Scoring an 'Almost There' or 'Struggling' in a category with the auto fail flag means the student is unable to pass the exam.

Java Belt Rubric

Competency	Objective	Binary	Auto Fail	Weight	PASS		INCOMPLETE	
					Mastery	**Proficient**	Almost There	Struggling
					Demonstrated mastery of skills and ability to synthesize concepts.	**Did the work as requested to the level needed in the field.**	Minimal or few errors. A bit more time studying or tinkering may fix it.	Many errors or an indication of misunderstanding of concepts, or clear student copy/pasted code from outside code base. A bit more time with the instructor requested.
					100.00%	**80.00%**	50.00%	0.00%
Competency	**Objective**							
Programming Concepts	Apply knowledge of datatypes and operations to solve problems and achieve the MVP features as designated by the wireframe		x	1	Flawlessly delivers on all features requested by the wireframe to achieve the MVP (including Stretch Goals). (1.0)	Successfully delivers on most features requested by the wireframe to achieve the MVP. Any features missing are minor and do not affect CRUD functionality. (0.8)	Falls just short of achieving MVP due to 1-3 missing or broken features that affect the CRUD functionality of the website. (0.5)	Missing 4 or more MVP features that affect the functionality of the website. (0.0)
Java Basics	Apply common programming concepts using Java		x	0.4	Demonstrates understanding of core Java concepts (0.4)		The web app does not run (0.2)	Code is very difficult to follow and does not make sense (0.0)
OOP	Correctly implement member variables, getters, setters and constructors as necessary.			1	Flawless implementation of models with getters, setters, and model level validations. (1.0)	Implements models with getters, setters, generally uses appropriate data types, access modifiers and validation annotations. (0.8)	One or two models are missing one member variable (0.5)	Models have fatal errors, missing key elements, fails to compile due to entity creation errors. (0.0)
Routing, Rendering & Basic Backend Logic	Create Routes to serve responses from client requests			0.4	Utilize GET and POST requests to industry standard (0.4)	All routes are functional, but may use GET routes where POST routes are preferable (0.32)	Missing 1-2 routes that affect functionality (0.2)	Missing all or several routes that affect functionality (0.0)
	Session: Manipulate session to access user data across requests.		x	1	User is unable to access any pages they shoudn't have access to. Each page should check for the user id in session. (1.0)	Able to log in and log out, retaining the user's ID. Successfully retrieves and utilizes session id to display data correctly according to the wireframe. (0.8)	Logout does not clear the user ID from session. Or uses session inappropriately, ie. logged in user is displayed when it should be another user. (0.5)	ID is not saved or saved and accessed incorrectly in session. (0.0)
ORM & Database	Implement CRUD functionality using JPA queries		x	1.6	CRUD functionality required for wireframe is completely implemented. Advanced queries are used in the Repository. (1.6)	CRUD functionality required for wireframe is mostly implemented. Appropriate queries are used in the Repository. DELETE functionality may not work correctly. READ must return something. (1.28)	CREAT, READ, or UPDATE is incomplete or broken. For example, when attempting to UPDATE, creates a new record. CREATE and/or READ are not functional. (0.8)	Multiple CRUD functionalities are missing or broken. Unable to CREATE or READ. (0.0)
	Implement required relationships between user and exam specific entity/entities			1.6	Successfully implements multiple one-to-many relationships or a many-to-many relationship indicated on the wireframe with full CRUD functionality. (1.6)	Successfully implements a one-to-many relationship with full CRUD functionality. (1.28)	When attempting to edit, loses one-to-many relationships or one-to-many relationships have other functional issues. (0.8)	Unable to query successfully for one-to-many relatioinships (0.0)
Full-Stack MVC	Include basic user authentication with password hashing		x	1	Implements BCrypt hashing. Login and registration is fully functional. Login required to access all pages. (1.0)	Implements BCrypt hashing. Login and registration is fully functional. Login required to access dashboard/home page. (0.8)	Login and registration is mostly functional. Users are able to access all pages without logging in. (0.5)	Login and registration is broken. (0.0)
	Include login and registration validation			0.5	All validation errors are present (0.5)	80% of validation errors are present (0.4)	60% of validation errors are present (0.25)	Fewer than 60% of validation errors are present (0.0)
	Use the controller-service-repository modularization	b		0.5	Project contains models, repositories, services, and controller(s) (0.5)		Project does not contain models, repositories, services, and controller(s) (0.0)	
	Build in pattern, logic, and type validation on CRUD features.			1	Prevents non-owning users from altering data, and/or has more difficult or custom validations, such as date, time, or pattern-matching (1.0)	Core validations are fully functional as per the wireframe on create and edit (0.8)	Validations work but do not appear on the page, or one of either create or edit does not have full validations (0.5)	Little or no validations, or critical errors on submission regarding validations (0.0)