

MODUL PRAKTIKUM LINEAR.APP

Step-by-step dari nol sampai project siap dipresentasikan

Manajemen Proyek Perangkat Lunak secara Agile

Studi kasus: Pengembangan Aplikasi Pemesanan Makanan Kampus

URL: <https://linear.app/>

Target akhir: mahasiswa mampu membuat project, issue, workflow, cycle, dependency, dan memantau progress tim.

Capaian Pembelajaran

Mahasiswa dapat membuat dan mengatur Workspace/Team.

Mahasiswa dapat membuat Project dan mendefinisikan milestone.

Mahasiswa dapat mengubah requirement menjadi Issue yang terukur.

Mahasiswa dapat menggunakan status, priority, label, assignee, dan due date.

Mahasiswa dapat membuat Cycle/Sprint dan mengatur backlog.

Mahasiswa dapat mengelola dependency, sub-issue, dan progress.

Mahasiswa dapat menjelaskan hubungan Linear dengan Git dan AI.

Persiapan Praktikum

Siapkan akun email untuk login Linear.

Bentuk kelompok 4–6 mahasiswa.

Tentukan 1 orang sebagai Project Lead/Coordinator.

Tentukan minimal: Product Owner, Developer, UI/UX, QA/Tester.

Siapkan nama aplikasi yang akan dikembangkan.

Contoh studi kasus: 'CampusFood – Aplikasi Pemesanan Makanan Kampus'.

STEP 1 — Membuka Linear

1. Buka browser.
2. Masuk ke <https://linear.app/>
3. Pilih Sign up atau Log in.
4. Gunakan akun yang tersedia.
5. Setelah masuk, perhatikan sidebar kiri dan area workspace.

OUTPUT: mahasiswa berhasil masuk ke Linear.

STEP 2 — Membuat Workspace

1. Pada saat onboarding, buat workspace baru jika diperlukan.
2. Masukkan nama workspace, contoh: 'Kelompok 03 – CampusFood'.
3. Pilih pengaturan yang sesuai kebutuhan tim.
4. Selesaikan onboarding.
5. Pastikan workspace tampil di bagian atas/sidebar.

OUTPUT: 1 workspace untuk satu kelompok.

STEP 3 — Membuat / Memilih Team

1. Buka menu Team.
2. Buat team baru jika belum tersedia.
3. Nama team: 'CampusFood Engineering'.
4. Tentukan identifier/key team sesuai pilihan, misalnya CF.
5. Tambahkan anggota kelompok.

OUTPUT: satu team berisi seluruh anggota proyek.

STEP 4 — Menentukan Role Tim

Product Owner → menentukan kebutuhan dan prioritas.

Project Lead/Scrum Master → mengatur pekerjaan dan cycle.

Developer → implementasi fitur.

UI/UX → desain antarmuka dan pengalaman pengguna.

QA/Tester → pengujian dan pelaporan bug.

Catatan: satu mahasiswa boleh memegang lebih dari satu peran.

STEP 5 — Membuat PROJECT

1. Masuk ke Projects.
2. Pilih Create Project.
3. Nama: 'CampusFood – Aplikasi Pemesanan Makanan Kampus'.
4. Tambahkan deskripsi singkat.
5. Tentukan lead dan anggota.
6. Hubungkan project dengan Team yang sudah dibuat.

OUTPUT: satu project aktif.

STEP 6 — Menentukan Scope Project

Tuliskan outcome project, contoh:

'Membangun aplikasi yang memungkinkan mahasiswa melihat menu, memesan makanan, membayar, dan memantau status pesanan.'

Tentukan batasan: fitur yang dikerjakan dan fitur yang tidak dikerjakan.

Tentukan target release/prototipe.

OUTPUT: scope project jelas dan dapat dipahami seluruh anggota.

STEP 7 — Membuat MILESTONE

Buat minimal 4 milestone:

M1 — Requirement & Analysis

M2 — UI/UX Design

M3 — Development

M4 — Testing & Release

Jika versi Linear yang digunakan menyediakan milestone pada project, gunakan fitur tersebut untuk mengelompokkan target.

STEP 8 — Memahami ISSUE

Issue = satu unit pekerjaan yang harus dikerjakan.

Contoh Feature: 'Mahasiswa dapat login'.

Contoh Task: 'Membuat tabel users'.

Contoh Bug: 'Tombol checkout tidak merespons'.

Issue sebaiknya spesifik, memiliki hasil yang jelas, dan dapat ditugaskan kepada anggota tertentu.

STEP 9 — Membuat ISSUE Pertama

1. Buka project/team.
2. Pilih New Issue / Create Issue.
3. Title: 'Membuat halaman Login'.
4. Description: tuliskan tujuan dan kriteria selesai.
5. Pilih Assignee.
6. Pilih Priority.
7. Tambahkan Label, misalnya Frontend.
8. Hubungkan dengan Project.

OUTPUT: issue pertama berhasil dibuat.

STEP 10 — Menulis Description Issue

Gunakan format sederhana:

Tujuan: pengguna dapat masuk ke aplikasi.

Input: email/username dan password.

Proses: validasi kredensial.

Output: pengguna masuk ke dashboard.

Acceptance Criteria: login berhasil; login gagal menampilkan pesan kesalahan.

Tujuan: developer dan tester memahami pekerjaan tanpa penjelasan berulang.

STEP 11 — Priority, Label & Assignee

Priority: tentukan tingkat kepentingan pekerjaan.

Label: kelompokkan berdasarkan tipe/area, misalnya Frontend, Backend, Database, UIUX, Testing, Bug.

Assignee: siapa yang bertanggung jawab.

Jangan memberi semua issue kepada satu orang.

OUTPUT: setiap issue memiliki pemilik dan konteks yang jelas.

STEP 12 — Membuat Backlog

Buat minimal 15 issue untuk CampusFood.

Contoh kelompok:

- Requirement: 2 issue
- UI/UX: 3 issue
- Database: 2 issue
- Backend/API: 3 issue
- Frontend: 3 issue
- Testing/Bug: 2 issue

OUTPUT: backlog minimal 15 issue.

Contoh Backlog CampusFood

CF-01 — Analisis kebutuhan pengguna

CF-02 — Membuat use case

CF-03 — Wireframe halaman login

CF-04 — Wireframe katalog makanan

CF-05 — Desain halaman checkout

CF-06 — Membuat database users

CF-07 — Membuat database products

CF-08 — API autentikasi

CF-09 — API katalog makanan

CF-10 — API pemesanan

CF-11 — Frontend login

CF-12 — Frontend katalog

CF-13 — Frontend checkout

CF-14 — Testing login

CF-15 — Testing checkout

STEP 13 — Mengatur WORKFLOW

Gunakan status workflow yang tersedia pada workspace/team.

Contoh alur praktikum:

Backlog → Todo → In Progress → In Review → Done

Pindahkan issue sesuai kondisi sebenarnya.

Jangan langsung mengubah issue menjadi Done sebelum acceptance criteria terpenuhi.

OUTPUT: mahasiswa memahami Kanban/workflow.

STEP 14 — Membuat SUB-ISSUE

Gunakan sub-issue untuk pekerjaan yang terlalu besar.

Contoh parent: 'Membuat fitur Checkout'.

Sub-issue:

1. Membuat desain checkout
2. Membuat tabel orders
3. Membuat API checkout
4. Membuat halaman checkout
5. Testing checkout

OUTPUT: satu pekerjaan besar terpecah menjadi pekerjaan yang dapat ditangani.

STEP 15 — Mengatur DEPENDENCY

Tentukan hubungan antar-issue.

Contoh: 'Frontend Checkout' membutuhkan 'API Checkout'.

Gunakan relasi dependency/blocked by yang tersedia.

Tujuannya agar tim mengetahui pekerjaan yang belum dapat dimulai.

OUTPUT: minimal 3 dependency terdokumentasi.

STEP 16 — Membuat CYCLE / SPRINT

1. Buka menu Cycles pada Team.
2. Buat Cycle 1.
3. Tentukan periode, misalnya 1–2 minggu.
4. Masukkan issue yang realistis dikerjakan.
5. Pastikan beban kerja anggota seimbang.
6. Selama cycle, update status issue.

OUTPUT: Cycle 1 aktif dengan issue terpilih.

STEP 17 — Daily Update Progress

Setiap anggota memperbarui issue yang dikerjakan.

Jika mulai bekerja → In Progress.

Jika menunggu review → In Review.

Jika selesai dan memenuhi kriteria → Done.

Jika terhambat → tuliskan dependency/blocker.

Tujuan: project lead dapat melihat kondisi proyek tanpa meminta laporan manual setiap saat.

STEP 18 — Review Cycle

Pada akhir Cycle 1:

1. Lihat issue yang Done.
2. Identifikasi issue yang belum selesai.
3. Catat alasan pekerjaan tertunda.
4. Evaluasi pembagian workload.
5. Pindahkan pekerjaan yang masih relevan ke cycle berikutnya sesuai proses tim.
6. Tentukan perbaikan untuk Cycle 2.

STEP 19 — Integrasi GitHub/GitLab

Jika tersedia pada workspace:

1. Hubungkan repository dengan Linear.
2. Hubungkan issue dengan branch/pull request menggunakan ID issue.
3. Developer mengerjakan kode pada repository.
4. Pull request direview.
5. Setelah pekerjaan selesai, status issue diperbarui.

OUTPUT: pekerjaan coding dapat ditelusuri dari issue ke repository.

STEP 20 — Menggunakan Linear + AI

Eksplorasi fitur AI/Agents yang tersedia pada akun/workspace.

Contoh penggunaan pembelajaran:

- Membantu merangkum update project.
- Membantu membuat atau mengolah issue.
- Membantu memahami pekerjaan yang belum selesai.
- Eksplorasi integrasi AI/MCP jika tersedia.

Catatan: fitur AI dapat berbeda menurut paket, konfigurasi workspace, dan versi Linear.

Praktikum Terpadu — Dari Requirement ke Done

Requirement: 'Mahasiswa dapat memesan makanan melalui aplikasi.'



Breakdown → UI/UX + Database + API + Frontend + Testing



Buat Issue → Assign → Priority → Label



Masukkan ke Cycle



Kerjakan → Review → Testing → Done



Pantau progress Project

Checklist Praktikum

- ☐ Workspace dibuat
- ☐ Team dibuat
- ☐ Anggota dan role ditentukan
- ☐ Project dibuat
- ☐ Minimal 4 milestone dibuat
- ☐ Minimal 15 issue dibuat
- ☐ Label dan priority digunakan
- ☐ Workflow digunakan
- ☐ Minimal 1 Cycle dibuat
- ☐ Minimal 3 dependency/sub-issue dibuat
- ☐ Progress project dapat ditunjukkan

Tugas & Output Pengumpulan

1. Screenshot Workspace + Team.
2. Screenshot Project + milestone.
3. Screenshot backlog 15 issue.
4. Screenshot Kanban/workflow.
5. Screenshot Cycle.
6. Screenshot dependency/sub-issue.
7. Laporan PDF 3–5 halaman berisi proses dan evaluasi penggunaan Linear.
8. Presentasi kelompok: 5–10 menit.

Nama file: Kelompok##_Praktikum_Linear.pdf

Rubrik Penilaian

Workspace & Team — 10%

Project & milestone — 15%

Kualitas backlog/issue — 25%

Workflow & Cycle — 15%

Dependency/sub-issue — 10%

Kerapian dan konsistensi pengelolaan proyek — 10%

Laporan & presentasi — 15%

Total — 100%

Troubleshooting Praktikum

Menu yang terlihat dapat berbeda tergantung versi UI, role, workspace, dan paket Linear.

Jika tidak dapat membuat fitur tertentu, gunakan fitur inti Issue/Project/Cycle terlebih dahulu.

Jika integrasi Git tidak tersedia, praktikkan alur issue secara manual dan dokumentasikan.

Jika fitur AI tidak tersedia pada akun, gunakan Linear untuk workflow inti; AI menjadi materi eksplorasi lanjutan.

Selalu gunakan dokumentasi resmi Linear untuk perubahan fitur terbaru.

Referensi Resmi

Linear: <https://linear.app/>

Start Guide: <https://linear.app/docs/start-guide>

Documentation: <https://linear.app/docs>

Learn: <https://linear.app/learn>

AI: <https://linear.app/ai>

Catatan: instruksi menu pada modul dapat berubah mengikuti pembaruan aplikasi.