Dispatching dashboard

ekkie.ai needs a dashboard based on the data we have on solved tickets. This dashboard will be made for the customer to look at and it will **gamify* ticket-solving for support engineers. It will do so by adding a point system to solving tickets, which we base on the line the ticket is sorted in (e.g., line 1 ticket is 100 point, line 2 is 200 points, so forward). This way difficulty should get rewarded with extra points. The dashboard will display the score of the engineers individually and grouped based on their line, this should be with filters.
For the implementation of databases use mocks, make sure to maintain loose coupling.

****gamify*** *(gamification) = the process of enhancing non-game activities by integrating game design elements and principles. It aims to make tasks more engaging and enjoyable, often by adding rewards and challenges to motivate participation and productivity.*

Here is a MoSCoW of the requirements:

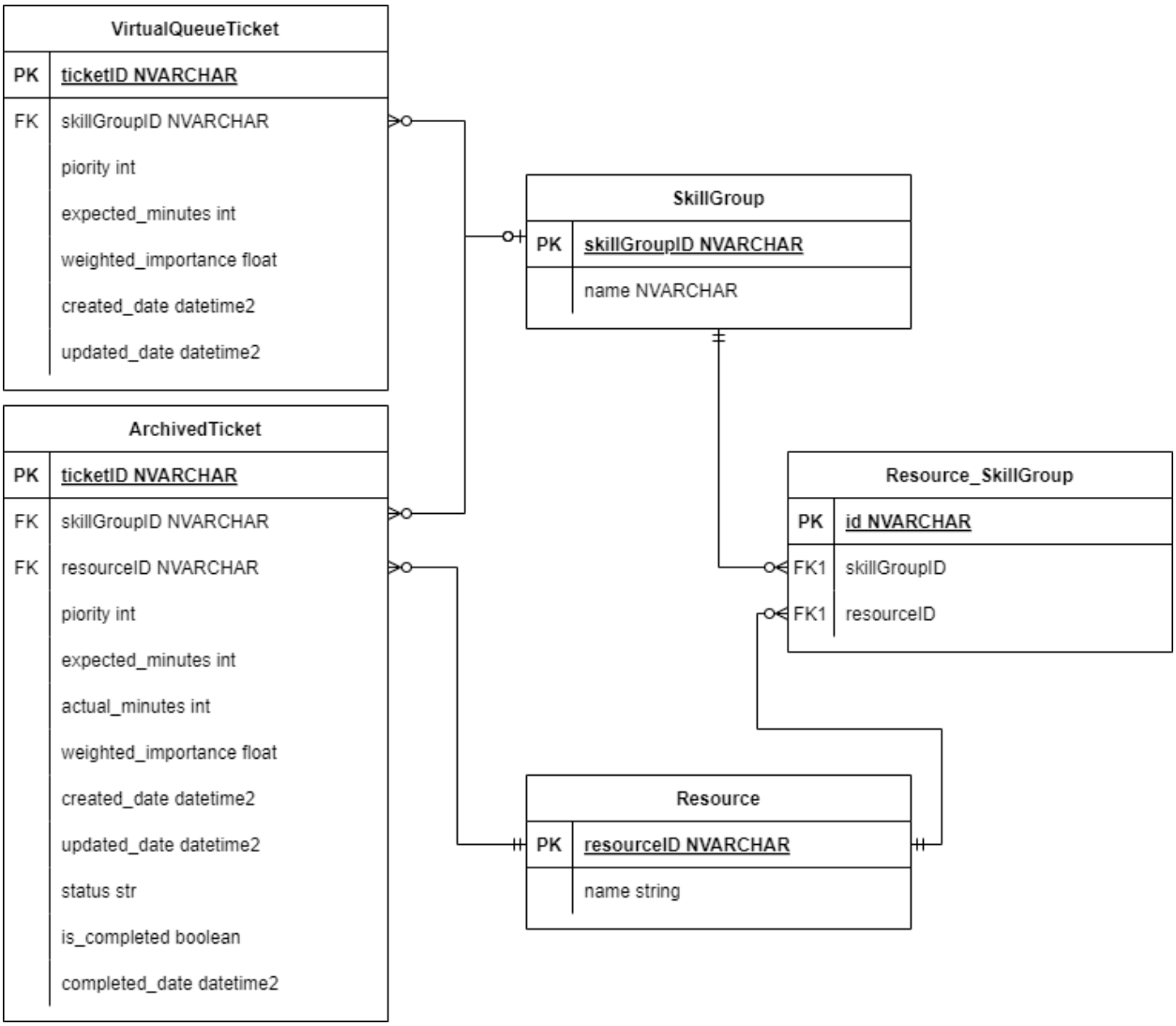
EPIC	User story	Requirements	MoSCoW
Settings	As a user I want to be able to switch between dark and light mode, so my eyes don't burn unless I want them to.	Simple settings page, with a switch.	C
Dashboard	As a support engineer, I want to see my standing on the leaderboard, so I can compare my standing with the others.	Display: • A rank number • Amount of points Show other engineers	M
	As a support engineer manager, I want a quick overview of how many tickets		M

	have been solved by each engineer, so I can judge them very harshly on their performance.		
Authtentication	As a security officer, I want authentication to use OAuth 2.0 and OpenID Connect, so that the system complies with organizational security standards.	Use an app registration to authenticate users across tenants The token should be validated against Entra ID Role based	M

Non-functionals

Security	Authentication must comply with OAuth 2.0 and OpenID Connect specifications.
Maintainability	The application shall define external dependencies (e.g., data access, external services, APIs) behind interfaces to allow loose coupling.
Testability	The system shall provide the ability to substitute real implementations with mocks or stubs at runtime or during testing.
Maintainability	The system architecture shall support dependency injection to simplify the replacement of service implementations with mocks.

ERD


VirtualQueueTicket
PK ticketID NVARCHAR
FK skillGroupID NVARCHAR
piority int
expected_minutes int
weighted_importance float
created_date datetime2
updated_date datetime2
SkillGroup
PK skillGroupID NVARCHAR
name NVARCHAR
ArchivedTicket
PK ticketID NVARCHAR
FK skillGroupID NVARCHAR
FK resourceID NVARCHAR
piority int
expected_minutes int
actual_minutes int
weighted_importance float
created_date datetime2
updated_date datetime2
status str
is_completed boolean
completed_date datetime2
Resource_SkillGroup
PK id NVARCHAR
FK1 skillGroupID
FK1 resourceID
Resource
PK resourceID NVARCHAR
name string

MOCKUP - frontend