**CS673 Software Engineering** 

**Team 5 - CVCoach**

**Software Test Document**

| Team Member | Role(s) | Signature | Date |
| --- | --- | --- | --- |
| [Linchen Xu](mailto:xlc98@bu.edu) | Team Leader, Design and Implementation Leader | *Linchen Xu* | 09/14/2024 |
| [Lin Ma](mailto:adamma@bu.edu) | Configuration Leader | *Lin Ma* | 09/14/2024 |
| [Zihan Zhou](mailto:zhzhjycs@bu.edu) | Security Leader | *Zihan Zhou* | 09/14/2024 |
| [Zhen Cao](mailto:caoz229@bu.edu) | QA Leader | *Zhen Cao* | 09/14/2024 |
| [Haochen Sun](mailto:haocsun@bu.edu) | Requirement Leader | *Haochen Sun* | 09/19/2024 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Revision history**

| **Version** | **Author** | **Date** | **Change** |
| --- | --- | --- | --- |
| **1.0.0** | [Linchen Xu](mailto:xlc98@bu.edu)  [adamma@bu.edu](mailto:adamma@bu.edu)  [caoz229@bu.edu](mailto:caoz229@bu.edu)[zhzhjycs@bu.edu](mailto:zhzhjycs@bu.edu) | **10/17/2024** | **Create initial document** |
| **2.0.0** | [caoz229@bu.edu](mailto:caoz229@bu.edu) | **11/07/2024** | **Update document** |

[Testing Summary](#_heading=h.gjdgxs)

[Manuel Tests Reports](#_heading=h.30j0zll)

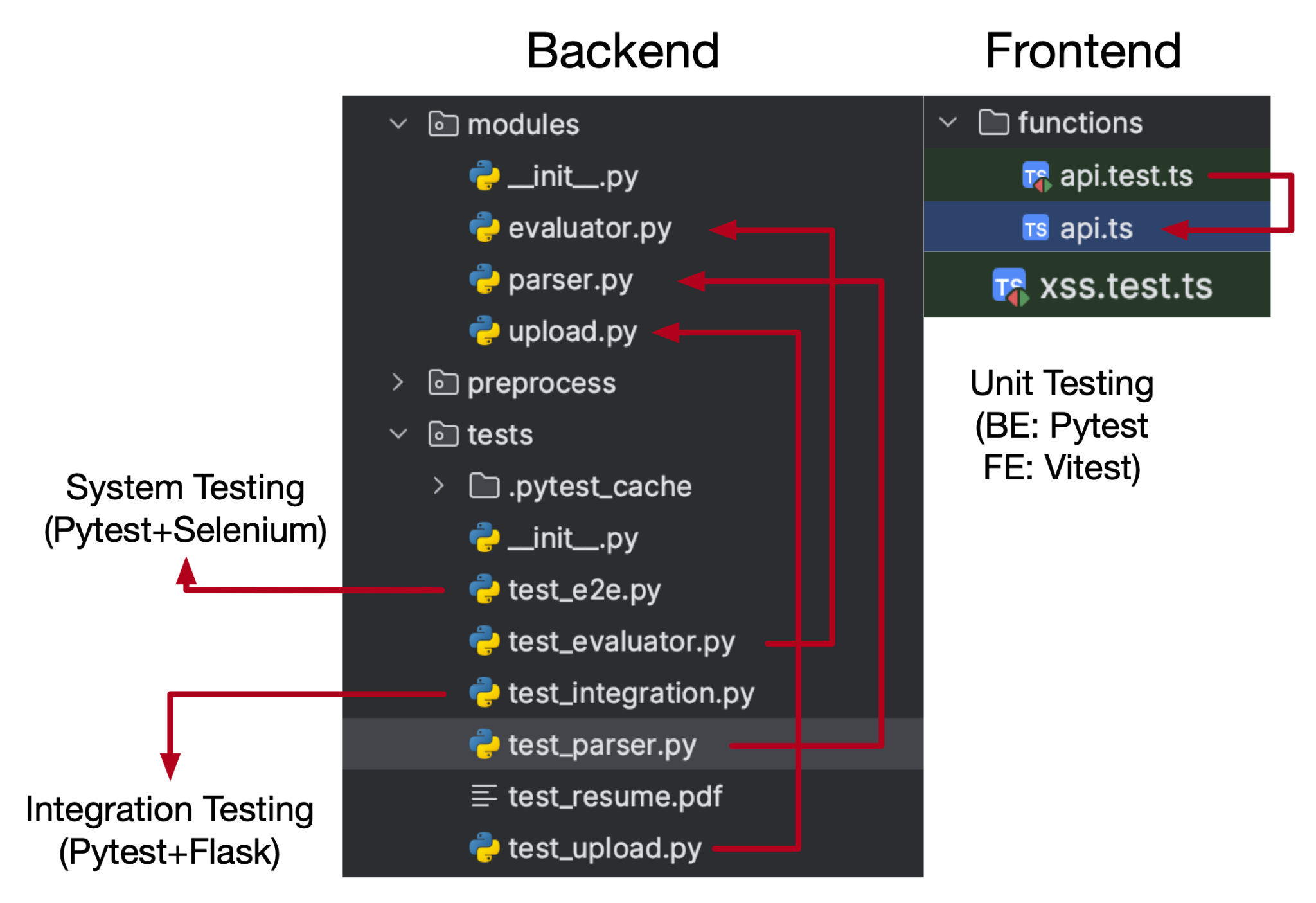
[Automated Testing Reports](#_heading=h.1fob9te)

[Testing Metrics](#_heading=h.3znysh7)

[References](#_heading=h.2et92p0)

[Glossary](#_heading=h.tyjcwt)

# Testing Summary

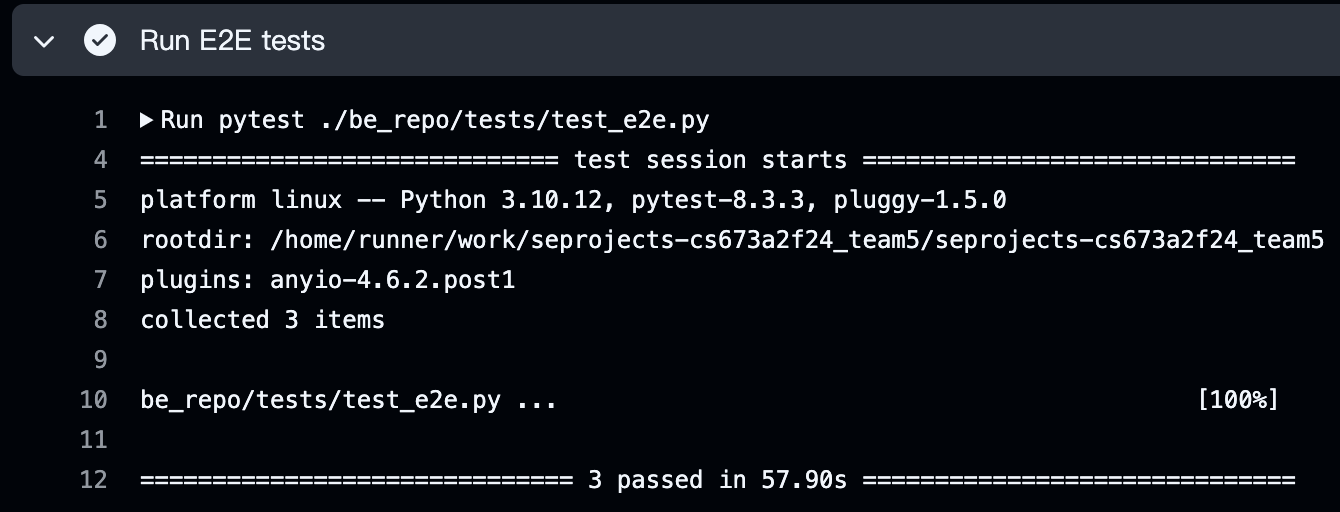


**Testing Levels and Techniques**

1. **Unit Testing (Each Team Member)**
   1. Backend: Conducted with **pytest**
   2. Frontend: Conducted with **Vitest**
   3. Target files for unit testing include evaluator.py, parser.py, upload.py in the backend and api.test.ts, xss.test.ts in the frontend.
2. **Integration Testing (QA Leader)**
   1. Uses **pytest** in combination with **Flask** for integration between backend modules, ensuring that components work together well/
   2. Test files is test\_integration.py.
3. **System Testing (QA Leader)**
   1. Uses **Pytest** combined with **Selenium** to do the end-to-end testing.
   2. System Testing is automated now on github action.
   3. Environment and Configuration
      1. **Operating System**: Ubuntu (latest version on GitHub Actions)
      2. **Frontend Service**: Image Node.js 18
      3. **Backend Service**: Image Python 3.9
      4. ​​**Google Chrome**: Latest version available during the CI run
   4. This level of testing aims to validate the end-to-end functionality of user interaction with browsers.
   5. Test files is test\_e2e.py.
4. **Regression Testing** 
   1. All the unit testing, integration testing and system testing is automated done on github action when code with specific functionality changes.
5. **Acceptance Testing**
   1. Based on the user story, it is done manually. We upload the resume and input the job and evaluate the results including the score and analysis by our experience.

**Testing Results**

1. **Unit Testing and Integration Testing (11 passed, 0 failed)**
2. **System Testing (3 passed, 0 failed)**

****

# Manual Testing Report

Since all unit tests, integration tests and system tests are automated, we only do manual test for acceptance testing.

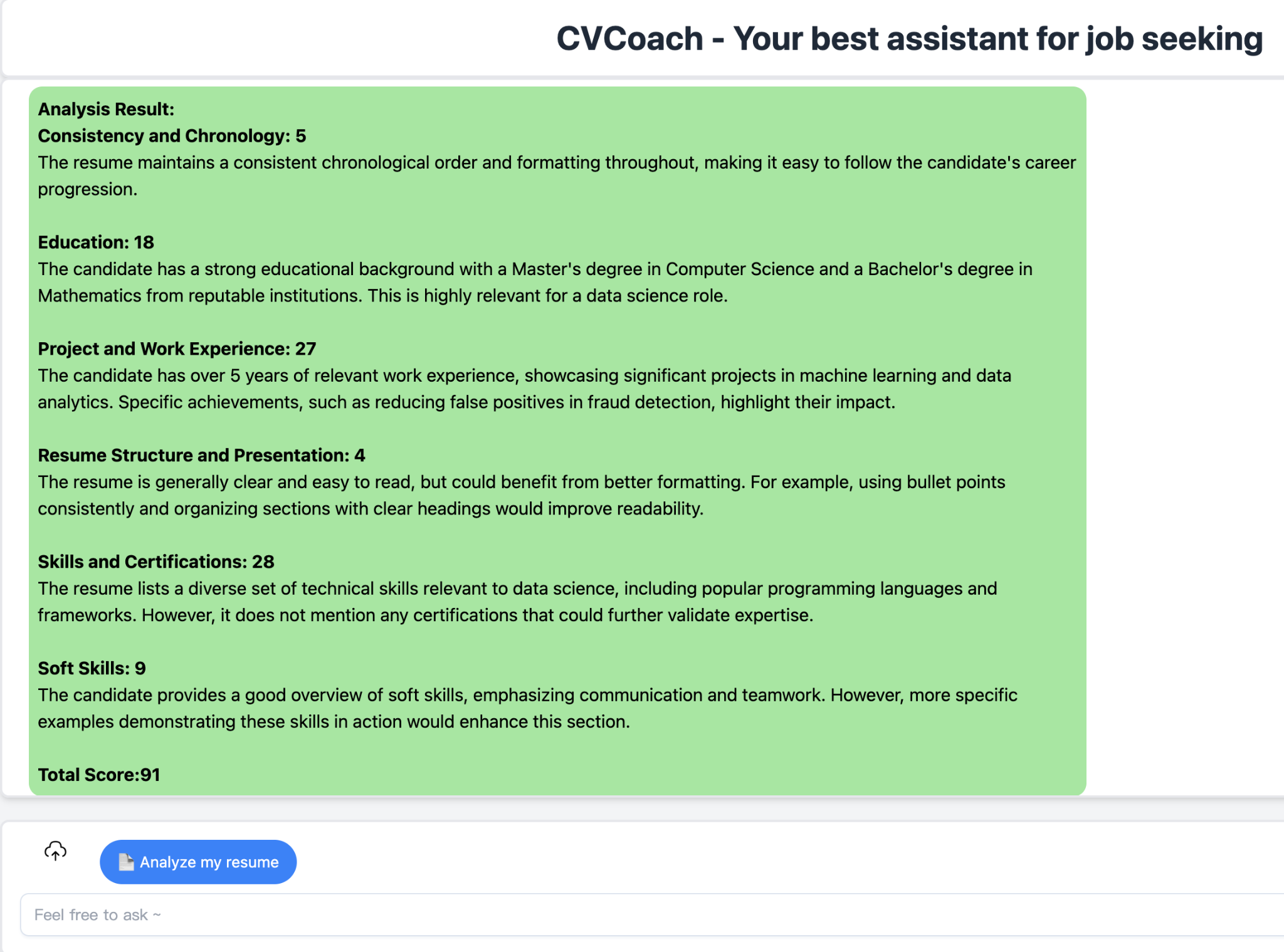
#### **Test Case ID: TC-001**

#### **Name: Verify Resume Analysis**

**New or Old**: New  
**Test Items**: Resume analysis functionality  
**Test Priority**: High  
**Dependencies**: None  
**Preconditions**: User must be registered in the system  
**Input Data**: resume.pdf

**Test Steps**:

1. Navigate to the login page.
2. Google Login
3. Upload resume.pdf
4. Click analysis button and submit

**Postconditions**: None  
**Expected Output**: Resume analysis with both weighted score and analysis for each criteria including Education, Project and Work Experience, Skills and Certifications, Soft Skills, Resume Structure and Presentation and Consistency and Chronology.  
**Actual Output**:   
**Pass or Fail**: Pass  
**Bug ID/Link**: N/A  
**Additional Notes**: None

—------------------------------------------------------------------------------------------------------------------

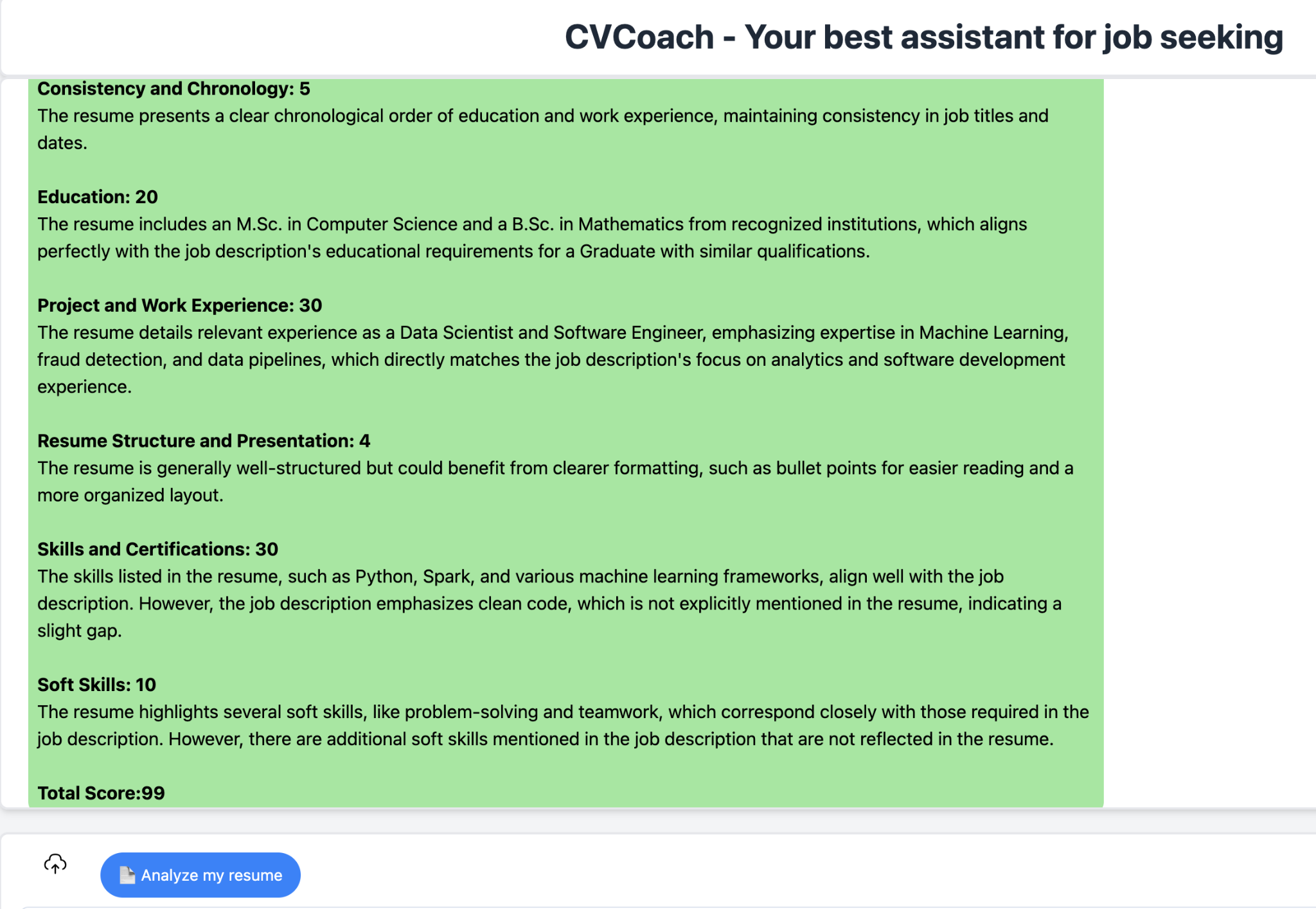
#### **Test Case ID: TC-002**

#### **Name: Verify Resume Analysis with JD**

**New or Old**: New  
**Test Items**: Resume analysis with JD functionality  
**Test Priority**: High  
**Dependencies**: None  
**Preconditions**: User must be registered in the system  
**Input Data**: resume.pdf and job description

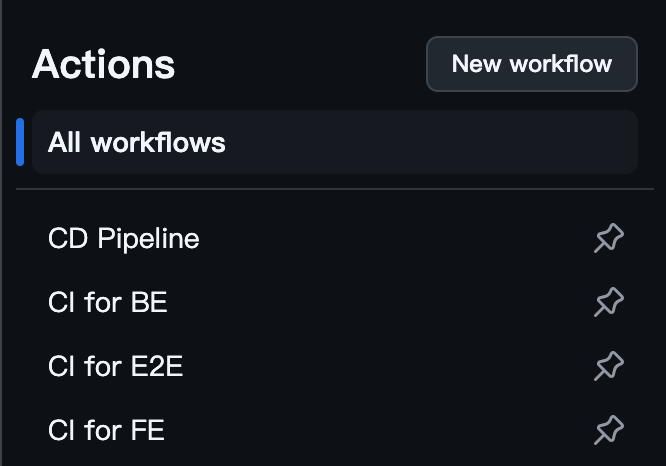
**Test Steps**:

1. Navigate to the login page.
2. Google Login
3. Upload resume.pdf
4. Click analysis button
5. Enter the job description and submit

**Postconditions**: None  
**Expected Output**: Resume analysis with both weighted score and analysis for each criteria including Education, Project and Work Experience, Skills and Certifications, Soft Skills, Resume Structure and Presentation and Consistency and Chronology. Also the analysis is more job related compared to the resume analysis without JD.  
**Actual Output**:  
**Pass or Fail**: Pass  
**Bug ID/Link**: N/A  
**Additional Notes**: None

# Automated Testing Report

Automated testing is at the core of our CI/CD pipeline and enables us to maintain high-quality standards with every code change.



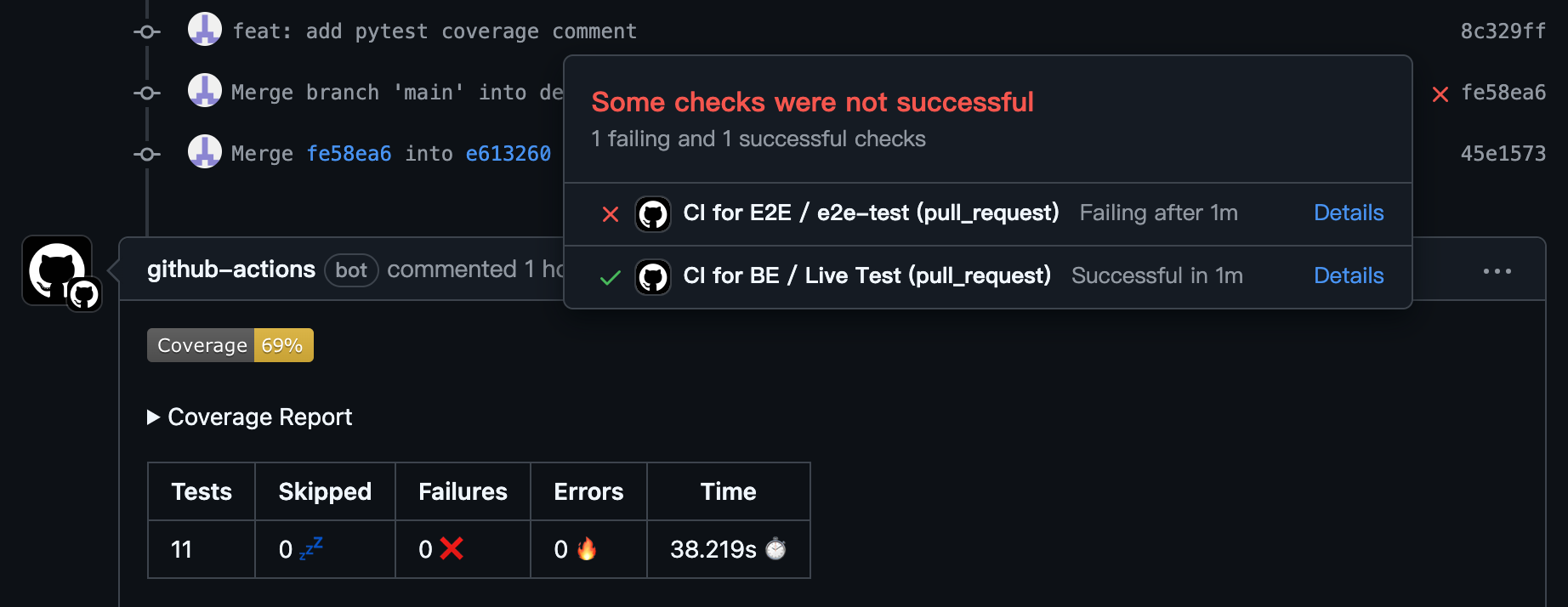
**Testing Levels and Techniques**

1. **Unit Testing (Each Team Member)**
   1. Backend: Conducted with **pytest**
   2. Frontend: Conducted with **Vitest**
   3. Test files include test\_evaluator.py, test\_parser.py, test\_upload.py in the backend and api.test.ts, xss.test.ts in the frontend.
2. **Integration Testing (QA Leader)**
   1. Uses **pytest** in combination with **Flask** for integration between backend modules, ensuring that components work together well/
   2. Test files is test\_integration.py.
3. **System Testing (QA Leader)**
   1. Uses **Pytest** combined with **Selenium** to do the end-to-end testing.
   2. System Testing is automated now on github action.
   3. Environment and Configuration
      1. **Operating System**: Ubuntu (latest version on GitHub Actions)
      2. **Frontend Service**: Image Node.js 18
      3. **Backend Service**: Image Python 3.9
      4. **Google Chrome**: Latest version available during the CI run
   4. Test files is test\_e2e.py.

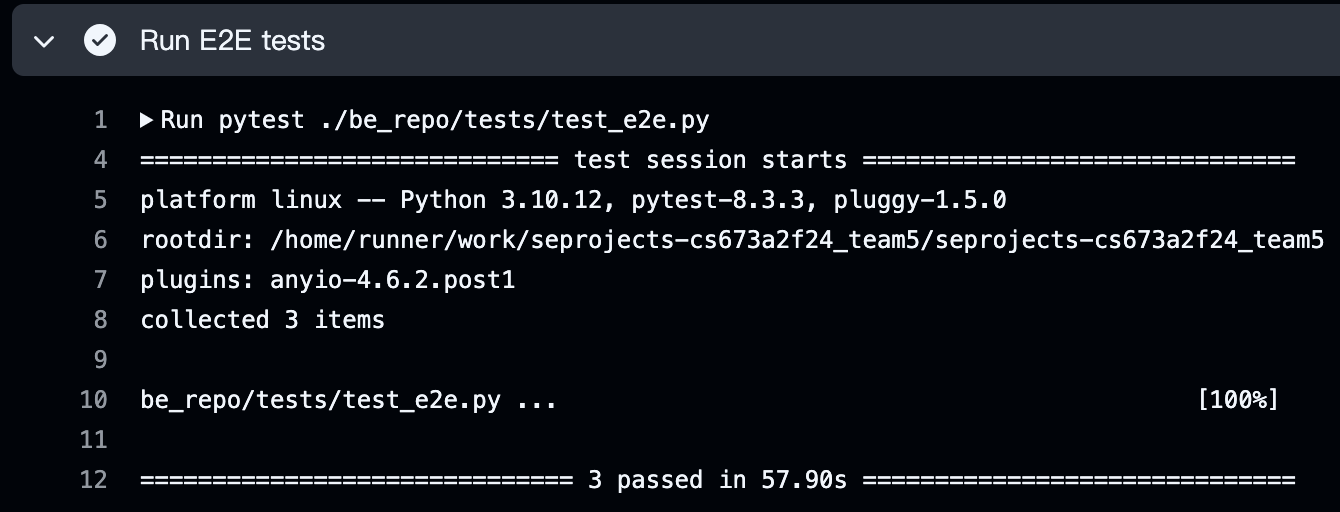
**Testing Results**

1. **Unit Testing and Integration Testing (11 passed, 0 failed)**

As part of our CI/CD process, we use **Pytest Coverage Comments** to monitor our code coverage and test results for each pull request. This integration with GitHub Actions provides immediate feedback. In this example, you can see the coverage is reported at **69%**. Additionally, we get a breakdown of the number of tests run, along with any failures, errors, and the time taken.

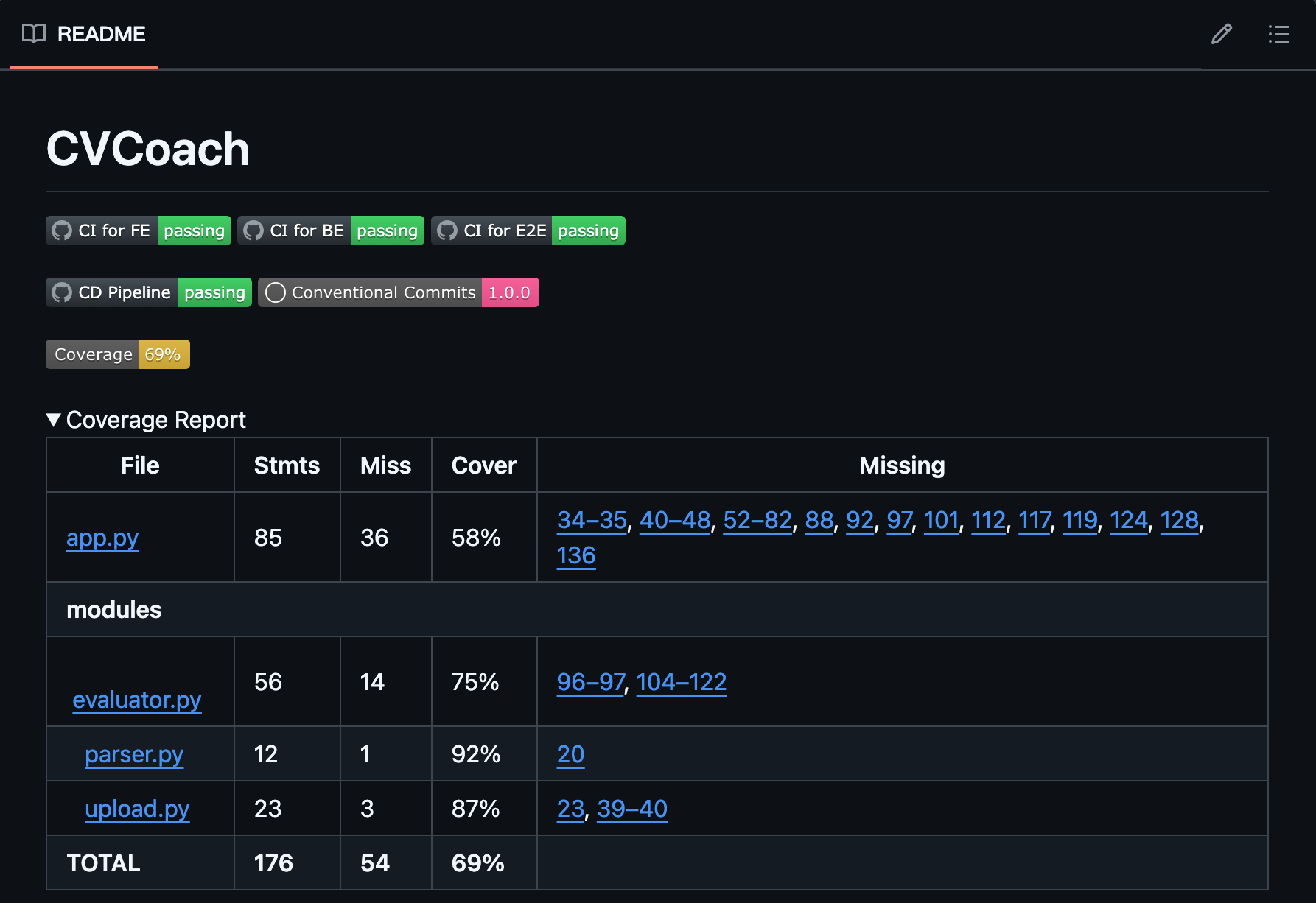
****

1. **System Testing (3 passed, 0 failed)**

****

1. **Auto Updating Badge and coverage report in README**

To provide quick insights into our project’s health, we have an **Auto Updating Badge and Coverage Report** directly on our **README** file. This setup allows anyone visiting our repository to understand the state of our tests and code coverage.

****

# Testing Metrics

| Metric Name | Results |
| --- | --- |
| Code Coverage | Stmts 176, Miss 54, Cover 69% |
| # of Test Cases  Test Case Pass Rate | 11 passed, 0 failures |
| # of User Stories Completed | 4 |
| Time Taken | 38s each unit+integration testing,  2m1s each system testing. |
| # defects rate | 0 |

# References

<https://github.com/MishaKav/pytest-coverage-comment/tree/main>

<https://github.com/ad-m/github-push-action>