**Prioritization of Dynamic Content Development**

Based on the provided files and the list of editable content, prioritize development to focus on high-impact sections that drive user engagement and require frequent updates. Below is a prioritized list of sections to make dynamic, along with how to manage their development in Git branches:

1. **Hero Section**
   * **Why**: The hero section is the first impression for users, with critical content like the title, subtitle, and call-to-action buttons.
   * **Dynamic Content**:
     + hero\_title, hero\_subtitle, begin\_journey, explore\_how (from home.php).
     + Typing effect phrases (currently hardcoded in JavaScript; consider moving to home.php for admin control).
   * **Branch**: feature/dynamic-hero-content
   * **Tasks**:
     + Create a database model (e.g., HeroContent) to store title, subtitle, button texts, and typing phrases.
     + Build an admin interface to edit these fields.
     + Update home.blade.php to fetch data dynamically (e.g., {{ $heroContent->title }}).
     + Ensure translations are loaded dynamically from home.php.
   * **Translation**:
     + Support multiple languages (e.g., en/home.php, si/home.php, ta/home.php).
     + Branch: translation/hero-<language> for each new language.
2. **Stats Section**
   * **Why**: Stats build trust and are likely to change frequently (e.g., user count updates).
   * **Dynamic Content**:
     + users\_count, users\_label, accuracy\_rate, accuracy\_label, careers\_matched, careers\_label, success\_stories, stories\_label (from home.php).
   * **Branch**: feature/dynamic-stats
   * **Tasks**:
     + Create a Stats model to store stat values and labels.
     + Develop an admin panel to update stats.
     + Update home.blade.php to fetch stats dynamically.
     + Ensure JavaScript counters (in <script>) use dynamic values.
   * **Translation**:
     + Update home.php for each language.
     + Branch: translation/stats-<language>.
3. **Testimonials Section**
   * **Why**: Testimonials enhance credibility and may need frequent updates to reflect new user stories.
   * **Dynamic Content**:
     + testimonials\_title, testimonials array (quotes and authors) (from home.php).
   * **Branch**: feature/dynamic-testimonials
   * **Tasks**:
     + Create a Testimonial model for quotes and authors.
     + Build an admin interface to add/edit/delete testimonials.
     + Update home.blade.php to loop through testimonials dynamically.
   * **Translation**:
     + Support translated quotes and authors in home.php.
     + Branch: translation/testimonials-<language>.
4. **Features Section**
   * **Why**: Features highlight the platform’s value and may evolve with new offerings.
   * **Dynamic Content**:
     + features\_title, features array (icon, title, description) (from home.php).
   * **Branch**: feature/dynamic-features
   * **Tasks**:
     + Create a Feature model for icons, titles, and descriptions.
     + Develop an admin interface to manage features.
     + Update home.blade.php to fetch features dynamically.
   * **Translation**:
     + Translate feature titles and descriptions.
     + Branch: translation/features-<language>.
5. **How It Works Section**
   * **Why**: Guides users through the process; steps may change as the platform evolves.
   * **Dynamic Content**:
     + how\_it\_works\_title, steps array (title, description) (from home.php).
   * **Branch**: feature/dynamic-how-it-works
   * **Tasks**:
     + Create a Step model for step titles and descriptions.
     + Build an admin interface for steps.
     + Update home.blade.php to loop through steps dynamically.
   * **Translation**:
     + Translate step titles and descriptions.
     + Branch: translation/steps-<language>.
6. **Community Section**
   * **Why**: Encourages user engagement; call-to-action text and links may change.
   * **Dynamic Content**:
     + community\_title, community\_description, join\_now (from home.php).
   * **Branch**: feature/dynamic-community
   * **Tasks**:
     + Create a CommunityContent model for title, description, and button.
     + Develop an admin interface for editing.
     + Update home.blade.php for dynamic content.
   * **Translation**:
     + Branch: translation/community-<language>.
7. **Newsletter Section**
   * **Why**: Drives user retention; form text and button may need updates.
   * **Dynamic Content**:
     + newsletter\_title, newsletter\_description, email\_placeholder, subscribe (from home.php).
   * **Branch**: feature/dynamic-newsletter
   * **Tasks**:
     + Create a NewsletterContent model.
     + Build an admin interface for editing.
     + Update home.blade.php for dynamic content.
   * **Translation**:
     + Branch: translation/newsletter-<language>.
8. **Navigation (Header)**
   * **Why**: Navigation links and user-specific options (e.g., login/logout, dashboards) are critical and role-dependent.
   * **Dynamic Content**:
     + features, how\_it\_works, testimonials, community, start\_free, login, register, admin\_dashboard, booker\_dashboard, provider\_dashboard, logout (from header.php).
   * **Branch**: feature/dynamic-navbar
   * **Tasks**:
     + Ensure dynamic role-based links (e.g., admin, student, mentor dashboards).
     + Allow admin to edit link text via an interface.
     + Update navbar.blade.php to handle dynamic translations.
   * **Translation**:
     + Branch: translation/navbar-<language>.
9. **Footer**
   * **Why**: Footer content (links, descriptions) may need updates for branding or legal reasons.
   * **Dynamic Content**:
     + Content from footer.php (e.g., company description, links, copyright).
   * **Branch**: feature/dynamic-footer
   * **Tasks**:
     + Create a FooterContent model.
     + Build an admin interface for footer content.
     + Update footer.blade.php for dynamic content.
   * **Translation**:
     + Branch: translation/footer-<language>.
10. **Chatbot**
    * **Why**: Chatbot functionality may expand; alt text and messages need to be editable.
    * **Dynamic Content**:
      + chatbot\_alt (from home.php), interaction message (currently hardcoded in JavaScript).
    * **Branch**: feature/dynamic-chatbot
    * **Tasks**:
      + Create a ChatbotContent model for alt text and messages.
      + Develop an admin interface for editing.
      + Update home.blade.php and JavaScript for dynamic content.
    * **Translation**:
      + Branch: translation/chatbot-<language>.
11. **General Settings (Lower Priority)**
    * **Why**: Colors, fonts, and logos are less frequently updated but still important for branding.
    * **Dynamic Content**:
      + Logo text (career\_text, compass\_word from header.php and home.php).
      + CSS variables (e.g., --primary, --secondary).
      + Background images.
    * **Branch**: feature/dynamic-settings
    * **Tasks**:
      + Create a Settings model for logo text, colors, and images.
      + Build an admin interface for branding settings.
      + Update CSS and Blade files to use dynamic values.
    * **Translation**:
      + Branch: translation/settings-<language>.

**Git Branching Workflow Example**

1. **Setup**:
   * Initialize main and develop branches.
   * Ensure main is deployed to production, and develop is the default branch for development.
2. **Development Process**:
   * **Create Feature Branch**: For the hero section, run:

bash

CollapseWrapRun

Copy

git checkout develop

git checkout -b feature/dynamic-hero-content

* + **Develop**: Implement the HeroContent model, admin interface, and update home.blade.php.
  + **Commit**: Use clear commit messages, e.g., Add dynamic hero content with admin interface.
  + **Test**: Test locally with multiple languages (e.g., en, si, ta).
  + **Push**: Push to remote: git push origin feature/dynamic-hero-content.
  + **Pull Request (PR)**: Create a PR to merge into develop. Include reviewers and test translations.
  + **Merge**: After approval, merge into develop and delete the branch:

bash

CollapseWrapRun

Copy

git checkout develop

git merge feature/dynamic-hero-content

git push origin develop

git branch -d feature/dynamic-hero-content

1. **Translation Updates**:
   * For Sinhala translations of the hero section:

bash

CollapseWrapRun

Copy

git checkout develop

git checkout -b translation/hero-sinhala

* + Add resources/lang/si/home.php with translated hero content.
  + Test in home.blade.php.
  + Create a PR to merge into develop.

1. **Release**:
   * When ready, create a release branch:

bash

CollapseWrapRun

Copy

git checkout develop

git checkout -b release/v1.0.0

* + Update translations, fix bugs, and test.
  + Merge into main and develop:

bash

CollapseWrapRun

Copy

git checkout main

git merge release/v1.0.0

git checkout develop

git merge release/v1.0.0

git tag v1.0.0

git push origin main --tags

* + Deploy main to production.

1. **Hotfix Example**:
   * For a typo in hero\_title:

bash

CollapseWrapRun

Copy

git checkout main

git checkout -b hotfix/hero-title-typo

* + Update resources/lang/en/home.php.
  + Merge into main and develop, then deploy.