General Code Quality & Best Practices: Checklist Prompts

Category: Architecture & Bootstrap

1. Prompt: Middleware Configuration Order

```
**Goal:**
Analyze `Program.cs` to verify that the middleware pipeline is configurec

**Context:**
You are an SRE agent reviewing application startup and configuration. In

**Source Code to Analyze:**
- `InventoryManagement.API/Program.cs`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the correctness of the middleware registratior
3.  **Evidence:** Quote the relevant section of the middleware pipeline f
4.  **Suggestion:** If `Fail`, provide the corrected order of method call
```

2. Prompt: CORS Policy Security

```
**Goal:**
Review the Cross-Origin Resource Sharing (CORS) policy defined in `Progra

**Context:**
You are a security review agent. The CORS policy is a critical security f

**Source Code to Analyze:**
- `InventoryManagement.API/Program.cs`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the security and specificity of the CORS polic
3.  **Evidence:** Provide the code snippet defining the CORS policy.
4.  **Suggestion:** If the policy is too permissive, recommend specific c
```

Category: Models & Types

3. Prompt: Model Validation Consistency

```
**Goal:**
Audit all classes in the `Models` folder to ensure consistent and appropr

**Context:**
You are a data integrity agent. To ensure data quality at the earliest st

**Source Code to Analyze:**
- All files in `InventoryManagement.API/Models/`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the completeness of the model validation.
3.  **Evidence:** Provide an example of a model property that is well-val
4.  **Suggestion:** If validation is missing, specify the model and prope
```

4. Prompt: EF Core Relationships & Naming

```
**Goal:**
Verify that relationships between EF Core models are correctly defined ar

**Context:**
You are a database schema review agent. Well-defined relationships (one-t

**Source Code to Analyze:**
- All files in `InventoryManagement.API/Models/`
- `InventoryManagement.API/Data/InventoryDbContext.cs`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the correctness and consistency of the data mo
3.  **Evidence:** Quote a correctly defined relationship or an example of
4.  **Suggestion:** If issues are found, recommend specific changes, such
```

Category: Services & Data Flow

5. Prompt: Efficient Database Queries

```
**Goal:**
Analyze the database queries in `InventoryController.cs` to identify pote

**Context:**
```

```
You are a performance optimization agent. When retrieving related data, c

**Source Code to Analyze:**
- `InventoryManagement.API/Controllers/InventoryController.cs`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the efficiency of the database queries.
3.  **Evidence:** Provide a code snippet of a query.
4.  **Suggestion:** If a query is inefficient, provide the optimized vers
```

6. Prompt: Service Layer Abstraction

```
**Goal:**
Re-evaluate `InventoryController.cs` to confirm that it contains no direc

**Context:**
You are an architectural review agent. As a follow-up to the initial revi

**Source Code to Analyze:**
- `InventoryManagement.API/Controllers/InventoryController.cs`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** 10 if the controller is completely free of `DbConte
3.  **Evidence:** Show the controller's constructor. It should inject a s
4.  **Suggestion:** If `DbContext` is still present, reiterate the need t
```

Category: Error Handling

7. Prompt: Global Exception Handling

```
**Goal:**
Check `Program.cs` for the presence of a global exception handling middle

**Context:**
You are a reliability agent. To prevent sensitive application details fro

**Source Code to Analyze:**
- `InventoryManagement.API/Program.cs`

**Expectations:**
```

```
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the implementation of global error handling.
3.  **Evidence:** Quote the code that registers the exception handling mi
4.  **Suggestion:** If no global handler is present, recommend adding one
```

8. Prompt: Specific Exception Handling

```
**Goal:**
Analyze the `try-catch` blocks within `InventoryController.cs` to ensure

**Context:**
You are a code correctness agent. Catching a generic `System.Exception` i

**Source Code to Analyze:**
- `InventoryManagement.API/Controllers/InventoryController.cs`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the specificity and correctness of the excepti
3.  **Evidence:** Provide a `try-catch` block from the source code.
4.  **Suggestion:** If a generic `catch (Exception e)` is found, recommer
```

Category: .NET Project & Build

9. Prompt: Project File (.csproj) Analysis

```
**Goal:**
Review the `InventoryManagement.API.csproj` file for obsolete package ref

**Context:**
You are a project maintenance agent. The `.csproj` file defines the proje

**Source Code to Analyze:**
- `InventoryManagement.API/InventoryManagement.API.csproj`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the health and configuration of the project fi
3.  **Evidence:** Quote relevant sections from the `.csproj` file, such a
4.  **Suggestion:** If issues are found, recommend specific changes like
```

10. Prompt: Environment Configuration

```
**Goal:**
Verify that the application correctly uses environment-specific configura

**Context:**
You are a deployment readiness agent. For security and correctness, certa

**Source Code to Analyze:**
- `InventoryManagement.API/Program.cs`

**Expectations:**
1.  **Verdict:** `Pass` or `Fail`.
2.  **Score (1-10):** Rate the implementation of environment-specific log
3.  **Evidence:** Quote the `if (app.Environment.IsDevelopment())` block.
4.  **Suggestion:** If production-unsafe features (like detailed error pa
```

10. Low Score changes to agent

```
can you do this changes and rerun the above prompt and check the scores
```