

```

test("ashley banks", async () => {
  const response = await BelAir.request("ashley");

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

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

  expect(response.data.user).toMatchSnapshot();
});

test("carlton banks", async () => {
  const response = await BelAir.request("carlton");

  expect(response.data.user).toEqual({
    id: "carlton",
    role: "student",
    name: { first: "Carlton", last: "Banks" },
    grades: {
      literature: "A",
      math: "A",
      gym: "A",
      health: "A",
      chemistry: "A"
    }
  });
});

```

```

test("ashley banks", async () => {
  const response = await BelAir.request("ashley")

  response.data.user.grades =
    Object.keys(response.data.user.grades)
      .map(course => BelAir.computeGpa(response.data.user.grades[course]))
      .reduce((total, grade) => (total += grade), 0)
    Object.keys(response.data.user.grades).length

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

  expect(response.data.user).toMatchSnapshot();
});

```

```

import * as BelAir from "../bel-air-academy";

```

```

test("uncle phil", async () => {
  const response = await BelAir.request("phil")

  expect(response.data.user.name.first).toBe("Phil");
  expect(response.data.user.name.last).toBe("Phillips");
  expect(response.data.user.role).toBe("parent");
});

```

```

test("carlton banks", async () => {
  const response = await BelAir.request("carlton")

  expect(response.data.user).toEqual({
    id: "carlton",
    role: "student",
    name: { first: "Carlton", last: "Banks" },
    grades: {
      literature: "A",
      math: "A",
      gym: "A",
      health: "A",
      chemistry: "A"
    }
  });
});

```