# Exercise: Development Workflow

Exercises for the ["Software Engineering and DevOps"](https://softuni.bg/modules/133/devops-for-developers-may-2024/1467) module @ SoftUni

## Scary Story

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

Graphical user interface

Description automatically generated

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.**

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

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

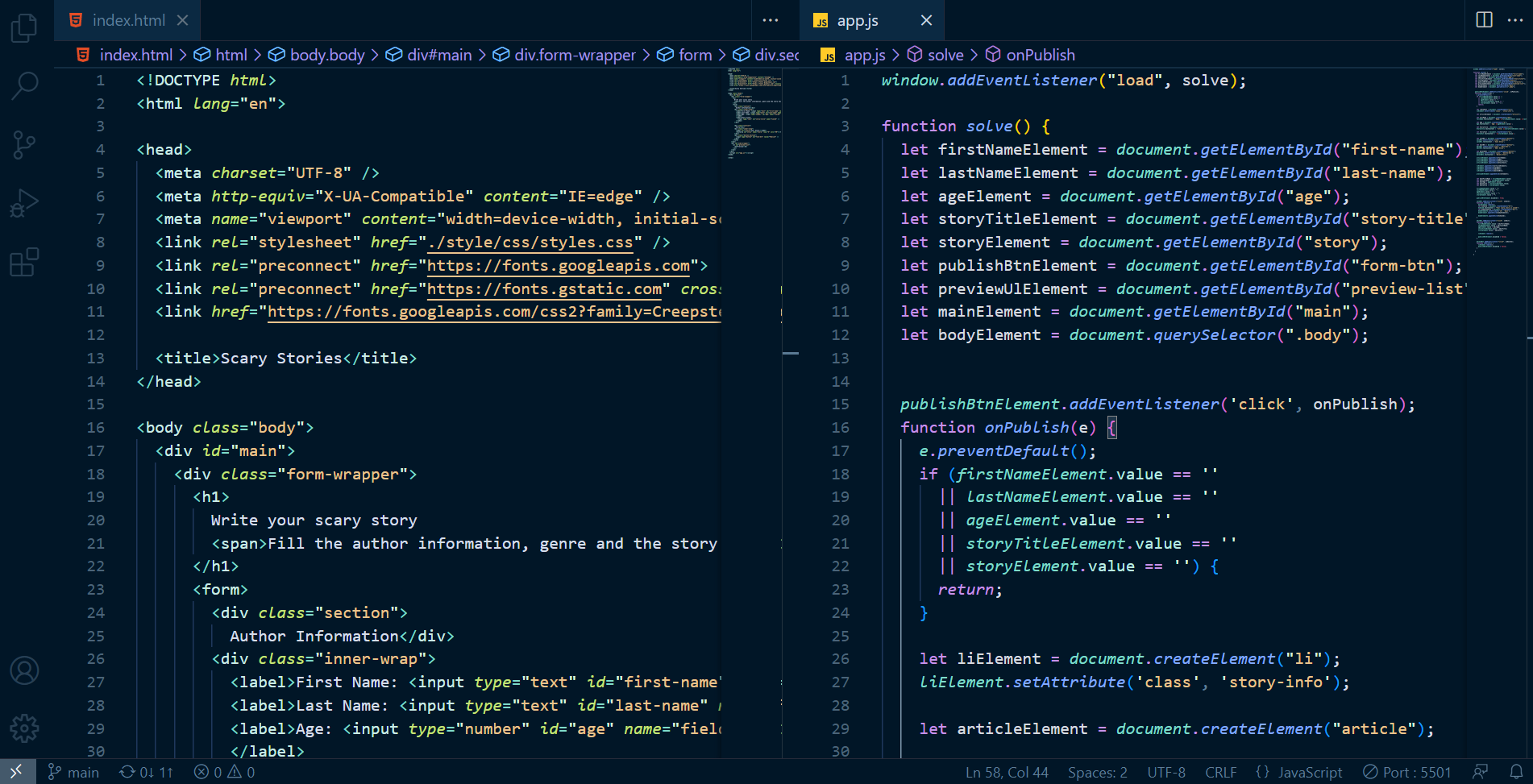
## Senior

### 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**.

Graphical user interface, text, application, email

Description automatically generated



### Step 2: Creating Branches

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

~~Graphical user interface, application

Description automatically generated~~

### Step 3: Creating a Branch Protection Rule

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

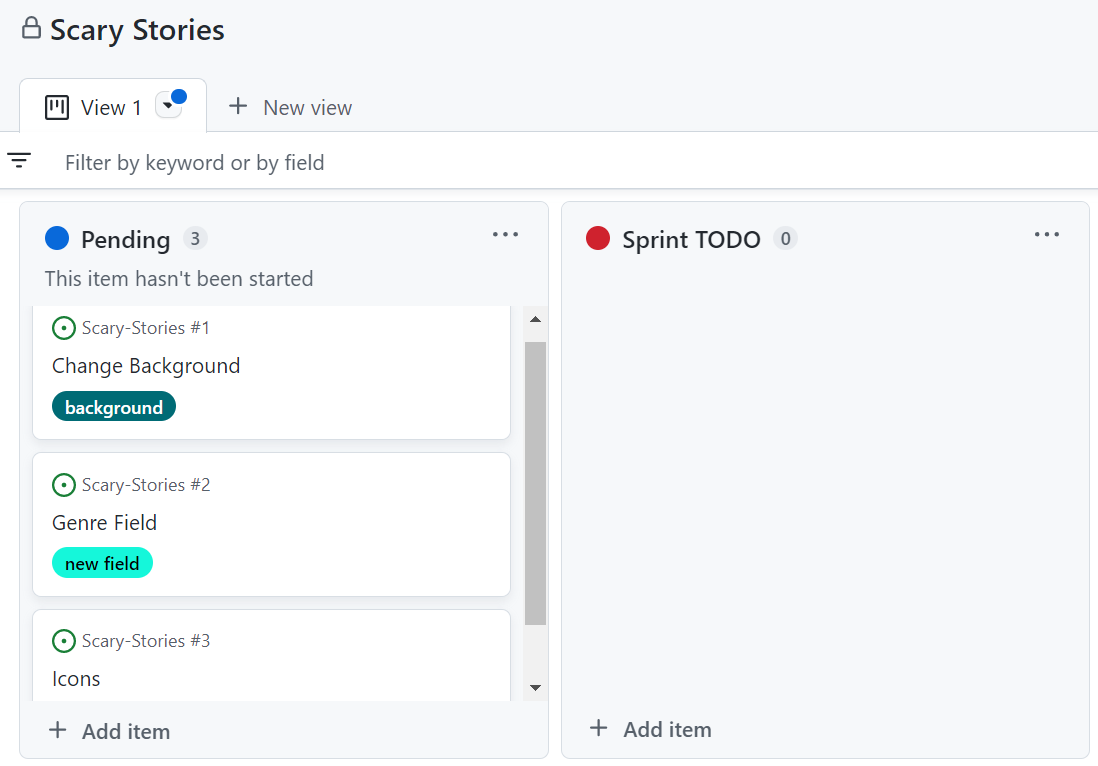
1. On **GitHub.com**, navigate to the **main** **page** of the **repository**.
2. **Under** your **repository name**, click [Settings]. If you **cannot see** the [Settings] tab, select the […] **dropdown menu**, then click [Settings].



1. In the "**Code and automation**" section of the sidebar, click [Branches].
2. Next to "**Branch protection rules**", click [Addrule].
3. **Under** "**Branch name pattern**", type the **branch name** or **pattern** you **want** to **protect**.
4. Optionally, enable **required pull requests**.
5. 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")** inthe **"Pending" section**. After that they move them tothe **"Sprint TODO" section.**



Graphical user interface, text, application

Description automatically generated

### Step 5: Deploying site

**Senior deploys** **site** from the **staging** **branch** to the **staging environment** on Render.com.

Graphical user interface, application

Description automatically generated

**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**.

Graphical user interface, text, application

Description automatically generated

**Senior** **deploy** site from the **main** **branch** to the **production** **environment** on **Render.com**.

## QA

### Step 1: Deploying QA Branch

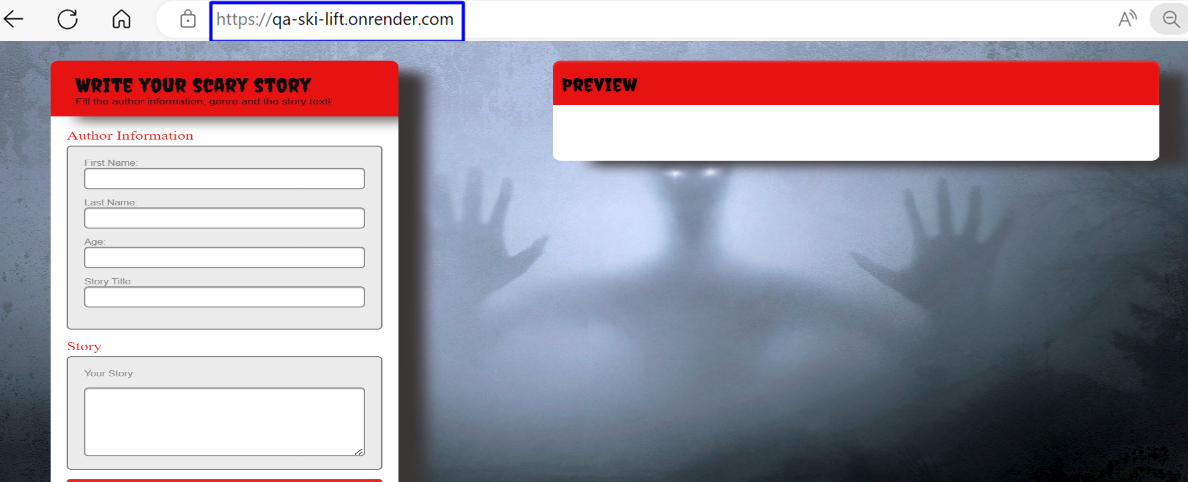
QA **deploys** **site** from the **qa** **branch** to the **QA** **environment** on **Render**.**com** (**first** **time**, then it will be **automated**)

Graphical user interface, application

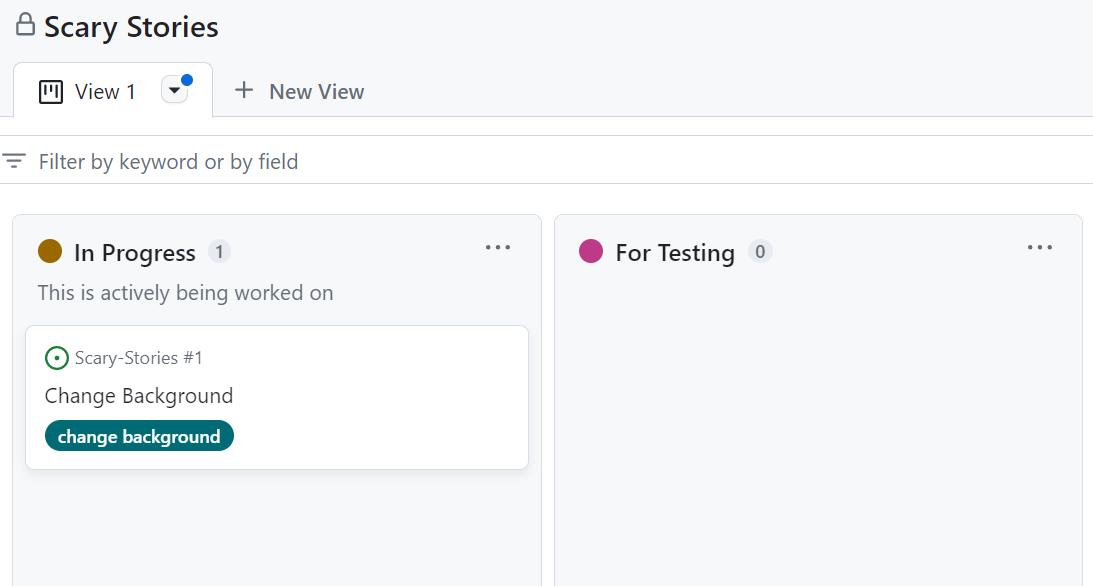
Description automatically generated

### Step 2: QA Environment Check

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



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



## Junior

### Step 1: Taking the Tasks

**Junior** takesthe **task "Change Background"** fromthe **"Sprint TODO" section** andtransfers itto **"In Progress"**.

Graphical user interface, text, application, chat or text message

Description automatically generated

### Step 2: Creating a Branch

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



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

* **Change Background Issue**

Text, application

Description automatically generated

**Junior commits** intobackground **branch** andafterthatmovethe **"Change Background" issue** from "**In Progress**" to "**For Testing**" **column**.

**Text

Description automatically generated**

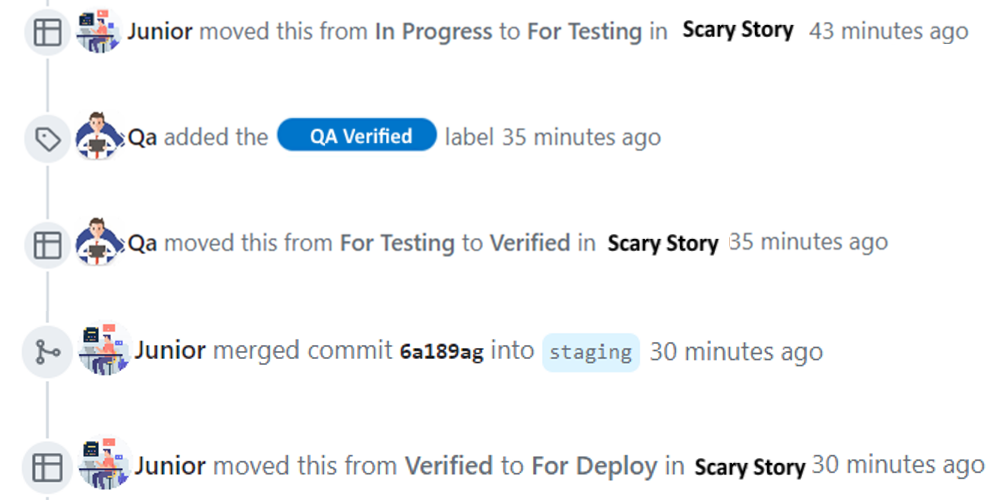
Graphical user interface, text, application, chat or text message

Description automatically generated

### Step 3: Open a Pull Request

Then, if the **issue** is "**Verified**", merge its **feature** **branch** to the **staging** **branch** with a **pull** **request.** Graphical user interface, text, application, email

Description automatically generated



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

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

Text

Description automatically generated

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

Text

Description automatically generated