```
def skills_123(self):

    # Load the skills data file
    skills = self.skills

    # Load data from pyresparser
    pyres_data, pyres_text = self.extract_resume()
    self.data = pyres_data
    self.text = pyres_text

    ocr_ser = pd.Series(pyres_text)
    cleaned_words = hero.clean(ocr_ser)

    main_df = pd.DataFrame(cleaned_words[0].split(), columns = ['text'])
    self.clean_data = main_df

    words = len(main_df)

    # Details
    columns = ['filename', 'name', 'mobile_number', 'email', 'company_names',
            'college_name', 'experience', 'skills', 'experience_age',
            'degree', 'words',
            'primary_score', 'primary_match',
            'secondary_score', 'secondary_match',
            'no_of_pages', 'document_similarity']
    details = pd.DataFrame(columns = columns)

    # Add the primary match and score
    pri_score, pri_match = self.find_match(main_df, skills[['Primary']])
    sec_score, sec_match = self.find_match(main_df, skills[['Secondary']])

    # Add the document similarity score
    doc_sim = self.resume_cosine_score(cleaned_words[0])

    # Add details in a dataframe
    details.loc[0, 'filename'] = self.resume
    details = self.fill_data(details, pyres_data, 'name')
    details = self.fill_data(details, pyres_data, 'mobile_number')
    details = self.fill_data(details, pyres_data, 'email')
    details = self.fill_data(details, pyres_data, 'company_names')
    details = self.fill_data(details, pyres_data, 'college_name')
    details = self.fill_data(details, pyres_data, 'degree')
    details = self.fill_data(details, pyres_data, 'experience')
    details = self.fill_data(details, pyres_data, 'skills')
    details.loc[0, 'words'] = words

    if pyres_data['no_of_pages'] == None:
```

```
            details.loc[0, 'no_of_pages'] = 0
        else:
            details = self.fill_data(details, pyres_data, 'no_of_pages')
        details.loc[0, 'primary_score'] = pri_score
        details.loc[0, 'primary_match'] = str(pri_match)
        details.loc[0, 'secondary_score'] = sec_score
        details.loc[0, 'secondary_match'] = str(sec_match)
        details.loc[0, 'document_similarity'] = int(doc_sim)

        if pyres_data['total_experience'] > 0:
            details.loc[0, 'experience_age'] = pyres_data['total_experience']
        else:
            details.loc[0, 'experience_age'] = np.NaN

        details['no_of_pages'] = details['no_of_pages'].astype(int)

        return details
```

Rest of the code is available here: **https://github.com/YashZauwar/SoftwareDevelopmentforAI**