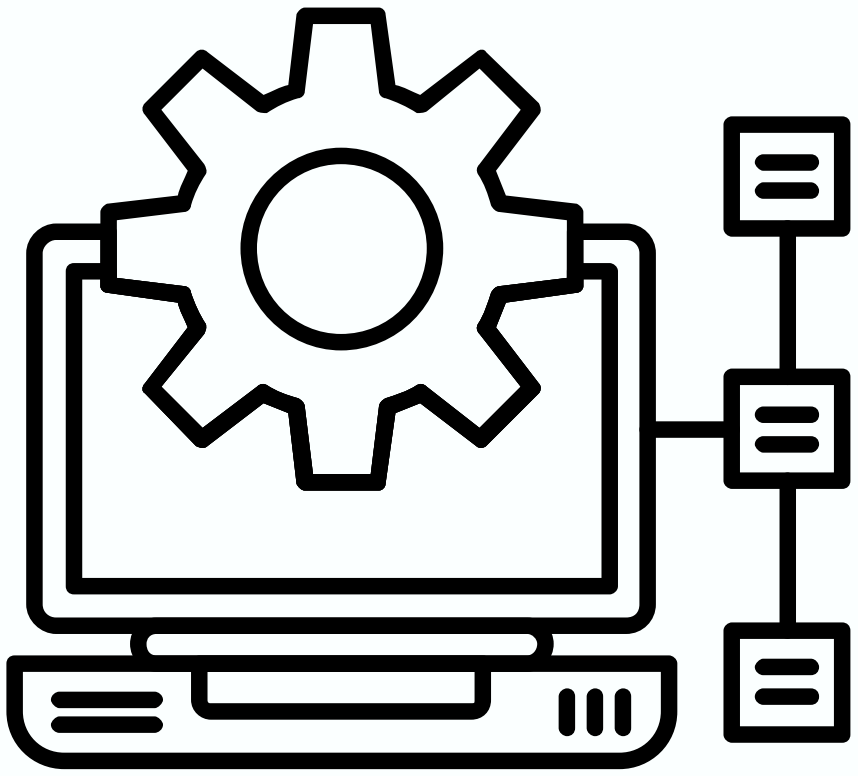
DOHE



Objective 3.2

Value Proposition

Set Up Basic Product Systems

Establish simple, organized systems for building, testing, and improving your product efficiently while capturing feedback and learning fast.

Early Stage/MVP

Objective Description

Put simple tools and routines in place to help you build, test, and improve your product without chaos. This means deciding where your code lives, how you track updates or bugs, and how you collect feedback from users. The goal is to stay organized so you can move fast and learn quickly.

Why This Objective Matters For Your Startup

Strong systems are what turn a good idea into a reliable, evolving product. Simple tools and routines — for tracking progress, managing updates, and collecting feedback — keep your team organized, efficient, and responsive. Setting these foundations early prevents chaos as your product and team grow.

Improves Organization and Visibility – Centralizing your code, tasks, and bugs ensures everyone knows what's being worked on and why it matters.

Increases Development Speed – Structured workflows and version control make it easier to release updates regularly without disruption.

Reduces Errors and Downtime – A clear process for testing and tracking issues improves product reliability and user trust.

Connects Feedback to Action – Linking user insights directly to your roadmap ensures improvements are based on real needs, not guesswork.

Builds a Scalable Foundation – A simple but well-structured system makes it easier to onboard new team members and expand operations smoothly.

Key Concepts & Resources

- Organization: Use simple tools to manage product builds, bugs, and updates.
- Consistency: Establish routines for releases and continuous improvement.
- Feedback Integration: Turn user insights into fast, actionable product updates.
- Speed with Stability: Stay agile while keeping processes reliable and transparent

Agile Product Development Framework

Adopt lightweight Agile principles to stay flexible and responsive.

- Break work into short sprints (1–2 weeks) with clear goals and deliverables.
- Review progress frequently and adapt plans based on results.
- Encourage cross-functional collaboration to move faster and learn continuously.

Agile ensures steady improvement, faster iteration, and better alignment with user needs.

https://www.youtube.com/watch?v=Z9QbYZh1YXY

Continuous Integration / Continuous Deployment (CI/CD) Framework

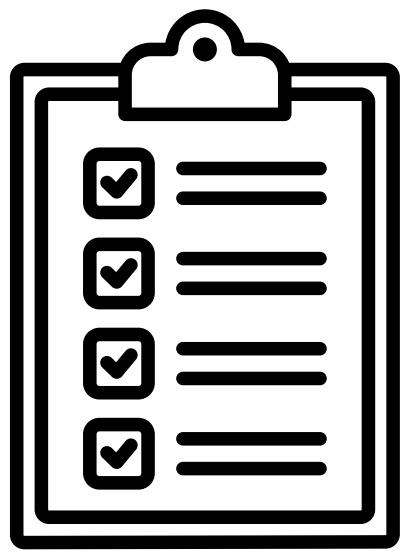
Streamline how you build, test, and release updates.

- Automate code integration, testing, and deployment to reduce errors and save time.
- Use simple tools like GitHub Actions or Jenkins to manage builds.
- Release small, frequent updates for faster feedback loops.

CI/CD helps teams maintain product quality while delivering updates quickly and efficiently.

https://www.youtube.com/watch?v=scEDHsr3APg

Objective Action List



Tick each action as you complete it during your 20-week accelerator programme.

1. I selected and set up one main place to store product files or code (e.g., GitHub, Google Drive, Notion) and shared access with all team members.

2. I created a simple tracker (e.g., Trello, Notion board, Google Sheet) with three columns — To Do, Doing, Done — and added all current product tasks within 2 days.

3. I made a clear list or form for reporting product bugs or issues and tested it by logging at least 3 sample bugs to ensure it worked smoothly.

4. I confirmed that everyone on the team could open, edit, and update product tools without confusion (target: 100% confirmed access.

5. I agreed with the team on a short update cycle (e.g., every 1–2 weeks) and added it to our shared calendar?

6. I created a simple “Release Log” where we recorded each fix or feature added — including date, person responsible, and short summary.

7. I tested each release on at least one real device or environment before calling it complete (e.g., web, mobile).

8. After every 2–3 releases, I reviewed what went well and what slowed us down, and listed at least two improvements for the next cycle.

9. I set up one place (e.g., Google Form, Typeform, Intercom chat, or Notion page) to collect user feedback and tested it with at least 5 users).

10. I connected every piece of feedback to the task tracker and marked whether it became a “Fix,” “Feature,” or “Idea,” ensuring no feedback was lost or ignored.