Overview

Table of Contents

1. Welcome
2. Project Summary
3. Core Learning Outcomes
4. Timeline and Key Dates
5. Requirements
6. Implementation Guides
   * Getting Started
   * Stand-Ups
   * Sprint Planning & Scoping
   * Design Proposals
   * Refactors
   * Retrospectives
7. Bug & Feature Tracker
8. Final Deliverables

**Welcome**

Welcome to Tech.Dive 2025! This sprint experience is designed to simulate a real-world software engineering team environment. You'll practice essential engineering skills like scoping, debugging, implementing features, running stand-ups, and participating in retrospectives. The goal is to strengthen your readiness for a future technical internship or full-time engineering role.

**Project Summary**

You will be working on a simulated Pinterest-like application. Everyone will receive the same codebase with bugs and missing features. In self-directed teams of 3–6, you will:

* Triage and self-assign 10+ tickets to fix/implement
* Scope your own work, track your time, and improve estimation skills
* Conduct stand-ups, sprint planning, and retrospectives
* Write design proposals and go through a refactor

**Core Learning Outcomes**

* **Working in a Sprint Environment:**
  + Weekly stand-ups
  + Bi-weekly sprint planning & retrospectives
  + Implementing features and fixing bugs in new and foreign environment
* **Ticket Scoping & Self-Assessment:**
  + Point your tickets and track estimated vs actual time
  + Improve ticket estimation and planning skills
* **Stand-Up Communication:**
  + Share completed work, current tasks, blockers, and help needed
* **Agency and Self-Direction:**
  + Learn when to ask for help vs. when to self-research
* **Design Proposals:**
  + Propose and explain implementation approach for new features
* **Refactor Practice:**
  + Clean up and improve existing code while maintaining functionality

**Timeline and Key Dates**

* **Week of April 14**: Kickoff & Setup
* **April 21**: Sprint 1 Begins
* **May 5**: Sprint 2 Begins
* **May 19**: Sprint 3 Begins
* **May 31**: Final Deliverables Due

Weekly stand-ups (15–30 mins) Bi-weekly sprint planning and retrospectives Each group must invite Bertrand to observe 1 stand-up early and 1 near the end

***Please use*** [***this link***](https://calendar.app.google/tFqvnQQrq6qf6rjF7) ***to set time with Bertrand Week of April 14th. Align with your team on a time, have one team member fill this out and then add the rest of your team members to the calendar invite.***

**Requirements**

* Complete at least 10 tickets (bugs or features)
* Participate in all stand-ups, sprint planning, and retrospectives
* Submit:
  + At least 1 design proposal
  + 1 refactor plan & implementation
  + Time-tracking vs scoping log
  + Final project reflection

**Implementation Guides**

**Getting Started**

1. Clone [the project repo](https://github.com/HackDotDiversity/2025-Tech-Dive)
2. Follow the README for setup
3. Join your team Slack channel and schedule your meetings

**Stand-Ups**

* Create a team doc using the Stand-Up Template
* Update it for each meeting with:
  + What I did yesterday
  + What I’m doing today
  + Am I blocked?
  + Do I need help?
* Time each update (aim for 15 min total)

**Sprint Planning & Scoping**

* Review the bug/feature tracker
* Use the Sprint Planning Template to:
  + Assign tickets to each person
  + Estimate time in hours/points
  + Track progress

**Design Proposals**

* Use the Design Proposal Template:
  + Problem Statement
  + Proposed Solution
  + Tradeoffs/Alternatives
  + Why this design?

**Refactors**

* Use the Refactor Plan Template:
  + Identify what’s messy/redundant
  + Plan to improve readability or structure
  + Include tests if possible

**Retrospectives**

* Use the Retro Template to:
  + Share what went well, what didn’t
  + Reflect on scoping, blockers, teamwork
  + Suggest process improvements

**Bug & Feature Tracker**

* See the Bug & Feature Tracker table for the full list
* Includes:
  + Ticket Title
  + Type (Bug / Feature / Refactor)
  + Description

**Final Deliverables**

Due May 31:

* Completion of 10+ tickets (each scoped and logged)
* 1 design proposal
* 1 refactor plan and execution
* Stand-up log
* Sprint planning log
* Final retrospective entry

We’re excited to see what you build. Go wild, experiment, learn from one another, and grow your skills!

Stand-Up Guide

**Tech Dive Stand-Up Log – Instructions for Use**

This shared Google Doc is your daily space to document and reflect on your team’s progress. Please read the instructions below carefully.

**How to Use This Document**

**Make a Copy of the Daily Template**

• Scroll down to the **Stand-Up Template** section below.

• Select the entire template (one page’s worth), copy it, and paste it **above** or **at the top** of the document.

• Rename the heading for today’s date (e.g., Stand-Up: Tuesday, April 1, 2025).

**Take Turns Facilitating**

• Each stand-up must have a **rotating facilitator**.

• Facilitator responsibilities:

• Start on time.

• Keep the total stand-up to **15 minutes or less**.

• Encourage all voices to be heard.

• **Track how long each person speaks** (record below their entry).

• Help teammates reflect on blockers and progress.

**Track Attendance**

• Add checkboxes for who was present each day.

• For example:

[x] Alex [x] Jada [ ] Jamal [x] Mei [x] Zara

**Complete Your Section**

• Each member fills out their stand-up answers in the format provided.

• Be honest about blockers and how you plan to move forward.

**Reflect Weekly**

• Each Friday, facilitators should summarize:

• Key improvements

• Recurring blockers

• Suggestions for better collaboration

**Why We Do This**

• Build real-world habits for effective team communication.

• Practice technical articulation and self-assessment.

• Create a written record of your growth and patterns over time.

• Strengthen your ability to scope, explain, and reflect on your work.

**Tip for Daily Duplication**

To keep your stand-up log clean and well-organized:

• Title each day clearly at the top (Stand-Up: [Day, Date])

• Insert a horizontal line between days (---)

• If needed, add a section for “Facilitator Notes” to reflect on timing, engagement, or improvement.

Stand-Up Template

**Tech Dive Stand-Up Template**

Each team member should fill out this document during stand-up.

Rotate who facilitates each day. Teams are composed of 3–6 people.

There is no specific lead—stand-ups should be co-led.

**Instructions for Facilitator**

• Make sure each person shares their updates.

• Ensure the group sticks to time (aim for 10–15 mins total).

• Encourage fellows to clarify blockers and ask for help if needed.

**Stand-Up Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Team Members:**

1.

2.

3.

4.

5.

6.

**Team Member 1**

**Name:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Yesterday:**

*What did you work on or complete?*

**Today:**

*What are you working on?*

**Blockers:**

*Are you stuck? What do you need?*

**Support Needed:**

*Do you need a tech lead or peer help?*

**Team Member 2**

*(Same format as above)*

(Repeat for Team Members 3 through 6)

Sprint Planning & Scoping Guide

**Tech Dive Sprint Planning & Scoping Guide**

**Purpose of Sprint Planning**

Sprint Planning helps engineering teams set a clear path forward for what they will achieve in a fixed period (in this case, 1–2 weeks). Fellows will simulate this practice by:

• Scoping and pointing their own tasks

• Discussing priorities and timelines

• Making realistic estimates of how long tasks will take

• Gaining ownership over their workload

**Sprint Duration**

Each sprint lasts **1 or 2 weeks**, depending on your facilitator’s direction.

Each team member should aim to complete **3–5 tickets** of varying size (bug fixes, refactors, feature builds).

**Sprint Planning Steps**

**1. Review the Project & Backlog**

• Read through the project repo and ticket board.

• Understand the overall product and the purpose of each ticket.

• Ask clarifying questions before scoping anything.

**2. Choose Tickets That Fit Your Skill Level + Challenge You**

• Start with a smaller ticket or bug fix to warm up.

• Select at least one stretch ticket to grow your skills.

• Balance is key: some quick wins, some deeper work.

**3. Scope Each Ticket You Plan to Work On**

Use the pointing system below and record your estimate before starting.

| **Points** | **Description** | **Est. Time** |
| --- | --- | --- |
| 1 | Very small fix, no unknowns | < 1 hour |
| 2 | Small fix or easy task | 1–2 hours |
| 3 | Moderate complexity, minor unknowns | 2–3 hours |
| 4 | Complex, may involve new logic or design | 4–5 hours |
| 5 | Very complex or large, includes research | 6–8+ hours |

**Note:** This is not about perfection—it’s about practicing how to estimate. You’ll reflect later on how accurate you were.

**4. Add Each Ticket to Your Personal Sprint Plan**

• Document the ticket title, point estimate, and your design proposal (for features).

• Keep this in a shared Google Doc, Notion board, or other team tracker.

**What to Ask Yourself When Scoping**

• Do I understand the requirements?

• What parts of this are familiar vs. new?

• Will I need to research something new?

• Can I break this down into smaller steps?

**During Sprint Planning Meeting**

Each team member should:

• Share which tickets they’re picking and why.

• Say their estimated points and why.

• Explain what their first steps will be.

**By the End of Sprint Planning You Should Have:**

• 3–5 scoped and self-assigned tickets

• Estimates for each one

• A clear plan for where you’ll begin

• A shared team plan with everyone’s tasks visible

Sprint Planning Template

**Tech Dive Team Sprint Planning Tracker**

Use this table to track each fellow’s sprint plan in a shared team document.

Each fellow should fill out their row during sprint planning.

**Sprint Start Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Team Members: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Instructions:**

• Each fellow fills in one row below.

• Use the estimated point values to scope your work.

• You can list up to 3–5 tickets depending on your capacity.

• Make a copy of this table for each sprint cycle.

**Sprint Planning Table**

| **Fellow Name** | **Overall Sprint Goal** | **Ticket 1: Title / Points** | **Ticket 2: Title / Points** | **Ticket 3: Title / Points** | **Ticket 4: Title / Points** | **Ticket 5: Title / Points** | **Total Points (Est.)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Retro Guide

**Tech Dive Sprint Retrospective Guide**

A sprint retrospective is a chance to reflect on how the sprint went—not just what was built, but how you worked, collaborated, and made decisions. It’s a key part of improving as an engineer and teammate.

**What Is a Retrospective?**

A retrospective (or “retro”) happens at the end of the sprint and gives fellows space to:

• Reflect on how their work went (scoping vs. actual time)

• Share what helped or hurt their progress

• Identify ways to improve their process for the next sprint

• Celebrate what went well!

**Why Retrospectives Matter**

• Build self-awareness around how you estimate time and solve problems

• Strengthen communication with your team

• Practice giving and receiving feedback

• Continuously improve your technical and workflow skills

**How to Run a Retrospective**

Retros can be done as a team or individually. In either case, you’ll answer a few structured questions. Below is a simple format you can follow:

1. **What went well?**

• Wins, things you’re proud of, teamwork highlights

2. **What was challenging?**

• Blockers, things that didn’t go as planned, bugs or tasks that took longer than expected

3. **What surprised you?**

• Anything unexpected—positive or negative

4. **What will you try differently next time?**

• Suggestions to improve how you work, plan, or communicate

5. **Compare Estimates vs. Actuals**

• Look at your sprint plan: Which tickets took more or less time than expected? Why?

**Optional: Team Retro Board (Digital or Whiteboard)**

Create columns for:

• 👍 Went well

• ⚠️ Challenges

• 💡 Ideas for next time

• 🎉 Shoutouts

**Tips for Great Retrospectives**

• Be honest and constructive

• Focus on the process, not people

• Share examples, not just opinions

• Reflect with a growth mindset

Retro Template

**Tech Dive Team Sprint Retrospective Template**

Use this shared document to reflect on your team’s sprint.

Each fellow should fill out a row in the table below. Complete it at the end of your sprint cycle.

**Sprint End Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Team Members: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Instructions:**

• Each team member fills in one row.

• Be specific and honest about your experience.

• Keep the focus on your process and learning.

• Use this to improve how you approach the next sprint.

**Retrospective Table**

| **Fellow Name** | **What Went Well** | **What Was Challenging** | **What Surprised You** | **What Will You Do Differently Next Time?** | **Estimate vs. Actual Reflection** |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Features, Bugs, & More

[**Tech Dive**](https://github.com/HackDotDiversity/2025-Tech-Dive) **Bugs, Features, & More**

***Make a copy of this document to track your progress.***

| **Title** | **Type** | **Description** | **Status** | **Expected Time / Points** | **Actual Time / Points** |
| --- | --- | --- | --- | --- | --- |
| **Remove test credentials from login screen** | Bug | Privacy issue – test credentials are visible on the login screen and must be removed. | Completed | 5 | 2 |
| **Implement default image for pins without images** | Feature | The pins don't have images; implement a default fallback image. | Not Started |  |  |
| **Fix broken or missing icons on main screen** | Bug | Several icons appear incorrect or are missing; ensure consistent iconography. | Not Started |  |  |
| **Fix create pin modal navigation** | Bug | Exiting the 'Create Pin' modal currently brings users to a blank screen instead of remaining on the main board. | Not Started |  |  |
| **Implement Create Pin** | Feature | Implement creating a pin and storing it in the database for retrieval. | Not Started |  |  |
| **Support Uploading an image for Pin** | Feature | Upload an image for a pin and store it in the database. | Not Started |  |  |
| **Improve search bar UI and contrast** | Bug | Text in the search bar is hard to see due to low contrast. | Not Started |  |  |
| **Display pin category tags under search bar** | Feature | Retrieve available categories from the database and display them as tag filters under the search bar. | Not Started |  |  |
| **Fix missing settings icon** | Bug | Settings icon next to the search bar is missing and needs to be displayed. | Not Started |  |  |
| **Implement pin detail page** | Feature | Selecting a pin should show its detail view; partially implemented code exists. | Not Started |  |  |
| **Refactor unused settings screen options** | Refactor | Remove or comment out options on the settings screen that won’t be implemented. | Not Started |  |  |
| **Implement dark mode / light mode switch** | Feature | Create a toggleable theme system to support light and dark modes. | Not Started |  |  |
| **Allow profile editing (username & password)** | Feature | Enable logged-in users to update their profile and credentials with backend integration. | Not Started |  |  |
| **Implement settings reset to default** | Feature | Add functionality to reset current settings (like theme) to default. | Not Started |  |  |
| **Update profile page to reflect current user** | Bug | Profile page currently shows dummy data instead of actual logged-in user info. | Not Started |  |  |
| **Implement 'Edit Profile' button** | Feature | Enable users to edit profile details with a modal, reusing components like 'Create Pin'. | Not Started |  |  |
| **Implement 'Create Board' button** | Feature | Allow users to create new boards from the profile page. | Not Started |  |  |
| **Implement three-dot menu on profile** | Feature | Add options like log out, toggle privacy, and other profile settings. | Not Started |  |  |
| **Enable 'Like' functionality for pins** | Feature | Add a 'like' field to the database and connect it to user interactions. | Not Started |  |  |
| **Add 'Save' functionality for pins** | Feature | Users should be able to save pins; update the database accordingly. | Not Started |  |  |
| **Add comments to pins** | Feature | Implement ability to post comments and save them to the database. | Not Started |  |  |
| **Delete comments on pins** | Feature | Allow users to delete their own comments and reflect it in the database. | Not Started |  |  |
| **Add edit/delete options to pin menu (3 dots)** | Feature | Implement modal or menu that allows editing or deleting a pin. | Not Started |  |  |
| **Implement board detail page from profile** | Feature | Clicking a board on the profile page should lead to a detailed view with matching styling. | Not Started |  |  |
| **Create a board using Create Pin modal** | Feature | Reuse the Create Pin modal logic to implement a board creation experience. | Not Started |  |  |
| **Delete a pin** | Feature | Allow users to delete their own pins and update the backend accordingly. | Not Started |  |  |
| **Delete a board** | Feature | Enable users to delete a board they own and reflect this change in the database. | Not Started |  |  |
| **Add unit test for retrieving all users** | Testing | Write a unit test to validate the functionality for fetching all users. | Not Started |  |  |
| **Add unit test for retrieving all pins** | Testing | Write a unit test to validate the functionality for fetching all pins. | Not Started |  |  |
| **Add unit test for creating a pin** | Testing | Write a unit test to ensure pin creation works as expected. | Not Started |  |  |
| **Add unit test for deleting a pin** | Testing | Write a unit test to ensure pin deletion functions correctly. | Not Started |  |  |
| **Add unit test for creating a user** | Feature | Write a unit test to verify user account creation. | Not Started |  |  |
| **Implement logout on homepage** | Feature | Add logout button and logic on homepage screen. | Not Started |  |  |
| **Share or hide profile via settings** | Feature | Implement share or hide profile functionality and adjust visibility accordingly. | Not Started |  |  |
| **Back button on profile page** | Feature | Add a back button on profile page to return to previous view. | Not Started |  |  |
| **Global back button across app** | Feature | Implement a consistent back navigation component throughout the app. | Not Started |  |  |
| **Match login page styling with rest of app** | Bug | Align visual styling of login page with the broader app or vice versa. | Not Started |  |  |
| **Add app name header above search bar** | Feature | Display 'Pinterest' or custom app name above search bar for branding. | Not Started |  |  |
| **Display comments on pin detail view** | Feature | Ensure pin comments appear under pin details. | Not Started |  |  |
| **Associate saved pins with user profile** | Feature | Link saved pins to user profiles and reflect in database. | Not Started |  |  |
| **Associate liked pins with user profile** | Feature | Link liked pins to user profiles and reflect in database. | Not Started |  |  |
| **Make pins private** | Feature | Add functionality for users to mark pins as private (may require database changes). | Not Started |  |  |
| **Create account functionality** | Feature | Enable account creation flow on existing sign-up page. | Not Started |  |  |
| **Fix text boxes on create account screen** | Bug | Ensure inputs on create account screen are styled and functional properly. | Not Started |  |  |
| **Settings: toggle show/hide email** | Feature | Allow users to choose if their email is visible in their profile. | Not Started |  |  |
| **Feature gating via toggle in settings** | Feature | Add a toggle in settings to enable/disable beta features without code changes (simulate A/B testing). | Not Started |  |  |
| **Fix grid layout rendering on home screen** | Bug | Resolve issues where the grid of pins does not dynamically render correctly on the home screen. | Not Started |  |  |
| **Discover and fix hidden bugs** | Bug | Investigate the codebase to find and fix any bugs not explicitly listed in the tracker. | Not Started |  |  |
| **Identify, propose, and implement new features** | Feature | Explore the codebase to identify potential new features. Propose their design and implement them to improve the user experience. | Not Started |  |  |
| **Propose and Implement a refactor** | Refactor | Restructure the codebase in a way that is more efficient, making it easier to build new features. | Not Started |  |  |

**Data**

| **Title** | **Type** | **Description** | **Status** | **Expected Time / Points** | **Actual Time / Points** |
| --- | --- | --- | --- | --- | --- |
| **Audit missing or inconsistent analytics fields** | Data Quality | Identify nulls, negative values, or invalid entries in analytics fields such as device type, views, or saves. | Not Started |  |  |
| **Validate referential integrity in analytics data** | Data Quality | Ensure all foreign keys (e.g., pin.userId, pin.boardId) refer to valid users and boards. | Not Started |  |  |
| **Detect duplicate or anomalous records** | Data Quality | Look for duplicate users, pins with unusually high metrics, or impossible stats (e.g., more saves than views). | Not Started |  |  |
| **Segment users by engagement metrics** | User Analysis | Cluster users into segments based on login frequency, pin activity, or follower counts. | Not Started |  |  |
| **Identify top creators and influencers** | User Analysis | Use engagement and network data to determine which users are most influential. | Not Started |  |  |
| **Perform retention analysis** | User Analysis | Track retention rates for users after 1, 7, and 30 days post-signup. | Not Started |  |  |
| **Analyze pin and board performance** | Content Analysis | Rank pins and boards by views, saves, or engagement rate. | Not Started |  |  |
| **Map the social network** | Network Analysis | Visualize the follower/following graph and identify clusters or central nodes. | Not Started |  |  |
| **Detect viral pins and echo chambers** | Network Analysis | Find pins that spread widely and detect isolated or insular user communities. | Not Started |  |  |
| **Build device and location dashboards** | Device/Geo Insights | Break down engagement and usage by device type and region. | Not Started |  |  |
| **Analyze onboarding funnel performance** | Growth Analysis | Track how many users complete actions (create board/pin/comment) after signing up. | Not Started |  |  |
| **Recommend features based on data insights** | Feature Strategy | Use usage and retention data to propose new or improved features. | Not Started |  |  |
| **Identify underused features** | Feature Strategy | Find features that are rarely used and assess whether they should be improved or removed. | Not Started |  |  |
| **Create dashboards for weekly reporting** | Dashboarding | Build and automate dashboards for active users, top content, and network health. | Not Started |  |  |
| **Run churn prediction or user clustering models** | Advanced Analysis | Apply machine learning or clustering to anticipate user churn or group similar users/pins. | Not Started |  |  |