expected	Result	Auto	Notes
30.1	Navigate to submodules	3 → w x3	Submodules view shown	[ ]	[Y]	Submodule configured
30.2	Enter submodule	Enter	Repo context changes, status shows submodule	[ ]		
30.3	Submodule data loads	—	Commits, files, branches for submodule	[ ]		

## Summary

Category	Total	Auto [Y]	Manual Only	Pass	Fail	Skip
Startup & Display	10	7	3			
Navigation	17	0	17			
Staging	5	3	2			
Commit	6	0	6			
File Operations	15	7	8			
View Cycling	9	1	8			
Branches Local	17	6	11			
Branches Remotes	6	1	5			
Tags	6	1	5			
Submodules	4	1	3			
Commits Panel	14	3	11			
Interactive Rebase	10	0	10			
Bisect	5	0	5			
Stash Panel	2	2	0			
Search/Filter	6	0	6			
Diff Options	3	0	3			
Push/Pull/Fetch	3	3	0			
Undo/Redo	2	0	2			
Merge Conflicts	5	1	4			
Config Viewer	4	0	4			
Clipboard/Browser	5	5	0			

Category	Total	Auto [Y]	Manual Only	Pass	Fail	Skip
Shell Command	3	0	3			
Screen Mode	3	0	3			
Recent Repos	2	1	1			
Git-Flow	4	1	3			
Create PR	2	1	1			
Custom Commands	2	1	1			
Ignore File	3	1	2			
External Diff	2	1	1			
Enter Submodule	3	1	2			
<b>TOTAL</b>	<b>177</b>	<b>48</b>	<b>129</b>			

## Notes & Issues

*Use this section to record any bugs, unexpected behavior, or improvement ideas found during testing.*

### Bugs Found

#	Test ID	Description	Severity	Fixed?

### Improvement Ideas

#	Description	Priority