QUERST MATCHMAKING SYSTEM - DETAILED OVERVIEW

Purpose and Vision

The QuERST Matchmaking System (QuERST-SyMM) is an end-to-end platform that connects students with qualified tutors and guides them through assessments, scheduling, and academic programs. It aims to streamline the entire tutoring lifecycle-onboarding, pairing, scheduling, exam delivery, and progress tracking-while giving administrators transparency and control over the process.

Core Product Pillars

- **Role-Based Experience** - Students, tutors, and administrators receive tailored dashboards and permissions that surface the actions most relevant to them.
- **Pairing and Programs** - Administrators can structure tutoring programs, assign participants, orchestrate pairings, and leverage recommendation tooling.
- **Scheduling and Availability** - Both students and tutors maintain detailed weekly availability, enabling the system to identify overlaps and assign sessions.
- **Assessment and Analytics** - Exams, questionnaires, and grading workflows underpin personalized recommendations and performance tracking.
- **Data-Backed Insights** - Analytics artifacts and recommender prototypes inform continuous improvement to matching quality.

Platform Architecture

Frontend
- Next.js App Router with React 19 and TypeScript for composable UI.
- Tailwind CSS plus shadcn/ui for component styling parity.
- Client/server component split with React Query for mutation and cache management.
- Route grouping for authentication (‘src/app/(auth)‘), dashboards, schedules, exams, and admin consoles.

Backend
- MongoDB with Mongoose models handling user accounts, schedules, matches, programs, exams, and derived analytics artifacts.
- Server Actions and REST endpoints provide CRUD access, share validation with Zod schemas, and enforce permission boundaries.
- NextAuth Credentials provider authenticates users and injects role metadata into JWT-backed sessions.

Supporting Assets
- Docker Compose spins up a local MongoDB instance for development parity.

- ‘recommender_system python/‘ contains Pydantic models, notebooks, and data transformations to prototype ML-driven pairing logic.
- ‘data analysis/‘ houses statistical studies on BFI/VARK data, tutor-student relationships, and regression findings.

Feature Walkthrough

Authentication & Onboarding
- Users self-register, log in, and receive role-specific routing guarded by middleware and dashboard checks.
- Onboarding status determines whether users must complete profile setup before accessing core features.

Dashboards
- ‘src/app/dashboard/page.tsx‘ selects a dashboard layout based on the authenticated user’s role.
- Students focus on assigned assessments and completion metrics.
- Tutors view paired students alongside exam attempts and status summaries.
- Admins access high-level stats, quick links to program/exam tooling, and user management utilities.

Program & Participant Management
- Admins create tutoring programs with scoped timelines and descriptions (‘src/app/admin/programs/page.tsx‘).
- Participants can be bulk-assigned or removed, with safeguards to cascade pairing data when needed (‘src/app/admin/programs/assign/page.tsx‘).
- Program-specific pairings (‘src/app/admin/programs/pairings/page.tsx‘) replace the legacy global pairing flow, offering:
- Tutor and student selection via searchable comboboxes.
- Automated similarity scoring and suggestion refreshing.
- Batched confirmation of recommended matches.
- Removal workflows with confirmation gates.

Scheduling and Paired Availability
- Tutors and students configure weekly availability using a drag-select calendar component (‘src/components/schedule/ScheduleSelector.tsx‘).
- Schedules persist to MongoDB via server actions (‘src/lib/actions/scheduleActions.ts‘) and support removal or reassignment.
- When pairings are approved, administrators can assign intersecting timeslots to lock in tutoring sessions and push them to both schedules.

Assessment Lifecycle
- **Creation** - Admins and tutors build exams with Markdown-enabled content, multiple question types, and grading metadata (‘src/app/exams/create/page.tsx‘).
- **Assignment & Tracking** - React Query-backed dashboards expose assigned exams grouped

by assessment type, completed status, and scoring (‘src/components/exams/AssignedExams.tsx‘).
- **Delivery** - During exam attempts, ‘ExamContext‘ manages autosave, navigation, timer tracking, and submission (‘src/app/exam/[id]/page.tsx‘).
- **Grading** - Server actions compute scores, tally correctness, and store results for later review.
- **Specialized Questionnaires** - Dedicated admin UIs map Big Five Inventory and VARK learning-style attributes to exam questions and choices, driving richer learner profiles.

User Directory and Profiles
- Admins can search, filter, and inspect user accounts (‘src/app/admin/users/page.tsx‘) to verify onboarding, role assignments, and activity.
- Detailed profile pulls combine schedules, pairings, exam attempts, and special-exam status while enforcing access restrictions (‘src/lib/actions/getUserProfileAction.ts‘).

Data Model Highlights

- **Account** - Stores authentication credentials, role type, and onboarding state.
- **Match** - Captures the lifecycle of tutor-student pairings, including subject focus and status transitions.
- **Schedule** - Persists weekly availability down to day-level intervals and session assignments.
- **Exam** / **Question** / **Choice** - Compose assessments and metadata for grading.
- **ExamStatus** / **ExamAnswers** - Track user attempts, multiple attempts per exam, and granular responses.
- **Program** - Groups students and tutors, centralizing pairings and admin oversight.
- **SpecialExam** - Flags exams tied to BFI or VARK for quick lookup by analytics surfaces.

Analytics & Recommendation Initiatives

- Python notebooks (‘recommender_system python/‘) explore feature engineering for pairing models, combining availability, competencies, learning styles, and personality traits.
- Data analysis outputs (‘data analysis/‘) document regression experiments and pairings datasets to validate heuristics or inform ML pipelines.
- Future roadmap includes real-time messaging, payment integration, advanced analytics, LMS integrations, and mobile outreach (‘docs/project_overview.md‘).

Operational Considerations

- **Environment Variables** - ‘AUTH_SECRET‘ and ‘MONGO_DB_URI‘ power authentication and persistence. Docker automation supplies local credentials.
- **Testing & Verification** - React Query patterns and server actions are ready for

integration tests; exam workflows include autosave to mitigate connectivity issues.
- **Security & Permissions** - Role checks guard server actions, NextAuth session callbacks refresh data, and pairing actions validate tutor/admin access.

Summary

QuERST-SyMM delivers a modular tutoring ecosystem that spans from onboarding and scheduling through assessments and analytics. Its Next.js + Mongo stack supports rapid iteration, while admin tooling, specialized questionnaires, and data science artifacts lay the groundwork for evidence-based matchmaking and student success insights.