



Walky Admin Portal Unit Test Plan

Status	Done
Assignee	Ⓜ Meagan
Due date	@08/11/2025
Task type	Work



Task Description





This page documents unit tests for the **Walky Admin Portal**. The goal is to ensure that core UI components render properly and behave as expected based on props, state, and user interactions.



Executive Summary

During my internship, I developed, executed, and documented **comprehensive automated tests** for the Walky Admin Portal using **Jest** and **React Testing Library**.

Impact at a glance:

-  Brought **all core pages and UI components** under automated test coverage, eliminating major untested areas in the portal
-  Streamlined QA by automating repetitive checks for login, tables, and forms, reducing reliance on repeated manual validation before releases
-  Strengthened **accessibility compliance** through tab navigation, keyboard input, and responsive layout checks
-  Ensured **light/dark mode** and **responsive design** consistency across the portal, preventing theme-related regressions



Testing Strategy

- **Framework:** Jest + React Testing Library — used to validate individual component behavior and integration within the UI.

- **File Format:** `.test.tsx` — each component has its own matching test file
- **Location:** All tests are inside the `tests` folder
- **What we're testing:** Just the *little parts* (like buttons, inputs, cards) — using fake data or simulating what a user would do (like typing or clicking)

Pages & Components Tested

Below is a **condensed version** of the detailed test breakdown.

The full appendix contains exact checks and edge cases tested per page.

Page	Key Components Tested	Focus Areas
Login	<code>LoginForm</code> , <code>Input</code> , <code>Button</code>	Input handling, error clearing, password toggle, routing, responsive layout
Dashboard	<code>StatCard</code> , <code>ChartSection</code> , <code>ExampleAdminLayout</code> , <code>Topbar</code> , <code>Sidebar</code>	Stat accuracy, graph rendering, layout responsiveness, theme toggles
Students	<code>InfoStatWidget</code> , <code>StudentTable</code>	API integration, data formatting, dropdown actions, tooltip behavior
Engagement	<code>Engagement Widgets</code>	API-driven stats, fallback handling, section styling, theme consistency
Review	<code>ReviewTable</code>	Data formatting, dropdown actions, error handling, table styling
Campuses	<code>CampusTable</code> , <code>CampusDetails</code>	Map toggling, polygon drawing, form validation, API mutation handling
Campus Sync	<code>SyncTable</code> , <code>PreviewModal</code> , <code>LogsModal</code>	Sync actions, loading states, error handling
Ambassadors	<code>AmbassadorDetails</code> , <code>AmbassadorView</code>	Form validation, CRUD actions, theme styling
Places	<code>PlacesList</code>	Campus filter dropdown, refresh logic, API failure handling
Place Types	<code>PlaceTypesTable</code> , <code>Add/Edit Modal</code>	Filtering, badge display, validation, API CRUD actions
Settings	<code>SettingsForm</code> , <code>ImageUpload</code>	Field validation, file handling, theme responsiveness
Verify Code / Forgot Password	<code>VerifyCodeForm</code> , <code>ResetPasswordForm</code> ,	Step-based UI flow, cooldown timers, validation rules

Page	Key Components Tested	Focus Areas
	ForgotPasswordForm	

✓ Appendix – Detailed Test Cases

▼ Test Login Page

- ✓ Renders email and password inputs
- ✓ Password visibility toggle functions correctly
- ✓ Clears error messages when user types valid input
- ✓ Calls `onLogin` function upon successful login
- ✓ Redirects to dashboard (or protected route) on successful login
- ✓ "Forgot password" link routes to `/forgot-password` + submits login form
 - Simulates clicking the "Forgot Password" link
 - Uses `MemoryRouter` to mock routing behavior
 - Confirms routing to `/forgot-password` by checking that the test component is rendered
- ✓ Responsive layout: Inputs/buttons render properly on smaller screens
- ✓ Inputs accept keyboard input and allow tab navigation (accessibility)

▼ Test Dashboard Page

- ✓ Renders all stat cards (Walks, Events, Ideas, Surprise)
- ✓ Shows correct label, number, trend %, and graph icon per card
- ✓ Handles missing/false stat data gracefully
- ✓ Renders graph for "Active Users & Walks"
- ✓ **Test** `ExampleAdminLayout.tsx`
 - ✓ Sidebar opens/closes based on screen size or toggle button
 - ✓ Topbar toggle button works
 - ✓ Correct page title is shown based on the current route
 - ✓ Breadcrumb renders when present
 - ✓ Theme styles apply correctly for light/dark mode

✓ Footer always renders at bottom of page

✓ **Test** `Tophar.tsx`

✓ Dark/light mode toggle switches themes

✓ Avatar/profile menu appears on click

✓ "Sign out" button routes to login or clears session

✓ **Test** `Sidebar.tsx`

✓ Shows correct nav links (Dashboard, Students, etc.)

✓ Highlights active route

✓ Closes on outside click in mobile view

▼ **Test** `Students Page`

✓ Renders all 4 stat widgets:

- Total Students
- Average Age
- Languages
- Parents

✓ Uses API calls to fetch stats from backend:

- `/users/count` → total students
- `/age/average` → average age
- `/language/count` → unique languages
- `/users/parent-count` → parent users

✓ Displays fallback (`=`) if API fails or data is missing

✓ **Test** `InfoStatWidget.tsx` (Stat card)

✓ Receives props correctly:

- `icon` , `value` , `label` , `tooltip`

✓ Renders layout consistently across light/dark mode

- Applies correct theme colors and tooltip styles

✓ Tooltip appears on hover with correct content

- ✓ Test `StudentTable.tsx`
 - ✓ Fetches and displays rows from `/users` API
 - ✓ Transforms API data into readable table format:
 - Name = first + last name
 - Dates formatted as `MMM D, YYYY`
 - ✓ Renders correct headers:
 - ID, Name, Email, Joined, Last Update
 - ✓ "Three dots" menu (dropdown) renders for each row
 - ✓ Dropdown actions appear:
 - Send email
 - Flag user / Unflag user
 - Request edit
 - Request to delete
 - ✓ Flag button toggles color (row turns red) and text (flag ↔ unflag)
-

- ▼ Test `Engagement Page`
- ✓ Renders all 3 engagement sections:
 - Walks (Purple)
 - Events (Blue)
 - Ideas (Orange)
 - ✓ Each section includes stat widgets with correct values and labels
 - `Walks` : Total, Pending, Active, Completed, Cancelled/Closed
 - `Events` : Total, Outdoor, Indoor, Public, Private
 - `Ideas` : Total, Active, Inactive, Collaborated
 - ✓ Stat values are fetched via API and rendered correctly
 - ✓ Displays fallback (e.g., `—` or 0) if API returns null/missing
 - ✓ Section background colors apply correctly (purple, blue, orange)
 - ✓ Layout remains responsive across screen sizes



- ✓ Makes correct API call per section:
 - `/walks/total` , `/walks/pending` , etc.
 - `/events?filter=outdoor` or similar
 - `/ideas/active` , `/ideas/collaborated` , etc.
 - ✓ Handles API errors gracefully (does not crash UI)
 - ✓ Test `Engagement Widgets` (Stat cards for each section)
 - ✓ Receives correct props: `value` , `label` , `icon` , `tooltip` , `color`
 - ✓ Renders the stat value and label
 - ✓ Icon renders with appropriate style
 - ✓ Tooltip appears on hover with the correct label/description
 - ✓ Applies correct theme styling in light/dark mode
-

▼ Test `Review Page`

- ✓ Renders the Review page container with proper padding
- ✓ Includes the `ReviewTable` component
- ✓ Test `ReviewTable.tsx`
 - ✓ Fetches and displays rows from `/users` API with:
 - Fields: `_id` , `first_name` , `last_name` , `reason` , `createdAt` , `reportedOn`
 - ✓ Transforms API data into readable table format:
 - Combines `first_name` + `last_name` into `Name`
 - Formats `createdAt` and `reportedOn` as `MMM D, YYYY`
 - ✓ Renders correct table headers:
 - `ID` , `Name` , `Reason` , `Joined` , `Reported On` , and dropdown
 - ✓ Dropdown menu renders for each row with actions:
 - Send email
 - View activity logs
 - Suspend user
 - Request to delete

- ✓ Displays fallback if `reason` is missing
- ✓ Handles API errors gracefully (e.g. logs error, UI doesn't crash)
- ✓ Applies consistent table styles and responsiveness (CoreUI)
- ✓ Opens dropdown menu and displays actions for each row



▼ Test **Campuses Page**

- ✓ Renders the Campuses page container with proper layout and spacing
- ✓ Displays title/header: **Campus Management**
- ✓ Renders the `+ Add Campus` button
- ✓ Includes the `CampusTable` component
- ✓ Test `CampusTable.tsx`
 - ✓ Fetches and displays campus data from `/campuses` API
 - Fields: `id` , `school_name` , `location` , `address` , `status`
 - ✓ Renders correct table headers:
 - `Campus` , `Location` , `Address` , `Status` , `Actions`
 - ✓ Displays each row with:
 - Icon + clickable school name
 - Shortened ID preview (`68613505...`)
 - Location (e.g., Miami, FL)
 - Full address
 - Status badge (e.g., Inactive)
 - Action buttons:  Edit,  Delete
 - ✓ Clicking the campus name toggles the **Campus Boundary map** below the row
 - Displays map with boundaries
 - Includes Map/Satellite toggle view
 - ✓ Edit button opens modal/form (test that the edit action is triggered)
 - ✓ Delete button renders (functional or not, just test presence)

- ✓ Handles API errors gracefully (logs error, doesn't crash UI)
- ✓ Applies consistent CoreUI styles and responsiveness
- ✓ Test `CampusDetails.tsx`
 - ✓ **Renders the CampusDetails form** with appropriate layout and theme styling
 - ✓ Displays correct title:
 - `Edit Campus` (when campus data is provided)
 - `New Campus` (when creating from scratch)
 - ✓ Pre-fills form inputs with `campusData` from navigation state
 - ✓ Handles form input changes:
 - ✓ `Campus Name`, `Phone Number`, `Address`, `City`, `State`, `ZIP`, `Time Zone`
 - ✓ `Dawn Time`, `Dusk Time`
 - ✓ `Campus Status` dropdown
 - ✓ Multi-select ambassador dropdown
 - ✓ Renders the **Campus Boundary** map section with draw polygon support
 - Displays initial polygon when provided
 - Fires `onBoundaryChange` when updated
 - ✓ Validates required fields before submission
 - ✓ Shows error messages for missing required fields
 - ✓ Blocks save if polygon is missing
 - ✓ Submits form via mutation:
 - ✓ `POST` request if new campus
 - ✓ `PUT` request if editing an existing one
 - ✓ Shows success toast on save
 - ✓ Shows error alert on failure
 - ✓ Handles cancel action (navigates back to Campuses)



▼ Test `CampusSync Page`

- ✓ **Renders the Campus Sync page** with proper layout and styling

- ✓ Displays header: [Campus Sync Management](#)
- ✓ Renders Sync All Campuses and View Logs buttons
- ✓ Fetches and displays rows with:
 - Campus name
 - Coordinates badge (Configured / Not Set)
 - Places count
 - Last sync (formatted date or "Never")
 - Sync status badge (Completed, Failed, Partial, etc.)
 - API calls used
 - Action buttons:  Sync,  Preview
- ✓ Handles Sync Campus button:
 - ✓ Triggers individual sync mutation
 - ✓ Shows loading spinner while syncing
 - ✓ Updates status and displays success or failure alert
- ✓ Handles Sync All Campuses button:
 - ✓ Triggers bulk sync mutation
 - ✓ Shows progress bar and summary message on completion
- ✓ Opens **Preview Modal** on button click
 - Displays area, center, bounds, and other metadata
- ✓ Opens **Logs Modal** on button click
 - Displays latest sync logs in table format
- ✓ Displays loading spinners and handles errors gracefully

▼ Test [Ambassadors Page](#)

- ✓ Renders the Ambassadors page container with proper layout and spacing
- ✓ Displays title/header: **Ambassador Management**
- ✓ Renders the [+ Add Ambassador](#) button
- ✓ Fetches and displays ambassador data from [/ambassadors](#) API

- ✓ Fields: `name`, `email`, `campus_name`, `major`, `status`
- ✓ Status badge (Active / Inactive)
- ✓ Action buttons:  Edit,  Delete
- ✓ Renders correct table headers:
 - ✓ `Ambassador`, `Email`, `Campuses`, `Major`, `Status`, `Actions`
- ✓ Edit button triggers navigation to `AmbassadorDetails` page
- ✓ Delete button:
 - ✓ Shows confirmation prompt
 - ✓ Triggers delete mutation and shows success/error alerts
- ✓ Renders loading skeleton while fetching
- ✓ Displays fallback message and call-to-action when ambassador list is empty
- ✓ Applies consistent CoreUI styles and responsiveness
- ✓ **Test** `AmbassadorDetails.tsx`
 - ✓ Renders the `AmbassadorDetails` form with proper layout and theme styling
 - ✓ Pre-fills form inputs with `ambassadorData` from navigation state
 - ✓ Handles form input changes:
 - ✓ Full Name
 - ✓ Email
 - ✓ Phone
 - ✓ Student ID
 - ✓ Major
 - ✓ Graduation Year
 - ✓ Bio
 - ✓ Profile Image URL
 - ✓ Status
 - ✓ Validates form:
 - ✓ Required: name, email

- ✓ Optional but validated: phone, student ID, graduation year
 - ✓ Real-time validation feedback
 - ✓ Save button:
 - ✓ Triggers `POST` (new) or `PUT` (edit) via mutation
 - ✓ Shows success alert and navigates to Ambassadors list
 - ✓ Cancel button navigates back to Ambassadors
 - ✓ Handles errors with contextual messaging (e.g., validation errors, server errors)
 - ✓ Displays loading state when fetching data for editing
 - ✓ Applies consistent CoreUI styling and dark/light theme support
 - ✓ **Test** `AmbassadorView.tsx`
 - ✓ Loads `AmbassadorDetails` with correct props based on route param
 - ✓ Passes `ambassadorId` and disables tab view
 - ✓ Renders `AmbassadorDetails` inside padded container
-

▼ Test `Places Page`

- ✓ Renders the Places page container with proper layout and spacing
- ✓ Displays title/header: **Places Management**
- ✓ Renders campus dropdown:
 - ✓ Fetches campus list from `/campuses` API
 - ✓ Displays each campus as: `campus_name - city, state`
- ✓ Renders `Refresh` button:
 - ✓ Disabled if no campus is selected
 - ✓ Triggers `refetchPlaces()` and shows "Refreshing..." alert
- ✓ Places list panel:
 - ✓ Renders only after a campus is selected
 - ✓ Shows loading spinner while fetching
 - ✓ Displays list of places with pagination (via `PlacesList`)

- ✓ Calls `placeService.getAll()` with filters
- ✓ Alerts:
 - ✓ Campus fetch failure alert
 - ✓ Places fetch failure alert
 - ✓ Info alert for refresh
- ✓ Handles empty states:
 - ✓ Before campus selection → "Select a campus"
 - ✓ After selection + no data → "No places found"
- ✓ Applies consistent CoreUI layout and responsiveness

▼ Test **Place Types Page**

- ✓ Renders the Place Types page with layout and CoreUI structure
- ✓ Displays title/header: **Place Types Management**
- ✓ Fetches place types from `/place-types` API
 - ✓ Query includes search + active toggle
- ✓ Search bar:
 - ✓ Filters place types by name or description
- ✓ "Show inactive" checkbox toggles visibility of inactive items
- ✓ Table headers:
 - ✓ `Name` , `Description` , `Google Types` , `Places Count` , `Status` , `Actions`
- ✓ Table row data:
 - ✓ Displays up to 3 Google types as badges
 - ✓ Extra types summarized as `+X more`
 - ✓ Badge for status (Active / Inactive)
 - ✓ Count of associated places
- ✓ Add/Edit Modal:
 - ✓ Form fields: name, description, Google types, active toggle
 - ✓ Displays scrollable Google types list

- ✓ Shows "Selected types" summary
- ✓ Validation:
 - ✓ Name required
 - ✓ At least 1 Google type required
- ✓ Triggers create or update mutation
- ✓ Success and error alerts
- ✓ Delete Modal:
 - ✓ Confirmation message with place count warning
 - ✓ Triggers delete mutation
 - ✓ Shows success/error alert
- ✓ Handles empty list fallback
- ✓ Shows loading spinner while fetching data

▼ Test **Settings Page**

- ✓ **Renders the Settings page with layout and styling**
- ✓ Contains title card and form elements for:
 - ✓ **School Name** input (with label and help text)
 - ✓ **Display Name (visible to students)** input (with label and help text)
 - ✓ **Email Domain** input (with label and help text)
 - ✓ **School Logo** uploader (ImageUpload component)
 - ✓ Accepts JPG, PNG, SVG formats
 - ✓ Displays recommended size and max size info
- ✓ Validates all required fields (school name, display name, email domain)
- ✓ Calls **handleSubmit** on Save button click
 - ✓ Assembles form data including file object for logo
 - ✓ Logs payload (simulated API submission)
- ✓ Cancel button present (no functionality yet)
- ✓ Adapts styling based on current theme (light/dark mode via **useTheme**)

▼ Test **Verify Code Page**

- ✓ **Initial render (step = 'verify')**
 - ✓ Renders logo and title: *Enter Verification Code*
 - ✓ Shows email and verification code inputs
 - ✓ Shows **Verify Code** button
 - ✓ Renders resend code button with default state
- ✓ **Verification step**
 - ✓ Calls `handleVerify` and logs payload
 - ✓ Transitions to `"reset"` step on success
 - ✓ Displays error on invalid code
- ✓ **Resend code logic**
 - ✓ Disables button when cooldown is active
 - ✓ Sends new request and starts 30s countdown
 - ✓ Displays updated text: *Resend available in XXs*
- ✓ **Reset Password step (step = 'reset')**
 - ✓ Title updates to *Reset Your Password*
 - ✓ Renders password and confirm inputs
 - ✓ Shows eye toggle for show/hide password
 - ✓ Shows **Reset Password** button
- ✓ **Password validation**
 - ✓ Error for mismatched passwords
 - ✓ Error for password < 8 characters
 - ✓ Successful POST logs in and navigates to `/login`

▼ Test **Forgot Password Page**

- ✓ Renders the Forgot Password page with layout and styling
- ✓ Displays title/header: **Forgot your password?**
- ✓ Renders the form when `submitted === false`

- ✓ Input: `EmailAddress` with label, placeholder, and required
- ✓ Submit button: `Send Reset Link`
- ✓ Secondary button: `Back to login`
- ✓ Shows error message if `error` is set
- ✓ Shows confirmation message when `submitted === true`
- ✓ Message: *If your email is registered...*
- ✓ Styled with green background and rounded card
- ✓ Calls `API.post('/forgot-password')` and navigates to `/verify-code` with email in state
- ✓ Handles API failure and displays fallback error

Visuals

• Login.tsx - July 1

```

[10:30] src/tests/Login.test.tsx
Walky Admin Portal - Login Component
  ✓ shows an error when fields are empty and Continue is clicked (227 ms)
  ✓ shows an error when user starts typing email (20 ms)
  ✓ calls onLogin function upon successful login (159 ms)
  ✓ handles password visibility when eye icon is clicked (158 ms)
  ✓ routes to Forgot Password page when link is clicked (73 ms)
  ✓ requests to backend on successful login (127 ms)
  ✓ renders login and button correctly on small screens (146 ms)
  ✓ allows keyboard input and tab navigation (20 ms)

Test Suites: 1 passed, 1 total
Tests:      8 passed, 8 total
Snapshots:  0 total
Time:       2.489 s, estimated 4 s
Ran all test suites matching src/tests/Login.test.tsx.
Done in 3.21s.

```

• NavSideBar.tsx - July 8

```

[10:30] src/tests/NavSideBar.test.tsx
Walky Admin - Sidebar Component
  ✓ highlights active route (64 ms)
  ✓ closes on outside click in mobile view (67 ms)

Test Suites: 1 passed, 1 total
Tests:      3 passed, 3 total
Snapshots:  0 total
Time:       2.297 s
Ran all test suites matching src/tests/NavSideBar.test.tsx.
Done in 4.67s.

```

• ReviewTable.tsx - July 9

```

[10:30] src/tests/ReviewTable.test.tsx
Walky Admin - ReviewTable Component
  ✓ fetches and displays user data from users API with required fields (428 ms)
  ✓ transforms API data into readable table format (33 ms)
  ✓ renders correct table headers (21 ms)
  ✓ renders correct table with correct columns for each row (124 ms)
  ✓ displays fallback (N/A) if reason is missing (31 ms)
  ✓ handles API errors gracefully (126 ms) (127 ms) (127 ms)
  ✓ applies correct table styles and is responsive (21 ms)
  ✓ shows disabled state and displays action for each row (186 ms)

Test Suites: 1 passed, 1 total
Tests:      8 passed, 8 total
Snapshots:  0 total
Time:       3.26 s
Ran all test suites matching src/tests/ReviewTable.test.tsx.
Done in 6.76s.

```

• CampusSyncc.test.tsx - July 16

• App.tsx - July 8

```

[10:30] src/tests/App.test.tsx (6.214 s)
Walky Admin Portal - Dashboard Page
  ✓ renders the widgets and chart title (346 ms)
  ✓ shows correct label, number, trend %, and graph icon per card (168 ms)
  ✓ handles missing/falsy stat data gracefully (92 ms)

Test Suites: 1 passed, 1 total
Tests:      3 passed, 3 total
Snapshots:  0 total
Time:       7.021 s
Ran all test suites matching src/tests/App.test.tsx.
Done in 18.19s.

```

• Topbar.tsx - July 8

```

[10:30] src/tests/Topbar.test.tsx
Walky Admin - Topbar Component
  ✓ calls toggleHome when toggle button is clicked (297 ms)
  ✓ shows avatar menu on click (152 ms)
  ✓ removes token and navigates to "/login" on logout (118 ms)

Test Suites: 1 passed, 1 total
Tests:      3 passed, 3 total
Snapshots:  0 total
Time:       5.763 s
Ran all test suites matching src/tests/Topbar.test.tsx.
Done in 8.75s.

```

• Students.tsx - July 8

```

[10:30] src/tests/Students.test.tsx
Walky Admin - Students Page
  ✓ renders all 4 stat widgets (375 ms)
  ✓ uses correct API calls to fetch stats (52 ms)
  ✓ displays fallback (N/A) if API fails or data is missing (124 ms)

Test Suites: 1 passed, 1 total
Tests:      3 passed, 3 total
Snapshots:  0 total
Time:       3.149 s
Ran all test suites matching src/tests/Students.test.tsx.
Done in 4.63s.

```

• StudentTable.tsx - July 9

```

[10:30] src/tests/StudentTable.test.tsx
Walky Admin - StudentTable Component
  ✓ fetches and displays rows from users API (167 ms)
  ✓ transforms API data into readable table format (52 ms)
  ✓ renders correct headers: ID, Name, Email, Joined, Last Update (32 ms)
  ✓ renders dropdown actions: Send email, Flag/Unflag user, Request edit, Request to delete (326 ms)
  ✓ toggles flag button text and row visual style when clicked (317 ms)

Test Suites: 1 passed, 1 total
Tests:      5 passed, 5 total
Snapshots:  0 total
Time:       8.311 s
Ran all test suites matching src/tests/StudentTable.test.tsx.
Done in 8.56s.

```

• ExampleAdminExample - July 8

```

[10:30] src/tests/ExampleAdminLayout.test.tsx
Walky Admin - ExampleAdminLayout.tsx
  ✓ shows sidebar on large screen and hides on small screens (326 ms)
  ✓ toggles sidebar when clicking the toggle button (57 ms)
  ✓ renders breadcrumb when present (127 ms)
  ✓ applies light theme styles correctly (65 ms)
  ✓ applies dark theme styles correctly (65 ms)
  ✓ renders footer with correct text (15 ms)
  ✓ renders correct title for route /
  ✓ shows "Dashboard" (18 ms)
  ✓ renders correct title for route /students
  ✓ shows "Students" (15 ms)
  ✓ renders correct title for route /engagement
  ✓ shows "Email" (11 ms)
  ✓ renders correct title for route /review
  ✓ shows "Review" (13 ms)
  ✓ renders correct title for route /campuses
  ✓ shows "Campuses" (9 ms)
  ✓ renders correct title for route /settings
  ✓ shows "Settings" (16 ms)

Test Suites: 1 passed, 1 total
Tests:      12 passed, 12 total
Snapshots:  0 total
Time:       3.393 s

```

• InfoStatWidget.tsx - July 8

```

[10:30] src/tests/InfoStatWidget.test.tsx
Walky Admin - InfoStatWidget Component
  ✓ renders props correctly and renders icon, value, and label (96 ms)
  ✓ applies correct dark mode styles (62 ms)
  ✓ displays tooltip with correct content on hover (85 ms)

Test Suites: 1 passed, 1 total
Tests:      3 passed, 3 total
Snapshots:  0 total
Time:       4.082 s, estimated 12 s
Ran all test suites matching src/tests/InfoStatWidget.test.tsx.
Done in 8.45s.

```

• Engagement.tsx - July 9

```

[10:30] src/tests/Engagement.test.tsx (5.413 s)
Walky Admin - Engagement Page
  ✓ renders all 3 engagement sections (487 ms)
  ✓ renders all stat widgets with correct values and labels (222 ms)
  ✓ fetches stat values from API and renders them correctly (132 ms)
  ✓ displays fallback (N/A) when API returns null or missing values (121 ms)
  ✓ applies correct background colors to each engagement section (127 ms)
  ✓ shows remaining percentage across screen (178 ms)
  ✓ makes correct API calls per section (122 ms)
  ✓ renders a widget with correct props (label, label, tooltip, color) (118 ms)
  ✓ renders correct value and label for each widget (124 ms)
  ✓ renders all stat with correct style for each widget (122 ms)
  ✓ displays tooltip on hover with correct content (129 ms)
  ✓ renders correct light mode styles (121 ms)
  ✓ applies correct dark mode styles (124 ms)

Test Suites: 1 passed, 1 total
Tests:      14 passed, 14 total
Snapshots:  0 total
Time:       8.382 s
Ran all test suites matching src/tests/Engagement.test.tsx.
Done in 18.21s.

```

- *AmbassadorsDetails.test.ts*
 - July 22

[illegible]

- *Settings.test.tsx* - July 23

```

Tests: 100% Passed, Settings-test.tsx
  Only Admin - Settings Page
    renders the Settings page with Layout and styling (124 ms)
    contains the login card and Form container (32 ms)
    renders the School Name input with label and help text (44 ms)
    renders the Email input with label and help text (24 ms)
    renders the School Location and accesses correct forms (124 ms)
    displays recommended Line and size text for Logo upload (24 ms)
    displays required Form Validator and display name and email domains (336 ms)
    calls handleEmail on Show Click and Logo uploaded form data (444 ms)
    renders the Cancel button functionally not set help text (122 ms)
    applies theme styles based on current theme (Light/Dark) (32 ms)

Test Suites: 1 passed, 1 total
Tests:      20 passed, 20 total
Snapshots:  0 total
Time: 6.62 s, estimated 7 s
Ran all test suites matching /src/tests/Settings-test.tsx.
Done in 9.469s.

```

- *Ambassadors.test.tsx*
- July 17

[illegible]

- *VerifyCode.test.tsx*
- July 29

```

[10:25] root@kali:~# verifyCode.test.tac
Kali Admin - VerifyCode Page
- renders login and title: render Verification Code (100 ms)
- renders login verification code inputs (111 ms)
- shows the verify Code button (4 ms)
- renders the reset password button with default state (8 ms)
- calls handleVerifyCode and logs the payload (191 ms)
- attempts to reset login on successful verifyCode login (1 ms)
- displays error on invalid code (78 ms)
- displays the response button when checkbox is active (100 ms)
- displays updated countdown time (136 ms)
- renders login title when checkbox is active (66 ms)
- renders password and confirm password inputs when step is "reset" (108 ms)
- shows the login button (10 ms)
- shows the reset password button when step is "login" (77 ms)
- shows error if passwords do not match (14 ms)
- shows error if password is less than 8 characters (113 ms)
- submits valid password reset and verifies to login (139 ms)

Test Results: 1 passed, 1 total
Tests: 1 passed, 1 total
Snapshots: 0 total
Time: 0.212 s
Ran all test suites matching /src/test/verifyCode.test.tac/.

```

- *All Tests*

- *Review.tsx* - July 9

```

PASS: src/tests/Review.test.tsx
Walky Admin - Review Page
  ✓ renders the Review page container with proper padding (243 ms)
  ✓ includes the ReviewTable component (12 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        2.746 s
Ran all test suites matching src/tests/Review.test.tsx.
* Done in 4.60s.

```

- *Campuses.tsx* - July 9

```

PASS: src/tests/Campusus.test.tsx
Kelly Admin - Campusus Page
  ✓ renders the Campusus page container with proper layout and spacing (512 ms)
  ✓ displays title/header: Campus Management (61 ms)
  ✓ renders the + Add Button (36 ms)
  ✓ includes the CampusTable component (Table) (45 ms)

Test Suites: 1 passed, 1 total
Tests:       4 passed, 4 total
Snapshots:  0 total
Time:        4.803 s
Ran all test suites matching src/tests/Campusus.test.tsx.
Done: 4.83s.

```

- *CampusDetails.test.tsx* - July 16

[illegible]

- *AmbassadorView.test.tsx*
- July 22

```

PASS src/tests/AmbassadorView.test.tsx
  Walky Admin - Ambassadors Page: AmbassadorView
    ✓ loads AmbassadorDetails with correct props based on route param (67 ms)
    ✓ passes ambassadorId and disables tab view (19 ms)
    ✓ renders AmbassadorDetails inside padded container (13 ms)

Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:  0 total
Time:        4.26 s
Ran all test suites matching src/tests/AmbassadorView.test.tsx.

```

- *CampusTable.test.tsx* - July 16

```

PASS src/tests/CampusTable.test.ts
  MyCity Admin > CampusTable Component
    ✓ renders and displays campus data from /campuses API (289 ms)
    ✓ renders correct table headers (179 ms)
    ✓ renders each row with campus name, ID, location, address, status, and actions (83 ms)
    ✓ renders the Campus Boundary map with polygon boundaries and toggle button (297 ms)
    ✓ Triggers edit action and navigates to campus view with correct state (119 ms)
    ✓ renders the Delete button for each campus row (78 ms)
    ✓ Handles 404 error and displays error message and alert (77 ms)
    ✓ applies correct structure and renders responsive elements (96 ms)

Test Suites: 1 passed, 1 total
Tests: 6 passed, 6 total
Snapshots: 0 total
Time: 0.433 s
How all test results are matching src/tests/CampusTable.test.ts.
  ✓ Done in 8.58s.

```

- *Places.test.tsx* - July 23

[illegible]

- *ForgotPassword.test.ts*
 - July 29

```

root@kali:~/FergoPassword# ./FergoPassword Page
# renders the Fergo Password page with layout and styling (254 ms)
# displays titleHeader FergoYour password (28 ms)
# renders the form when submitted is false (19 ms)
# shows confirmation message when submitted is true (235 ms)
# calls API.post and switches to verifyCode with mail in state (146 ms)
# handles API failure and displays fallback error (282 ms)

Test Suites: 1 passed, 1 total
Tests:       6 passed, 6 total
Snapshots:   0 total
Time:        4.75 s
Ran all test suites matching src/tests/FergoPassword.test.jsx.
Done in 6.56s

```

- *PlaceTypes.test.tsx* - July 23

[illegible]


```
Test Suites: 23 passed, 23 total
Tests:       187 passed, 187 total
Snapshots:   0 total
Time:        6.673 s
Ran all test suites.
✨ Done in 7.56s.
```