Functional Requirements

FR1: User Authentication and Roles

- **FR1.1**: System shall support user authentication for Admin, Tutor, and Student roles.
- **FR1.2**: System shall restrict access to features based on user roles.

FR2: Module Management

- **FR2.1**: Admin/Tutor shall be able to create new modules.
- **FR2.2**: Admin/Tutor shall be able to edit module details.
- **FR2.3**: Admin/Tutor shall be able to delete modules.

FR3: Assignment Management

- **FR3.1**: Admin/Tutor shall be able to create assignments for a module.
- **FR3.2**: Admin/Tutor shall be able to edit assignment details.
- **FR3.3**: Admin/Tutor shall be able to delete assignments.

FR4: Marking Script Management

- **FR4.1**: Admin/Tutor shall be able to create marking scripts using:
  - **FR4.1.1**: GATLAM
  - **FR4.1.2**: Random Number Generator
  - **FR4.1.3**: Coverage-based algorithm
- **FR4.2**: Admin/Tutor shall be able to delete marking scripts.
- **FR4.3**: Admin/Tutor shall be able to edit marking scripts.
- **FR4.4**: Admin shall be able to upload custom marking logic (not use interpreter).
- **FR4.5**: Interpreter shall translate marking script into executable code.

FR5: Grammar Input

- **FR5.1**: Admin shall be able to define terminals.
- **FR5.2**: Admin shall be able to define nonterminals.

FR6: Input Data Management

- **FR6.1**: Admin/Tutor shall be able to create input data:
  - **FR6.1.1**: Manually
  - **FR6.1.2**: Automatically (supporting seeding)
- **FR6.2**: Admin/Tutor shall be able to delete input data.
- **FR6.3**: Admin/Tutor shall be able to edit input data.

FR7: Code Submission

- **FR7.1**: Students shall be able to upload their code files.
- **FR7.2**: System shall hash uploaded student files using md5sum or quantum-safe hash.

FR8: Code Viewer and Runner

- **FR8.1**: System shall allow viewing code without downloading.
- **FR8.2**: System shall support syntax highlighting for code.
- **FR8.3**: System shall allow running code without downloading.
- **FR8.4**: System shall show output and stacktrace of execution.

FR9: Execution Environment

- **FR9.1**: Student submissions shall be run in containerized environments.
- **FR9.2**: Student output shall be matched to marker output outside of the container.

FR10: Plagiarism Detection

- **FR10.1**: System shall support plagiarism detection per assignment.

- **FR10.2**: System shall allow modular swapping of plagiarism algorithms (e.g., MOSS).
- **FR10.3**: System shall compare ASTs before invoking MOSS.
- **FR10.4**: System shall match code using GitHub Search API.

FR11: AI Assistance

- **FR11.1**: System shall provide AI-generated summaries of exceptions.
- **FR11.2**: System shall provide AI-generated summaries of incorrect outputs.

FR12: Gamification and Progression

- **FR12.1**: System shall support achievements and other gamified elements.
- **FR12.2**: System shall support unlocking tasks by completing previous tasks.

FR13: Grading System

- **FR13.1**: System shall calculate grades per assignment.
- **FR13.2**: System shall allow different grade weights per task.
- **FR13.3**: System shall display grades to students.
- **FR13.4**: System shall support time and space complexity analysis.

FR14: Submission Rules

- **FR14.1**: Admin shall be able to configure:
  - **FR14.1.1**: Submission deadlines (date and time)
  - **FR14.1.2**: Late submission policy
  - **FR14.1.3**: Submission count limit (including infinite)

FR15: Reporting and Statistics

- **FR15.1**: System shall provide live statistics per assignment.
- **FR15.2**: Statistics shall be available as downloadable reports.
- **FR15.3**: Statistics shall be displayed in graph form.

FR16: Security

- **FR16.1**: System shall restrict student access to memo content.
- **FR16.2**: System shall isolate containers to prevent memo leakage.

FR17: Support System

- **FR17.1**: System shall have a ticketing system (Feature Flag enabled).

Non-Functional Requirements (NFR)

NFR1: Performance

- **NFR1.1**: Code execution and result feedback should occur within 3 seconds for average code.
- **NFR1.2**: Plagiarism detection should complete within 60 seconds for assignments under 100 submissions.

NFR2: Scalability

- **NFR2.1**: System shall handle concurrent submissions from up to 1000 students.
- **NFR2.2**: Database and container systems shall scale horizontally.

NFR3: Availability

- **NFR3.1**: System shall have 99.9% uptime during the academic term.

NFR4: Usability

- **NFR4.1**: System UI shall support accessibility standards (WCAG 2.1).
- **NFR4.2**: Students and Tutors should be able to perform basic tasks with ≤ 3 clicks.

NFR5: Security

- **NFR5.1**: All code execution must occur in sandboxed containers.

- **NFR5.2**: Communication between client and server shall be encrypted via TLS.
- **NFR5.3**: Student files and sensitive data must follow role-based access control (RBAC).

NFR6: Maintainability

- **NFR6.1**: Codebase shall be modular and support hot-swappable plugins for AI, plagiarism, and grading engines.

NFR7: Compatibility

- **NFR7.1**: Web app shall support modern browsers (Chrome, Firefox, Edge).
- **NFR7.2**: Mobile responsiveness is optional (FF-based toggle).