Project Overview: **SkillBridge**

**Overview:**

**SkillBridge is a modern freelance marketplace designed to connect skilled professionals with businesses and individuals who need their expertise. Unlike traditional platforms, SkillBridge emphasizes transparency, skill verification, and user-friendly design tailored for both clients and freelancers. Our MVP will include essential features like registration, job posting, bidding, and messaging.**

**Project Details**

Name: SkillBridge

Type: Freelance Marketplace Web Application.

Team Size: 5 Members

Team Skills: Intermediate Django, Expert HTML

Duration: 7 Days

Framework: Agile (Daily Stand-ups, Iterative Development)

Version Control: GitHub

Main Languages: Django (Python), HTML, CSS (Bootstrap optional)

| **Team Member** | **Roles** | **Responsibilities** |
| --- | --- | --- |
| Project Manager | Team Lead | Planning, Stand-ups, Task Assignment, Reviewing, Testing |
| Dev 1 | Backend Lead | User auth, Profile system, Job posting APIs |
| Dev 2 | Backend Support | Proposal system, messaging, notifications |
| Dev 3 | Frontend Lead | HTML screens, page styling, user flows, Logo Creation |
| Dev 4 | Frontend Support | Forms, layout integration, UX fixes, Responsiveness |

Main Features(must - haves)

**User System**

* Sign up / Login (Client & Freelancer)
* User Profiles with editable info
* Skill tags and brief portfolio section

**Job Posting & Proposals**

* Clients post job listings
* Freelancers browse & send proposals
* Proposal listing per job with status

**Hiring**

* Client can “Accept” a proposal
* Job status updates to “In Progress” or “Completed”

**Dashboard**

* Separate dashboards for client & freelancer

Optional  
**Messaging**

* Private chat between clients and freelancer
* E.t.c

1. Day Agile Development Plan

* **Day 1 – User Auth & Profile**

Objective:

Implement login/registration and create distinct user profiles (Client & Freelancer).

1. Backend: Django authentication system
2. Create models for User, ClientProfile, FreelancerProfile
3. Front-end: Login, Register, Profile pages (static)
4. Testing login/register flow

✅ Milestones:

1. Create custom user model (email login)
2. Setup authentication views & routes (login, logout, register)
3. Create ClientProfile and FreelancerProfile models
4. Build profile creation/edit pages (HTML + backend forms)
5. Test user signup flow for both account types
6. Integrate profile info into dashboard template

* **Day 2 – Job Posting System**

Objective:

Allow clients to create, view, and manage job listings. Freelancers can browse available jobs.

1. Models: Job, Category, JobTag
2. Client dashboard to post/edit/delete jobs
3. Job detail view (backend + frontend)
4. Freelancer job listing page with filters

✅ Milestones:

1. Create Job model (title, description, budget, category, tags)
2. Client can post/edit/delete jobs from their dashboard
3. Create job listing and job detail pages
4. Freelancers can view all jobs (public job board)
5. Filter jobs by keyword/category (basic)

* **Day 3 – Proposal/Bidding System**

Objective:

Build a basic messaging system between freelancers and clients.

1. Proposal model (Job ↔ Freelancer)
2. Freelancers can submit proposal form
3. Clients view proposals on their jobs
4. Accept/Reject proposals
5. Front-end pages for viewing and sending proposals

✅ Milestones:

1. Create Proposal model (job, freelancer, description, price, status)
2. Add proposal submission form on job detail page (if freelancer)
3. Client dashboard displays proposals per job
4. Accept/Reject proposals (status update logic)
5. Display proposal status on freelancer dashboard

* **Day 4 – Messaging System**

Objective:

Build a basic messaging system between freelancers and clients.

1. Message model (User ↔ User, related to Job)
2. Basic chat interface (HTML only)
3. Integrate with jobs & proposals (only visible if proposal accepted)
4. Styling updates

✅ Milestones:

1. Create Message model (sender, receiver, job, message\_text, timestamp)
2. Restrict messaging to accepted proposals
3. Build conversation view per job
4. Style messaging page to show sent/received messages
5. Display notifications or message previews in dashboards

* **Day 5 – Hiring Workflow + Status**

Objective:

Track hiring status and allow both users to update job progress.

1. Client marks proposal as accepted → job status = “In Progress”
2. Freelancer marks job as completed
3. Add status tracking on dashboard (e.g., “Pending”, “In Progress”, “Completed”)
4. HTML UI for status updates

✅ Milestones:

1. Add job status: “Open”, “In Progress”, “Completed”
2. Client marks proposal as “Hired” → job status changes
3. Freelancer marks job as “Completed”
4. Dashboard cards reflect job/proposal status
5. Add visual status indicators (badges or labels)

* **Day 6 – Testing, Styling & Dashboard**

Objective:

Test all features, fix bugs, and polish UI for usability.

1. Manual testing of all flows (auth, posting, proposals, messaging)
2. Frontend polish: fix layout, responsiveness
3. Enhance dashboard views (summaries, action buttons)
4. Add error handling (e.g., for empty states)

✅ Milestones:

1. Test full user flows (register → post job → proposal → message → hire)
2. Fix layout issues, broken links, and validation errors
3. Improve dashboard UI (e.g., summary counts, action buttons)
4. Add error messages, success alerts
5. Optimize responsiveness for mobile/tablets

* **Day 7 – Final Review & Documentation**

Objective:

Finalize the project, review with team, and prepare for submission/demo.

1. Team bug hunt & final UI review
2. Write README.md, include:
   1. App features
   2. Setup guide
   3. User login credentials (if seeded)
3. Team demo walkthrough
4. Deliver final version

✅ Milestones:

1. Team-wide bug hunt and final edits
2. Update README with setup instructions, features, and user roles
3. Add seed data or dummy accounts for demo
4. Document any remaining known issues or limitations
5. Run through live demo walkthrough as a team

Final Notes

* Give updates and reports daily in the Whatsapp group.
* It is Most Essential to hit ALL the daily Milestones.
* Use GitHub issues for task tracking.
* Label every edited commit on GitHub.
* Use a Virtual Environment when coding.
* Stick to the main features first– skip complex features like reviews or real payments for now.
* Prioritize completing functional flows over perfect styling.
* Any Further Questions pertaining to this Charter should be taken to the Project Manager