

Portfolio Stage 4 Report

MVP Development and Execution
1447 (Summer 2025)

Darek – Students accommodation

| NAME |
|------------------------------|
| <i>Abdulrahman Al-Fawzan</i> |
| <i>Abdulelah Al-Shehri</i> |
| <i>Mohammad Al-Omar</i> |
| <i>Meshari Al-Abdullah</i> |

Introduction

This stage transformed the **Darek** concept into a working minimum viable product (MVP). Following the project charter and technical documentation from Stages 2 and 3, the team built a secure student-housing platform aligned with Vision 2030. Although the plan included Agile sprints, the small team adopted a flexible, availability-driven workflow. Nonetheless, they maintained iterative development, version control, and continuous testing to ensure progress and quality.

Implementation Approach

Technology Stack

- **Backend – Django:** implemented a secure, scalable REST API; deployed on **Railway** for straightforward continuous deployment.
- **Frontend – TypeScript with React:** provided responsive, bilingual user interfaces; deployed on **Netlify**.
- **Development tools:** used **PyCharm** (backend) with shared sessions for pair programming, and **VS Code** (frontend) with ESLint/Prettier for code consistency.
- **Collaboration platforms:** used **GitHub** for version control and issue tracking, **Trello** for task management, and **Discord** for daily communication and ad-hoc stand-ups.
- **Deployment platforms:** we used Railway for our back-end and Netlify for the front-end

Team Roles and Responsibilities

| Member | Role(s) | Responsibilities |
|-------------|--|---|
| Abdulrahman | Project Manager, Backend Developer | Oversaw planning and tracked tasks; implemented core API endpoints, authentication, and database models. |
| Mohammad | Backend Developer, QA Support | Developed listing, search and messaging APIs; contributed to unit and integration testing; coordinated integration with the frontend. |
| Meshari | Source Control Manager, Frontend Developer | Designed React components such as search, listing detail, chat and dashboard; enforced Git branching strategy; reviewed pull requests and ensured code quality. |
| Abdulelah | UI/UX Lead, Frontend Developer | Implemented responsive UI with; handled user acceptance testing and UX refinements (manual testing). |

The team worked on backend and frontend tasks **concurrently**. Integration occurred when major features reached stability (done testing), minimizing merge conflicts and enabling early feedback by working on the same device utilizing pair programming for better quality.

Sprint Planning and Task Breakdown

While formal two-week sprints were not strictly followed, tasks were organized into three **iterations** reflecting major milestones. A MoSCoW-like prioritization ensured critical features (Must-have) were completed first. Dependencies were identified up front; for example, the frontend listing pages depended on backend CRUD endpoints. The table below summarizes the iterations and their focus:

| Iteration | Key features (priority) | Contributors |
|-------------|---|--|
| Iteration 1 | User authentication & verification (Must); Listing CRUD & search filters (Must); basic homepage and property detail page | Abdulrahman (API); Mohammad (DB models & tests); Meshari & Abdulelah (UI) |
| Iteration 2 | Messaging / chat (Should); Roommate matching (Could); dashboards for landlords and students; deployment pipelines | Abdulrahman (API & WebSockets); Mohammad (Security); Meshari (chat UI); Abdulelah (dashboards) |
| Iteration 3 | Admin tools (Should); Review & rating system (Could); accessibility and bilingual refinements; performance optimization | Mohammad (admin APIs); Abdulrahman (reviews); Meshari & Abdulelah (UI/UX enhancements, localisation) |

Development Execution

Source Control & Branching

- Adopted **Git** with a *main* branch for production, *develop* branch for integration, and feature branches for individual tasks.
- Pull requests were peer reviewed by the Source Control Manager (must be reviewed by all 4 of us) to maintain code quality and enforce the branching strategy.
- Continuous deployment pipelines on **Railway** and **Netlify** automatically built and deployed each commit to staging environments.

Workflow & Communication

- Rather than formal daily stand-ups, team members posted progress updates on Discord; blockers were resolved via ad-hoc discussions.
- Trello cards (using Jira) tracked tasks; labels indicated their priority (Must, Should, Could).

Testing & Quality Assurance

- **Automated testing:** used Django's test framework for backend unit and integration tests; test suites ran after significant edits. Frontend components were tested with **Jest** and React Testing Library. Critical user flows (login, listing search, chat) were tested with **Cypress**.
- **Manual testing:** conducted by Abdulelah to ensure the user interface met design expectations and supported right-to-left and bilingual functionality.
- **Bug tracking:** if a bug is present all the focus goes to fix it immediately

Monitoring Progress & Metrics

The team monitored progress through key metrics:

- **Velocity:** on average 8 tasks were completed per week, based on Trello cards.
- **Completion rate:** approximately 85 % of planned tasks were completed within their iteration; non-critical (Could-have) features were deferred.
- **Bug resolution:** total recorded bugs remained below 15 during Stage 4; the majority were minor UI glitches or integration issues.

When unexpected challenges arose—such as configuring WebSockets on Railway for real-time chat—the team re-prioritized and adjusted the scope of iterations accordingly.

Reviews & Retrospectives

At the end of each iteration the team conducted an informal review:

- New features were demoed end-to-end, from user registration through listing search and chatting with potential roommates.
- Feedback was gathered and documented; for example, early UI designs lacked sufficient contrast, prompting updates for accessibility.
- Retrospectives identified improvements such as clearer documentation for local environment setup and more detailed pull-request descriptions.

These sessions improved subsequent iterations and strengthened team collaboration.

Final Integration & QA Testing

After the third iteration the team performed comprehensive **integration testing**:

- Verified that React components correctly consumed Django APIs for user management, listings, messaging, reviews and roommate matching.
- Conducted performance tests on search filters to ensure acceptable response times under typical loads.
- Confirmed that the deployment pipeline delivered stable builds on **Railway** and **Netlify** and that environment variables were correctly managed.

Automated test coverage exceeded 80 %, and manual testing confirmed that the product was responsive, and accessible across devices.

Deliverables

- **Iteration planning document:** lists iterations, priorities, dependencies, deadlines, and team assignments.
- **Source code repository:** <https://github.com/MoeAlomar/UniStay>
- **Bug and task tracking board:** Jira board showing tasks, priorities, statuses, and completion metrics.
- **Deployment links:**
 - Backend on Railway:
 - Frontend on Netlify: