

```
expect.extend({
  toHaveGPA(actual, GPA) {
    const { grades } = actual.data.user;

    const actualGPA =
      Object.keys(grades)
        .map(course => BelAir.computeGpa(grades[course]))
        .reduce((total, grade) => (total += grade)) /
      Object.keys(grades).length;

    const { first, last } = actual.data.user.name;
    const name = `${first} ${last}`;

    const pass = actualGPA >= GPA;

    const message = pass
      ? () => `${name} should not have GPA ${gpa}` // message for .not
      : () => `${name} should have a ${GPA} GPA, not ${actualGPA}`;

  }
});
```

```
expect.extend({
  toHaveGPA(actual, GPA) {
    const { grades } = actual.data.user;

    const actualGPA =
      Object.keys(grades)
        .map(course => BelAir.computeGpa(grades[course]))
        .reduce((total, grade) => (total += grade)) /
      Object.keys(grades).length;

    const { first, last } = actual.data.user.name;
    const name = `${first} ${last}`;

    const pass = actualGPA >= GPA;

    const message = pass
      ? () => `${name} should not have GPA ${gpa}` // message for .not
      : () => `${name} should have a ${GPA} GPA, not ${actualGPA}`;

    return { actual: actual, message, pass };
  }
});
```