Plugins and extras:

Particle JS:

Login page on the web application uses particle JS, which is a light weighted JavaScript library for creating particles. This is a NPM package and can be installed with below command

**npm install particles.js**

This library is only just for a good UI and it can be customized based on our requirements like number of particles, type of interaction, colour, etc.

   "particles": {

                "number": {

                    "value": 150,

                    "density": {

                        "enable": true,

                        "value\_area": 700

                    }

                },

                "color": {

                    "value": "#00d4ff"

                },

                "shape": {

                    "type": "image",

                    "image": {

                        src: "./img/uol-logo-png.png",

                        width: 60,

                        height:50

                    },

                },

I have used university of Leicester logo here instead of predefined particles for better UI. 

Tailwind CSS -A utility-first CSS framework for rapidly building custom designs.

“They come with all sorts of predesigned components like buttons, cards, and alerts that might help you move quickly at first, but cause more pain than they cure when it comes time to make your site stand out with a custom design.

Tailwind is different.

Instead of opinionated predesigned components, Tailwind provides low-level utility classes that let you build completely custom designs without ever leaving your HTML.” [<https://tailwindcss.com/#what-is-tailwind>].

By default tailwind imports all its predefined classes into final stylesheet which increases the file size and also makes page loading very slow

“Why do we use PurgeCSS? Tailwind includes a lot of bloat by default so we need to remove anything that isn't being used.

CSS file size before purge -> 414KB

After purge for this default template -> 3KB” [<https://github.com/SteinIAS/Noah>]

. To overcome this, we have installed a plugin called purgeCSS. This will check all the classes used in out markup and add only them to final output and makes file size to 1/10th of initial file size. If in case we have some classes defined in css and not included into the markup which were supposed to added through Javascript, in such case we need to add ‘pa-’ before the class name. Example: pa-active, which means purge allow and it makes sure to include this class in final outputted css.

Noah web development template:

This is Stein IAS' Web Development Template, it has all the plugins added to it and ready to use web development template for any sized application.

SASS:

On this project, we are using sass for styling the web pages. SASS stands for Syntactically Awesome Style Sheets. This is a stylesheet language that will be converted to css.” It allows you to use variables, nested rules, mixins, functions, and more, all with a fully CSS-compatible syntax. Sass helps keep large stylesheets well-organized and makes it easy to share design within and across projects.” [<https://sass-lang.com/documentation>].

This can be added through NPM with this command: **Install node**-**sass**.

SCSS SYNTAX:

$font-stack: Helvetica, sans-serif;

$primary-color: #333;

body {

font: 100% $font-stack;

color: $primary-color;

}

CSS OUTPUT:

body {

font: 100% Helvetica, sans-serif;

color: #333;

}

BEM methodology:

This method is used while naming classes in our style sheet. BEM stands for Block,Element,Modifier. When we are creating a module on a web page, we need to divide them into blocks and use single class name for each block and divide it into elements. Now the class name for the element will be BLOCKNAME\_\_ELEMENT-NAME, by which we can know that the element belongs to that particular block. In case we need some more styling to the same class or using same class with some extra styles, we need to use a modifier. The syntax for using modifier is BLOCKNAME\_\_ELEMENT-NAME—MODIFIER.

A block can have only block name and can have any number of elements and modifiers, but nested naming is not allowed in this. Example: ‘BLOCKNAME\_\_ELEMEMT1\_\_ELEMENT2—MODIFIER’.

“BEM Avoids inheritance and provides some sort of scope by using unique CSS classes per element (like .my-component\_\_list-item).

Reduces style conflicts by keeping CSS specificity to a minimum level.” [<https://blog.decaf.de/2015/06/24/why-bem-in-a-nutshell/>].