



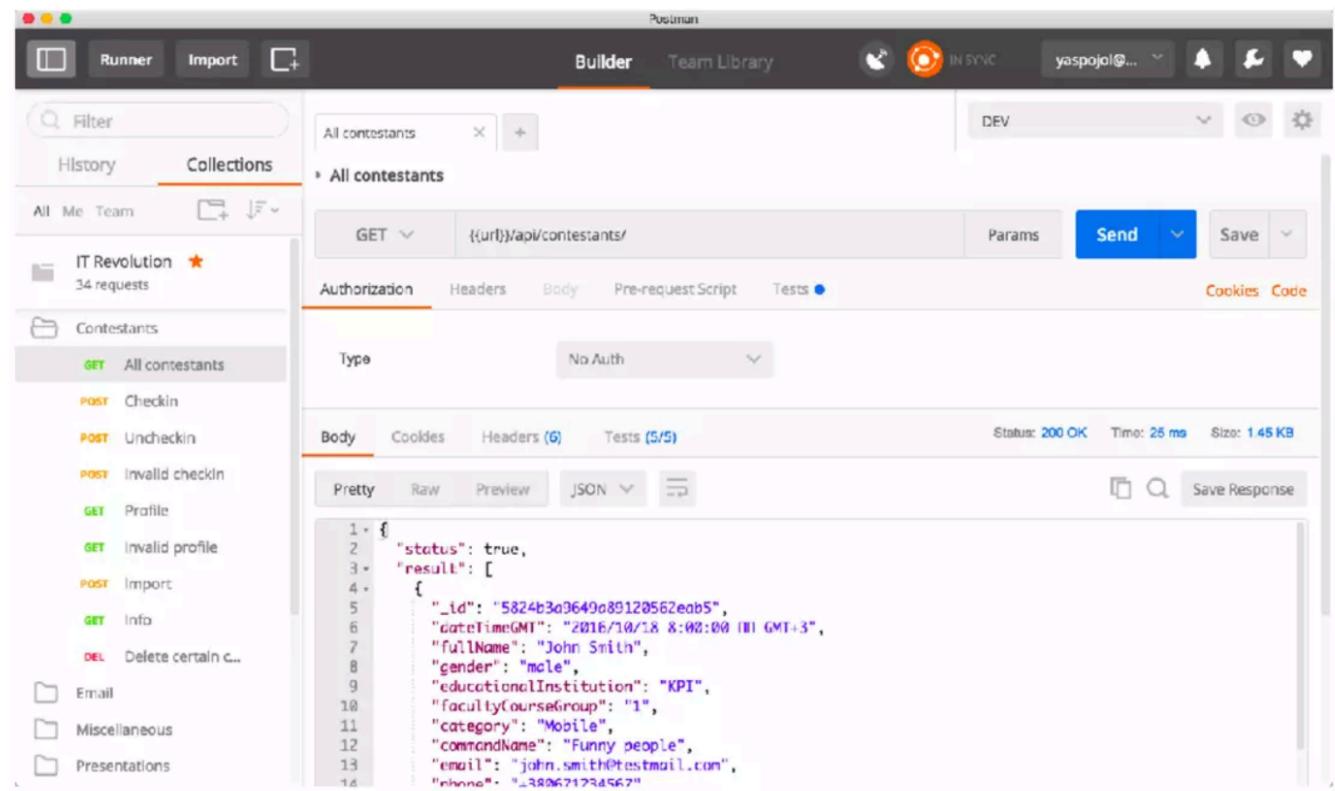
POSTMAN

An Introduction for Developers



Postman - API development tool

- Design
- Manage
- Run
- Test
- Documentation
- Share

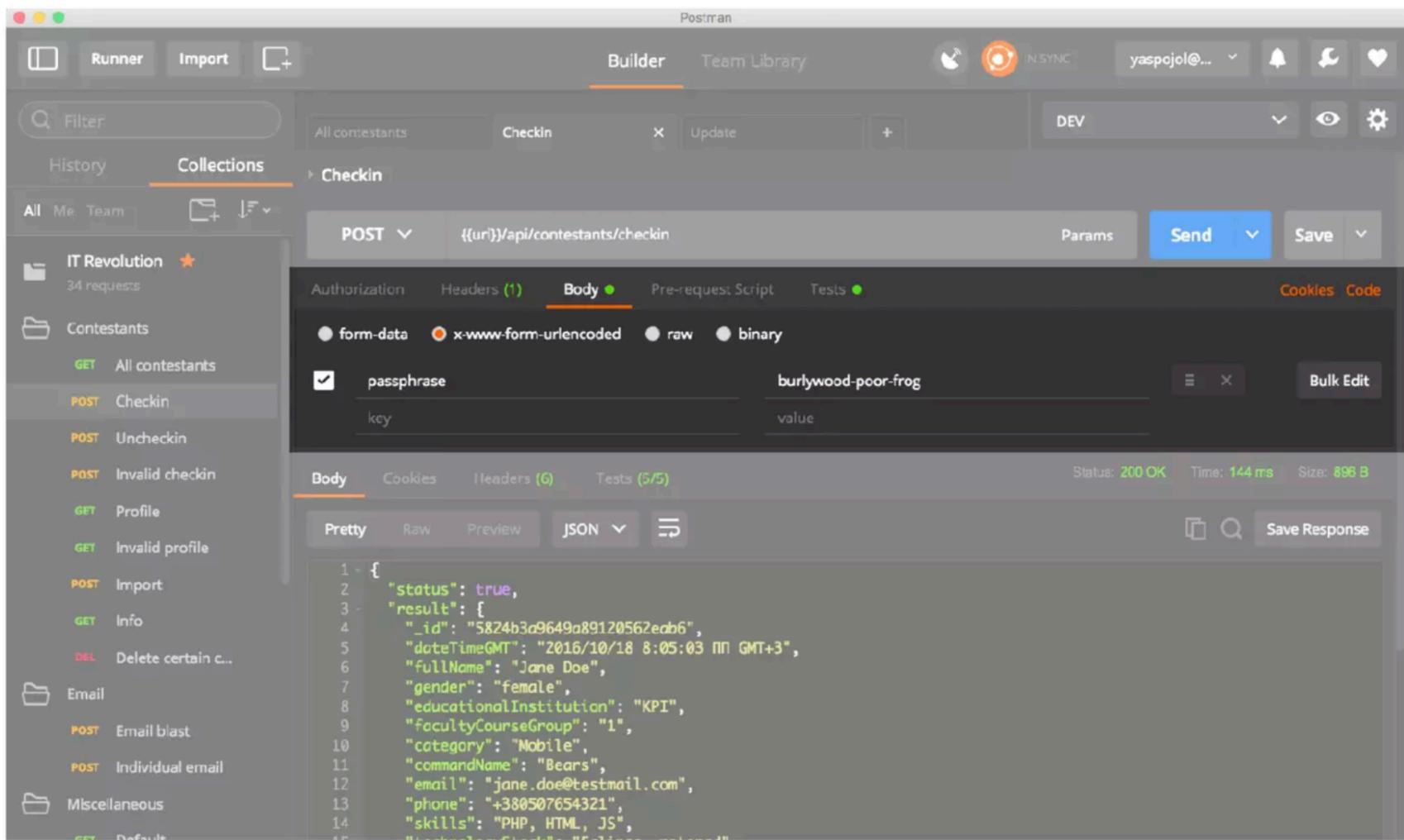


Learning Objectives

At the end of this session, you will be able to:

- Know what Postman is and what it's used for
- Send a request in Postman using CRUD operations
- Inspect a response
- Test a response
- Save example responses to a collection
- Get and set variables
- Basic authorization
- Debug and troubleshoot unexpected behavior

Request forms



Response

The screenshot shows the Postman application interface. The top navigation bar includes 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and user information 'yaspojol@...'. The left sidebar shows 'Collections' with 'IT Revolution' containing various requests like 'All contestants', 'Checkin', 'Uncheckin', etc. The main area displays a 'GET' request to '{{url}}/api/contestants/'. The 'Authorization' tab is selected, showing 'No Auth'. The 'Body' tab displays a JSON response:

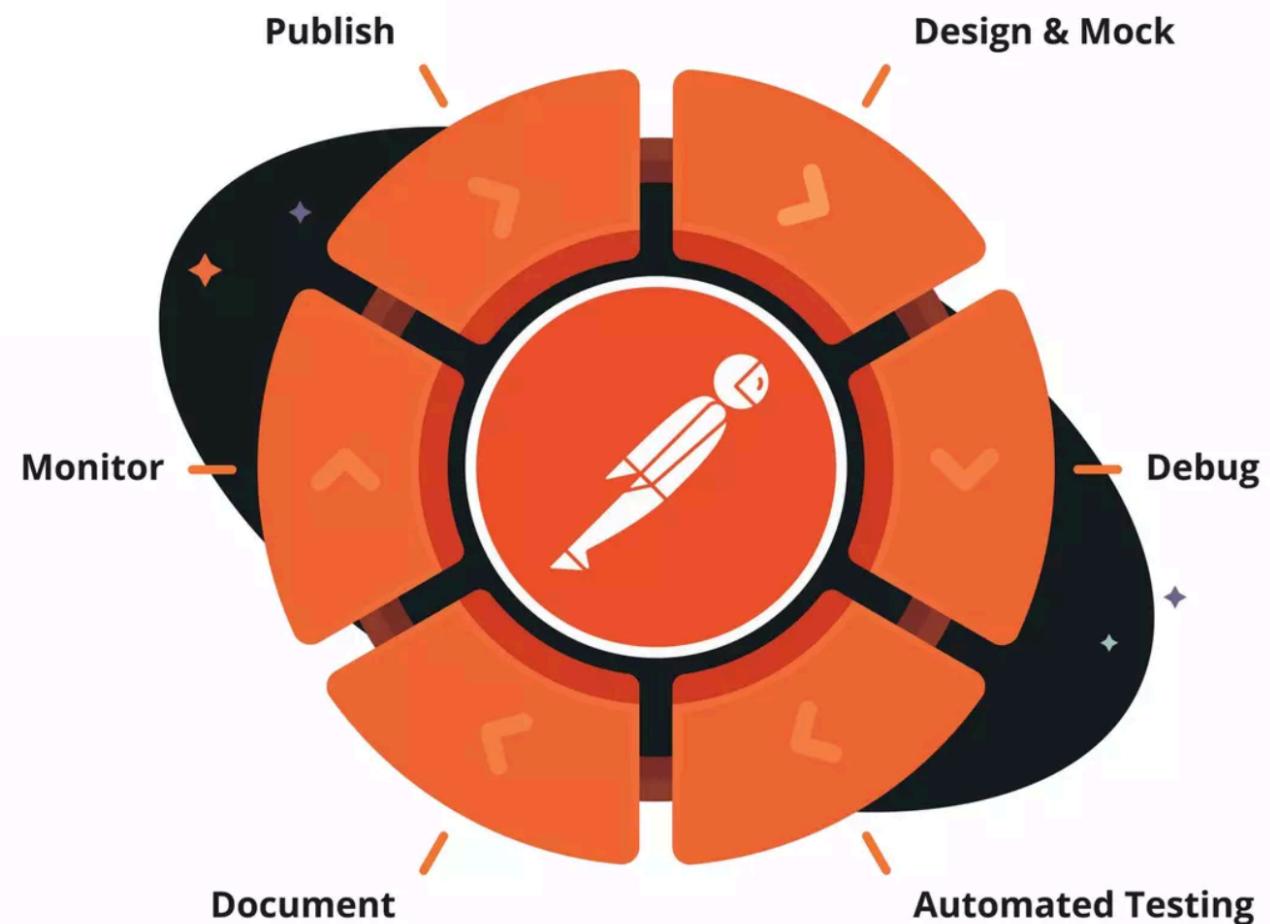
```
1 - {
2 -   "status": true,
3 -   "result": [
4 -     {
5 -       "_id": "5824b3a9649a89120562eab5",
6 -       "dateTimeGMT": "2016/10/18 8:00:00 GMT+3",
7 -       "fullName": "John Smith",
8 -       "gender": "male",
9 -       "educationalInstitution": "KPI",
10 -      "facultyCourseGroup": "1",
11 -      "category": "Mobile",
12 -      "commandName": "Funny people",
13 -      "email": "john.smith@testmail.com",
14 -      "phone": "+380671234567",
15 -      "skills": "Painting, design",
16 -      "technologyStack": "Adobe Photoshop",
17 -      "whoAreYouInTheCommand": "leader"
}
```

The status bar at the bottom indicates 'Status: 200 OK', 'Time: 51 ms', and 'Size: 1.45 KB'.

What is Postman?

Postman is an API Collaboration Platform used by more than **15 million users.**

Postman helps people work with APIs more efficiently.



Request headers

The screenshot shows the Postman application interface. The top navigation bar includes 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and various status indicators like 'IN SYNC' and 'DEV'. The left sidebar displays 'Collections' with an 'IT Revolution' folder containing several API requests: 'All contestants' (GET), 'Checkin' (POST, selected), 'Uncheckin' (POST), 'Invalid checkin' (POST), 'Profile' (GET), 'Invalid profile' (GET), 'Import' (POST), 'Info' (GET), and 'Delete certain c...' (DELETE). The main workspace shows a 'Checkin' collection with a single 'Checkin' request. The request method is 'POST' to the URL '{{url}}/api/contestants/checkin'. The 'Headers' tab is active, showing a single header 'Content-Type: application/x-www-form-urlencoded'. The 'Body' tab shows a JSON response with the following content:

```
1 {  
2   "status": true,  
3   "result": {  
4     "_id": "5824b3a9649a89120562eab6",  
5     "dateTimeGMT": "2016/10/18 8:05:03 ПП GMT+3",  
6     "fullName": "Jane Doe",  
7     "gender": "female",  
8     "educationalInstitution": "KPI",  
9     "facultyCourseGroup": "1",  
10    "category": "Mobile",  
11    "commandName": "Bears",  
12    "email": "jane.doe@testmail.com",  
13    "phone": "+380507654321",  
14    "skills": "PHP, HTML, JS",  
15    "technologyStack": "Eclipse, notepad",  
16    "whoAreYouInTheCommand": "team player",  
17    "workExperience": "student"  
}
```

The response status is 200 OK, time is 144 ms, and size is 896 B. There are tabs for 'Pretty', 'Raw', 'Preview', and 'JSON' at the bottom of the body panel.

Request

The screenshot shows the Postman application interface. The top navigation bar includes 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and other account and settings icons. The left sidebar has 'History' and 'Collections' tabs, with 'Collections' selected. Under 'Collections', there is a folder named 'IT Revolution' containing 34 requests, and a section for 'Contestants' with various methods listed: GET All contestants, POST Checkin, POST Uncheckin, POST Invalid checkin, GET Profile, GET Invalid profile, POST Import, GET Info, and a partially visible DEL Delete certain c...'. The main workspace shows a 'All contestants' collection with a single 'Checkin' item. The 'Builder' tab is active, showing a 'GET' request to {{url}}/api/contestants/. The 'Authorization' tab is selected, showing 'No Auth'. The 'Body' tab displays a JSON response:

```
1 {  
2   "status": true,  
3   "result": [  
4     {"  
5       "_id": "5824b3a9649a89120562eab5",  
6       "dateTimeGMT": "2016/10/18 8:00:00 GMT+3",  
7       "fullName": "John Smith",  
8       "gender": "male",  
9       "educationalInstitution": "KPI",  
10      "facultyCourseGroup": "1",  
11      "category": "Mobile",  
12      "commandName": "Funny people",  
13      "email": "john.smith@testmail.com",  
14      "phone": "+380671234567",  
15      "skills": "Painting, design",  
16      "technologyStack": "Adobe Photoshop",  
17      "whoAreYouInTheCommand": "leader"  
]
```

The status bar at the bottom right indicates 'Status: 200 OK', 'Time: 51 ms', and 'Size: 1.45 KB'.

Collections and folders

The screenshot shows the Postman application interface. On the left, there's a sidebar titled 'Collections' which contains several items:

- IT Revolution** (marked with a star) containing:
 - All contestants** (GET request)
 - Checkin** (POST request)
 - Uncheckin** (POST request)
 - Invalid checkin** (POST request)
 - Profile** (GET request)
 - Invalid profile** (GET request)
 - Import** (POST request)
 - Info** (GET request)
 - Delete certain c...** (DELETE request)
- Email**
- Miscellaneous**
- Presentations**
- Schedule**
- Support requests**

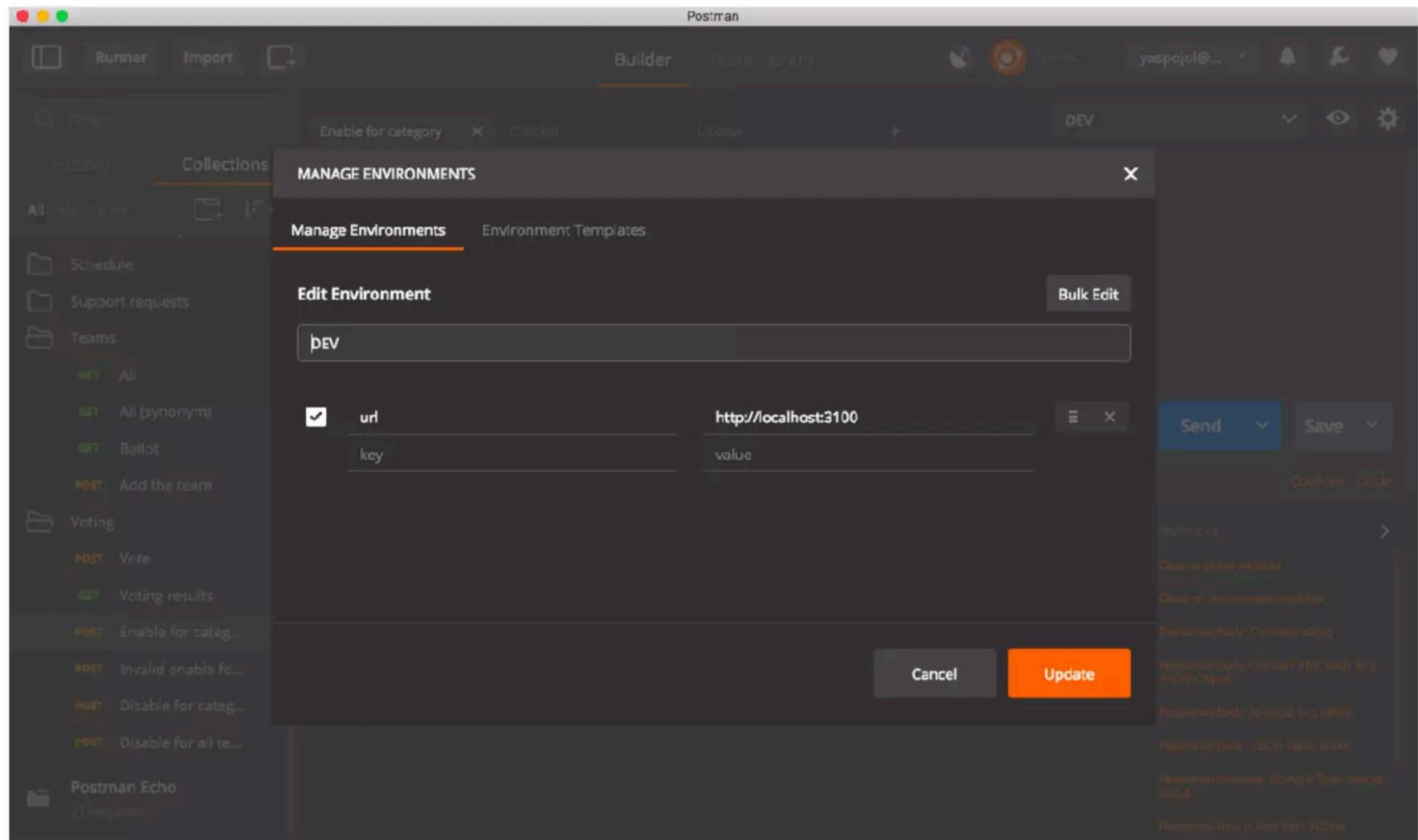
In the main area, the 'All contestants' collection is selected. A request is being built for the endpoint `{url}/api/contestants/`. The 'Authorization' tab is selected, showing 'No Auth'. The 'Body' tab displays the JSON response received from the server, which includes fields like _id, dateTImeGMT, fullName, gender, etc. The status bar at the bottom right indicates a successful 200 OK response.

```
1 {  
2   "status": true,  
3   "result": [  
4     {"  
5       "_id": "5824b3a9649a89120562eab5",  
6       "dateTImeGMT": "2016/10/18 8:00:00 MIT GMT+3",  
7       "fullName": "John Smith",  
8       "gender": "male",  
9       "educationalInstitution": "KPI",  
10      "facultyCourseGroup": "1",  
11      "category": "Mobile",  
12      "commandName": "Funny people",  
13      "email": "john.smith@testmail.com",  
14      "phone": "+380671234567",  
15      "skills": "Painting, design",  
16      "technologyStack": "Adobe Photoshop",  
17      "whoAreYouInTheCommand": "leader"  
]
```

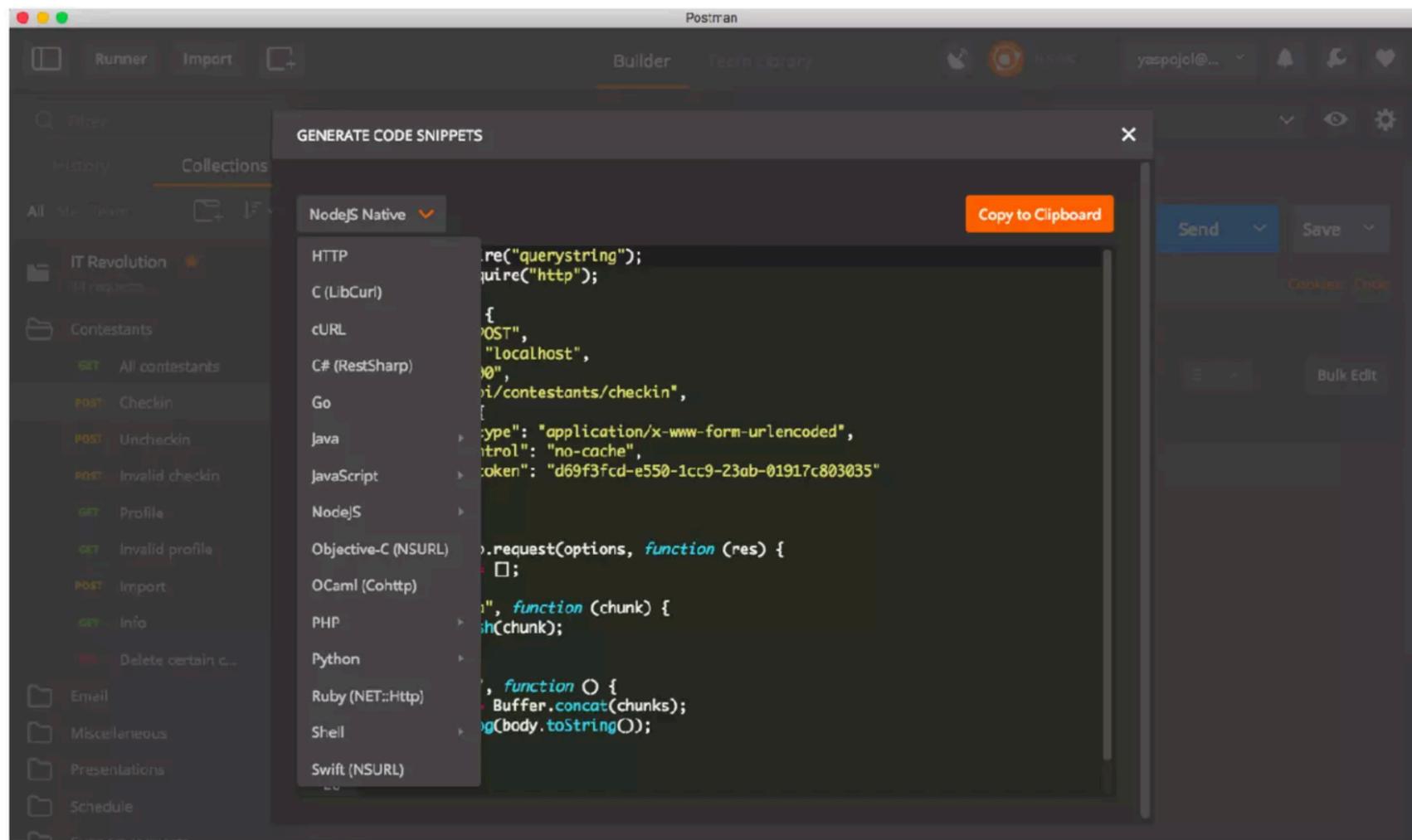
Environments

The screenshot shows the Postman application interface. The top navigation bar includes 'Runner', 'Import', 'Builder' (which is highlighted), 'Team Library', and various user and notification icons. The left sidebar features a 'Collections' tab selected, displaying a tree structure of API endpoints under categories like 'Schedule', 'Support requests', 'Teams', and 'Voting'. The main workspace shows a request builder for a 'POST' request to the endpoint {{url}}/api/voting/enable. The 'Tests' tab is active, containing a single test step labeled '1'. To the right of the request builder is a panel titled 'ENVIRONMENTS' which lists available environments: 'No Environment', 'Andrew API', 'DEV', 'Production (BE CAREFUL!!)', and 'Staging'. A 'Params' tab is also visible in the request builder area.

Environment variables



Code generation



Tests

The screenshot shows the Postman Builder interface. On the left, the sidebar displays collections like 'IT Revolution' and 'Contestants'. The main area shows a GET request to `{(url)}/api/contestants/`. The 'Tests' tab is selected, containing the following JavaScript code:

```
1 tests["Status code is 200"] = responseCode.code == 200;
2
3 var jsonData = JSON.parse(responseBody);
4 tests["Status is true"] = jsonData.status == true;
5 tests["Result is an array"] = Array.isArray(jsonData.result) == true;
6 tests["Result is a non-empty array"] = jsonData.result.length > 0;
7
8 tests["Response time is less than 200ms"] = responseTime < 200;
```

The status bar at the bottom indicates **Status: 200 OK**, **Time: 51 ms**, and **Size: 1.45 KB**.

Tests

The screenshot shows the Postman Builder interface. A GET request is defined with the URL `{(url)}/api/contestants/`. The 'Tests' tab is selected, displaying the following JavaScript code:

```
1 tests["Status code is 200"] = responseCode.code == 200;
2
3 var jsonData = JSON.parse(responseBody);
4 tests["Status is true"] = jsonData.status == true;
5 tests["Result is an array"] = Array.isArray(jsonData.result) == true;
6 tests["Result is a non-empty array"] = jsonData.result.length > 0;
7
8 tests["Response time is less than 200ms"] = responseTime < 200;
```

The right side of the interface shows a sidebar with various snippets and a status bar at the bottom indicating a successful response: Status: 200 OK, Time: 51 ms, Size: 1.45 KB.

Request

The screenshot shows the Postman application interface. The top navigation bar includes 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and user information 'yaspojol@...'. The left sidebar shows 'Collections' with 'IT Revolution' containing various requests like 'All contestants', 'Checkin', 'Uncheckin', etc. The main workspace displays a 'GET' request to '{{url}}/api/contestants/'. The 'Authorization' tab is selected, showing 'No Auth'. The 'Body' tab displays a JSON response:

```
1 {
2   "status": true,
3   "result": [
4     {
5       "_id": "5824b3a9649a89120562eab5",
6       "dateTimeGMT": "2016/10/18 8:00:00 GMT+3",
7       "fullName": "John Smith",
8       "gender": "male",
9       "educationalInstitution": "KPI",
10      "facultyCourseGroup": "1",
11      "category": "Mobile",
12      "commandName": "Funny people",
13      "email": "john.smith@testmail.com",
14      "phone": "+380671234567",
15      "skills": "Painting, design",
16      "technologyStack": "Adobe Photoshop",
17      "whoAreYouInTheCommand": "leader"
18    }
19  ]
20}
```

Collections and folders

The screenshot shows the Postman application interface. On the left, there's a sidebar with tabs for History, Collections, and a currently selected 'All contestants' folder. This folder contains several requests: GET All contestants, POST Checkin, POST Uncheckin, POST Invalid checkin, GET Profile, GET Invalid profile, POST Import, GET Info, and a partially visible DEL Delete certain c... request. On the right, the main workspace displays a 'Builder' tab for a 'All contestants' collection. A GET request is configured with the URL {{url}}/api/contestants/. The 'Authorization' tab is selected, showing 'No Auth'. The 'Body' tab displays a JSON response with status 200 OK, time 51 ms, and size 1.45 KB. The response body is a JSON object:

```
1 - {
2 -   "status": true,
3 -   "result": [
4 -     {
5 -       "_id": "5824b3a9649a89120562eab5",
6 -       "dateTimeGMT": "2016/10/18 8:00:00 GMT+3",
7 -       "fullName": "John Smith",
8 -       "gender": "male",
9 -       "educationalInstitution": "KPI",
10 -      "facultyCourseGroup": "1",
11 -      "category": "Mobile",
12 -      "commandName": "Funny people",
13 -      "email": "john.smith@testmail.com",
14 -      "phone": "+380671234567",
15 -      "skills": "Painting, design",
16 -      "technologyStack": "Adobe Photoshop",
17 -      "whoAreYouInTheCommand": "leader"
}
```

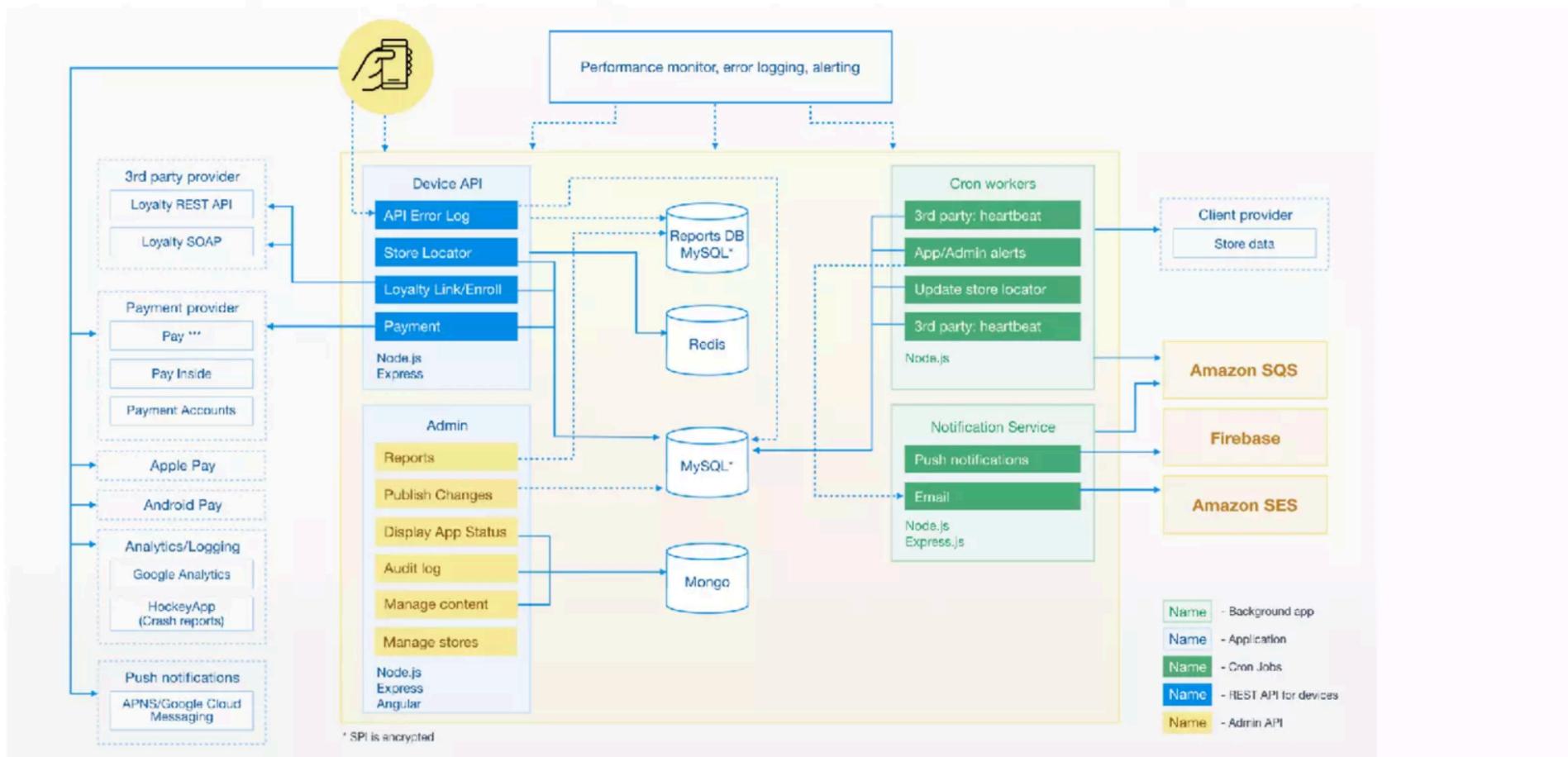
Test results

The screenshot shows the Postman Builder interface with a collection named "Contestants". A GET request is being built to the URL `{(uri)}/api/contestants/`. The "Tests" tab is selected, displaying five passed assertions:

- PASS Status code is 200
- PASS Status is true
- PASS Result is an array
- PASS Result is a non-empty array
- PASS Response time is less than 200ms

The status bar at the bottom indicates a successful response with **Status: 200 OK**, **Time: 51 ms**, and **Size: 1.45 KB**.

Complex systems



Response

The screenshot shows the Postman application interface in dark mode. The top navigation bar includes 'Runner', 'Import', 'Builder' (which is selected), 'Team Library', and user information 'yaspojol@...'. The left sidebar shows 'Collections' with 'IT Revolution' containing various requests like 'All contestants', 'Checkin', etc. The main workspace displays a 'GET /api/contestants/' request. The 'Authorization' tab is selected, showing 'No Auth'. The 'Body' tab is selected in the preview area, displaying a JSON response:

```
1 - {
2 -   "status": true,
3 -   "result": [
4 -     {
5 -       "_id": "5824b3a9649a89120562eab5",
6 -       "dateTImeGMT": "2016/10/18 8:00:00 GMT+3",
7 -       "fullName": "John Smith",
8 -       "gender": "male",
9 -       "educationalInstitution": "KPI",
10 -      "facultyCourseGroup": "1",
11 -      "category": "Mobile",
12 -      "commandName": "Funny people",
13 -      "email": "john.smith@testmail.com",
14 -      "phone": "+380671234567",
15 -      "skills": "Painting, design",
16 -      "technologyStack": "Adobe Photoshop",
17 -      "whoAreYouInTheCommand": "leader"
}
```

The status bar at the bottom right indicates 'Status: 200 OK', 'Time: 51 ms', and 'Size: 1.45 KB'.

Test results

The screenshot shows the Postman Builder interface with a collection named "All contestants". A GET request is defined with the URL `{(url)}/api/contestants/`. The "Tests" tab is selected, displaying five passed assertions:

- PASS Status code is 200
- PASS Status is true
- PASS Result is an array
- PASS Result is a non-empty array
- PASS Response time is less than 200ms

The status bar at the bottom indicates a successful response with **Status: 200 OK**, **Time: 51 ms**, and **Size: 1.45 KB**.