## Test Plan Items

Test Plan Item (TPI) is an issue created with label `testplan-item` for testing the feature during the endgame. It shall define the platforms it needs to be tested against and also how complex (1-5) it is to test.

**Note:** A Test Plan Item shall not take too long to test and scoped to a specific feature. Please note that **Complexity 5 is not unlimited and there shall be a time scope of one or two hours max of testing**. If your Test Plan Item has multiple features to test and also goes beyond complexity 5 and time scope, please break it down into multiple complexity (1-5) test plan items.

There is a tool that parses all these TPIs and generates user assignments. Endgame master uses this tool to assign these TPIs to others for testing. A Test Plan Item should contain **Header** and **Body** sections which are separated by `---` (line). Header section shall contain the meta information about the TPI that is used by the tool for generating TPI assignments. Body section shall include testing details for the user to test.

### Header Section

Header section should be separated using `---` (line)

```
<!-- Header Section. -->

---

<!-- Body Section. -->
```

It shall contain following:

- References to issues being tested

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->
```

- Platform assignments mentioning on which platforms this TPI shall be tested. You can pre-assign them to a user you would like to test by adding the user id next to platform. Refer to the examples for how to include this information.
  - macOS
  - windows
  - linux
  - wsl
  - ssh
  - dev container
  - web
  - anyOS
  - iPad
- Complexity of the test plan item which should be between 1 till 5.

```
Complexity: 4
```

- If there are more than one authors for it, all the authors should be specified so that they will not be assigned for testing

```
Authors: @user1, @user2
```

- If a TPI requires a certain skillset (e.g. the person should be a developer), then you can specify what roles people should hold to work on the TPI. See the examples no how to format such information.
  - Content Developer
  - Designer
  - Developer
  - Engineering Manager
  - Program Manager

## Examples

### Example 1 (Platforms)

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] macOS
- [ ] linux
- [ ] windows

Complexity: 2

---

<!-- Please write your test here. -->
```

### Example 2 (Any platforms)

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] anyOS
- [ ] anyOS

Complexity: 4

---

<!-- Please write your test here. -->
```

**Example 3 (One specific and other any platform)**

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] windows
- [ ] anyOS

Complexity: 4

---

<!-- Please write your test here. -->
```

**Example 4 (Same platform multiple times)**

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] windows
- [ ] windows

Complexity: 4

---

<!-- Please write your test here. -->
```

**Example 5 (Pre-assigned to users)**

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] macOS: @user1
- [ ] linux

Complexity: 4

Authors: @user1, @user2

---

<!-- Please write your test here. -->
```

**Example 6 (Remote)**

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] wsl
- [ ] ssh
- [ ] dev container

Complexity: 2

---

<!-- Please write your test here. -->
```

**Example 7 (Web)**

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] web
- [ ] web

Complexity: 2

---

<!-- Please write your test here. -->
```

**Example 8 (Multiple Authors)**

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] windows
- [ ] linux

Complexity: 4

Authors: @user1, @user2

---

<!-- Please write your test here. -->
```

**Example 9 (Additional note to assigned users)**

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] linux (test cross platform)
- [ ] windows: @user1 (test 32 bit)

Complexity: 4

---

<!-- Please write your test here. -->
```

**Example 10 (Assigning Roles)**

```
Refs: <!-- Refer to the issue that this test plan item is testing. -->

- [ ] linux
- [ ] windows

Complexity: 4
Roles: Developer, Engineering Manager

---

<!-- Please write your test here. -->
```

### How to: Testing Proposed and/or Latest API

0. Make sure to have `npx`, `yeoman`, and `generator-code` installed
1. Call `yo code` and create a new TypeScript extension (accept all defaults). You now have the default “Hello World” extension
2. Open the extension’s package.json-file and remove the `@types/vscode` dev-dependency
3. Delete the `node_modules/@types/vscode`-folder
4. **Latest API**: Run `cd src && npx vscode-dts main` to fetch latest stable API
5. **Proposed API**: Run `cd src && npx vscode-dts dev` to fetch `vscode.proposed.d.ts`. Also add `enableProposedApi: true` to the `package.json`-file
6. Press `F5` to run your extension