**Document for app.js**

This file is created in root directory and used to require and use all the dependencies we have installed in nodejs using npm packet manager.

1. **express-ejs-layouts:**  This module is a middleware which is used to enhance the capabilities of ejs templates. It allows for automatic layout injection.

**What is Automatic Layout Injection?**

Automatic Layout Injection means once we have created a main layout in which all the necessary cdn’s such as bootstrap, stylesheet, scripts are included, it will be automatically injected to other .ejs files we create.

Here, we have created a file main.ejs which is a whole html template. Now, this template is being added to every .ejs file we are creating without using that ejs <% include(‘main’) %> method.

In other .ejs file we directly have to write the respective code for that file.

1. **express-fileupload:**  This module is a middleware which allows us to easily upload files to database which are submitted to server using HTML forms.

Here, we have used it to upload image of recipe to database.

1. **Express-session:** This is a module which is used to manage user sessions.

**What is session?**

A session can be said to be a type of data which is user specific.

For e.g.: Imagine we have an e-commerce website which have a shopping cart feature.  
i) When user visits website adds some items to shopping cart. For each item they add to their cart the server creates or updates the user session with the information of that product.

ii) The session data is persisted across the different page visit. This allows user to navigate through website without losing their cart items. Each time the user interacts with the website, their session id is sent along with the request to the server, ensuring that their cart data is always accessible.

1. **Cookie-parser:** This module is a middleware which is used to parse and handle cookies that are sent between the client’s browser and the server. Cookies are the small piece of data that are stored on the user’s browser and are often used to store information about the user’s session, preferences and the interactions with the website.
2. **Connect-flash:** This is a middleware which is used to send temporary message to the user.

**What are flash messages/ temporary messages?**

These are the types of messages which are used to give immediate response to user for their actions. These messages are shown to user for short period of time.

In our website, when user submit the form for adding new recipes a message is shown at the top form when the form is successfully submitted or there’s an error. It is done with the help of this module.

**Knowing more about the code written here:-**

1. **app.use(cookieParser('CookingBlogSecure')):** This is how we use a cookie parser in our file.

‘CookingBlogSecure’ is a secret key passed to cookieParser which is used to sign the cookies, adding a layer of security to prevent tampering.

\*\*\*\*\*\*\*How it is done is a part of Cryptography, which is not discussed here.\*\*\*\*\*\*\*\*\*

1. **app.use(session({**

**secret:'CookingblogSecertSession',**

**saveUninitialized: true,**

**resave: true**

**})):** This is how we use express-session in our project.

**secret:'CookingblogSecertSession’ --- >** Here, we have set a secret value which is used to sign session data.

**saveUninitialized: true -- >** This allows uninitialized sessions to be stored in the session store. It's typically used when you want to establish a session even if the session data is not immediately populated.

**resave: true --- >** This force session to saved back to session store even if there are no changes.

1. **app.set('layout', './layouts/main'):**  This is how we use express-layouts. In this line of code we are setting the default layouts for our views. It means the file name main.ejs inside layout directory will automatically added to the all the .ejs file inside views directory. This is how we do automatic layout injection.
2. **app.set('view engine', 'ejs'):** This line sets the default view engine for rendering templates to EJS. The view engine is responsible for interpreting your template file and rendering them as HTML that the browser can display.
3. **const routes = require('./server/routes/recipeRoutes.js'):** We know that we are using MVC architecture to develop our web app so we define all the routes in different file. In this line of code we are exporting or requiring that routes file here and saving all the data received from requiring that file in routes constant.
4. **app.use('/', routes):** This line of code tells us that the file recipeRoutes.js which is stored inside routes constant as discussed above will handle all the request made to main or root route(“/”).