### TABLE OF CONTENTS

**CHAPTER- 1: ABSTRACT…….…………………………………………. . . . . . . . . .0**

**CHAPTER-2: INTRODUCTION ………………………………………… . . . . . . .. 0-0**

2.1 EXISTING SYSTEM………………………………………………0

2.2 PROPOSED SYSTEM……………………………………………….0

**CHAPTER-3 SYSYTEM REQUIREMENTS……………………………. . . . . . . . . 0-0**

3.1 SOFTWARE REQUIREMENTS…………………………………….0

3.2 HARDWARE REQUIREMENTS………………………………….0

3.3 FUNCTIONAL REQUIREMENTS………………………………..0

3.4 NON-FUNCTIONAL REQUIREMENTS…………………………0

**CHAPTER -4: DESIGN……………………………………………………… . . . . . .0-0**

4.1 DATA FLOW DIAGRAM……………………………………………0

4.2 USE CASE DIAGRAM………………………………………………0

4.3 CLASS DIAGRAM…………………………………………………...0

4.4 SEQUENCE DIAGRAM……………………………………………...0

4.5 ACTIVITY DIAGRAMS……………………………………………...0

**CHAPTER -5: IMPLEMENTATION …………………………………… . . . . . . . .0-0**

5.1SCREENSHOTS………………………………………………... . . . .0-0

5.2SAMPLE CODE……………………………………………. . . . . . . .0-0

**CHAPTER-6: FUTUREENHANCEMENT……………………………… . . . . . . . .0**

**CHAPTER -7: CONCLUSION …………………………………………….. . . . . . ..0**

**CHAPTER-8: REFERENCES………………………………………………………..0**

1. **ABSTRACT:**

Code editors are vital tools for developers, offering features like syntax highlighting, automatic indentation, error-checking, auto-completion, and code snippets to enhance coding efficiency. Mastery of these tools is essential for careers in software engineering and web development, as they simplify the coding process and improve productivity. While similar to Integrated Development Environments (IDEs), code editors are distinct. IDEs incorporate additional functionalities like debugging and compiling, whereas code editors are lighter, focusing on providing a streamlined space for writing and organizing code. This simplicity allows developers to concentrate on the core task of coding without the added complexity of managing other integrated features. Key functionalities such as error-checking help identify mistakes early, reducing the need for manual debugging. Auto-completion predicts what the developer may want to write next, speeding up the coding process and minimizing errors. This project introduces a webbased code editor that enhances developer productivity and code quality. By offering essential features like error detection and auto-completion, it provides a user-friendly environment suited for both novice and experienced developers.

1. **INTRODUCTION:**

In the rapidly advancing world of technology, coding has become a fundamental skill, underpinning everything from web development to complex software engineering. As the demand for innovative software solutions grows, so does the need for efficient coding environments. The ability to write clean, optimized code is crucial for developers, whether they are building websites, applications, or system software. However, the complexity of modern programming languages and the increasing demands of multi-language projects have made coding more challenging than ever.

Code editors play a pivotal role in simplifying the coding process by providing a userfriendly interface equipped with features like syntax highlighting, automatic indentation, and errorchecking. These features make writing code more intuitive and reduce the likelihood of errors. Traditionally, developers had to use different code editors for each programming language, which led to inefficiencies and slowed development. This fragmented approach made it difficult to work seamlessly across multiple languages or integrate various tools in one environment.

The integration of multiple programming languages and essential features like error detection, auto-completion, and code snippets into a single code editor solves this problem, making coding more accessible and productive. With an all-in-one platform, developers can focus on writing quality code without worrying about switching between editors or manually managing common tasks.

**2.1 Existing Systems**

The code editors in the previous generation are traditional code environments that are created to write the code in a particular or specified language and has lot of issues like

Problems with Traditional IDEs:

1. **Single-Language Limitation**:

Many IDEs are designed with a specific programming language in mind (e.g., PyCharm for Python, IntelliJ for Java), limiting flexibility for developers who need to work in multiple languages.

Switching between IDEs can be cumbersome.

2**.Heavy Resource Usage**:

IDEs often require significant CPU and RAM resources to run smoothly. This can slow down the overall system, especially when working with large projects or when multiple IDEs are open simultaneously.

3. **Large Installation Sizes**:

IDEs tend to be bulky and consume a lot of storage space. For instance, setting up an IDE like Visual Studio can take up several gigabytes of disk space, making them less ideal for lightweight or mobile setups.

4. **Complex Installation Processes**:

Developers often need to install different IDEs for different languages, which can lead to a fragmented development environment. Managing multiple IDEs can be time-consuming and errorprone.

5. **System Compatibility**:

Some IDEs require specific system configurations or operating systems. Cross-platform compatibility can be an issue, and developers may face limitations when switching between operating systems.

6. **Long Loading Times**:

The startup time for some IDEs is relatively slow, and working on large projects can lead to sluggish performance, further reducing productivity.

**2.2 Proposed Systems:**

The proposed system is a full-fledged web application that provides developers with a powerful, all-in-one platform supporting more than six different programming languages. This multi-language support allows users to code in diverse programming environments, including popular languages such as Python, Java, C++, JavaScript, and others, without the need for separate installations or switching between multiple platforms. Whether users are working on web development, software engineering, or scripting, the system offers a seamless experience by accommodating different languages under one roof.

**Key Features:**

1**)Support for Data Structures and Algorithms:**

* Beyond basic coding, the system empowers developers to work on complex data structures and algorithms. This feature is ideal for students, professionals preparing for coding interviews, or developers working on advanced software solutions.
* The editor provides specific functionalities like code suggestions and optimizations related to common data structures (e.g., arrays, linked lists, trees) and algorithms (e.g., sorting, searching, dynamic programming), enhancing coding efficiency.

2) **Real-Time Code Execution and Debugging:**

* The application allows developers to write, compile, and run their code directly within the browser, eliminating the need for external compilers or installations. Real-time error detection and debugging features provide immediate feedback, helping developers identify and correct issues on the go.
* Integrated debugging tools help track variables, step through code execution, and monitor memory usage, giving users complete control over their code’s behavior.

3)**Minimal Resource Usage:**

* Unlike traditional IDEs that consume significant CPU, RAM, and storage space, the web-based nature of the proposed system requires minimal resources, ensuring smooth performance even on lower-spec devices. This lightweight setup reduces system load and makes the editor accessible to a broader range of users, including those on older or less powerful machines.

**3.System Requirements**

As a online web application it does not require much of the software or hardware requirements

**3.1 Software Requirements**

* Operating System : Windows 7 or higher
* Programming language : Javascript, Node JS, React JS, HTML, CSS.
* Platform : VS CODE.

**3.2 Hardware Requirements**

* Processor: 1.3 gigahertz (GHz) frequency or above.
* RAM: 2 GB
* Hard disk: 100 GB.
  1. **Functional Requirements**

User/ Admin :-User can be able to fully utilize all the features to use the tool as a medium

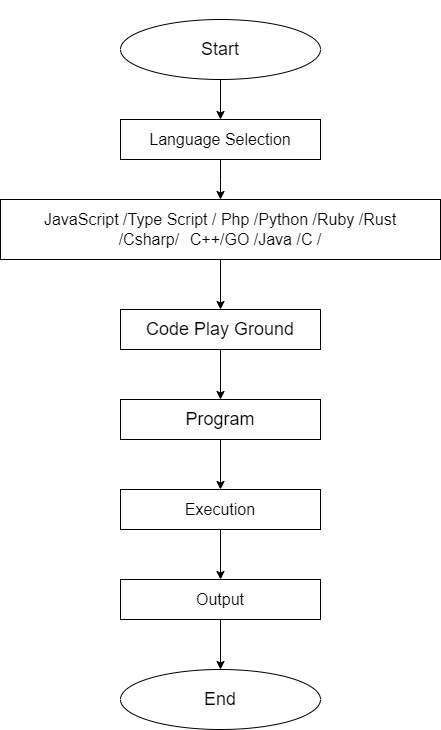
to complete the code .

* 1. **Non-Functional Requirements**
* **FLEXIBILITY-** The Website can be used easily without any trouble.
* Robust-Error checking has been built into the system to write the code clean and effectively.
* **SCALABILITY-**The system can be extended to integrate the modifications done in the present application to improve the quality over time in the future, it is possible to integrate many features.
* **RELIABILITY**-Since the application is being developed through the frameworks of javascript it made it possible by improving it in many ways because of development in javascript and it’s frameworks.
* **PORTABILITY**-System must be enough such that the user with an average background can quickly experiment with the system and learn how to use the project. The system had a friendly interface.

**4. DESIGN:**

**4.1. DATA FLOW DIAGRAM:**

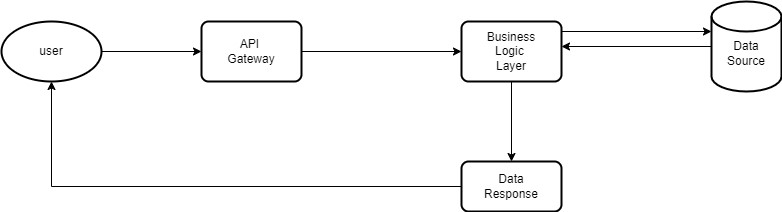
**ZERO LEVEL DFD**:



*Figure:1 Zero Level DFD*

As the name suggest above the zero level data flow diagram illustrates how data is processed by a system in terms of inputs and outputs. As it’s name indicates its focus in on the flow of information from on step to the other.

**First Level DFD:**



*Figure:2 First Level DFD*

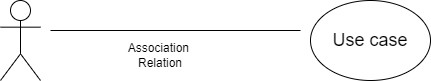
The above figure shows the movement of the data and describes the process that happening behind it.

**4.2.1 Use Case Diagram**

What is a Use Case Diagram?

The purpose of a use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as the other four diagrams (activity, sequence, collaboration, and State chart) also have the same purpose. We will look into some specific purpose, which will distinguish it from the other four diagrams.

A Use Case Diagram has the following Notations:



USE EASES: A use case describes a function that a system performs to achieve the user's goal. A use case must yield an observable result that is of value to the user of the system.

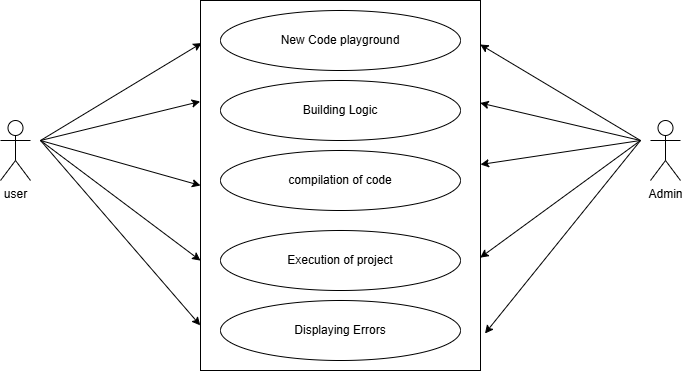
ACTORS: An actor represents a role of a user that interacts with the system that you are modelling. The user can be a human user, an organization, a machine, or another external system

ASSOCIATIONS: A line between actors and use cases that represents the relationship between an actor and a business use case. It indicates that an actor can use a certain functionality of the business system.

**Benefits of Use Case Diagram:**

* considering the system from the user's viewpoint.
* getting different angles on the system and the requirements
* deriving recommended requirements
* deriving test cases
* and through all that, developing the right system for users.

Use case diagram for the proposed system:



*Figure:3 Use Case Diagram*

The Above Figure describes a function that a system performs to achieve the user’s goal. A use case must yield an observable result that is of value to the user of the system.

**Explanation:**

In this proposed system we have 2 actors and 4 cases. The relation between them is association.

Here the admin has an association relation with the playground. Execution of code and displaying errors. The user has the association relation with the playground, Building logic and its execution.

**4.2.2. CLASS DIAGRAM:**

What is a class diagram?

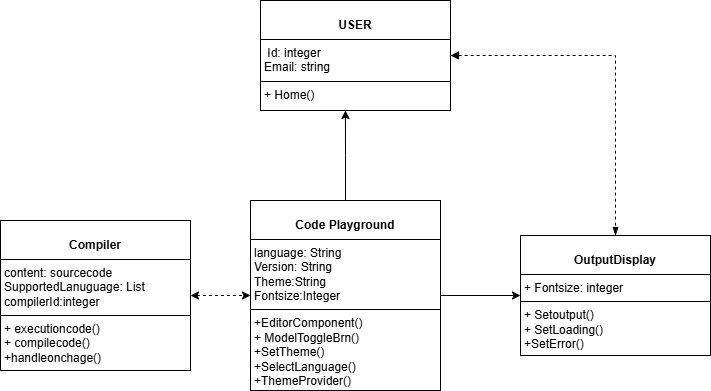
Class diagram is a static diagram. It represents the static view of an application, Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constrains executable code of the software application. It describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because they are the only UML. diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints.

The advantages of the class diagram: Class diagrams give you a sense of orientation. They provide detailed insight into the structure of your systems. At the same time, they offer a quick overview of the synergy happening among the different system elements as well as their properties and relationships. Class diagrams are simple and fast to read. With the right software they are also easy to create. They are the foundation for creating systems

CLASS DIAGRAM NOTATION:

* UML class is represented by the following figure. The diagram is divided into four parts.
* The top section is used to name the class.
* The second one is used to show the attributes of the class.
* The third section is used to describe the operations performed by the class
* The fourth section is optional to show any additional components.

**Class Diagram for the Proposed System:**



*Figure:4 class diagram*

In the above figure class diagram sense of orientation. That Provide detailed insight in to the structure of your systems. At the same time, they offer a quick overview of the synergy happening among the different system.

**4.2.3 SEQUENCE DIAGRAM:**

What is Sequence Diagram?

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

Uses of sequence diagrams:

* Used to model and visualise the logic behind a sophisticated function, operation or procedure.
* They are also used to show details of UML use case diagrams.
* Used to understand the detailed functionality of current or future systems
* Visualise how messages and tasks move between objects or components in a system.

**Benefits of a Sequence Diagram**

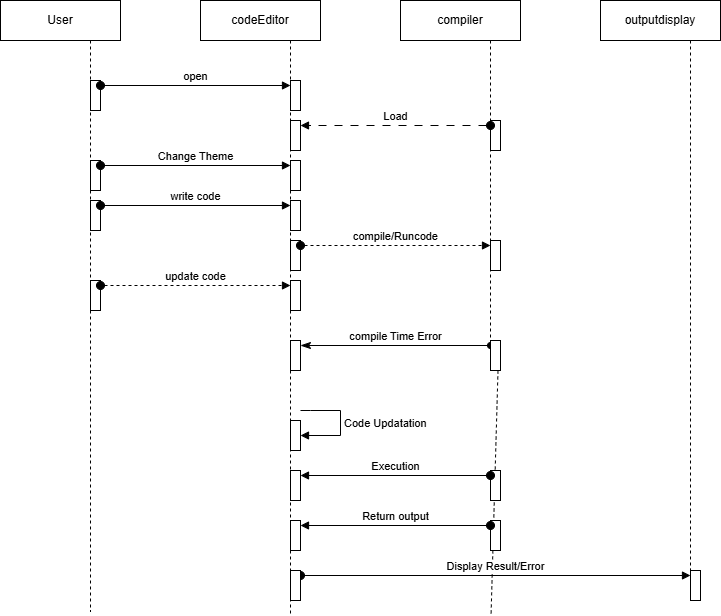
* Sequence diagrams are used to explore any real application or a system.
* Sequence diagrams are used to represent message flow from one object to another object.
* Sequence diagrams are easier to maintain.
* Sequence diagrams are easier to generate.
* Sequence diagrams can be easily updated according to the changes within a system.
* Sequence diagram allows reverse as well as forward engineering.

**Sequence Diagrams for the proposed system:**

**Explanation:**

In the above figure, the general system architecture's flow of the messages between different

operations is being displayed. There are 4 different classes and one actor.



*Figure :5 Sequence Diagram*

In the above figure, the general system architecture’s flow of the messages between different operations is being displayed.

**4.2.4 Activity Diagram**

What is an Activity Diagram?

In UML, the activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models the concurrent and sequential activities rather than the implementation. It models concurrent and sequential activities.

The activity diagram helps in envisioning the workflow from one activity to another. It put emphasis on the condition of flow and the order in which it occurs. The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc.

It is also termed as an object-oriented flowchart. It encompasses activities composed of a set of actions or operations that are applied to model the behavioural diagram.

**Components of an Activity Diagram**

Activities: The categorization of behaviour into one or more actions is termed as an activity. In other words, it can be said that an activity is a network of nodes that are connected by edges.

The edges depict the flow of execution. It may contain action nodes, control nodes, or object nodes

The control flow of activity is represented by control nodes and object nodes that illustrates the objects used within an activity. The activities are initiated at the initial node and are terminated at the final node

Activity

Activity partition / swimlane: The swimlane is used to cluster all the related activities in one column or one row. It can be either vertical or horizontal. It used to add modularity to the activity diagram. It is not necessary to incorporate swimlane in the activity diagram. But it is used to add more transparency to the activity diagram.

|  |
| --- |
| Swimlanee |
|  |

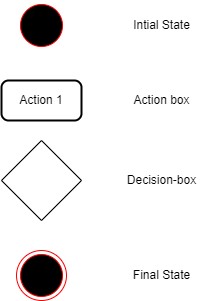
**Notation of an Activity diagram**

**Initial State**: It depicts the initial stage or beginning of the set of actions.

**Final State**: It is the stage where all the control flows and object flows end.

**Decision Box**: It makes sure that the control flow or object flow will follow only one path.

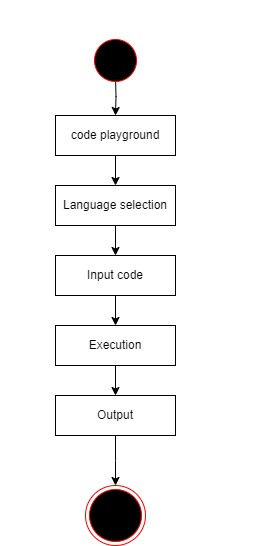
**Action Box**: It represents the set of actions that are to be performed.



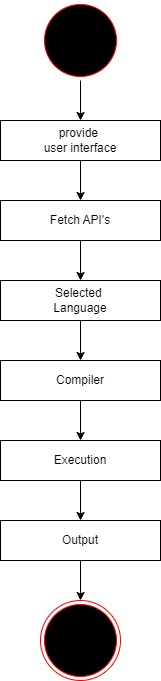
**Benefits of activity diagrams**

Activity diagrams present a number of benefits to users. Consider creating an activity diagram

* Demonstrate the logic of an algorithm.
* Describe the steps performed in a UML use case.
* Illustrate a business process or workflow between users and the system.
* Simplify and improve any process by clarifying complicated use cases.
* Model software architecture elements, such as method, function, and operation.



*Figure :6 Activity Diagram for User*



*Figure :7 Activity Diagram for Admin*

**5. IMPLEMENTATION**

This section outlines the development plan for the Evolve Studio Code Editor web application. The plan focused on creating a platform where users can seamlessly write, execute, and manage their code across multiple programming languages.

The development process was divided into several key phases:

**Phase 1: Setting up the Development Environment**

The first phase involved setting up the development environment and installing the necessary tools and libraries. We created a project skeleton and established the basic structure of the application. This phase was critical in laying the foundation for the entire development process.

**Phase 2: Building the Frontend Interface**

In the second phase, we focused on developing the frontend user interface using React.js,

Typescript, Node.js, Tailwind.CSS. We utilized various UI libraries and frameworks, such as Material-UI and Bootstrap, to create an attractive and responsive design. The goal was to ensure that users could easily access, write, and execute their code in a user-friendly environment.

**Phase 3: Fetching APIs**

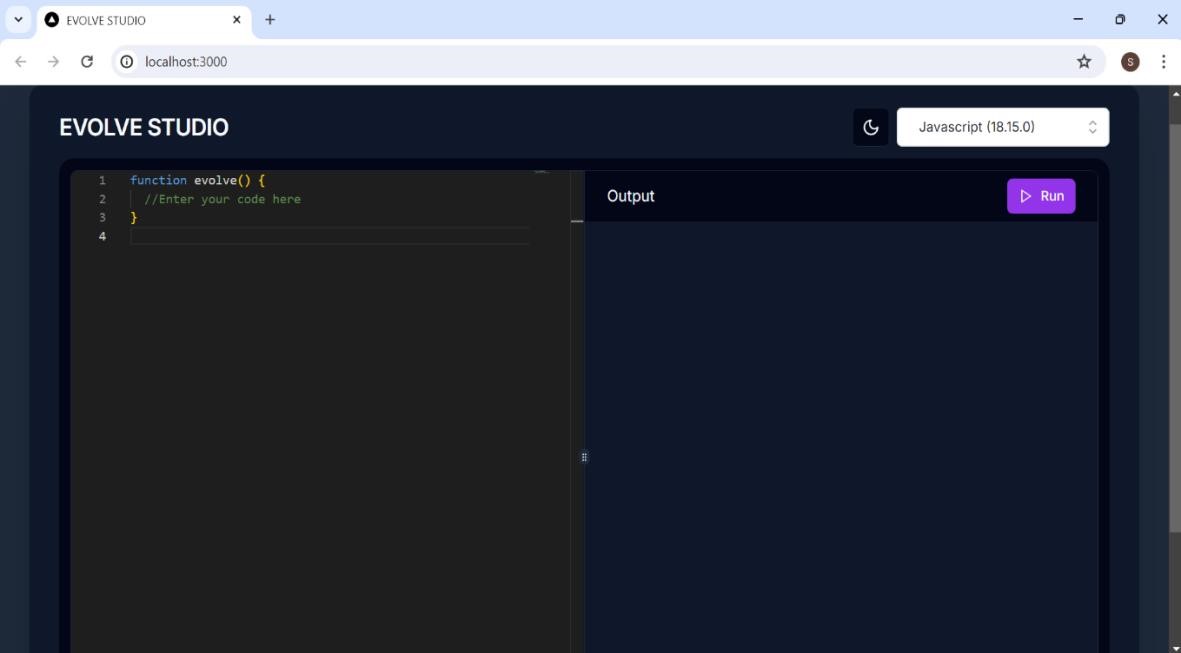
In the third phase, we fetched APIs to handle data retrieval and execution of programs in different programming languages. By using APIs, we eliminated the need for a dedicated database for program execution, streamlining the backend architecture while supporting multiple languages efficiently.

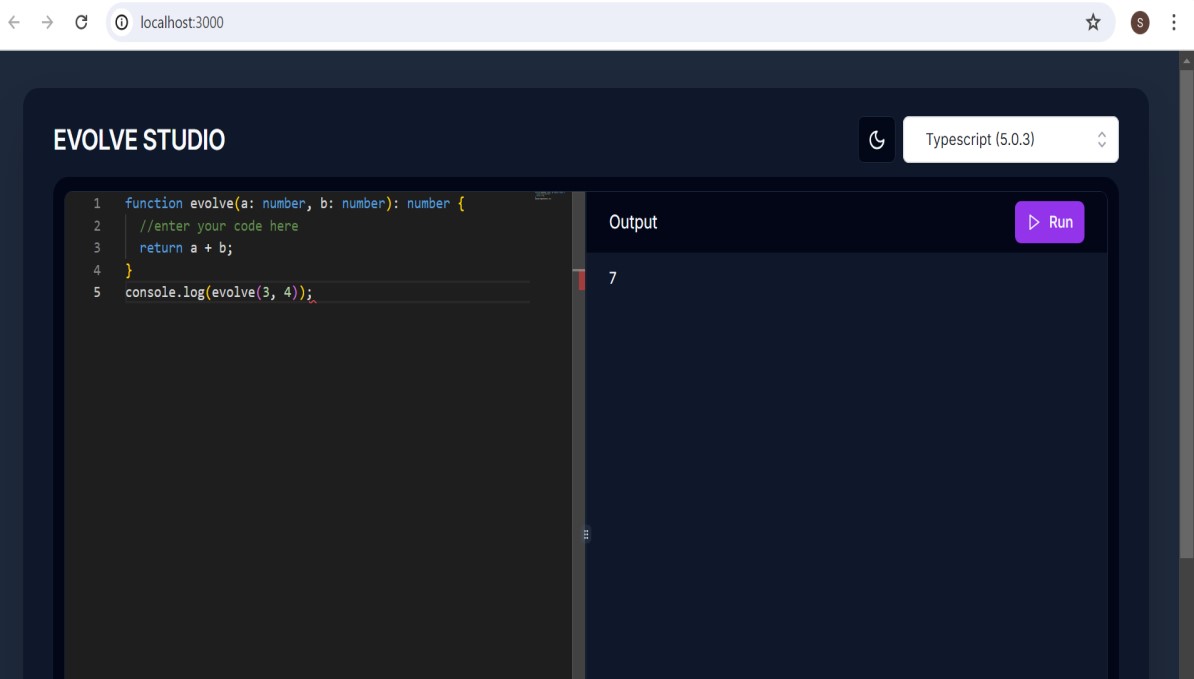
**Phase 4: Testing and Deployment**

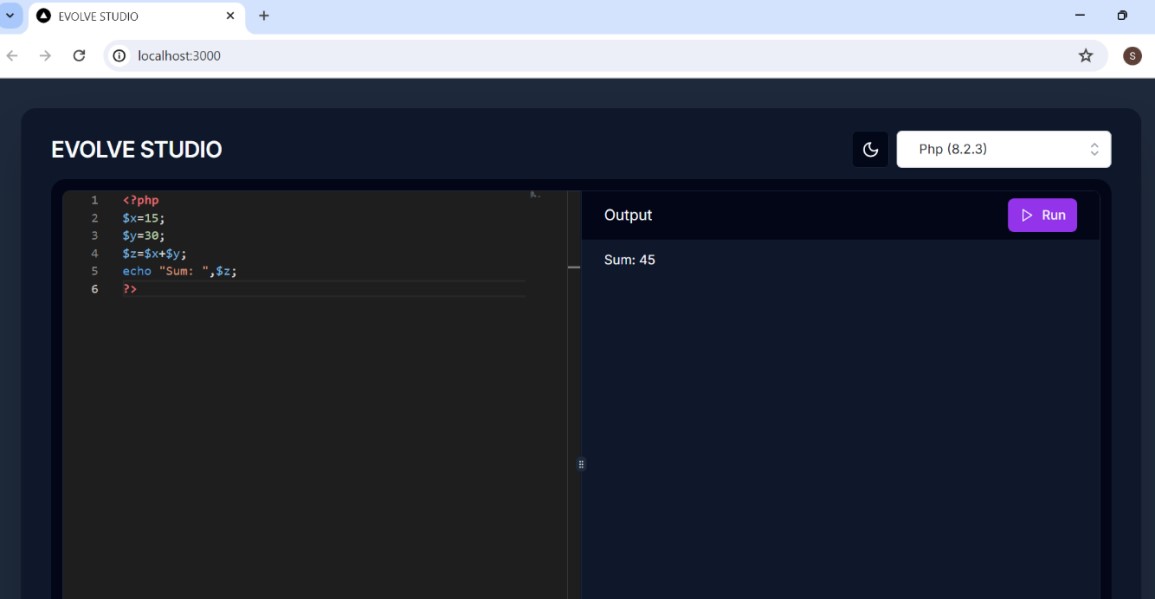
In the final phase, we conducted extensive testing and debugging to ensure that the application was error-free and performed optimally. Once testing was complete, we deployed the project on the web, making it accessible to users worldwide.

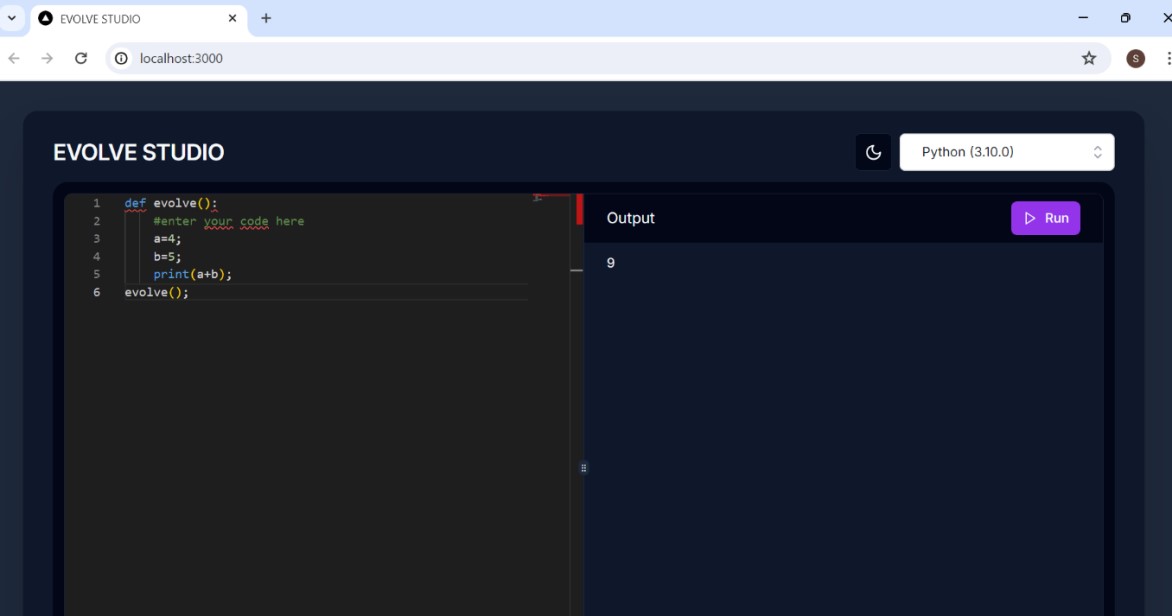
### 

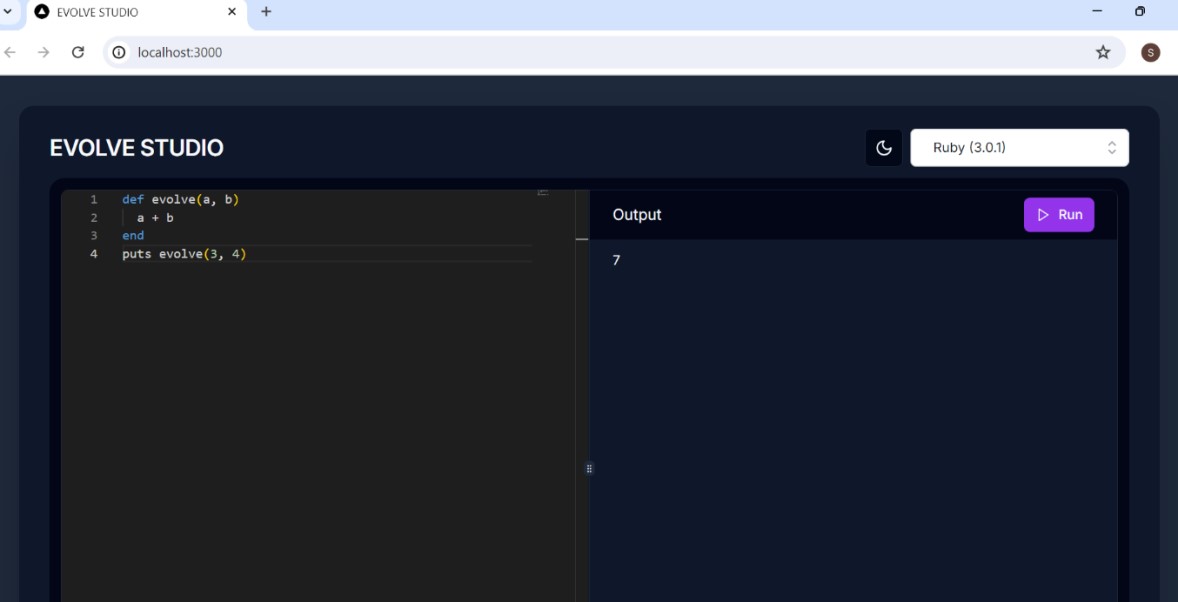
### 5.2 Screenshots

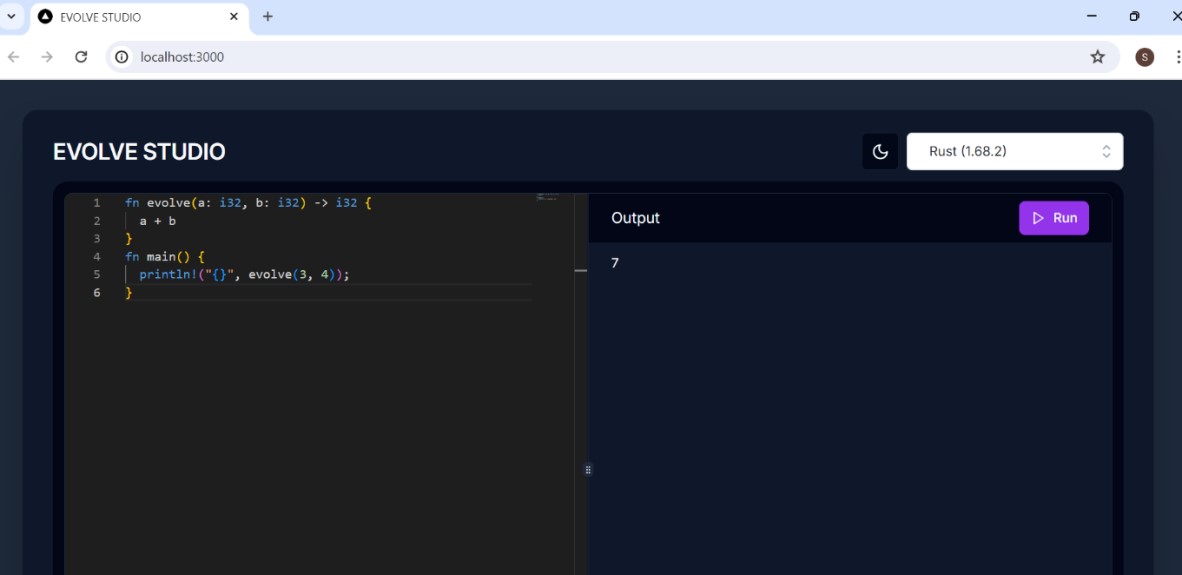


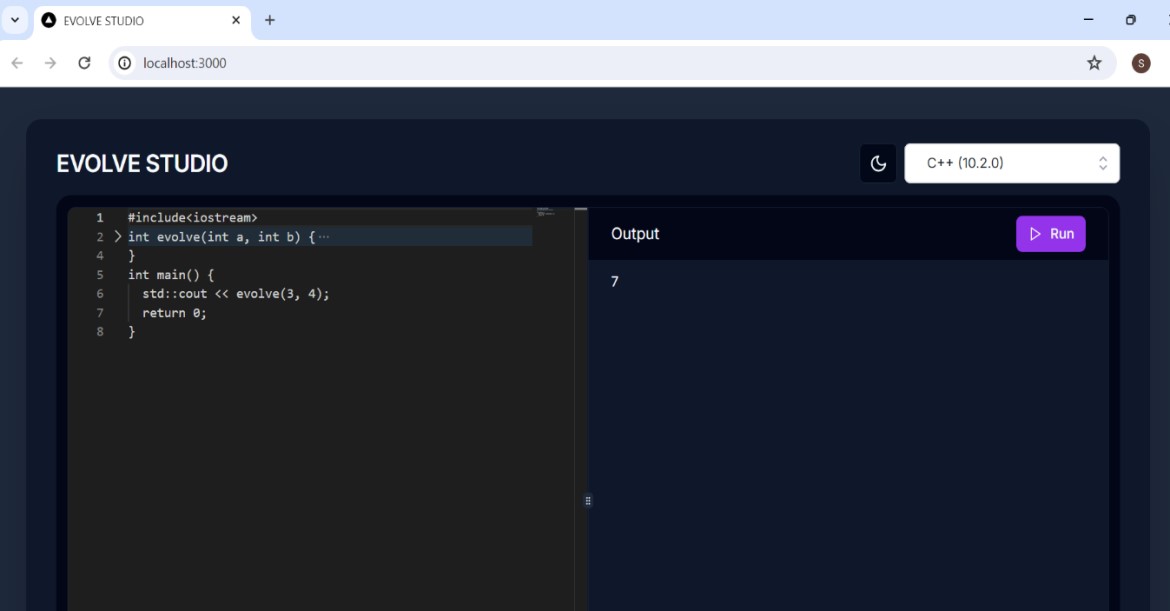


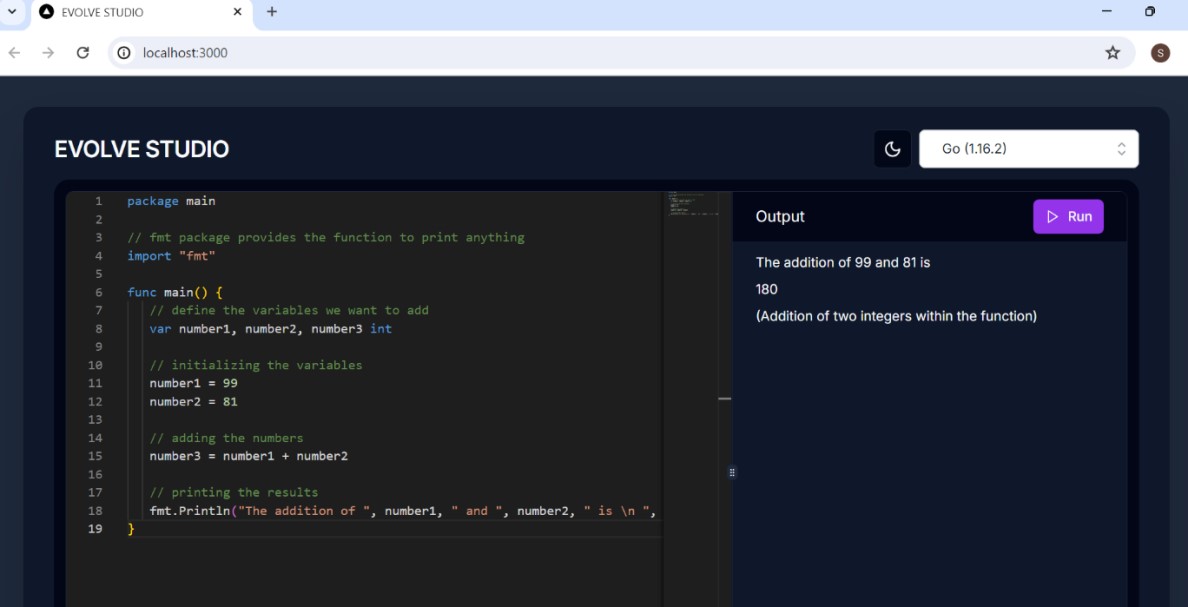


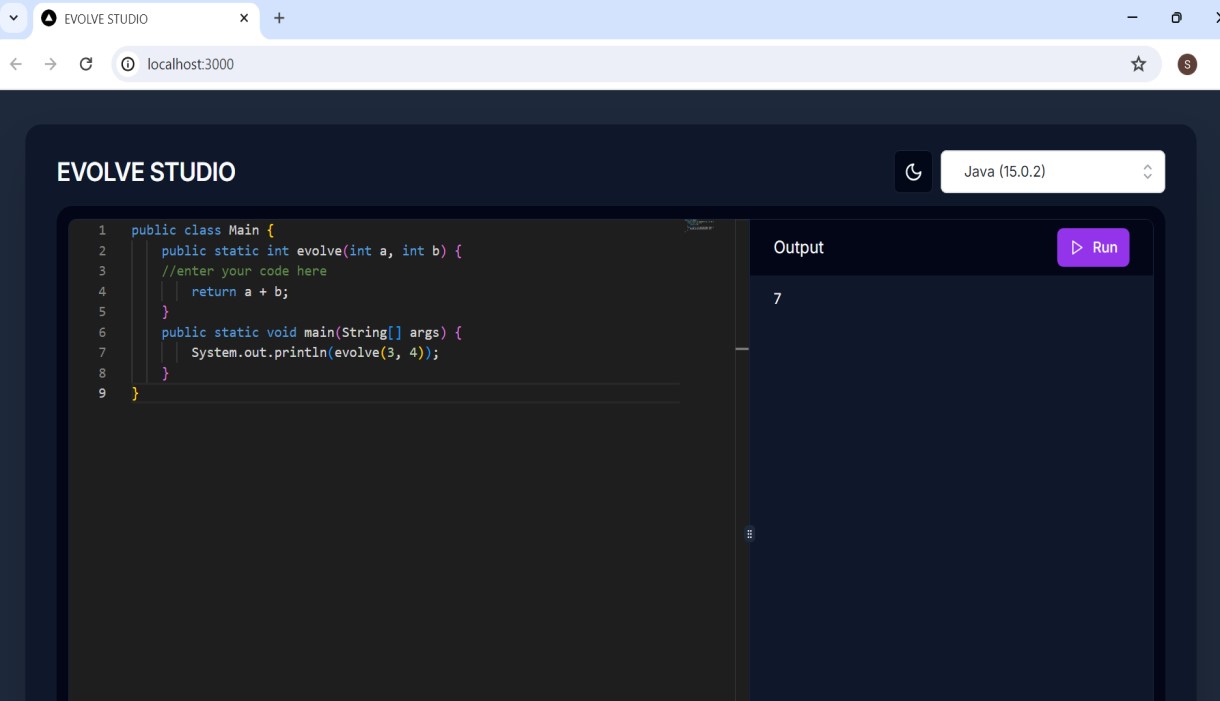


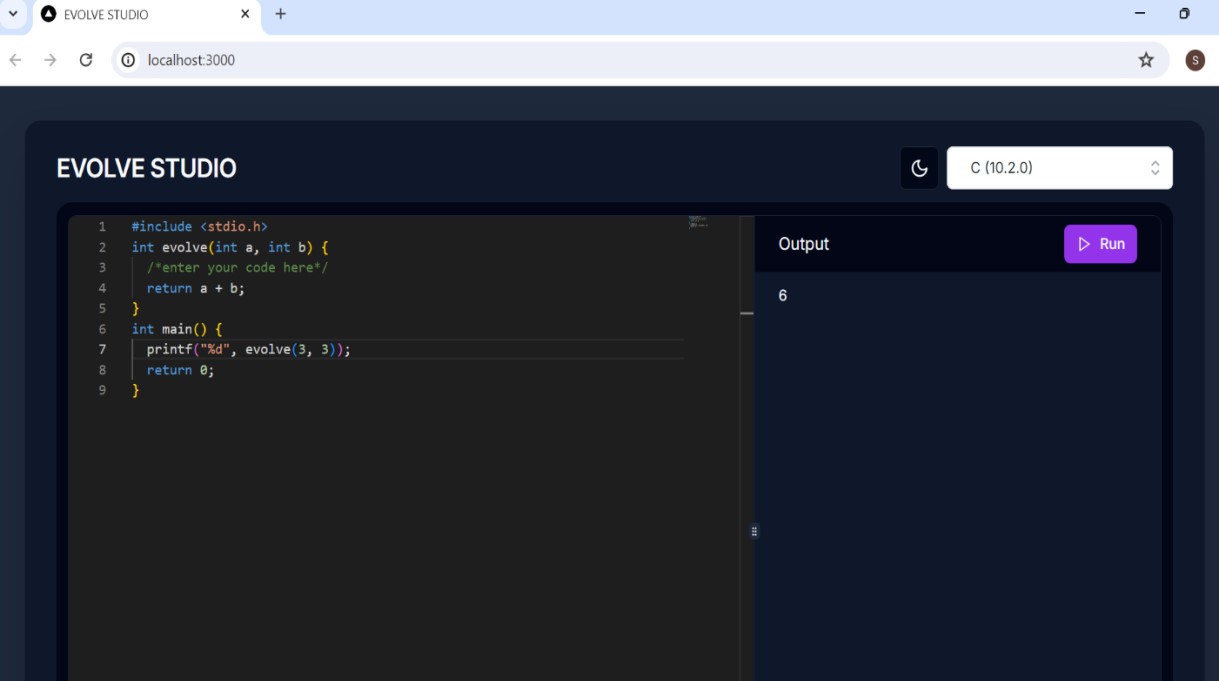






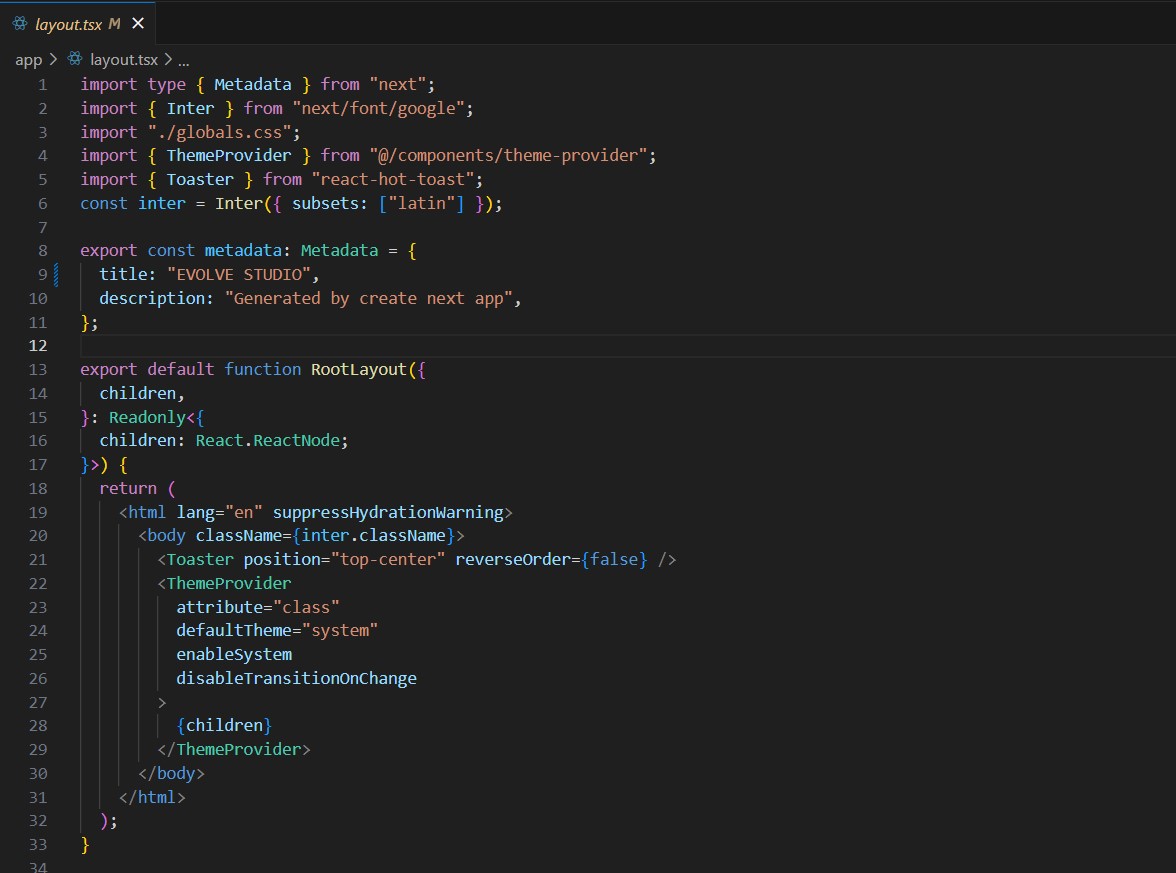




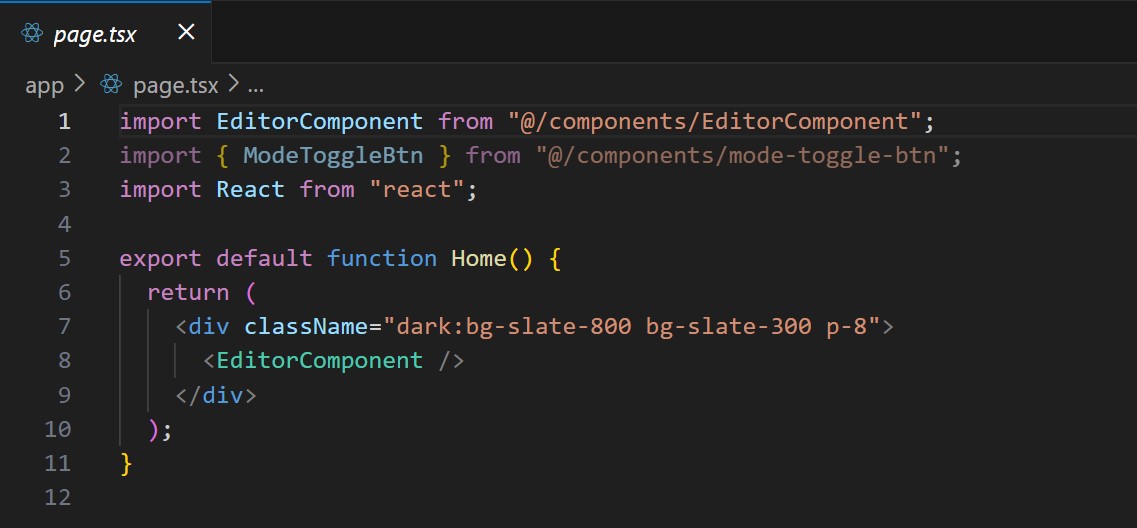


**5.3 Sample Code**

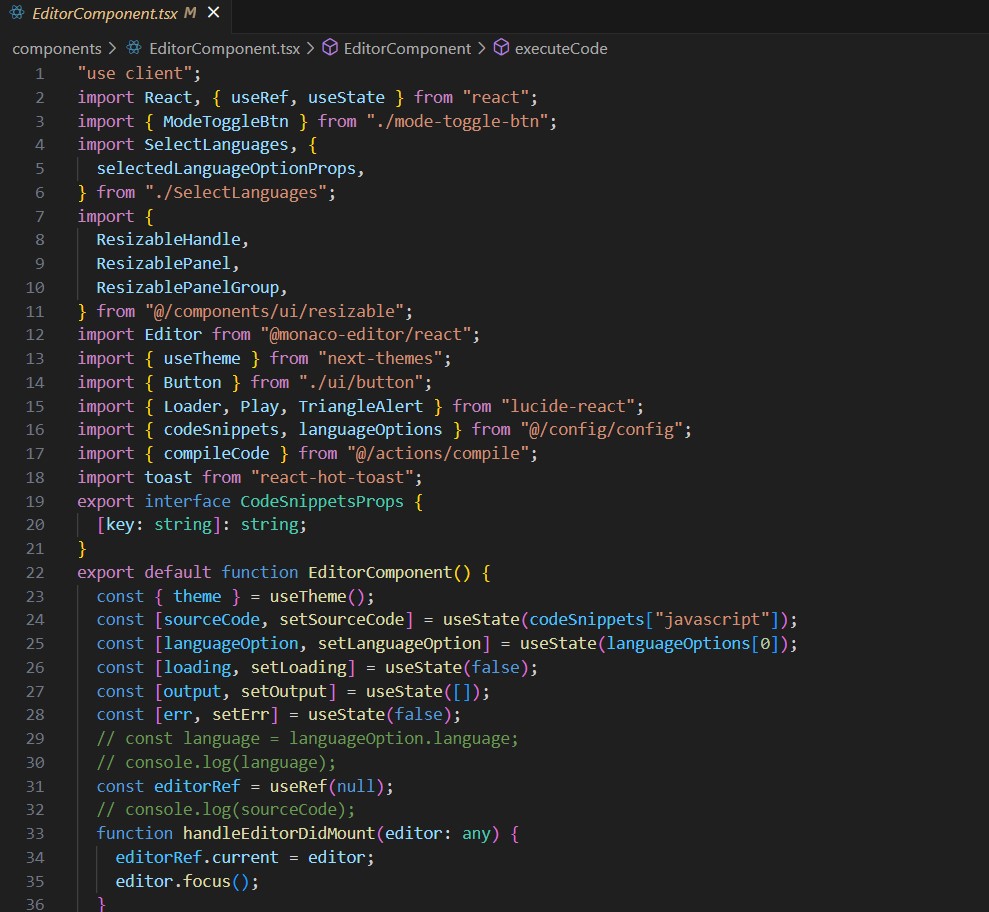
**Layout.tsx:**

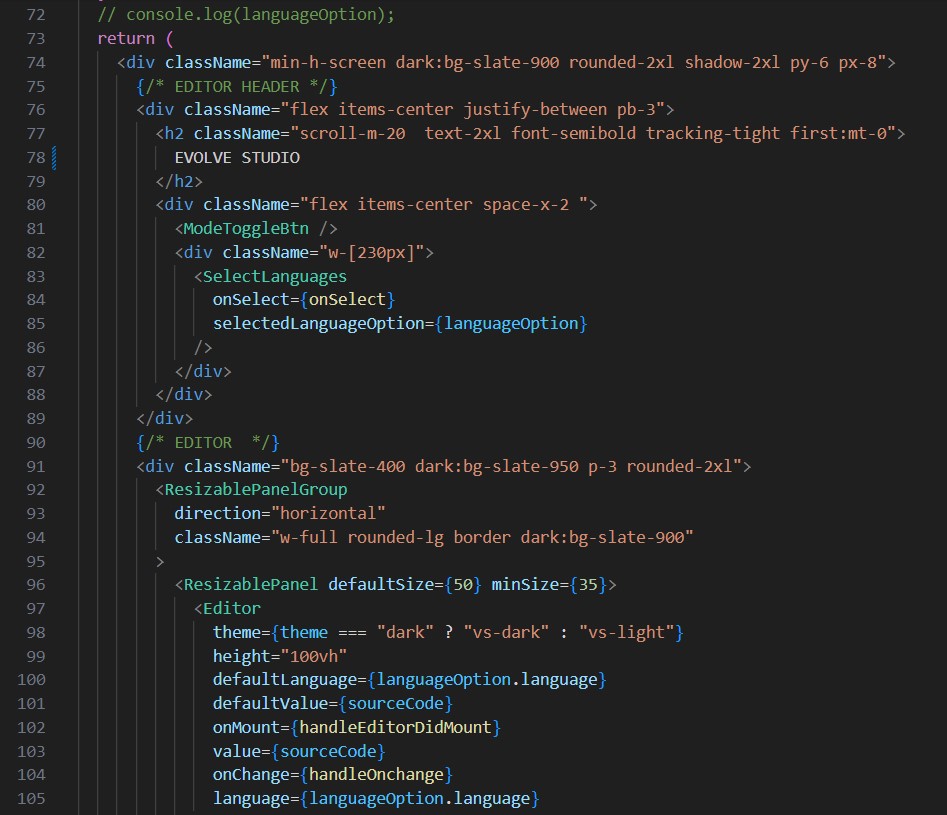


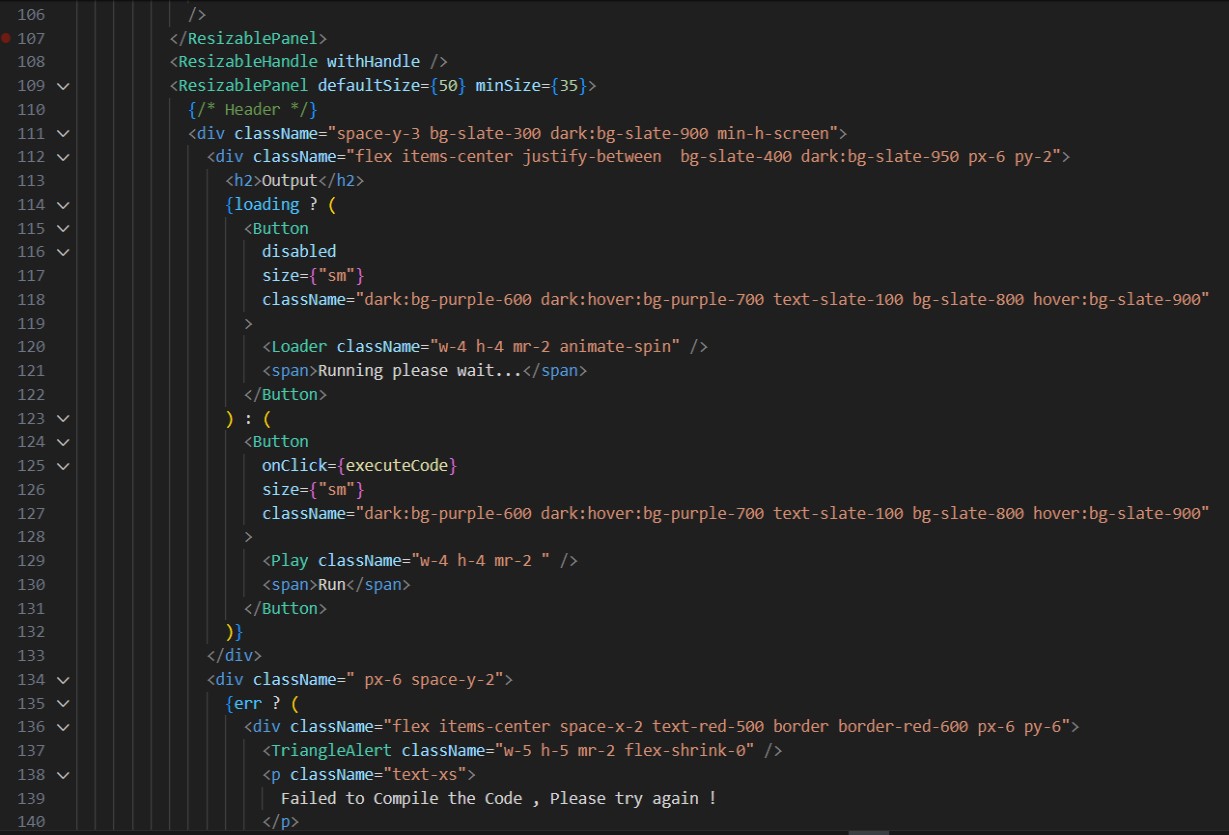
**Page.tsx:**



**EditorComponenet.tsx:**





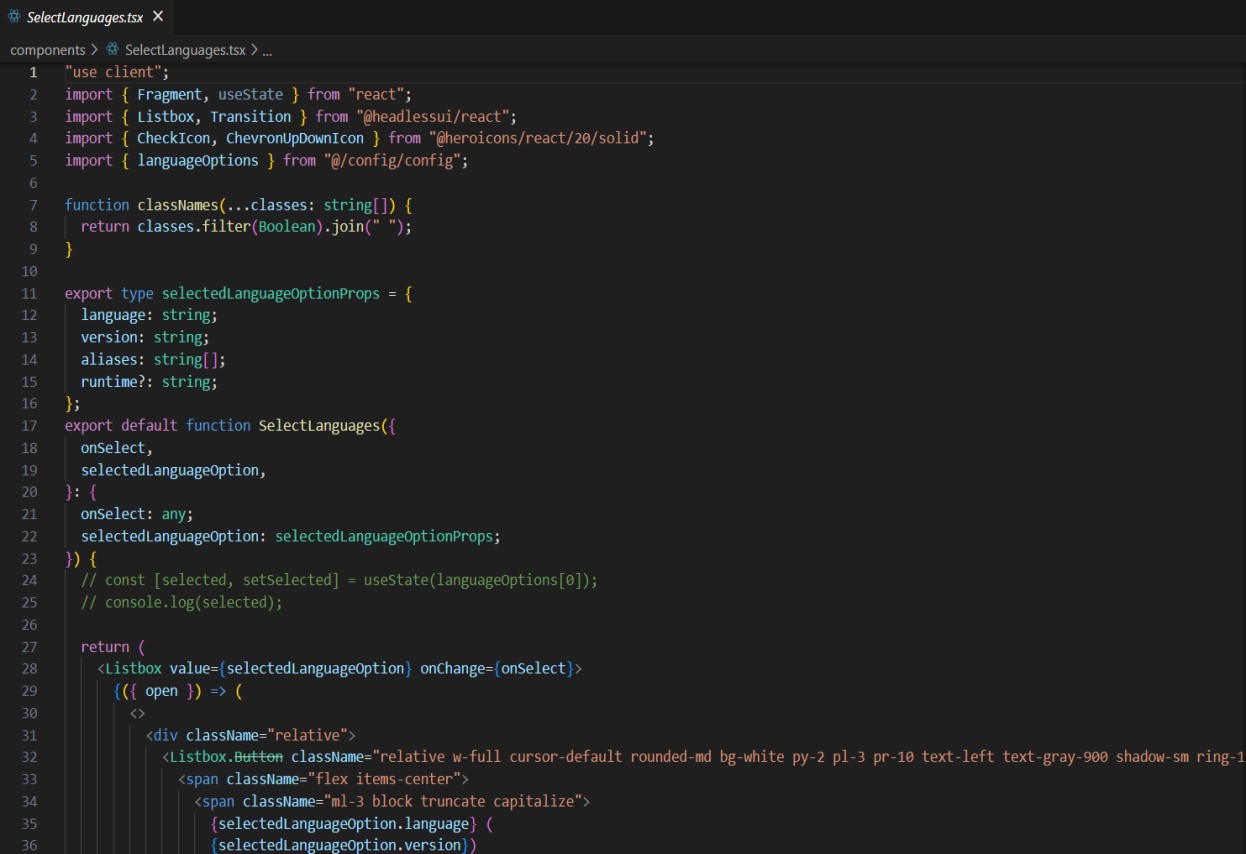


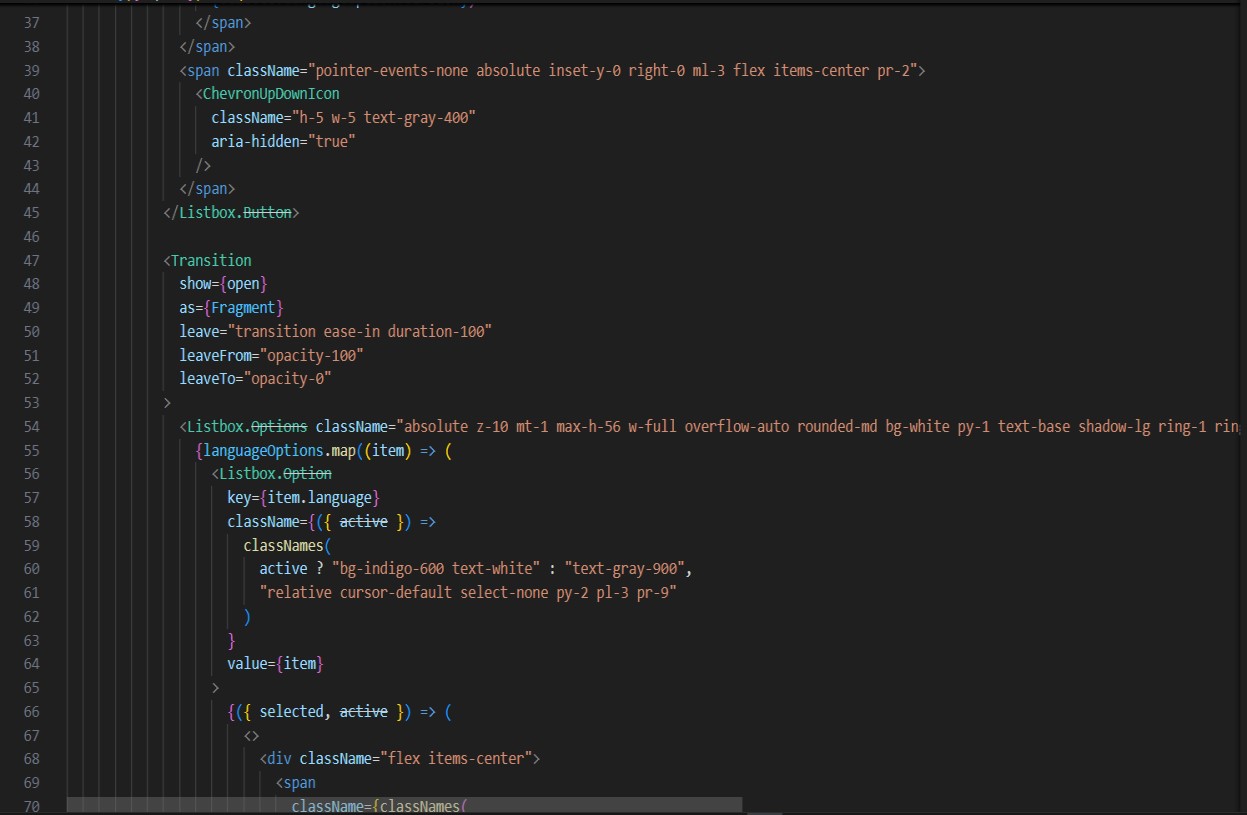


**mode-toggle-btn.tsx:**



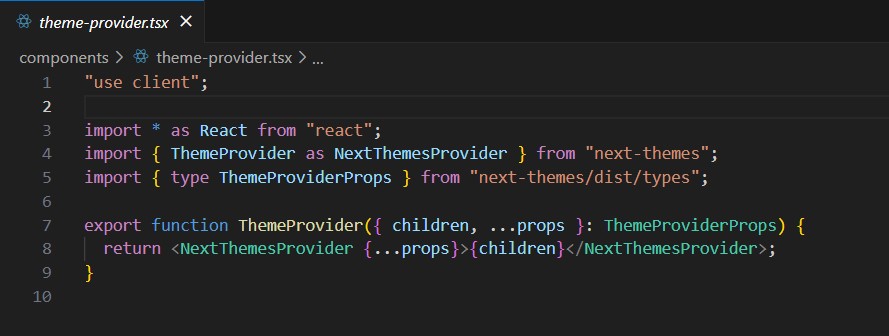
**SelectLanguages.tsx:**



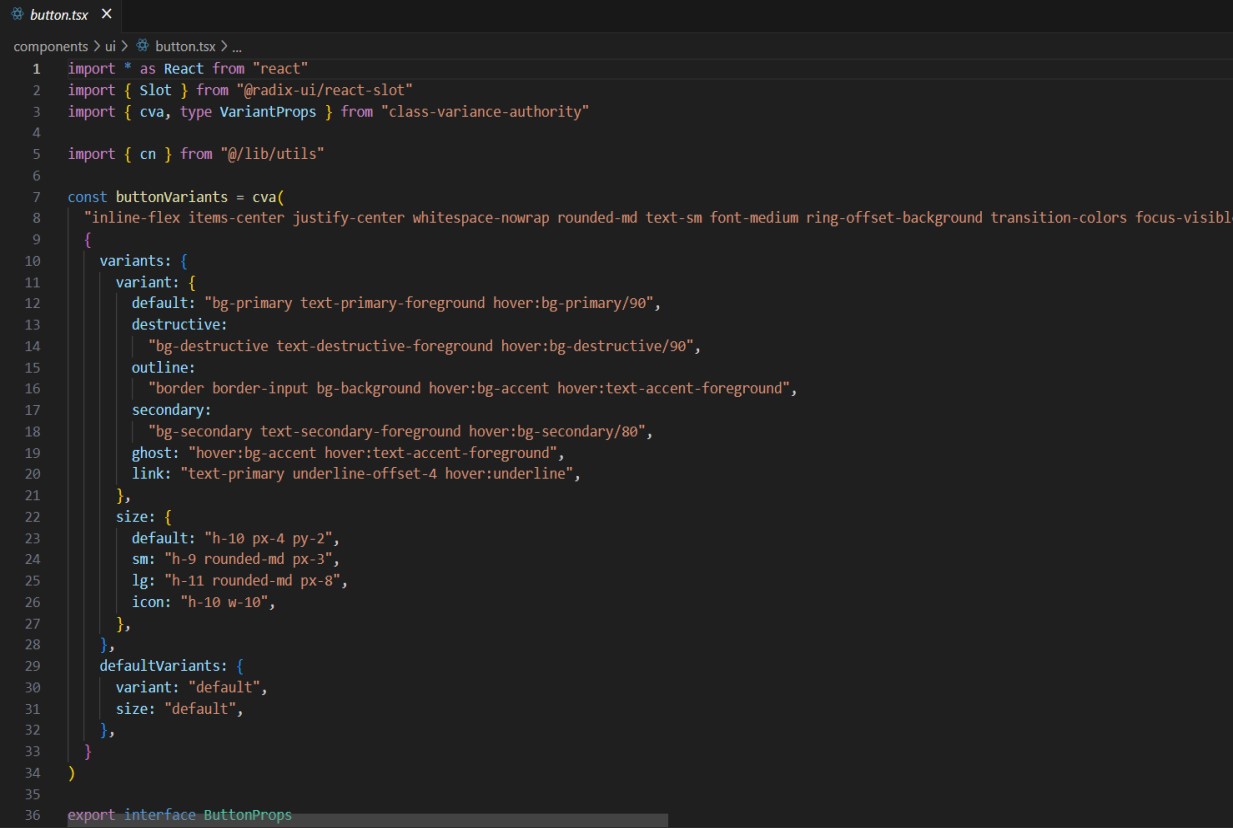


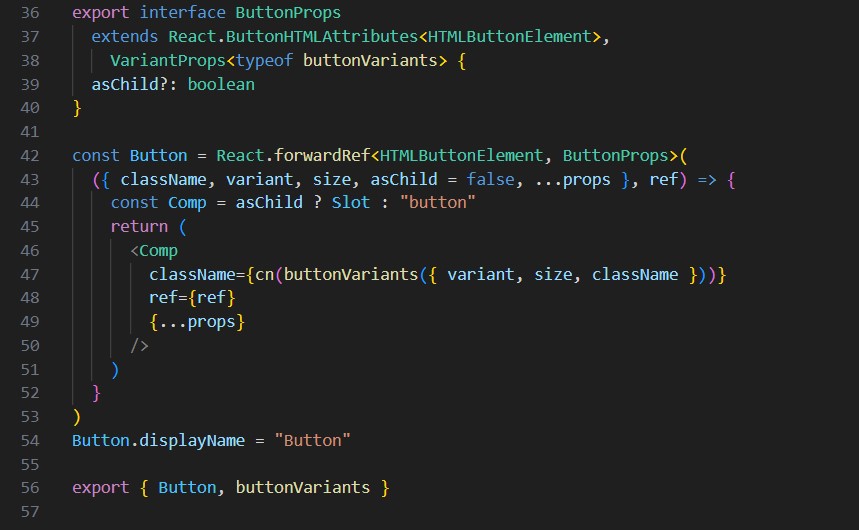


**Theme-provider.tsx:**

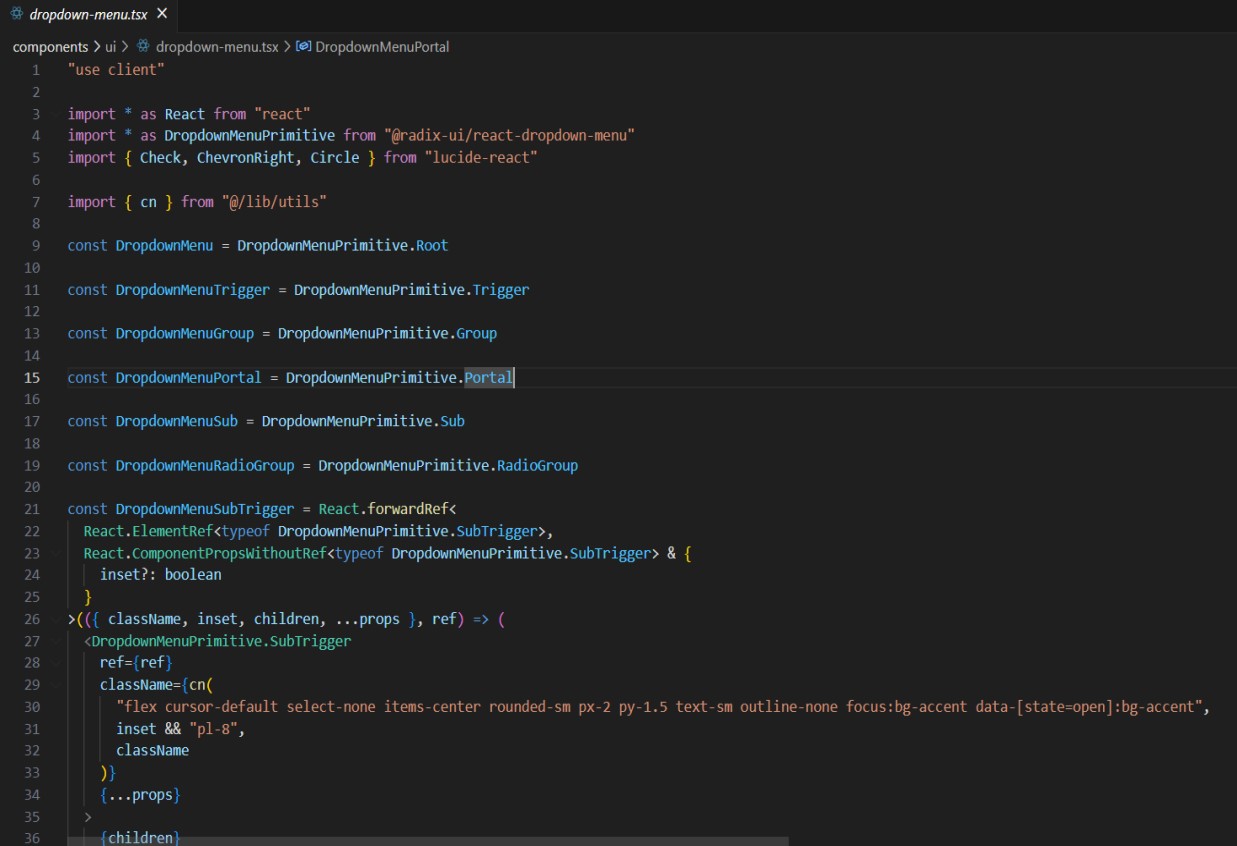


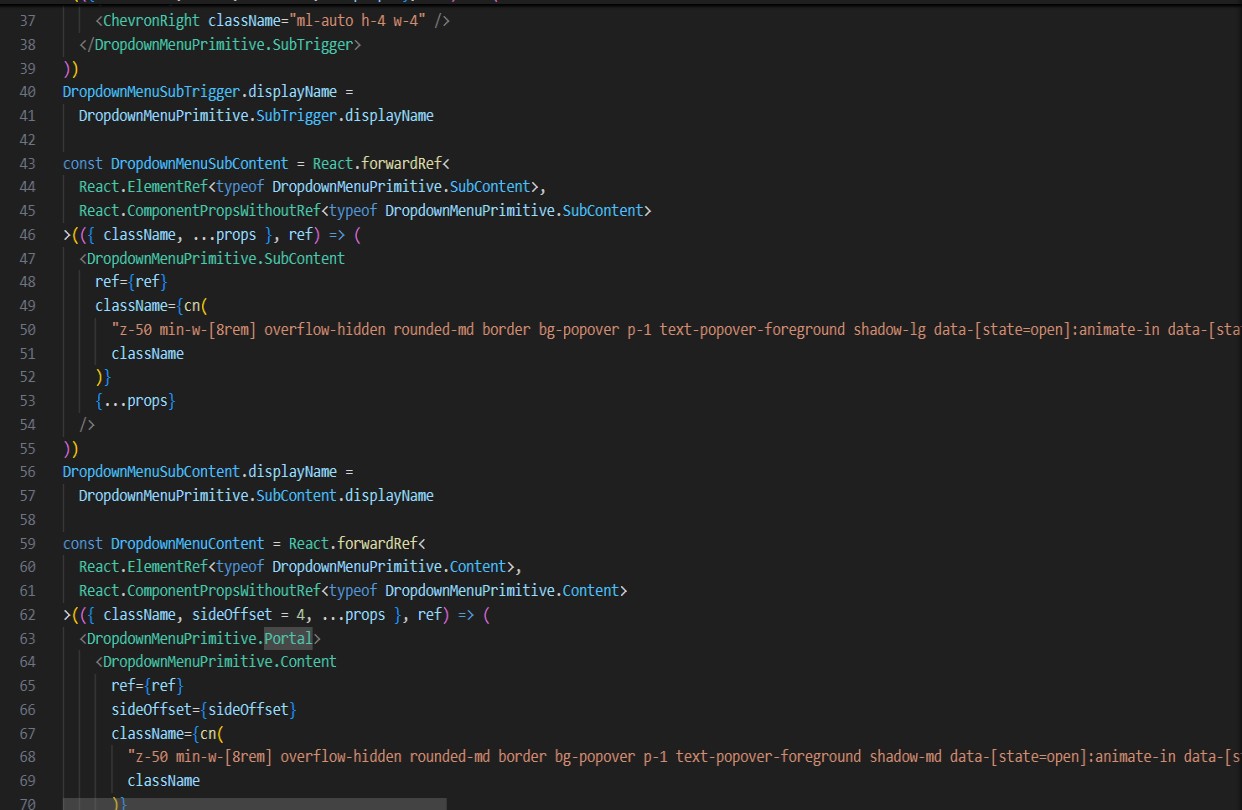
**Button.tsx:**

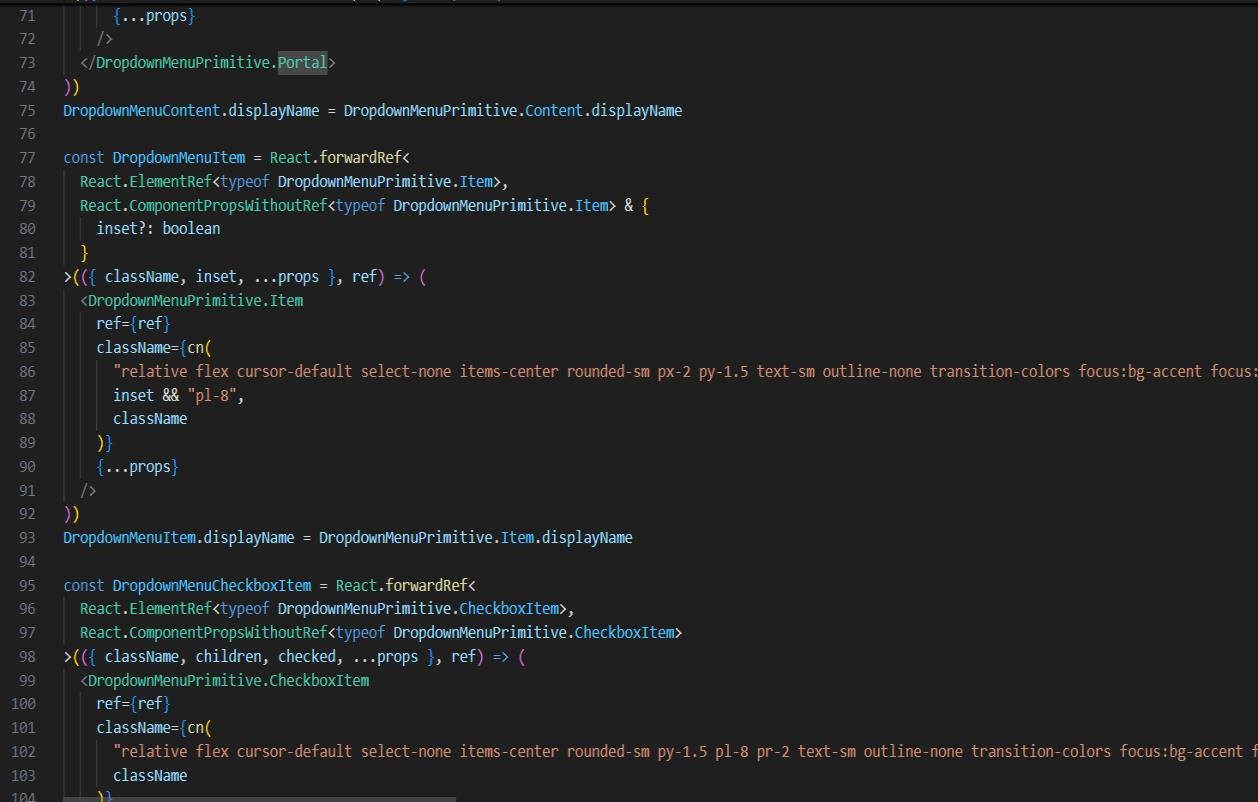


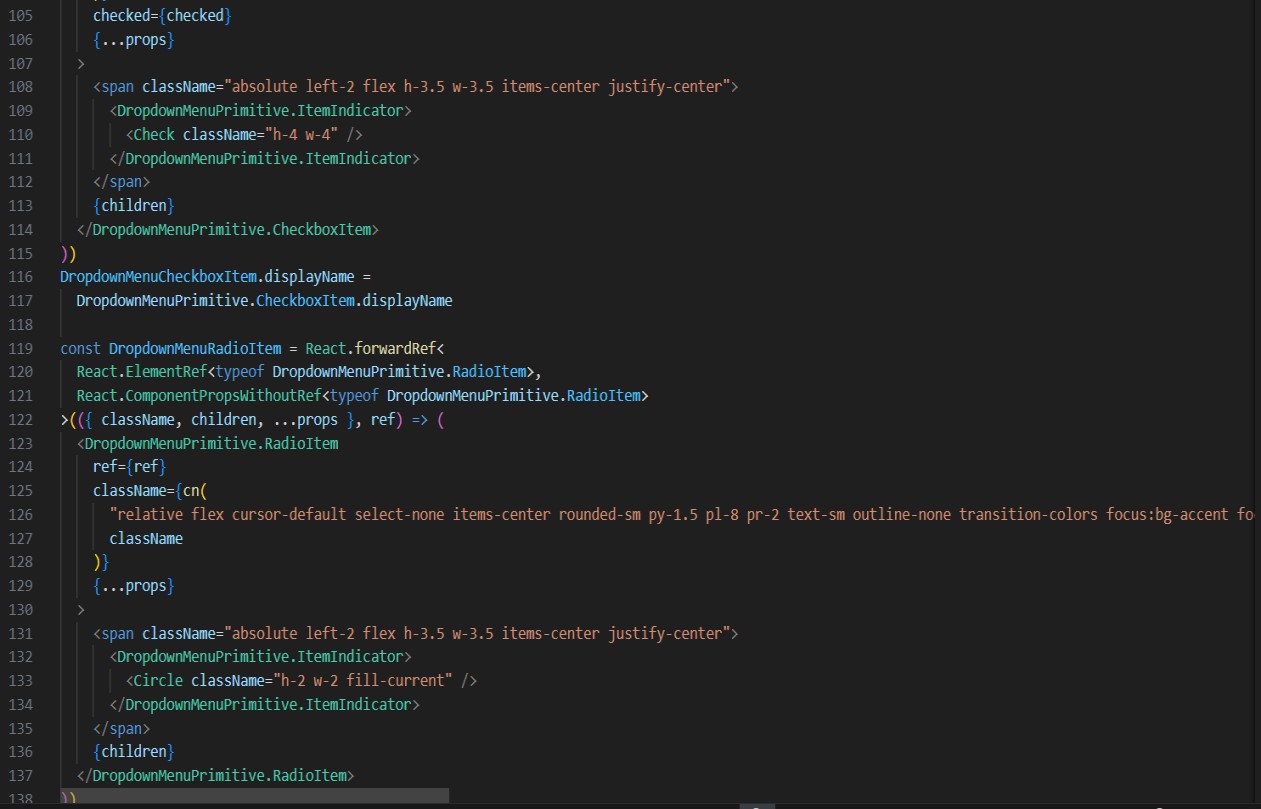


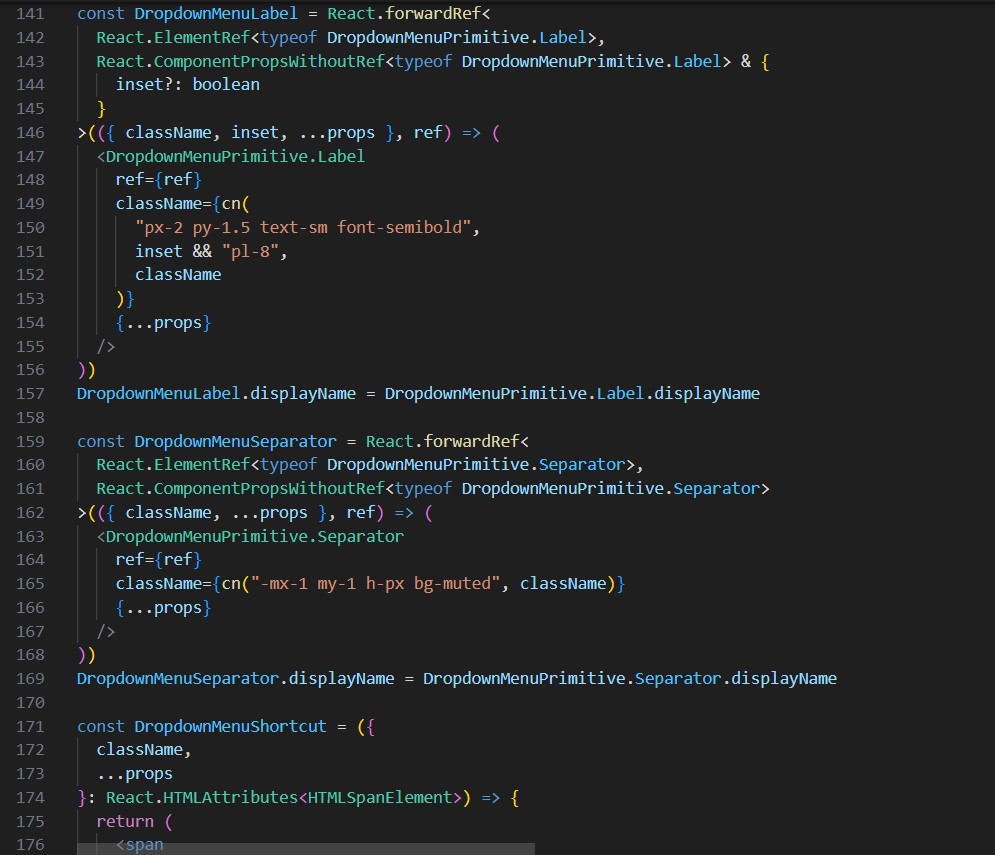
**Dropdown-menu.tsx:**

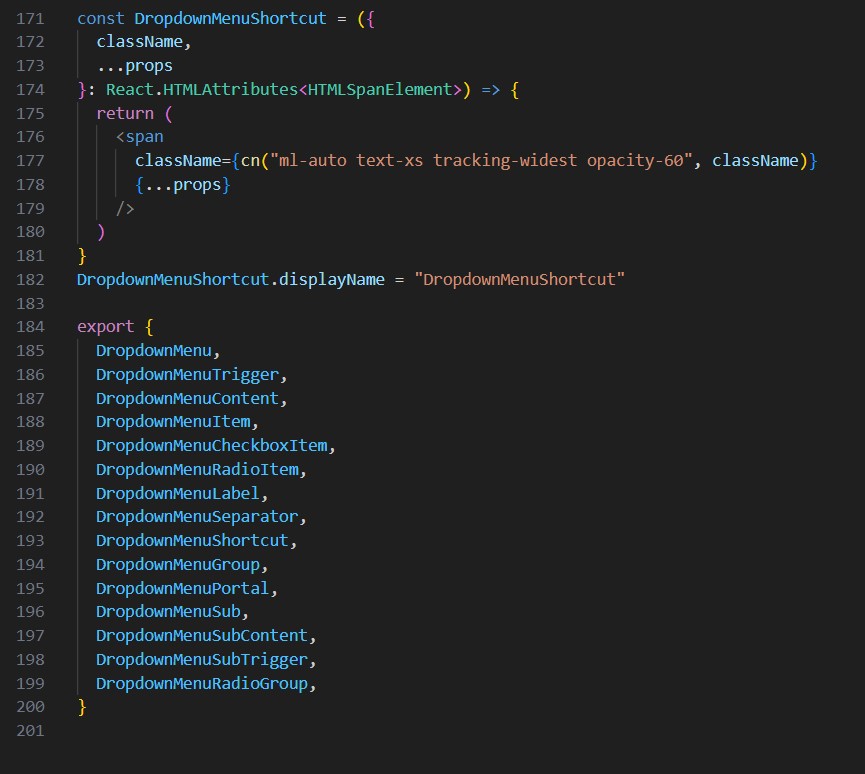




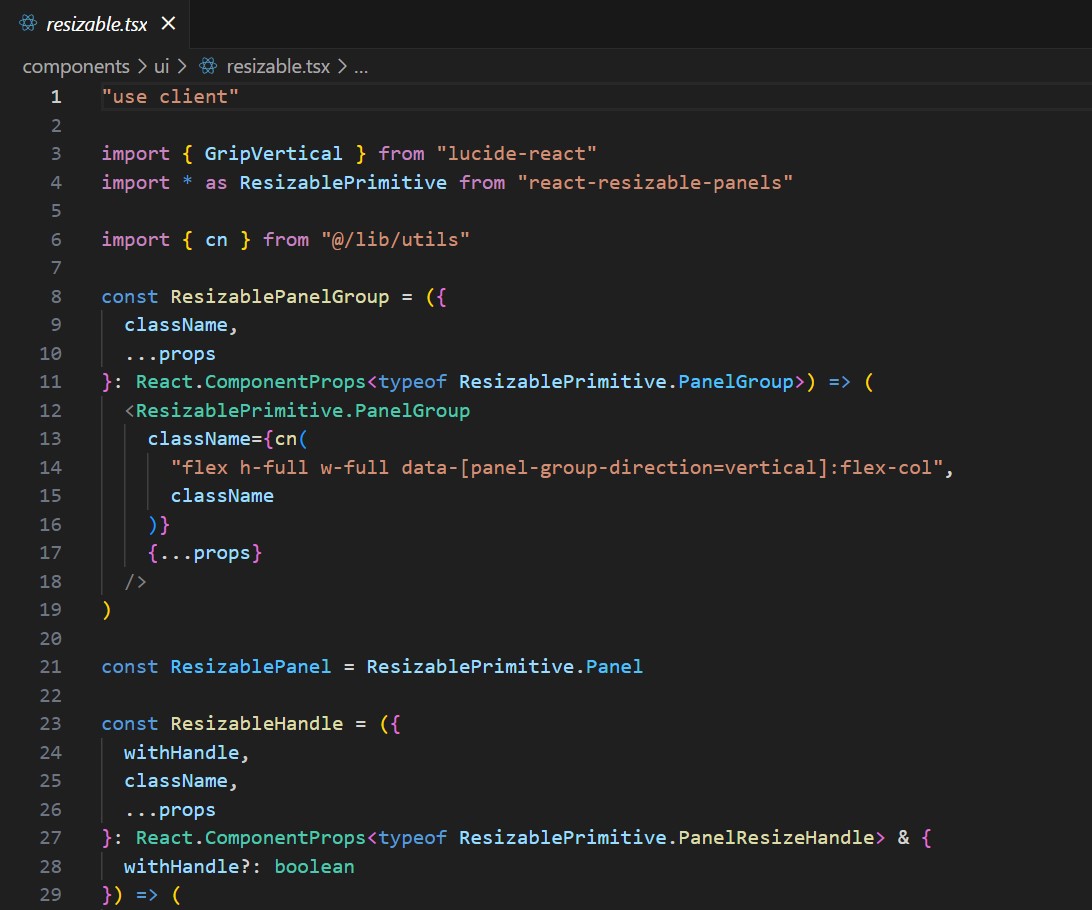


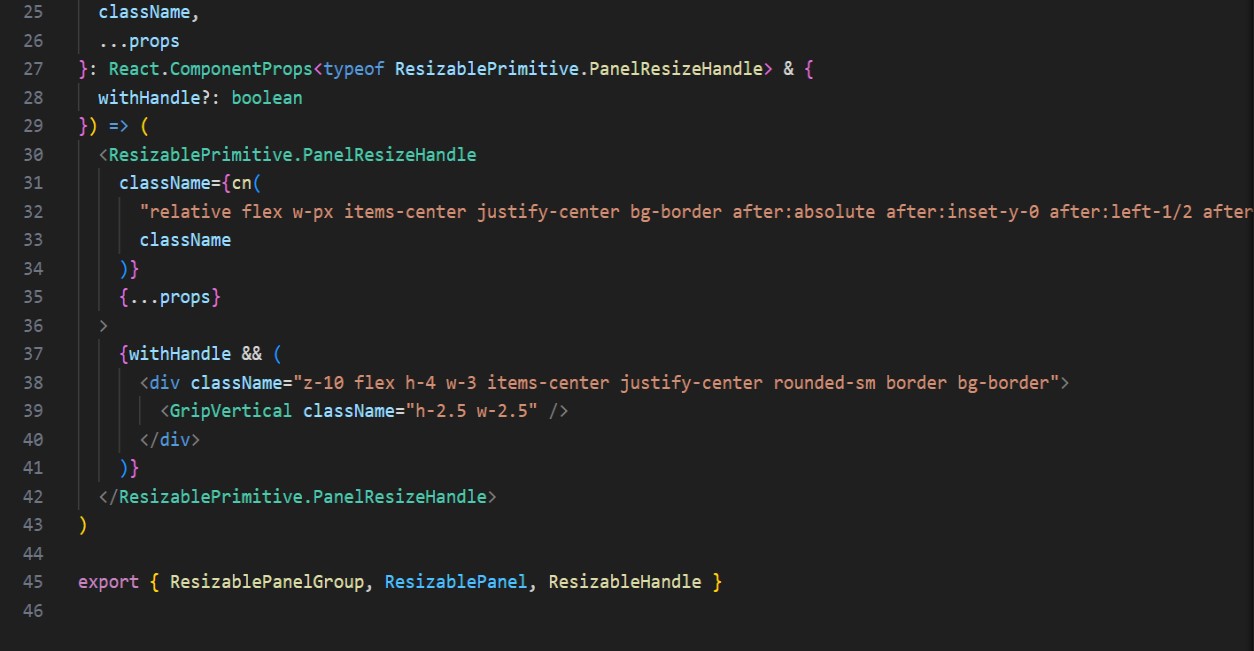




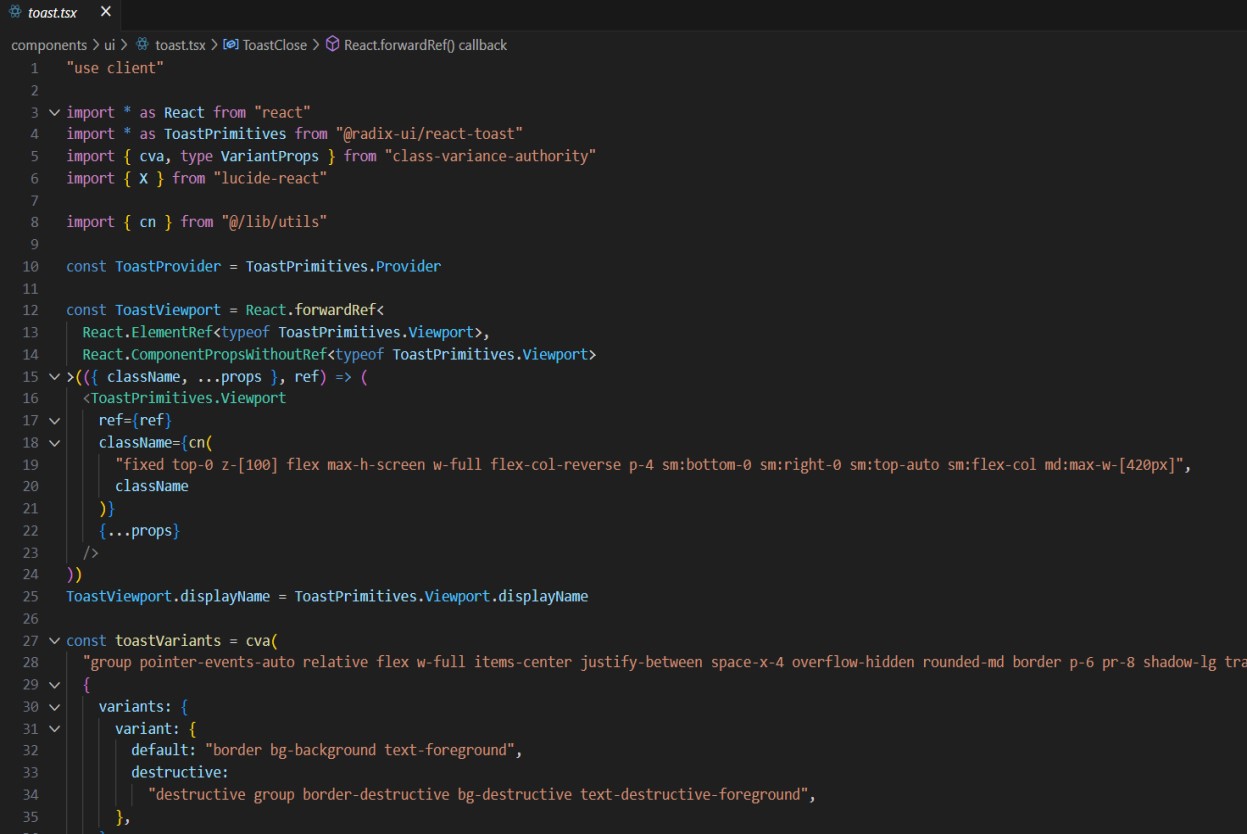


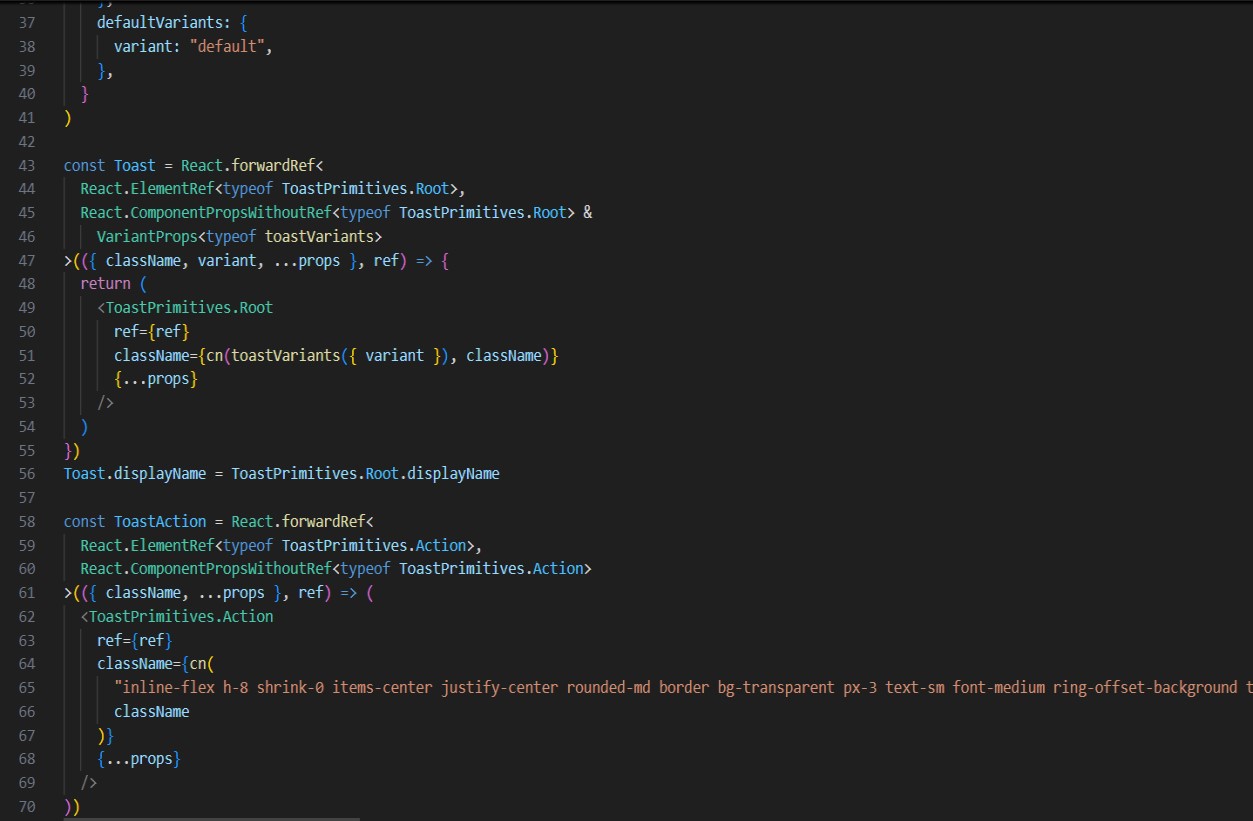
**Resizable.tsx:**

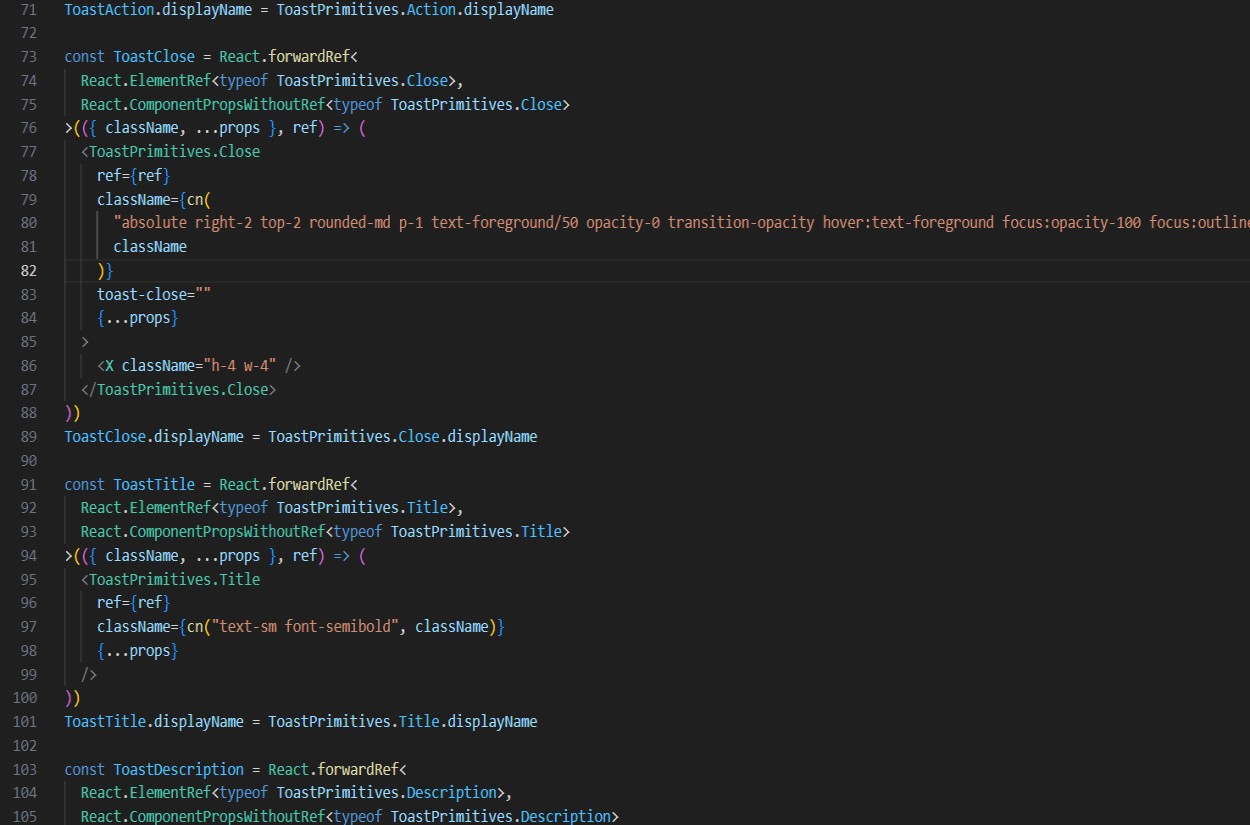


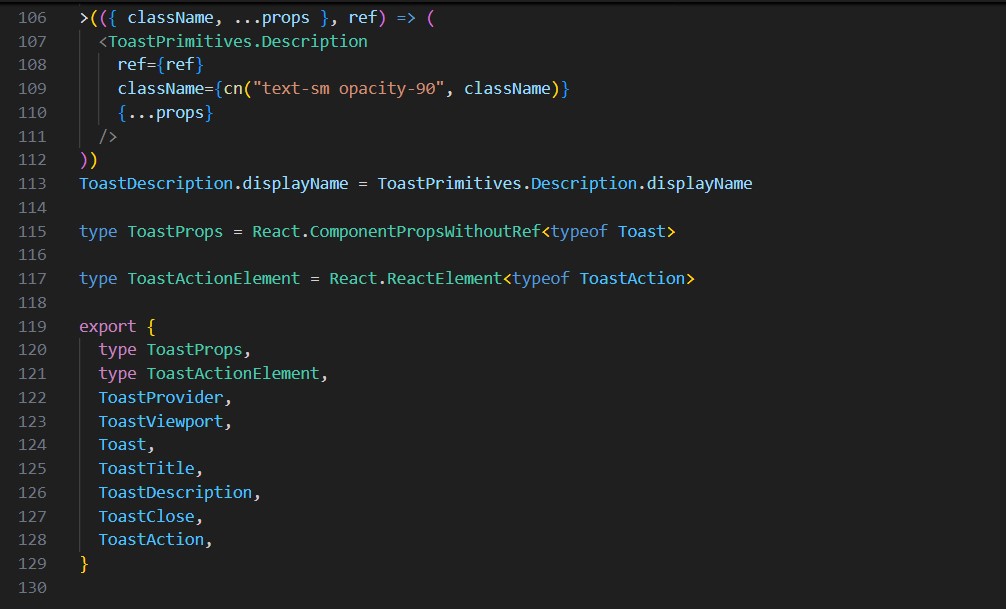


**Toast.tsx:**

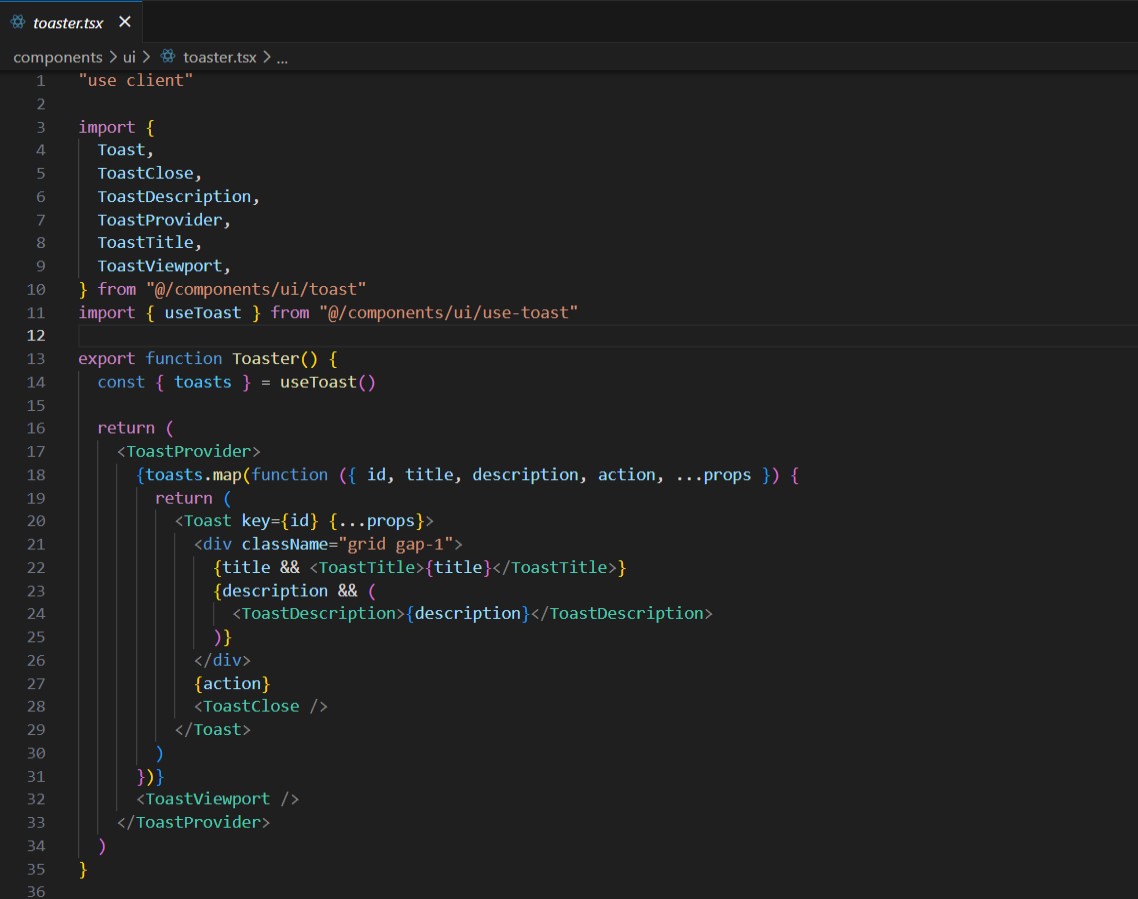




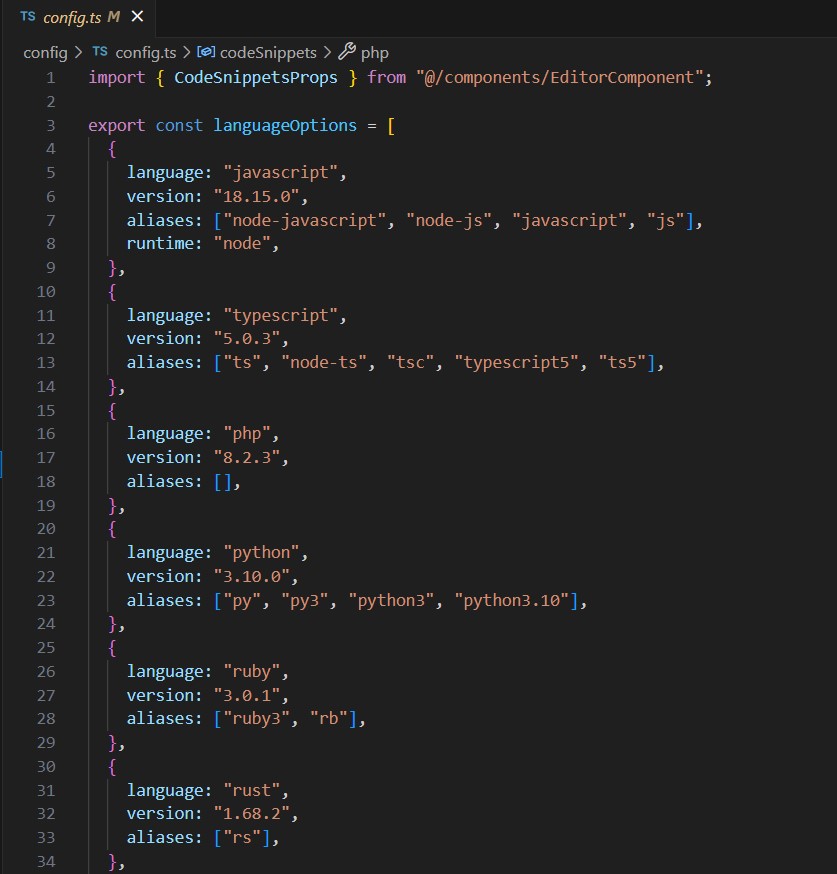


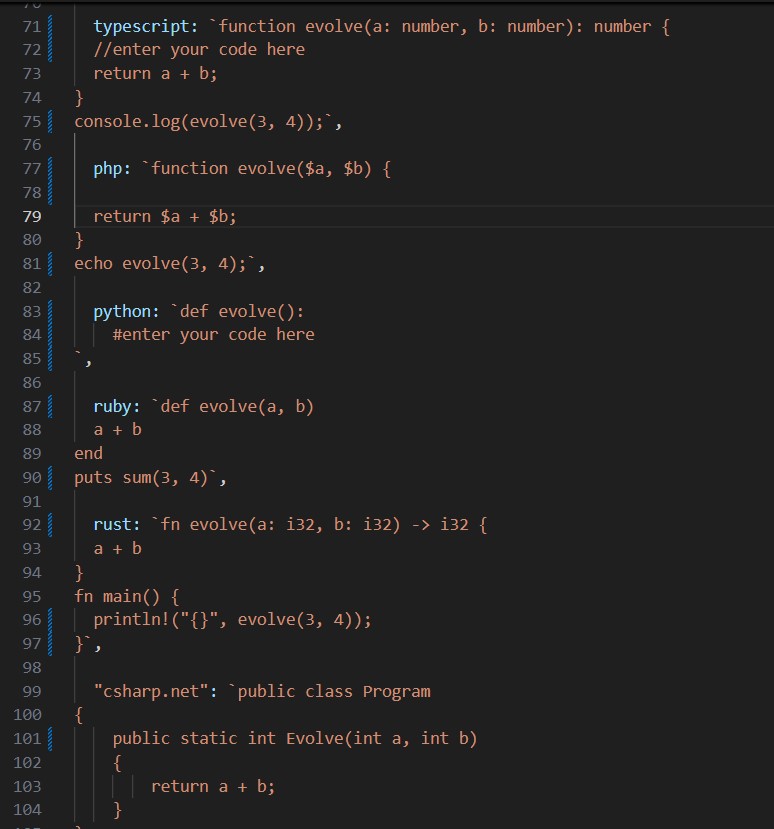
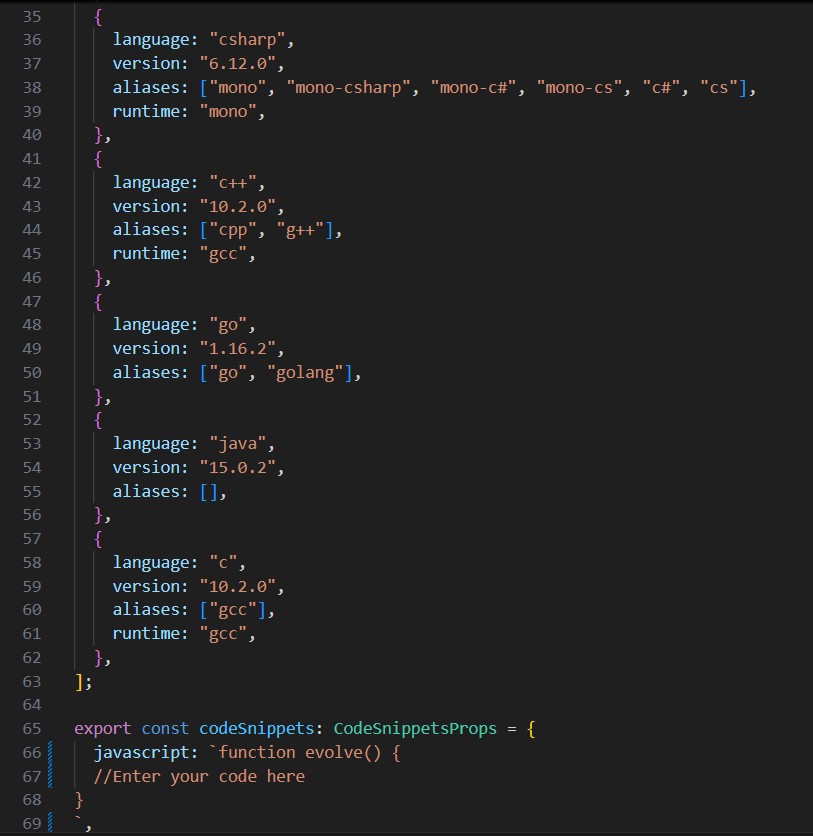


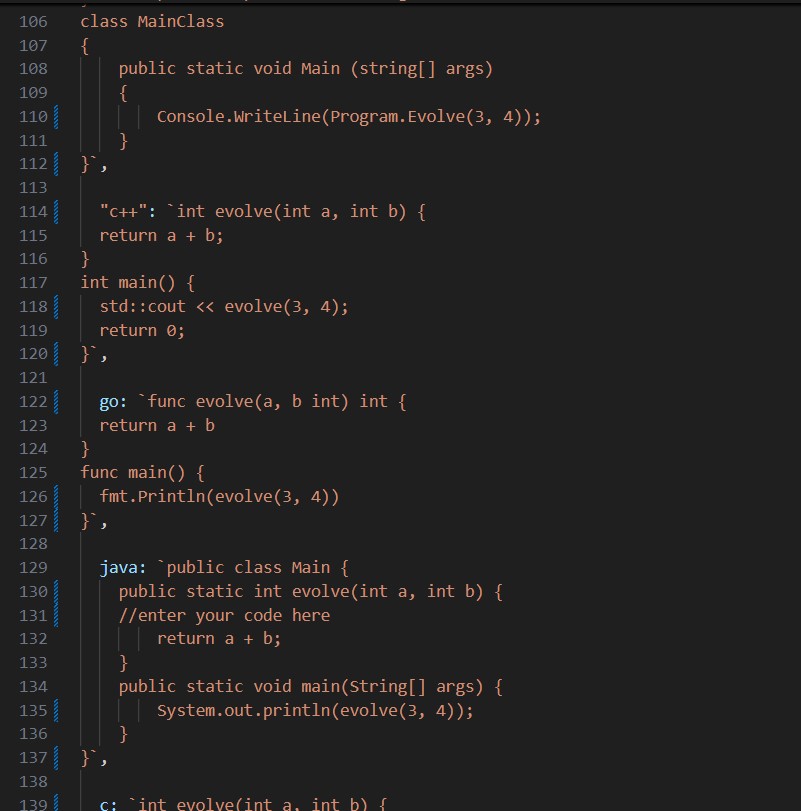
**Toaster.tsx:**

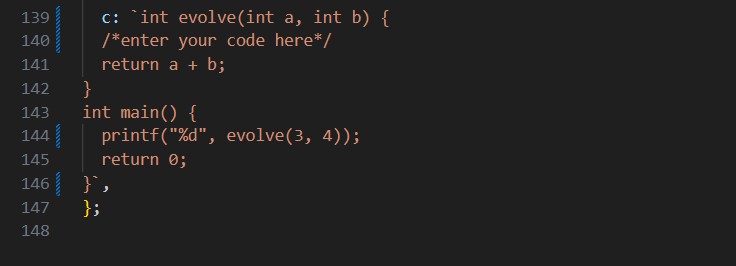


**Config.ts:**

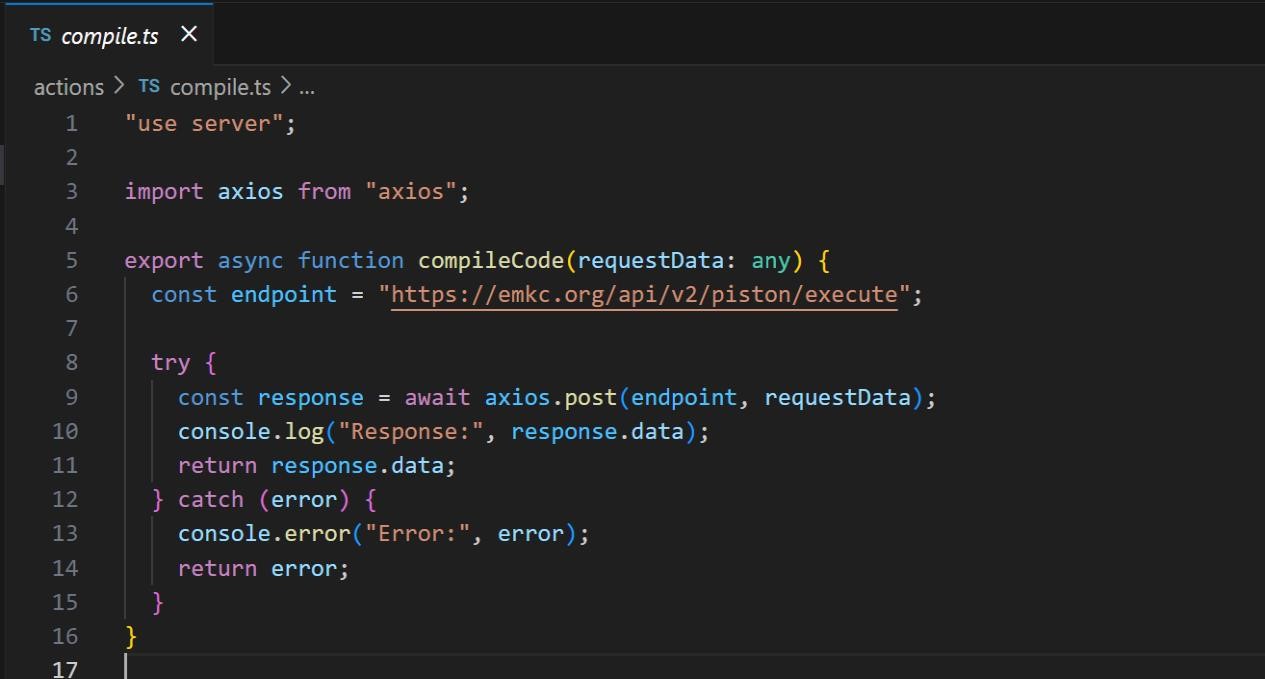




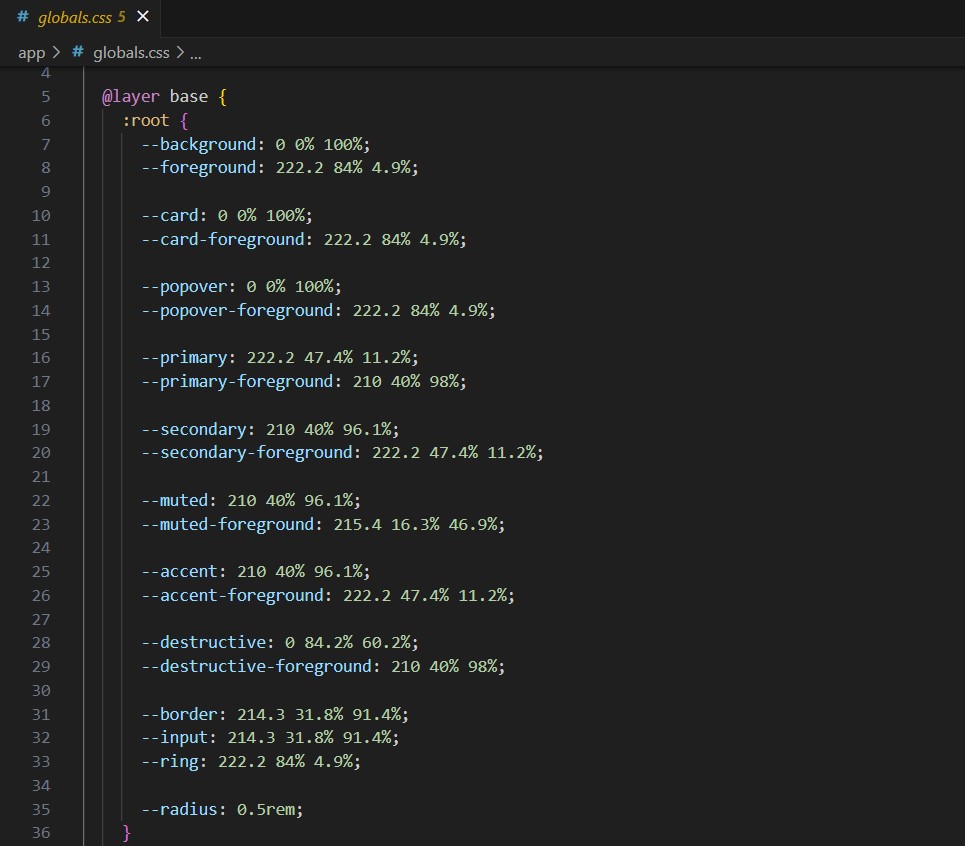


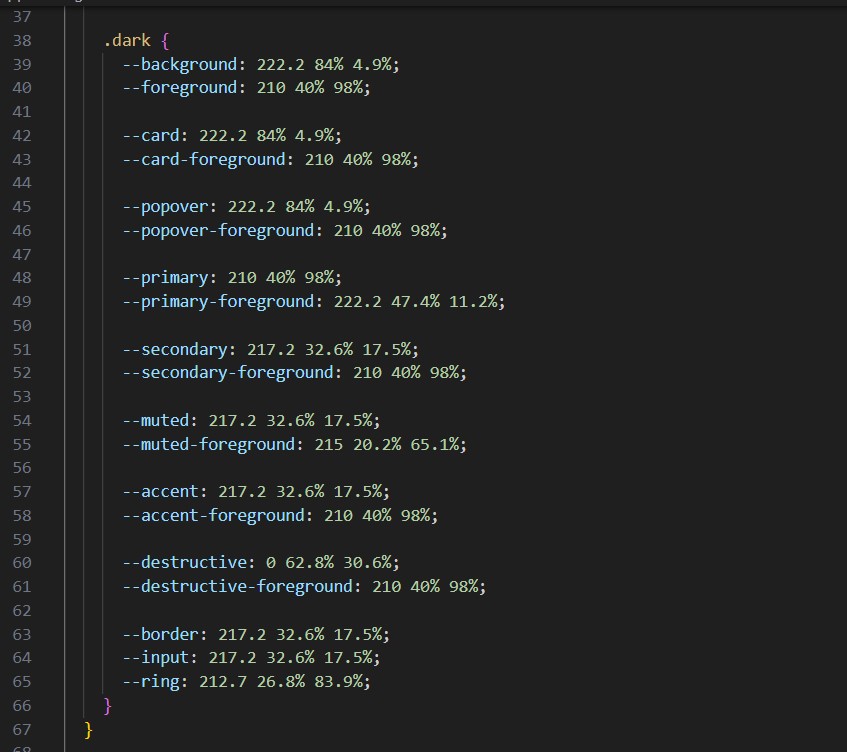


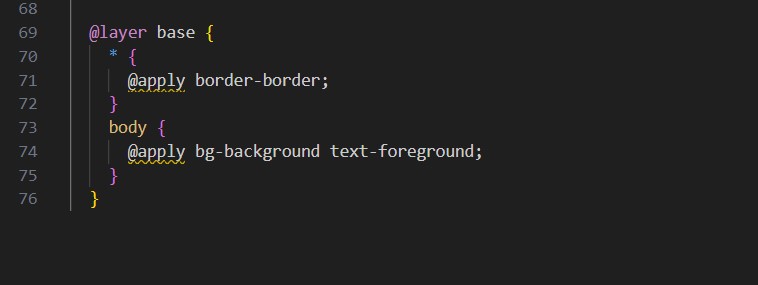
**Compile.ts:**



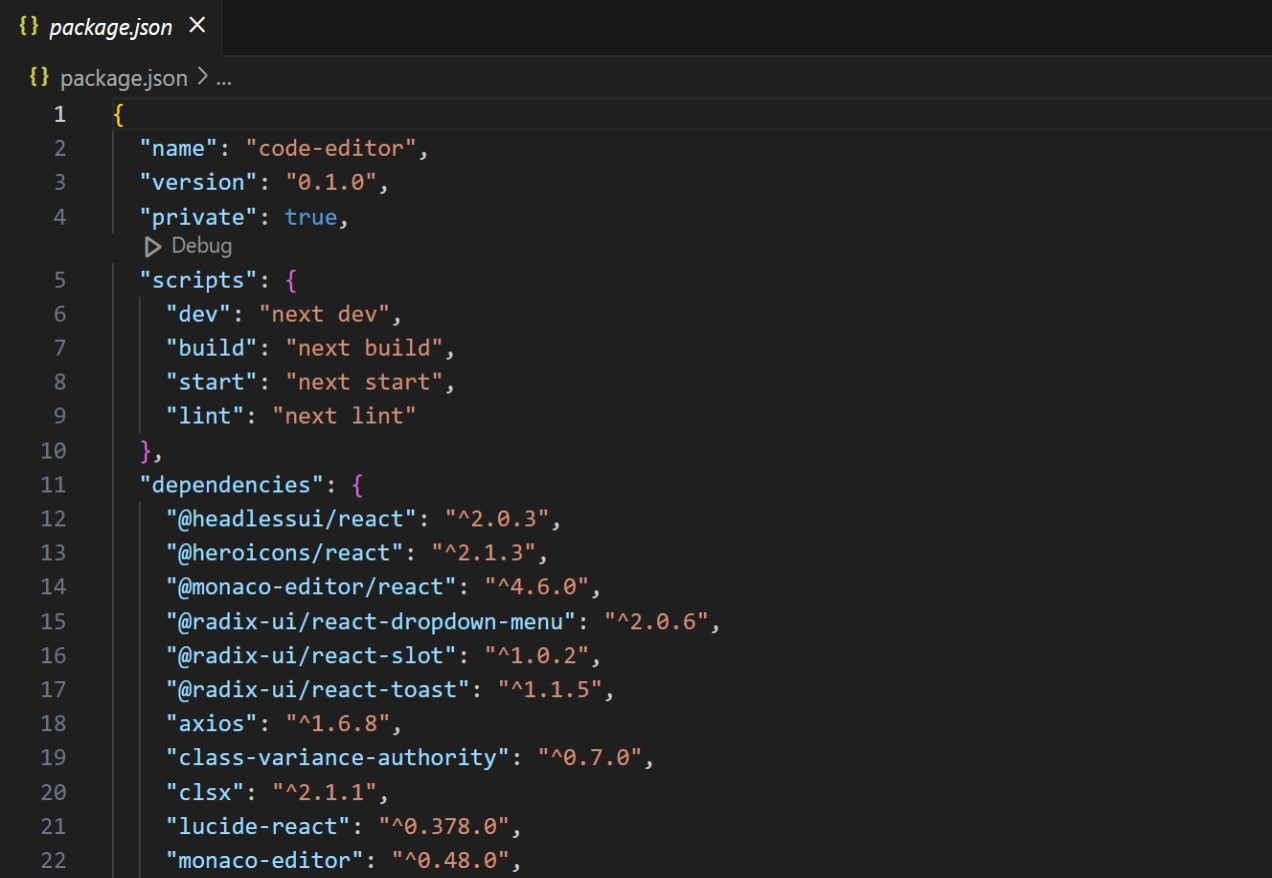
**Global.css:**







**Package.json:**





**6.FUTURE ENHANCEMENT:**

* We can integrate as many APIs as required, allowing the addition of multiple programming languages based on user needs.

o

* For future enhancements, we can incorporate Artificial Intelligence to enable automatic code generation, helping programmers complete tasks more efficiently.
* Personalized chatbots can also be integrated, allowing coders to track their daily progress and tasks.o The platform can easily be transformed into a collaborative website where multiple programmers can work together on a single interface.

**7. Conclusion:**

In conclusion, developing a web application like *Evolve Studio Code Editor* presents several challenges, but it also offers tremendous opportunities for innovation and enhancement in the coding experience. The primary goal of our project is to design a versatile, web-based code editor capable of supporting the successful implementation of programs across multiple programming languages. *Evolve Studio Code Editor* distinguishes itself by integrating advanced features that not only streamline the coding process but also improve developer productivity and code quality.

This application is more than just a traditional code editor; it incorporates cutting-edge technologies like intelligent code completion, real-time error detection to provide continuous feedback and guidance during development. These features enable developers to write cleaner, more efficient code while minimizing mistakes and reducing debugging time. Additionally, the editor's support for multiple programming languages makes it an invaluable tool for developers of all skill levels and backgrounds.

One of the key objectives of *Evolve Studio Code Editor* is to create an intuitive and user-friendly interface that promotes seamless interaction between the developer and the application. By focusing on accessibility and ease of use, we ensure that both novice and experienced developers can take full advantage of the editor's capabilities. In essence, the project not only aims to optimize the coding workflow but also to empower developers with the tools they need to produce highquality software efficiently.

Overall, the *Evolve Studio Code Editor* stands as a comprehensive solution for developers seeking to improve their coding proficiency, enhance productivity, and embrace modern development practices within a sophisticated and accessible environment.