Hustlr - (Sprint 3)

✅ Sprint Backlog

Task	Owner	Status	Due Date	Story Points
Refactor UI for Home page	Front-End	Completed	04/05/25	4
Design UI for Admin job retrieval and deletion	Front-End	Completed	30/04/25	4
Design UI for Accepted Jobs and Rejected Jobs	Front-End	Completed	19/04/25	5
Implement Recent Activities Page	Team	Completed	30/04/25	4
Implement Accepted Jobs and Rejected Jobs Logic	Team	Completed	03/05/25	6
Backend endpoint for applications	Back-end	Completed	30/04/25	5

Store Applications in DB	Back-end	Completed	30/04/25	3
Fetch Applications in DB	Back-end	Completed	01/05/25	3
Creation of UML diagrams	Team	Completed	30/04/25	2
Implement Logic for Freelancer applications and validation	Team	Completed	02/05/25	5
Deploy on Azure	Front-End	Completed	04/05/25	6
UATs in correct format, code coverage tool integrated on GitHub	Back-end	Completed	03/05/25	5
60% Code coverage	Team	Completed	04/05/25	7
Implement Logic for milestones.	Team	Completed	29/05/25	5
Implement Logic for “Payments”	Team	Pending	30/04/25	6

📱 Sprint User Stories

User Story 1: Create Milestones as Client ✨

Who : Client

What: Add milestone to a listing

Why : Ensures freelancers have knowledge of tasks

Acceptance Criteria

- ✓ **Given**: the client is on the jobs page. **When:** they fill in the job form with milestones.
  Then: the job gets posted with milestones.

User Story 2: Apply to jobs as Freelancer 🔐

Who : Freelancer

What : Can apply to job listings

Why : Ensures Client knows there's potential applicants

Acceptance Criteria

- ✓ **Given:** a freelancer is logged in and navigates to the job listings page. **When:** the job they want is shown **Then:** they click 'apply' to apply to the job

User Story 3: Client can access Applicants🔐

Who : Client

What : I want to view applicants to a job

Why : Ensures Client can keep track of applicants

Acceptance Criteria

- ✓ **Given:** the client is on their job dashboard. **When:** they click the view button on a job **Then:** they are show current applicants

---

User Story 4: Accept Applicant as Client 👨‍💼

Who : Client

What: Accept an applicant listing with details like budget, deadline, and skills required.

Why : Ensures a client can accept an applicant for a job

Acceptance Criteria

- ✓ **Given:** the client is on their job listings page. **When:** they view the list of applicants. **Then:** they can accept a particular applicant

User Story 5: Reject Applicant as Client 👨‍💼

Who : Client

What: Reject an applicant

Why : Ensures a client can reject an applicant for a job

Acceptance Criteria

- ✓ **Given:** the client is on their job listings page. **When:** they view the list of applicants . **Then:** they can reject a particular applicant.

User Story 6: View accepted jobs as a Freelancer

Who : Freelancer

What: I want to view the jobs I've been accepted for

Why : So I know what Jobs I can start working on

Acceptance Criteria

- ✓ **Given:** a freelancer accesses the jobs dashboard **When:** they view the page . **Then:** the user sees the jobs they have been accepted to do

User Story 7: View rejected jobs as a Freelancer🙋‍♂️

Who : Freelancer

What: I want to view the jobs I've been rejected for.

Why : So I know what jobs I can stop being hopeful to work on.

Acceptance Criteria

- ✓**Given:** a freelancer accesses the jobs dashboard .**When:** the page loads. **Then:** the user sees the jobs they have been rejected to do

User Story 8: View any jobs as an Admin🙋‍♂️

Who : Admin

What: I want to view all the jobs available on the platform

Why : **So I can monitor the jobs and their descriptions**

Acceptance Criteria

- ✓**Given:** an admin accesses the jobs dashboard .**When:** the page loads. **Then:** the admin sees all the jobs that have been posted.

User Story 9: Delete any jobs as an Admin 👨‍💼

Who : Admin

What: I want to be able to delete any job on the platform

Why : So I can remove any malicious or fake jobs on the platform

Acceptance Criteria

- ✓ **Given:** an admin accesses the jobs dashboard. **When:** view the page . **Then:** the admin can delete any job permanently.

Requirements Gathering

Client Page Requirements

- Should clients vet a freelancer(receive an application), or should it be first come basis?

Milestone Requirements

- What information is required for a milestone (budget, deadline, etc.)?

Admin Views

- Should admins be able to see all Jobs ?
- Should admins be able to remove any job ?

📝Product Backlog

Epic	User Story	Priority
UI Enhancements	As a user, I want a clean and navigable UI for admin and freelancer pages.	High
Application Management	As a freelancer, I want to apply to jobs and see accepted/rejected statuses.	High
Activity Monitoring	As a user, I want to view recent system activities.	Medium

Admin Tools	As an admin, I want to view and delete job postings.	Medium
CI/CD & Quality	As a developer, I want to track code coverage and ensure quality with UATs.	High
Deployment	As a user, I want the system to be live and deployed on Azure.	High
Payments	As a freelancer/client, I want to view and track payments.	High
Milestone Logic	As a project manager, I want the system to track task milestones.	Medium

🔁Sprint Retrospective

What went well:

- UI tasks were completed ahead of schedule.
- Smooth collaboration between front-end and back-end teams.
- Code coverage goal of 60% was achieved.
- Azure deployment had no blockers.

What didn't go well:

- "Payments" logic implementation was delayed.
- Slight misalignment on the due date for milestones (completed later than expected).

What can be improved:

- Early prioritization of complex logic-related tasks like payments.

📆Daily Stand-Up Summaries

Day	Summary
Day 1	Sprint kickoff; team assigned tasks; repo cloned and environment set up.
Day 2	UI refactoring started; backend endpoint design for applications planned.
Day 3	DB schema for applications finalized; activities page drafted.

Day 4	Azure deployment pipeline tested; initial UI testing.
Day 5	Applications stored in DB; UAT cases started.
Day 6	Freelancer application logic finalized; accepted/rejected job views working.
Day 7	Admin job deletion logic integrated; 60% code coverage passed.
Day 8	Final tests on milestones logic; UATs completed.
Day 9	Final tweaks; all completed tasks reviewed; payment logic pending.
Day 10	Sprint closing; deployment validated; retrospective prepared.

📈Sprint 3 Burndown Chart

Day	Goal	Done	Goal Velocity	Remaining Effort
0	0	0	75	75
1	8	7	67	68
2	8	10	59	58
3	8	8	51	50
4	8	6	43	44
5	8	8	35	36
6	8	10	27	26
7	8	8	19	18
8	8	9	11	9
9	8	8	3	1
10	1	0	0	1


Burndown - Sprint 3
80
70
60
50
40
30
20
10
0
1
2
3
4
5
6
7
8
9
10
11
Goal velocity
Remaining