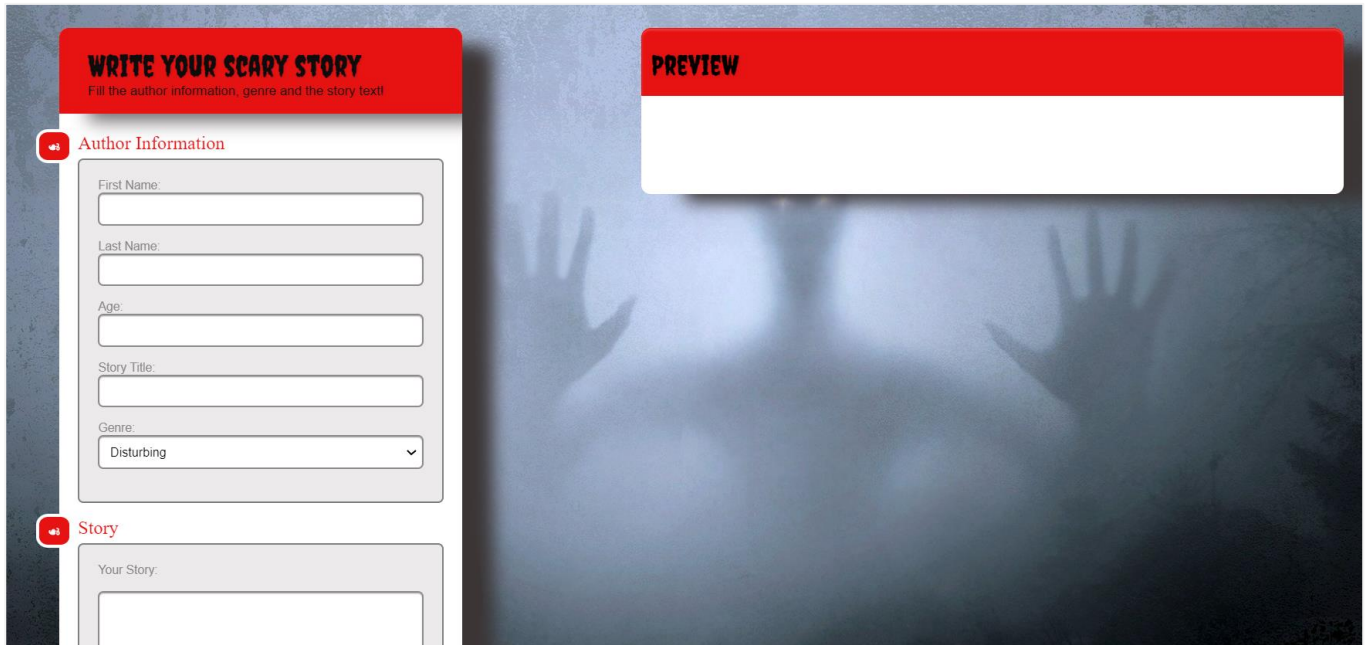# Exercise: Development Workflow

Exercises for the "Software Engineering and DevOps" module @ SoftUni

1. Scary Story

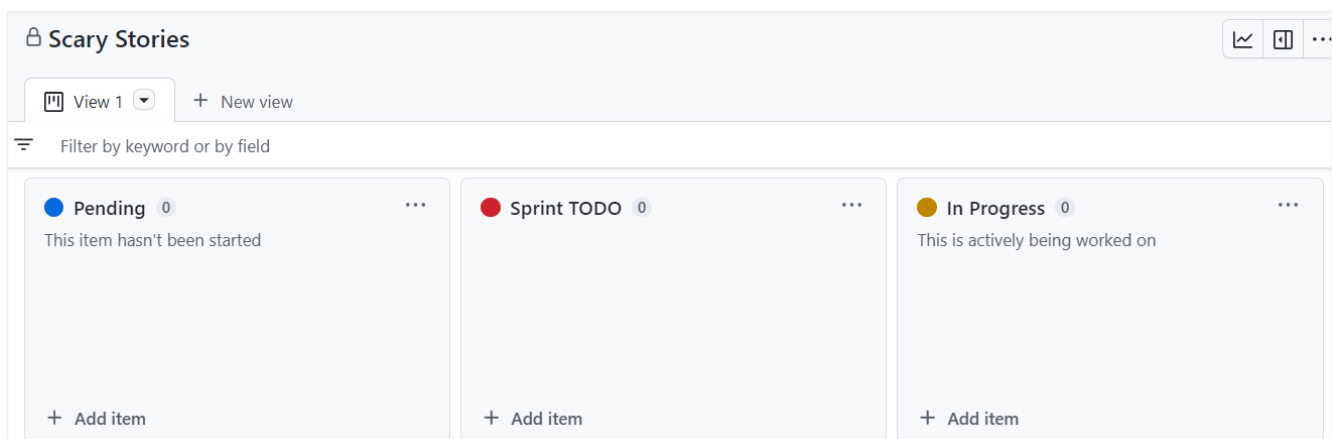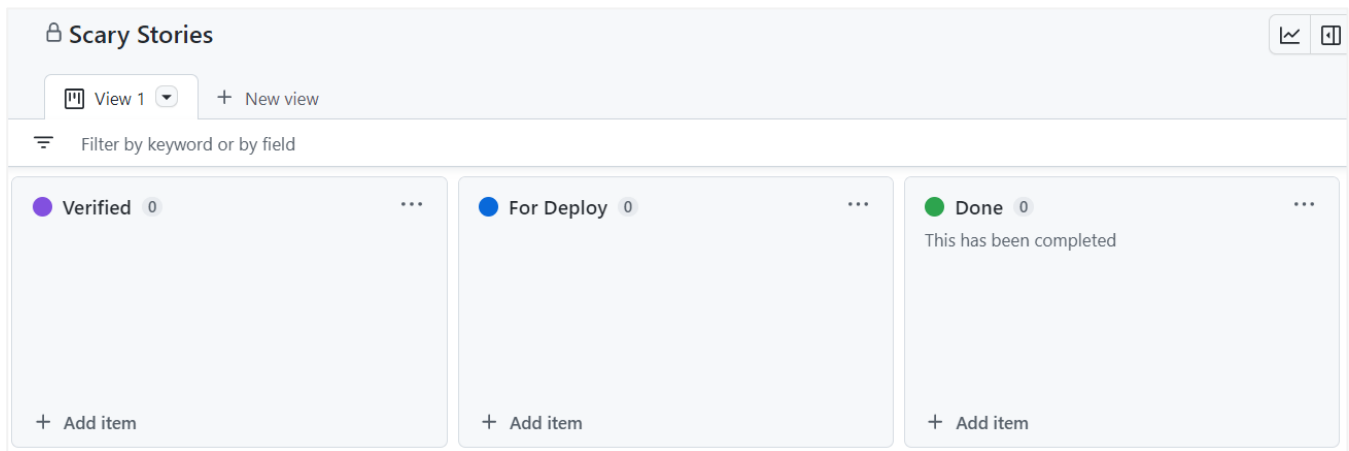Your task is to follow the entire process of **development workflow**.

You should have a **GitHub Projects board** with following **categories**:

- **Pending**
- **Sprint TODO**
- **In Progress**
- **For Testing**
- **Verified**
- **For Deploy**
- **Done**

GitHub Projects visualizes the **issues** (**tasks**) **workflow** as a **Kanban board.**

Follow us:

You should **create three issues:**

- **Change background**
- **Add genre field**
- **Add field icons**

NOTE: You can work in in teams of 3 students or work alone with several roles to simulate multi-user interaction, where each role follows the provided instructions for the given team member.

Senior

- Make an **empty GitHub repo** and **commit files** from **resources**.
- Create a **GitHub Projects board** and **add tasks**.
- **Deploy site** from the **staging branch** to the **staging environment** on `Render.com`.
- Conduct a **code review** when a **pull request** is created to the **staging branch**
- At the end, merge the **staging** to the **main branch**
- **Deploy site** from the **main branch** to the **production environment** on **Render.com**

QA

- **Deploy site** from the **qa branch** to the **QA environment** on `Render.com` (first time, then it will be automated)
- Looks **QA environment** to see if **changes** are **done correctly**
- If something does not pass, return the issue to "In Progress"

Junior

- Create a **branch** for each **issue**.
- Implement the **issue TODOs**.
- When ready, merge each of your feature branches to the qa branch
- Then, if issue is "Verified", merge its feature branch to the staging branch with a pull request

2. Senior

Step 1: Creating an Empty GitHub Repository

Senior makes an **empty GitHub repo** and **commits files** from **resources**. After that they add the **Junior** and the **QA** as **collaborators**.

Step 2: Creating Branches

Senior makes **2** more **branches** (**qa** and **staging**) that are **clones of the main branch**.

Step 3: Creating a Branch Protection Rule

Create a **branch protection rule** for committing to **main branch**, following the steps below:

1. On **GitHub.com**, navigate to the **main page** of the **repository**.
2. **Under** your **repository name**, click **[Settings]**. If you **cannot see** the **[Settings]** tab, select the **[…]** **dropdown menu**, then click **[Settings]**.
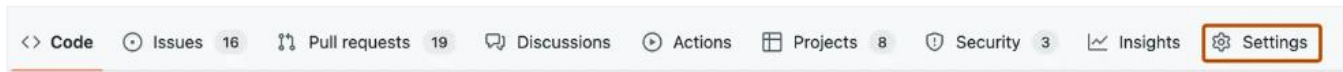
3. In the "**Code and automation**" section of the sidebar, click **[Branches]**.
4. Next to "**Branch protection rules**", click **[Add rule]**.
5. **Under** "**Branch name pattern**", type the **branch name** or **pattern** you **want** to **protect**.
6. Optionally, enable **required pull requests**.
7. For **more information** see here: https://docs.github.com/en/repositories/configuring-branches-and-merges-in-your-repository/managing-protected-branches/managing-a-branch-protection-rule

Step 4: Creating GitHub Projects Board

Senior makes the **GitHub Projects board "Scary Stories"** and adds **3 issues ("Change Background",** **"Add genre field", "Add field icons")** in the **"Pending" section**. After that they move them to the **"Sprint TODO"** **section**.
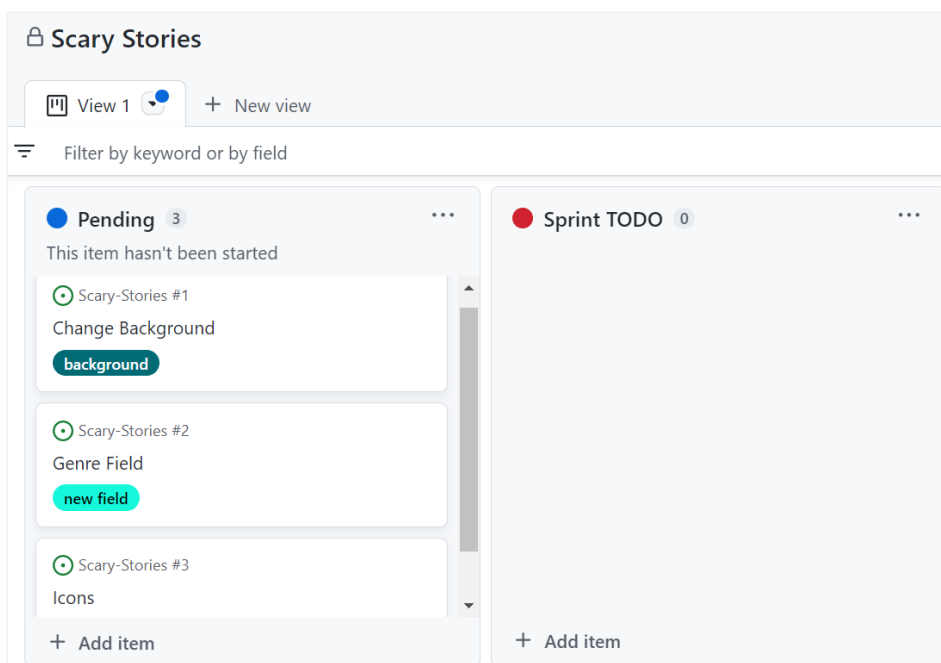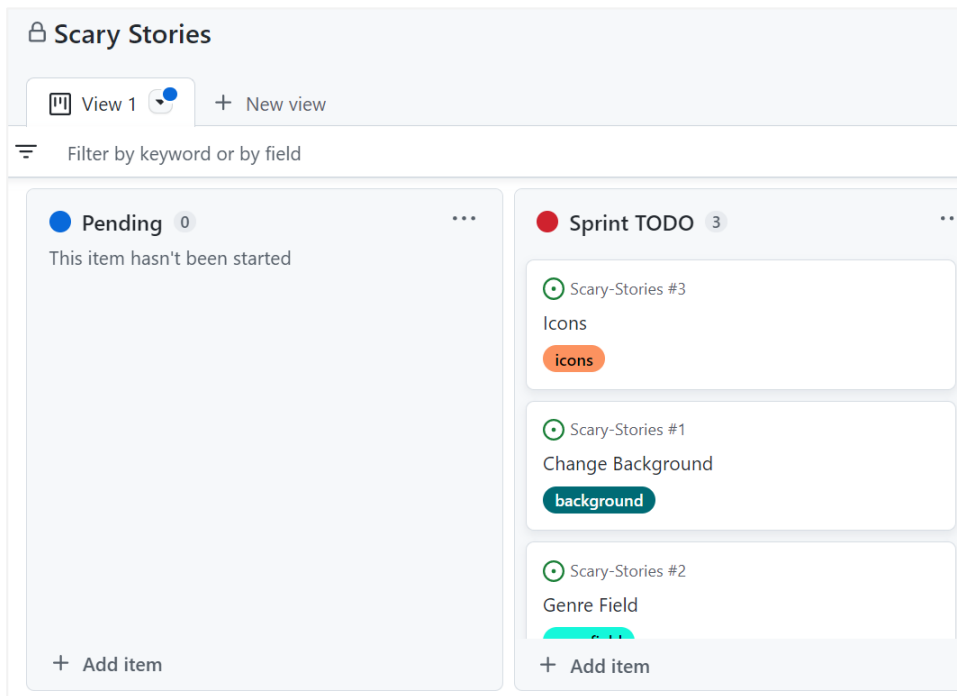
Step 5: Deploying site

Senior deploys site from the **staging branch** to the **staging environment** on **Render.com**.
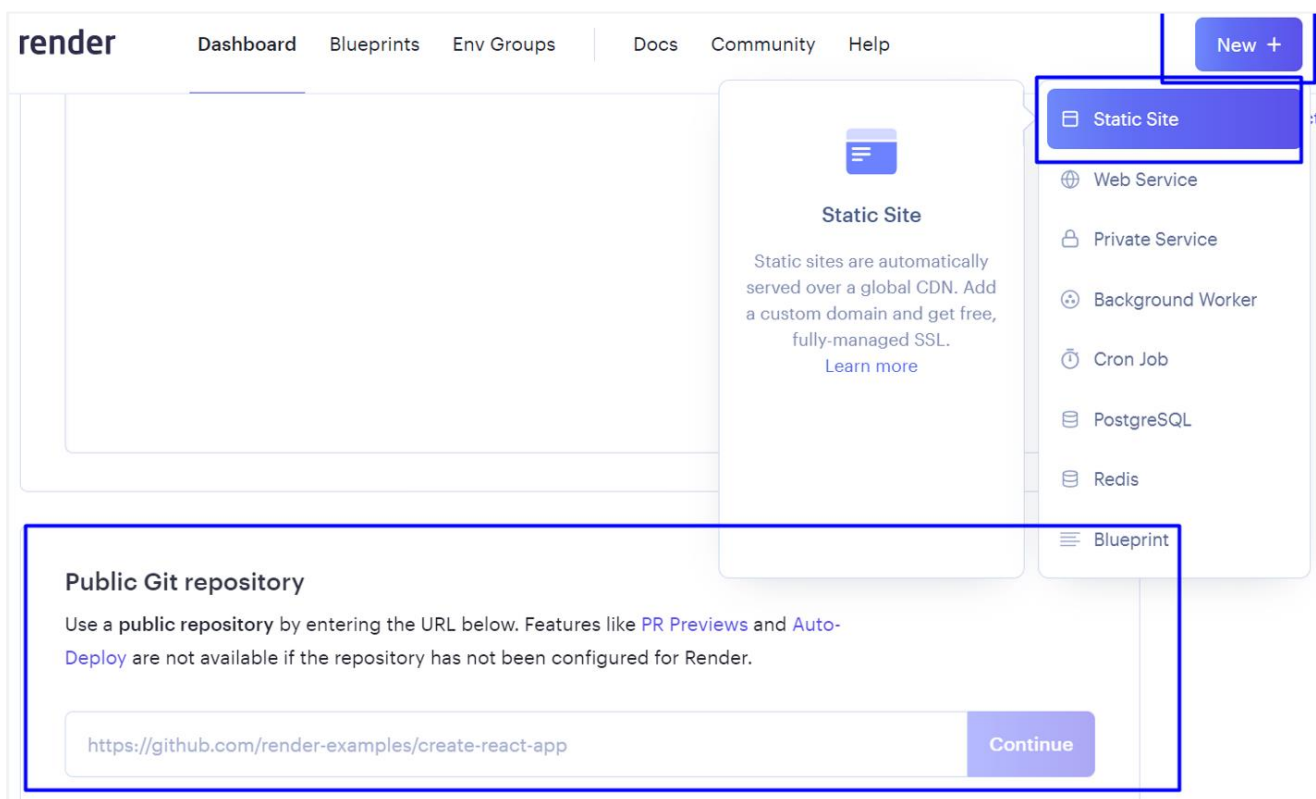
Senior conducts a **code review** when a **pull request** is created to the **staging branch**.

At the end, **Senior merges** the `staging` to the `main branch`.
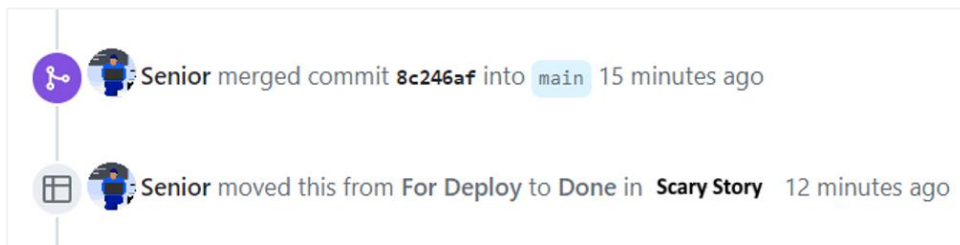
Senior deploy site from the **main branch** to the **production environment** on **Render.com**.

3. QA

Step 1: Deploying QA Branch

QA **deploys site** from the **qa branch** to the **QA environment** on **Render**.com (**first time**, then it will be **automated**)
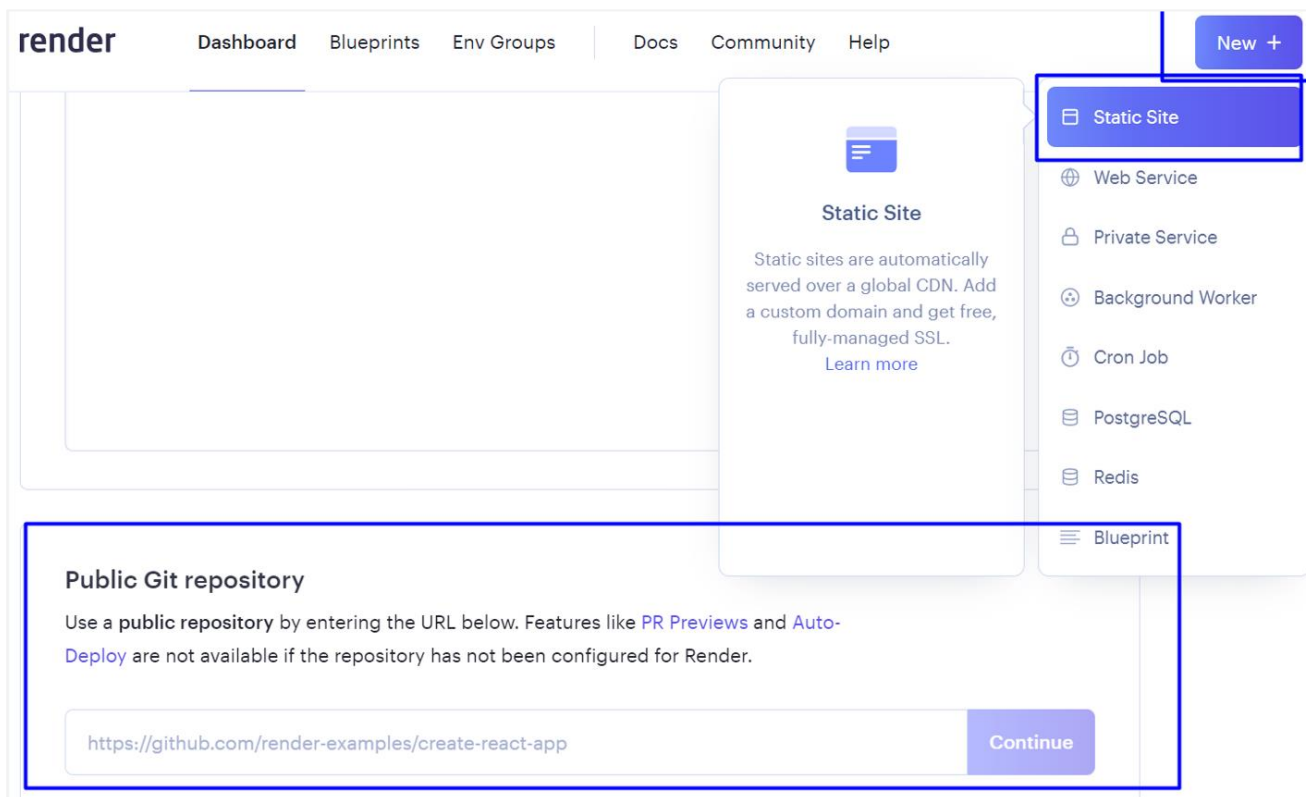
Step 2: QA Environment Check

QA looks **QA environment** to see if **changes** are **done correctly.**

Follow us:

If something **does not pass**, return the **issue** to "**In Progress**"

4. Junior

Step 1: Taking the Tasks

Junior takes the **task "Change Background"** from the **"Sprint TODO" section** and transfers it to **"In Progress"**.
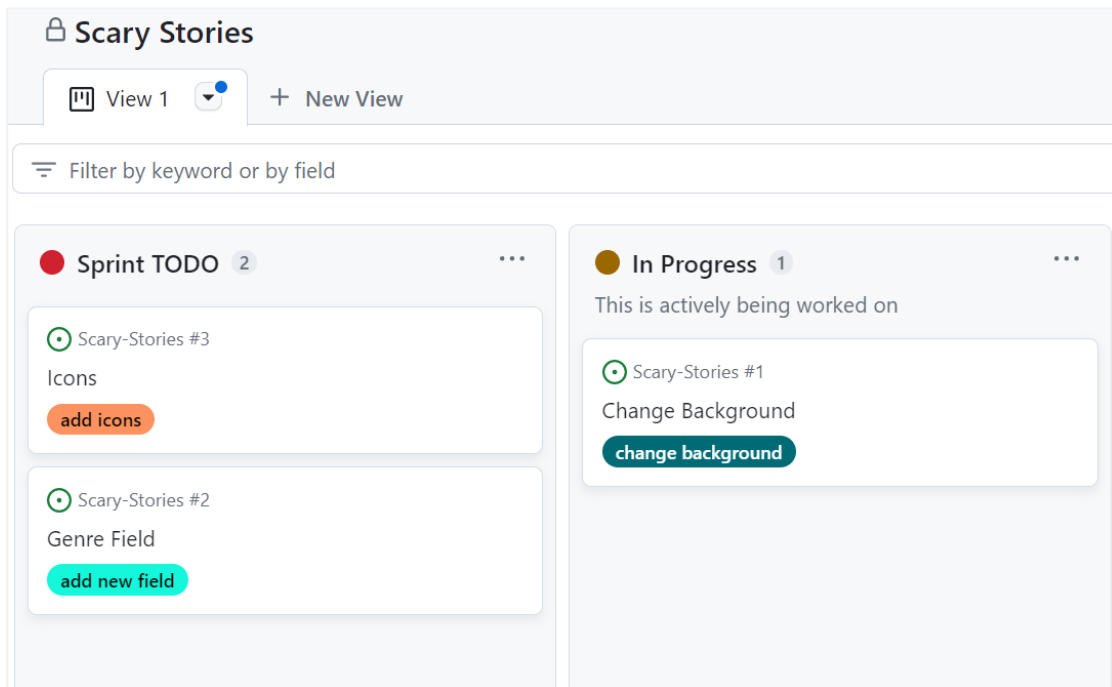
Step 2: Creating a Branch

Then they should create a new branch – **change**-**background**.

```
C:\Users\Desktop\demo\Scary-Stories>git branch change-background
```

After that they implement the **functionality** for **current issue**.

- **Change Background Issue**

Junior commits into **background branch** and after that move the **"Change Background" issue** from "**In Progress**" to "**For Testing**" **column**.

```
C:\Users\Desktop\demo\Scary-Stories>git add style/css/styles.css

C:\Users\Desktop\demo\Scary-Stories>git commit -m "Change background"
[main dd172c5] Change background
 1 file changed, 1 insertion(+), 1 deletion(-)
```

🔲 View 1 ⌄• | + New View

≡ Filter by keyword or by field

● In Progress 0 ···	● For Testing 1 ···
This is actively being worked on	⊙ Scary-Stories #1
	Change Background
	change background

Step 3: Open a Pull Request

Then, if the **issue** is "**Verified**", merge its **feature branch** to the **staging branch** with a **pull request.**

● For Testing 0 ···	● Verified 1 ···
	⊙ Scary-Stories #1
	Change Background
+ Add item	+ Add item

🔲 🖼️ **Junior** moved this from **In Progress** to **For Testing** in **Scary Story** 43 minutes ago

🏷️ 🖼️ **Qa** added the QA Verified label 35 minutes ago

🔲 🖼️ **Qa** moved this from **For Testing** to **Verified** in **Scary Story** 35 minutes ago

⑂ 🖼️ **Junior** merged commit **6a189ag** into staging 30 minutes ago

🔲 🖼️ **Junior** moved this from **Verified** to **For Deploy** in **Scary Story** 30 minutes ago

Follow us:

Junior does the **same** for the **rest** of the **tasks** (**"Add genre field", "Add field icons"**).

- The **necessary changes** for **"Add field icons"** are:

```html
17    <div id="main">
18      <div class="form-wrapper">
19        <h1>
20          Write your scary story
21          <span>Fill the author information, genre and the story text!</span>
22        </h1>
23        <form>
24          <div class="section"><span>&#9753;</span>
25            Author Information</div>
26          <div class="inner-wrap">
27            <label>First Name: <input type="text" id="first-name" name="field1" /></label>
28            <label>Last Name: <input type="text" id="last-name" name="field2" /></label>
29            <label>Age: <input type="number" id="age" name="field3" /></label>
30            </label>
31            <label>Story Title:
32              <input type="text" id="story-title" name="field5" />
33            </label>
34          </div>
35
36          <div class="section"><span>&#9753;</span>Story</div>
37          <div class="inner-wrap">
38            <label for="story">Your Story:</label>
39            <textarea id="story" name="story" rows="6" cols="50"></textarea>
40          </div>
```

- The **necessary changes** for **"Add genre field"** are:

```html
25          Author Information</div>
26        <div class="inner-wrap">
27          <label>First Name: <input type="text" id="first-name" name="field1" /></label>
28          <label>Last Name: <input type="text" id="last-name" name="field2" /></label>
29          <label>Age: <input type="number" id="age" name="field3" /></label>
30          </label>
31          <label>Story Title:
32            <input type="text" id="story-title" name="field5" />
33          </label>
34          <label>Genre:
35            <select name="field4" id="genre">
36              <option value="Disturbing">Disturbing</option>
37              <option value="Psychological">Psychological</option>
38              <option value="Killer">Killer</option>
39              <option value="Monsters">Monsters</option>
40              <option value="Paranormal">Paranormal</option>
41            </select>
42        </div>
```

Follow us: