



**Dhirubhai Ambani  
University  
Technology**

Formerly DA-IICT

**IT314 - Software Engineering  
Group - 20**



**KRUSHISETU**  
**PATH TO PROSPERITY**

**Assignment by :**  
**Prof. Saurabh Tiwari**

## **Group Members :**

<b>Student ID</b>	<b>Name</b>
202301205	Jainil Patel
202301215	Jay Patoliya
202301226	Priyanshi Gothi
202301255	Rudra Desai
202301207	Tarang Hirapara
202301208	Harshil Vasava
202301238	Jeel Prajapati
202301245	Ishan Thakkar
202301241	Jal Khunt
202301206	Meet Patel
202301248	Farzan Bhalara

**Github Repository Link:** **KrushiSetu**

**Introduction:** The objective of unit testing in the KrushiSetu project is to verify that individual components, functions, and modules of the system work as expected.

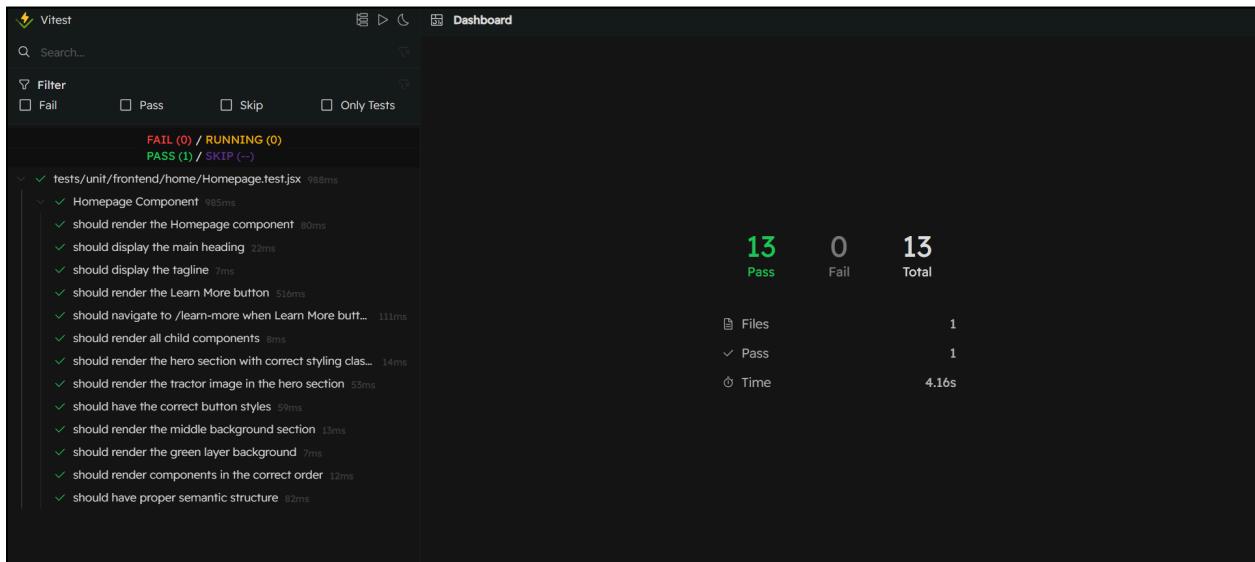
## Unit testing ensures:

- Correctness of UI components
- Reliable logic
- Prevention of regression bugs
- Improved maintainability

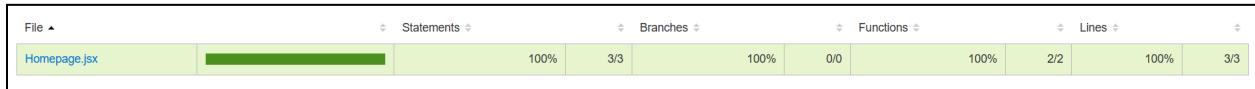
For the frontend, we use **Vitest + React Testing Library**.

## Test Execution and Coverage Report :

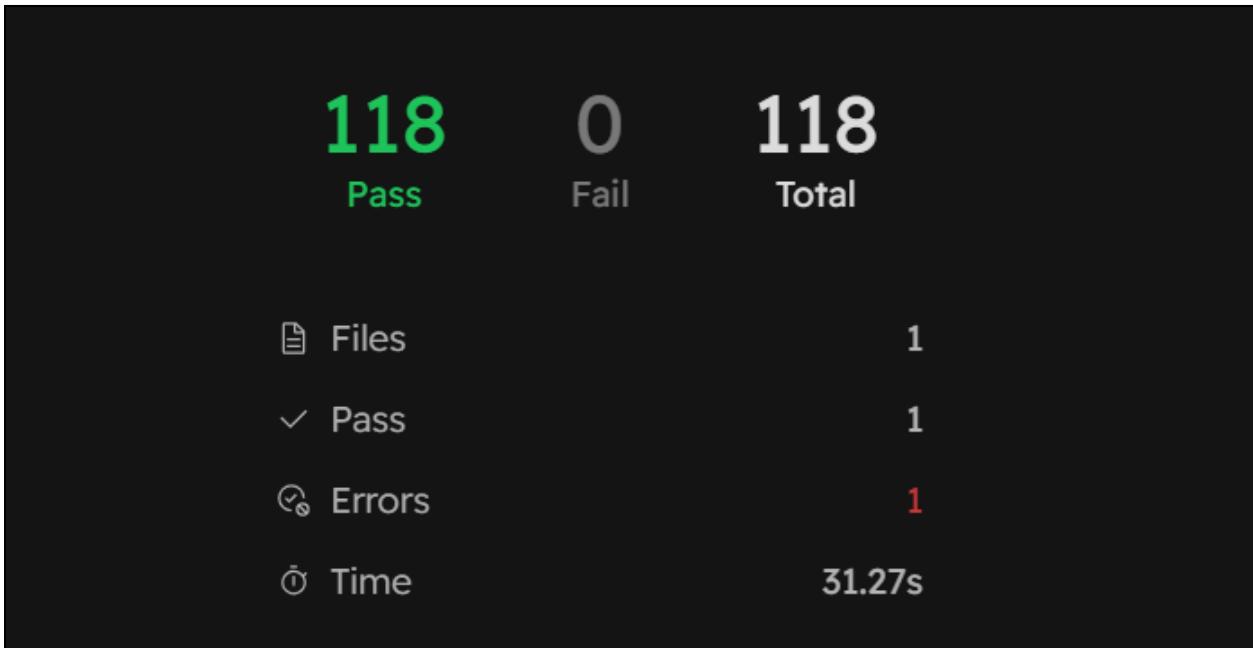
### 1. Homepage:



### Coverage Report:



## 2. ApplySubsidy (Application Form):



### Coverage Report :



We tested the full multi-step form and got **77.8% statement coverage** with **118 passing tests**. This means that most of the important parts of the form are fully tested.

### What we tested:

- All the fields the user fills in
- Validations (required fields, formats, limits, etc.)
- Moving between steps (Next/Back buttons)
- Final form submission
- Handling of files and uploads (Documents step)

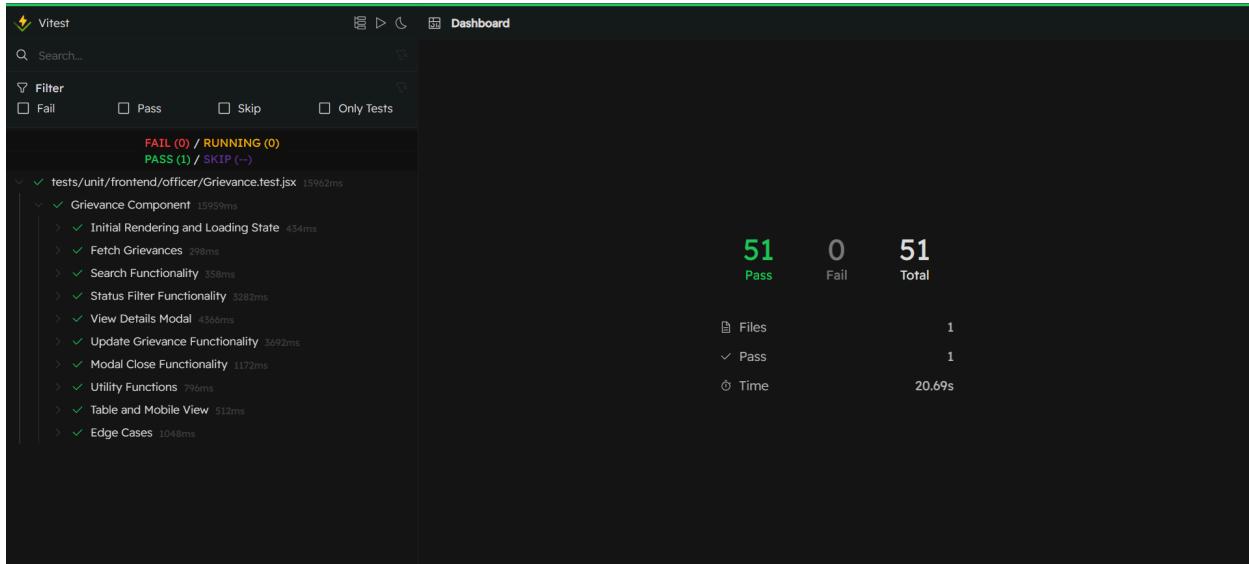
The remaining code includes:

- Extra safety/error handling conditions

- Rare cases that happen only if something goes wrong unexpectedly

Testing these parts can make the test code **complicated and unstable**, even though they rarely affect real users.

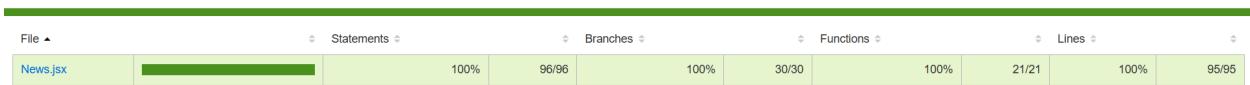
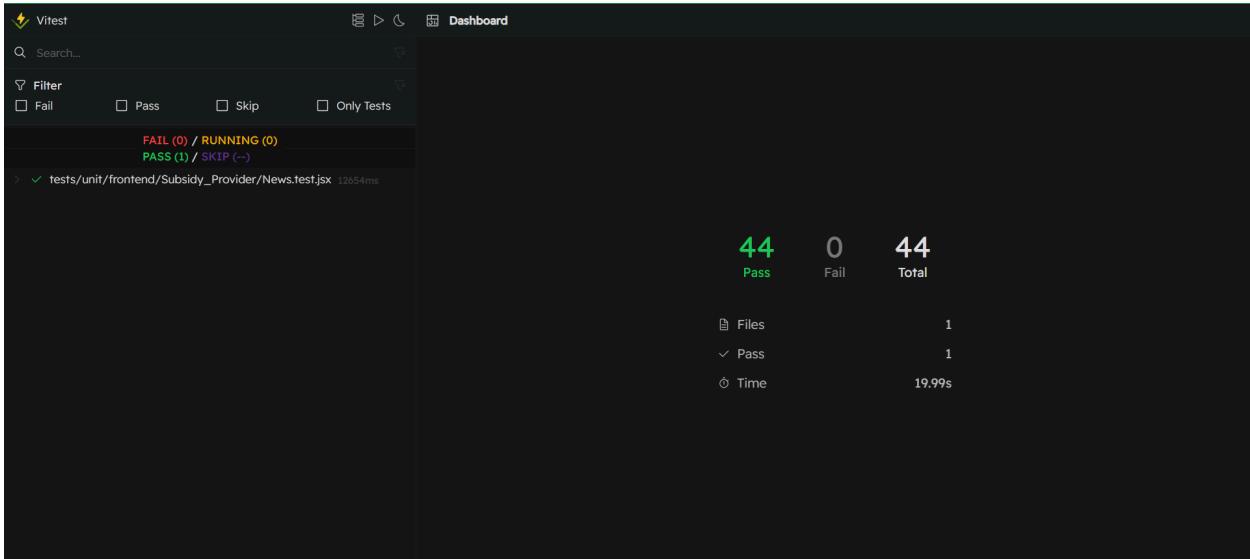
### 3. Grievance resolution page (Officer):



100% coverage could not be achieved because the remaining lines are **defensive checks** that are **not part of the actual user flow**. Testing them would add **unnecessary, unstable test cases without improving functionality**.

The current coverage already tests **all real features the user interacts with**, making further testing impractical and not meaningful.

## 4. Post News page (Provider)



## 5. Dashboard

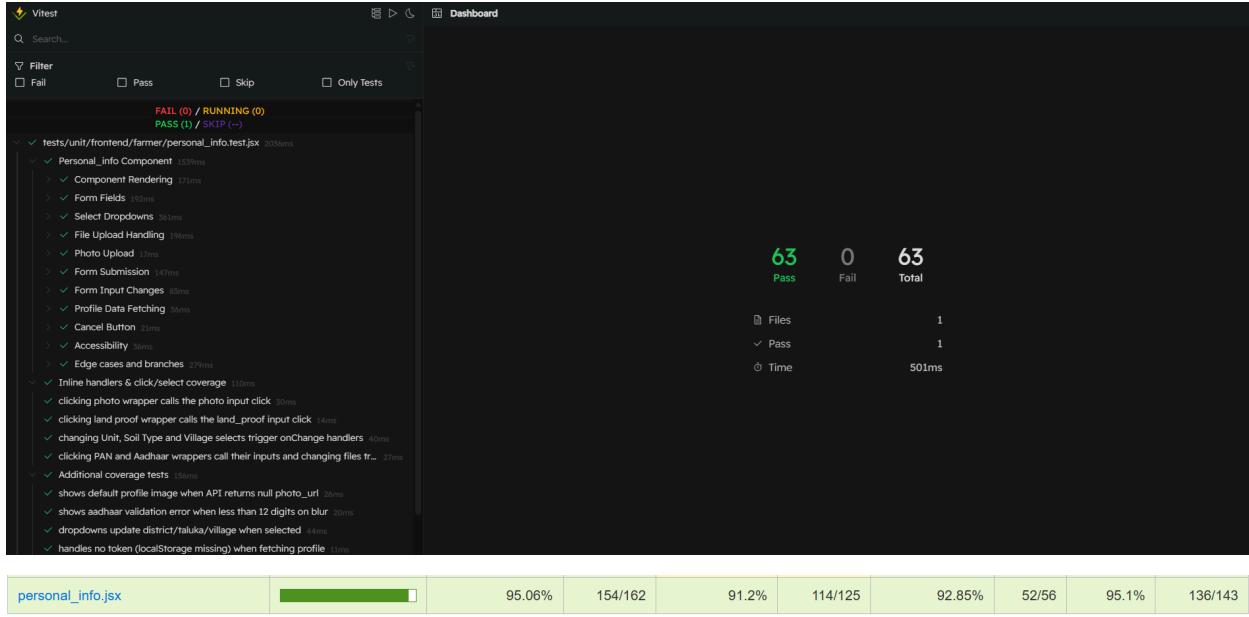
The screenshot shows the Vitest dashboard interface. On the left, a sidebar lists test files and their results. The main area displays summary statistics: 26 Pass, 0 Fail, and 26 Total tests. Below this, coverage details are shown for a single file, Dashboard.jsx. The coverage table includes columns for File, %Stmts, %Branch, %Funcs, %Lines, and Uncovered Line #s. The file Dashboard.jsx has 97.05% statement coverage, 93.33% branch coverage, 100% function coverage, 96.96% line coverage, and 60 uncovered lines. A message at the bottom indicates "Waiting for file changes...".

File	% Stmt	% Branch	% Func	% Lines	Uncovered Line #s
All files	97.05	93.33	100	96.96	
Dashboard.jsx	97.05	93.33	100	96.96	60

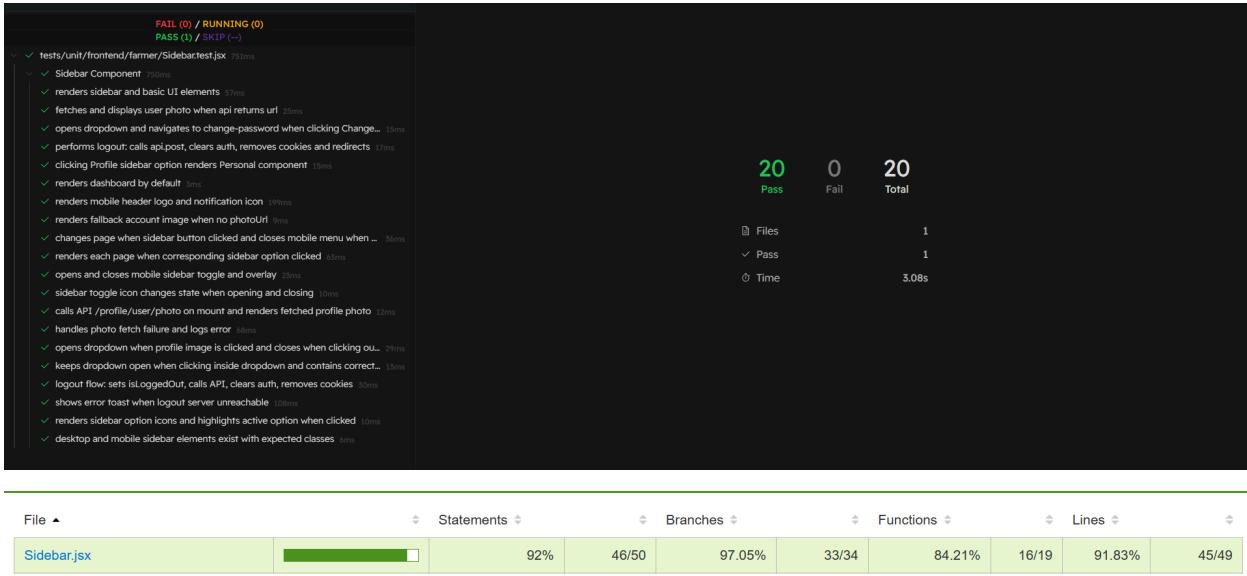
Branch coverage is slightly below 100% (93.33%) due to a small number of defensive branches in the data-fetching logic that are not executed during testing. These branches are triggered only under particular conditions such as non-OK responses with valid JSON or alternate environment configurations for backend URL selection. While functionally correct, testing these branches would require unrealistic mock setups that do not reflect actual usage and would reduce test stability. All core

execution paths and relevant UI behavior are thoroughly tested, so the uncovered branches have no impact on reliability.

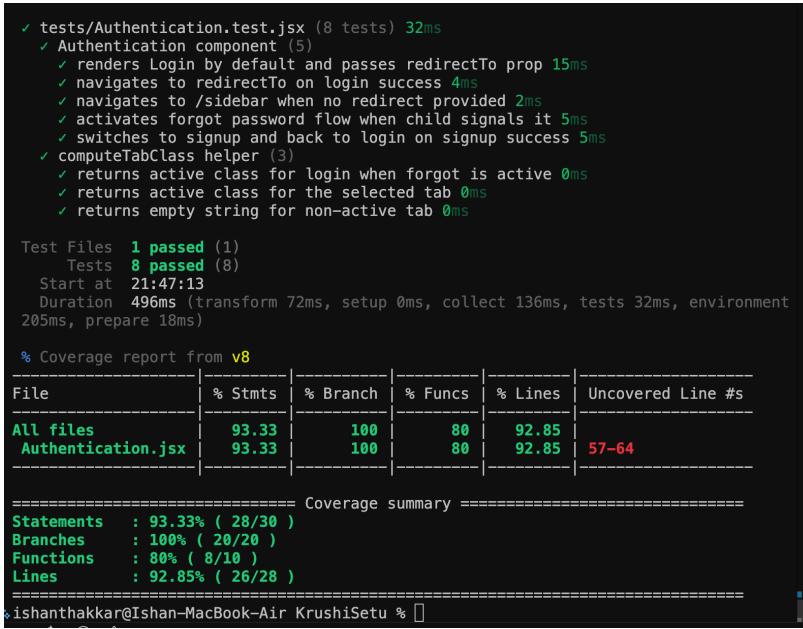
## 6. Personal\_info

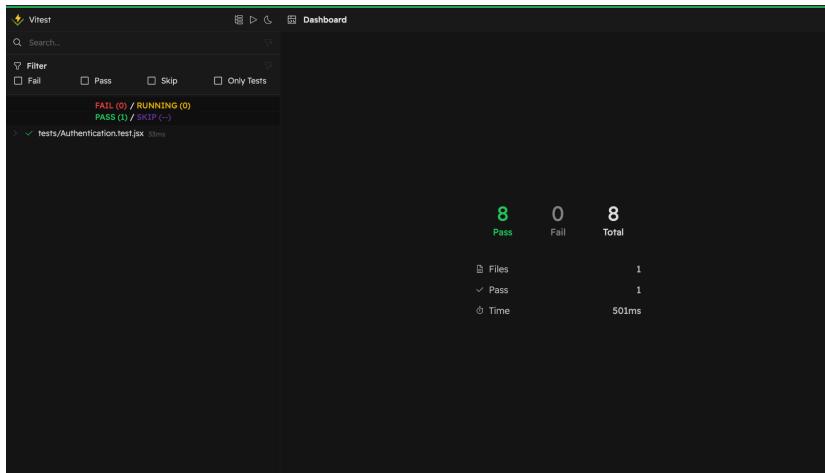


## 7. Sidebar

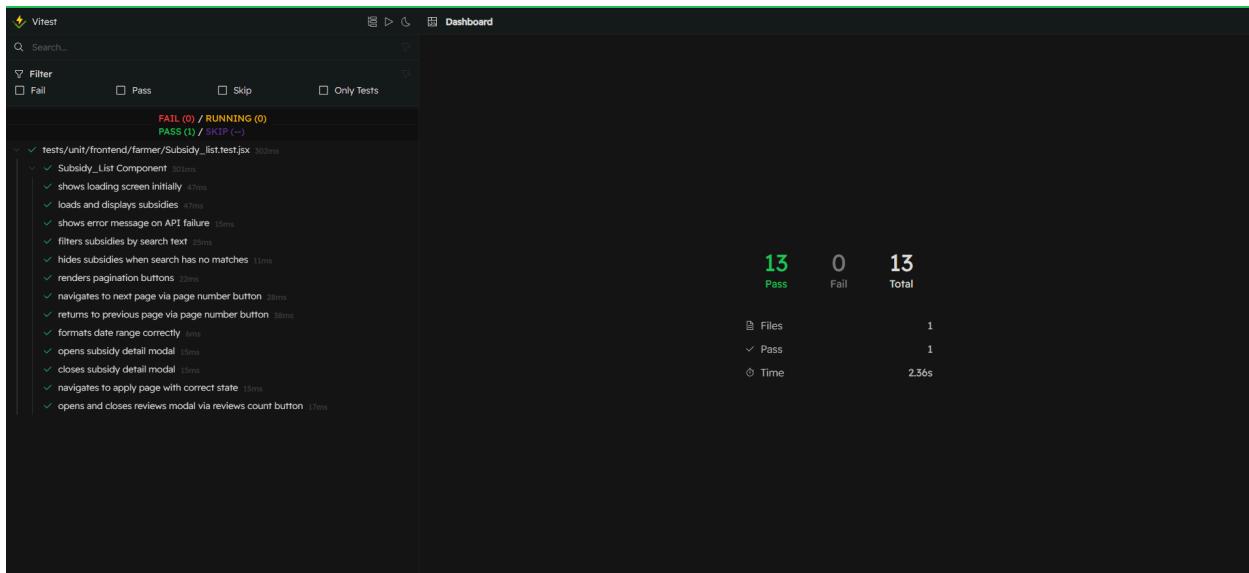


## 8. Authentication





## 9. Subsidy\_List



File ▲	Statements ▲	Branches ▲	Functions ▲	Lines ▲
Subsidy_List.jsx	100%	49/49	95%	19/20

## 10.SubsidyRecommendation.test

Vitest

Search...

Filter

Fail  Pass  Skip  Only Tests

**FAIL (0) / RUNNING (0)**  
**PASS (1) / SKIP (-)**

tests/unit/frontend/farmer/SubsidyRecommendation.test.jsx 1504ms

- ✓ RecommendSubsidy – STEP 1 367ms
  - ✓ renders main heading 44ms
  - ✓ renders Basic Information section 14ms
  - ✓ updates Step 1 inputs 172ms
  - ✓ validates Step 1 44ms
  - ✓ moves to Step 2 when valid 56ms
  - ✓ prefills profile data 5ms
  - ✓ tries alternate fallback endpoints 12ms
- ✓ RecommendSubsidy – STEP 2 791ms
  - ✓ renders Step 2 4ms
  - ✓ district is disabled when no state 48ms
  - ✓ enables district after selecting state 39ms
  - ✓ populates districts once state is selected 212ms
  - ✓ validates missing Step 2 fields 44ms
  - ✓ submits Step 2 and proceeds to results 65ms
  - ✓ shows loading while submitting 44ms
  - ✓ handles API error – Network 51ms
  - ✓ handles 400 API error 70ms
  - ✓ handles custom API error 53ms
  - ✓ handles API response with success:false 49ms
  - ✓ handles 503 API error 39ms
- ✓ RecommendSubsidy – STEP 3 345ms
  - ✓

**28** Pass   **0** Fail   **28** Total

Files 1  
Pass 1  
Time 4.36s

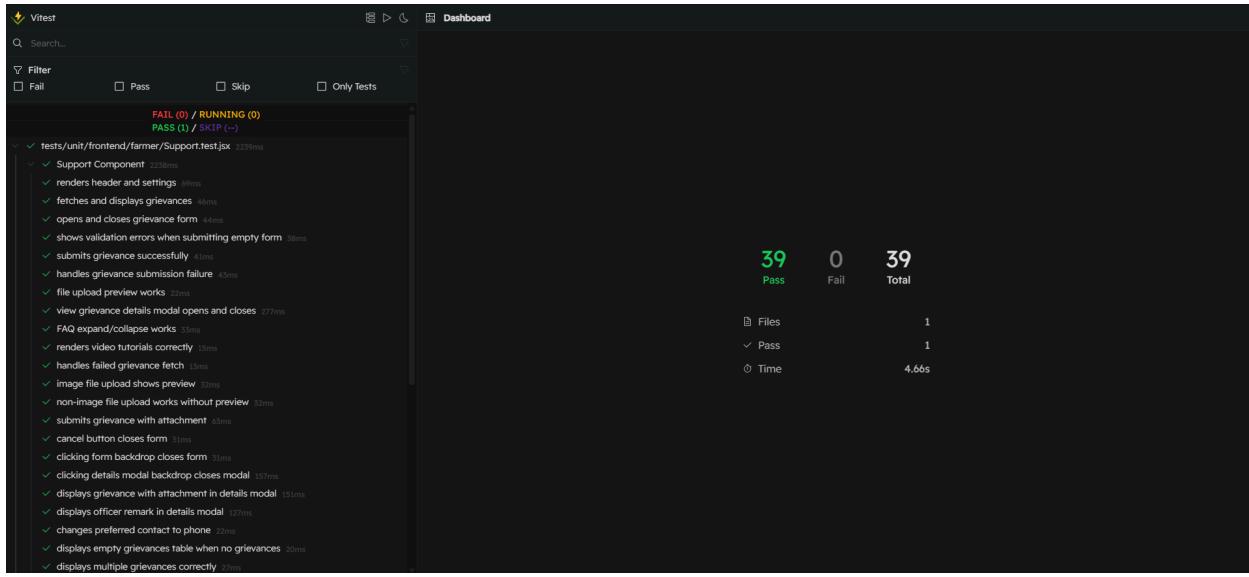
---

File ▲

Statements ▲ Branches ▲ Functions ▲ Lines ▲

SubsidyRecommendation.jsx	92.85%	117/126	72.66%	109/150	90.32%	28/31	93.27%	111/119
---------------------------	--------	---------	--------	---------	--------	-------	--------	---------

## 11. Support



File	Statements	Branches	Functions	Lines
Support.css	0%	0/0	0%	0/0
Support.jsx	90.74% ███████████	98/108	79.45% ███████████	88.88% ███████████

## 12. Forgot Password

The screenshot shows the Vitest dashboard interface. At the top, there's a search bar and filter options for 'Fail', 'Pass', 'Skip', and 'Only Tests'. The main area displays a tree view of test cases under 'ForgotPassword component'. Most tests are marked with a green checkmark, indicating they passed. The total summary at the bottom right shows 30 Pass, 0 Fail, and 30 Total tests, with a total execution time of 1.34s.

Category	Count
Pass	1
Fail	1
Total	30

Time: 1.34s

% Coverage report from v8

File	% Stmt	% Branch	% Funcs	% Lines	Uncovered	Line #s
All files	95.95	94	93.75	95.5		
ForgotPassword.jsx	95.95	94	93.75	95.5	54-55,84,125	

❖ ishanthakkar@Ishan-MacBook-Air KrushiSetu %